Asian Journal of Information Technology 15 (24): 5145-5153, 2016
ISSN: 1682-3915

Cooperative Media Streaming and Transcoding on Live Video Chat over VoIP

[1]J.T. Anita Rose, [2]A. Chandrasekar and [3]D. Frederick Swartz Daniel
[1]Manonmaniam Sundaranar University, Tamil Nadu, India
[2]St. Joseph's College of Engineering, Tamil Nadu, India
[3]Anna University, Tamil Nadu, India

Abstract: Recent growth in popularity of mobile services raises a demand for transcoding. Transcoding is the most important and a convenient method for delivering multimedia content to the heterogeneous devices over WLAN. However, currently existing mobile devices such as smart phones, tablet PCs and smart TVs involves an issue of separate video transcoding for each type. Thus, additional burden comes to media servers which pre-transcode the multimedia data for number of clients and hence the problem of media servers overload. A straight forward solution for relieving media servers can be to localize transcoding of a video to the client side, that is, to move it from a media server to the mobile device involved in the video call. Besides, transcoding process consumes a great amount of energy apparently, the capacity of the mobile device battery will not be enough to finish the video calls. To struggle against the problem we propose a cooperative media streaming where Media-Aware Network Elements (MANE) such as media gateways will transcode the received media stream to minimize the energy usage of mobile device. Thus, mobile device can receive personally customized live video stream from MANE meanwhile the overload of media servers and the process of transcoding overhead of mobile device is reduced. Our research in this study cooperated media streaming solves the issues of energy usage for transcoding in mobile device by giving transcoding and real-time delivery responsibility to the MANE. We consider the basic steps that the MANE takes to decide when and how to transcode the received media stream and work together to adapt and optimize the real-time delivery of the audio and video streams provided by the application.

Key words: Delay-constrained applications, cooperative media streaming, stream format on interest, energy efficiency, transcoding

INTRODUCTION

New technologies are transforming the world at a dizzying speed. People spend more and more time interacting with smartphones which at the beginning were thought for wireless mobile telephony but now are used to stay always-connected. One of the most promising and convenient ways to deliver multimedia content to the users in the internet is through multimedia streaming. Today multimedia streaming takes up about 80% of video data traffic and engages for even bigger share in future. Heterogeneity of modern mobile devices that is the difference in screen size and computational power of each concrete model of mobile devices along with their energy limitations, necessitates customized video for all end user devices with the unique parameters (Rho *et al*., 2005 mentioned can lead to several issues in work of media servers, namely, network overload, storage space shortage and computational overload of media servers. Although, the applications that involve smartphones are constantly growing, smartphones still derive their energy from batteries whose capacity is severely restricted due to constraints on size and usage of the device. This implies that, while the phone usage increases, the device batteries may become insufficient and managing energy efficiently is of paramount importance.

Real-time transport of live video call is the predominant part of Voice over Internet Protocol (VoIP). In the streaming mode, the live video content need to be transit and is being played out continuously when the content is being received and decoded. Due to its real-time nature, video streaming typically has band-width, delay and loss requirements. Presently smartphones are normally equipped with WLAN interfaces that can support real-time transport of live multimedia (Han *et al*., 2012). WLANs also facilitate online calls through the VoIP where audio and video data played out continuously on the mobile node (Jin and Yu, 2011;

Corresponding Author: J.T. Anita Rose, Manonmaniam Sundaranar University, Tamil Nadu, India

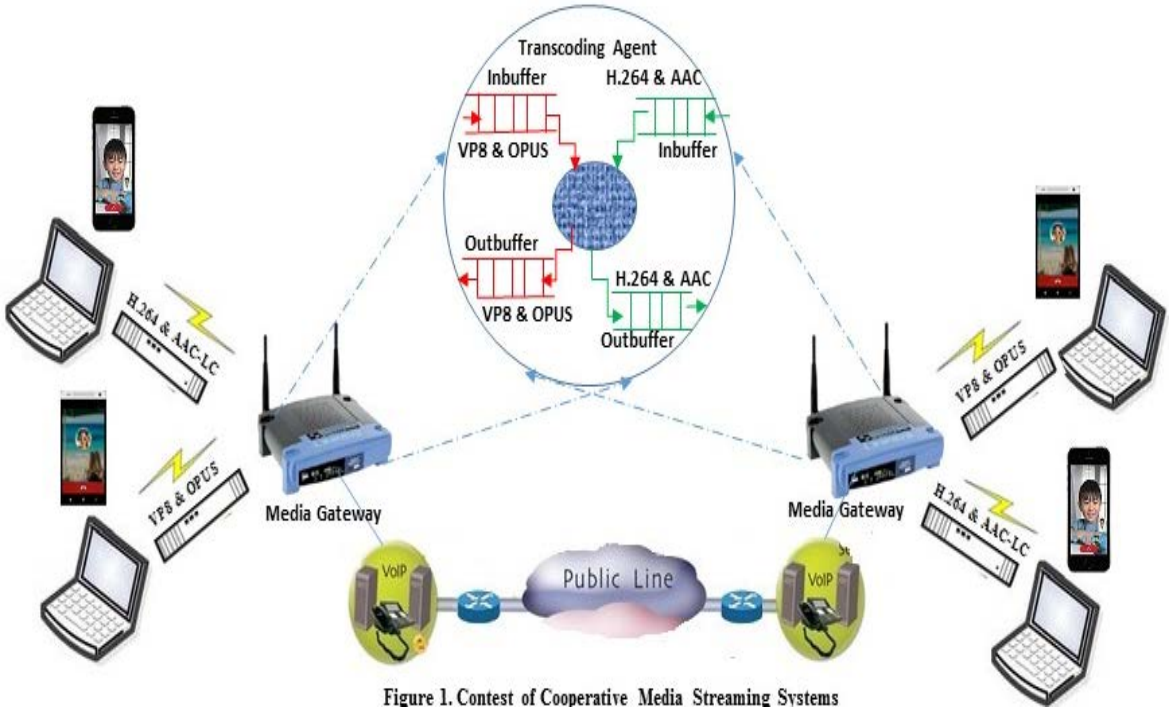

Figure 1. Contest of Cooperative Media Streaming Systems

Fig. 1: Contyest of cooprative media streaming systems

Goode, 2002). Each packet needs to arrive at its destination before a prescribed deadline; otherwise, it will be dropped.

Another critical challenge in wireless networks is saving energy (Repantis *et al*., 2009; Curran and Annesley, 2005; Shen *et al*., 2004; Zhang *et al*., 2013; Ahlehagh and Dey, 2013). The IEEE 802.11 standard defines a few power saving techniques for WLANs including Power-Saving Mode (PSM) (Huang *et al*., 2011) and Automatic Power-Saving Delivery (APSD) (Yang *et al*., 2011) which reduce energy consumption by switching the node from idle sensing mode to sleep mode. However, an energy-efficient design should also meet the network Quality-Of-Service (QoS) requirements such as end-to-end delay (Rosenberg *et al*., 2002). If sleep time is not properly scheduled, significant delays can occur which undesirable in live video streaming. Therefore, these power-saving techniques need to be enhanced to better accommodate sensitive delay constraints.

Original equipment manufacturers make devices with difference in screen size to display video and the speaker to play audio to its respective hardware codec acceleration. Google is working to complete its media engine that will use the VP8 video codec and OPUS audio codec but the apple and windows already paid the royalties to MPEG LA for the video codec H.264 and audio codec AAC. Mitigating between clients supporting different codecs requires transcoding because of varying computational power of each concrete model of mobile devices along with their energy limitations necessitates customized video for all end user devices with the unique parameters. In this study, we propose a cooperative media streaming where the media gateway will transcode the payload of Real Time Protocol (RTP) stream to minimize the energy usage of media servers and mobile node where our goal is to minimize the transmission delay, power usage and decoding complexity of end device.

Figure 1 show, the google WebRTC compliant devices generate the RTP payload which encoded with video codec VP8 and the audio codec OPUS. The apple and windows devices generate the RTP payload which encoded with video codec H.264 and the audio codec AAC. This RTP packet transmit through the network towards its destination and reaches the media gateway where the packet will convert into the format acceptable by the destination device. So that the destination device will playout the media smoothly with the available hardware codec.

Literature review: Heterogeneous modern mobile devices such as smart phones, tablet PCs and smart TVs have differences in screen size and computational power along with their energy limitations, necessitates customized video transcoding for all end user devices with the unique parameters (Rho *et al*., 2005). Presently smartphones are normally equipped with WLAN interfaces that can support real-time transport of live multimedia (Han *et al*., 2012). WLANs also facilitate online calls

through the Voice over Internet Protocol (VoIP) where audio and video data played out continuously on the mobile node (Jin and Yu, 2011; Goode, 2002). Adaptive bitrate streaming has become a prominent technology to improve the quality of video delivery over different network environments and support media consumption over heterogeneous devices (Repantis *et al*., 2009; Curran and Annesley, 2005). In terms of adaptive bitrate streaming, a transcoding-enabled proxy system which allows to perform content adaption per the network environment (Shen *et al*., 2004). To maximize the QoE for mobile users by storing content copies transcoded in different bitrates under constrained storage budget (Zhang *et al*., 2013). By introducing a joint video processing and caching framework to support adaptive bit rate streaming which can improve network capacity and maintain QoE (Ahlehagh and Dey, 2013). A SVC based video proxy to adapt to network dynamics by transcoding the original video into scalable codec at different bitrates (Huang *et al*., 2011). An online buffer fullness estimation method for adaptive video streaming (Yang *et al*., 2011).

SIP is an application level signaling protocol, widely used for VoIP and videoconferencing over IP networks (Rosenberg *et al*., 2002; Handley and Jacobson, 2015). Before establishing a peer-to-peer connection connectivity checks or session negotiation can occur we must find out if the other peer is reachable and if it is willing to establish the connection. We must extend an offer and the peer must return an answer. The signaling server can act as a gateway to an existing communications network in which case it is the responsibility of the network to notify the target peer of a connection offer and then route the answer back to the client initiating the exchange. if both peers are connected to the same signaling service then the service can shuttle messages between them.

WebRTC specification requires that all transferred data-audio, video and custom application payloads must be encrypted while in transit (Holmberg *et al*., 2015). WebRTC provides media acquisition and delivery as a fully managed service: from camera to the network and from network to the screen. The WebRTC application specifies the media constraints to acquire the streams and then registers them with the peer connection object. From there, the rest is handled by the WebRTC media and network engines provided by the browser: encoding optimization, dealing with packet loss, network jitter, error recovery, flow, control and more.

Secure Real-time Transport Protocol (SRTP) defines a standard packet format for delivering audio and video over IP networks. SRTP run directly over UDP and work together to adapt and optimize the real-time delivery of the audio and video streams provided by the application. The WebRTC application is never exposed to the internals of SRTP protocols but the browser implements all the necessary infrastructure. Another critical challenge in wireless networks is saving energy (Tsao and Huang, 2011; Xiang *et al*., 2013). The Carrier Sense Multiple Access with Collision Avoidance (CSMA/CA)-based IEEE 802.11 Medium Access Control (MAC) protocol requires the nodes to keep sensing the channel or staying in idle mode when they are neither transmitting nor receiving which consumes a significant portion of the energy resources of mobile nodes. Therefore, a promising strategy for power saving is to switch nodes when doing idle listening to sleep mode by turning off the wireless interfaces thus saving a considerable amount of energy. With the IEEE 802.11 PSM technique, a node periodically enters sleep mode and wakes up to retrieve buffered downlink packets from or report uplink packets to the associated Access Point (AP) (Liu *et al*., 2014; Namboodiri and Gao, 2010). Downlink packets suffer from both the buffering delay at AP and the delay due to channel contention of the destination nodes after waking up.

Problem statement: Rho *et al*. (2005) transcoding is done by adaptation engine which is a solution for multimedia data customization problem rather than the media server overload problem. Shen *et al*. (2004) adaptive transcoding performed at proxy servers or video adaptation nodes located on the edges of fixed networks, while in (Zhang *et al*., 2013) transcoding is fairly spread among existing neighbor servers. Although, those approaches reduce the overload of media servers they still might encounter difficulties in case huge number of video files need to be transcoded.

Presently smartphones are normally equipped with WLAN interfaces that can support real-time multimedia data offloading. Since energy is an important concern particularly for power constrained mobile nodes, VoIP nodes with WLAN interfaces want to work with low power while matching the playout deadline requirements.

Another, drawback of these studies is that if a user has several heterogeneous devices and needs the video for each of them, the only way to get customized video files is to access the media server several times which again emphasizes the media server overload problem. Thus, the more efficient way to address the problem would be to perform transcoding not at server side but at client side. However in view of the client side being a portable device with limited power and energy resources this solution does not seem viable. Therefore, to reduce the overload of media servers while keeping the transcoding process at client side a novel mobile offloading approach inspired by cloud computing is introduced in this study.

Mobile offloading (Huang *et al*., 2011; Yang *et al*., 2011) is a powerful technique helping to move most power

consuming computations from resource constrained portable devices to server. The main difference between mobile offloading and client-server approaches is that in case of the second one a client always migrates some part of work to the server (Rosenberg *et al*., 2002) while in case of mobile offloading it is decided dynamically which parts of code should be run at the server side.

MATERIALS AND METHODS

Device specific cooperative media streaming: As far as end-point device capabilities are concerned display size, processing power, memory size and the media codec must be considered. The received video stream resolution may be larger than the display size of end-point devices then the video stream not easily rendered by end-point devices. Hence, both the width and height of a video frame must be adjusted to fit in to the display size. However, it places an extra load on the end-point devices and is undesirable for devices where the processing power of the end-point device is low. The interoperation of media streams is unavoidable when both end-point device build with different hardware codec. In case of interoperation, when the end-point device is doing the transcode with the real-time media streams which will drain up the battery power much earlier so that user cannot complete the video calls. The amount of processing required for transcoding is related to the number of video frames in one second (frame rate) and total number of pixels in one frame (frame size). Therefore, the frame rate and size need to be adjusted to save the processing power of the end-point device.

Real time media processing: The media processing take up a lot of CPU usage when processed with the software codec. To minimize the general-purpose CPU usage of device, manufacturer creates the device with special chipset named as hardware codec. The H.264 hardware codec has the great market share and billions of end-point devices make use of it. To overcome the latency of existing streaming application, google has been creating the WebRTC application with new codec VP8. The chipset for VP8 also improved and released in the market with android based end-point devices. The hardware acceleration over the hardware codec is important to enhance the performance of the media processing on end-point devices. Currently WebRTC application supports both the codec so that the device performance can be maximized by selecting the supported hardware codec. Since the end-point devices are made with the diversity of supported codec hardware, the interoperation between the end-point devices are unavoidable to utilize the device's native codec hardware support. The live video chat initiated from origination end-point device with its expected media parameter such as audio codec, video codec and display size has passed to destination end-point device through Session Initiation Protocol (SIP) signaling. The destination end-point device validates the parameters and start the one to one communication when both the end-point devices agreed and ready to process with all the expected parameters. When both end-point devices working with same codec and display size then the media conversion or transcoding is not required otherwise transcoding is required in between both end-point devices. There are other cases where transcoding may be required even if both end-point devices support the same codec that may be differences in supported video formats and profile. For ex, Edge supports H.264 UC (the Microsoft version of H.264). This will not work with the common H.264 AVC so transcoding is required.

As discussed above the process can be separated in to signaling and media processing as shown in Fig. 2. Signaling is used to establish a peer-to-peer connection. Media processing is used to adapt and optimize the delivery of audio and video streams.

Cooperative media transcoding: In this study we present our proposed cooperative media transcoding system to transmit and receive the live Secure RTP (SRTP) streams through media gateway server. In media gateway the interface channel has open to transmit and receive the live SRTP stream after the preferred audio and video codec has finalized through SIP signaling. SRTP defines a standard packet format for delivering audio and video over IP networks. By itself, SRTP does not provide any mechanism or guarantees on timeliness, reliability or error recovery of the transferred data. Instead, it simply wraps the digitized audio samples and video frames with additional metadata to assist the receiver in processing each stream. In short, SRTP run directly over UDP and work together to adapt and optimize the real-time delivery of the audio and video streams provided by the application.

In end-point device the payloads of SRTP packets carrying the compressed RTP payload of video format either VP8 or H.264 and audio format either OPUS or AAC depends on hardware codec present on the end-point devices. Microphone generate the Pulse Code Modulation (PCM) data with resolution of 16 bits and a sampling rate of 44.1 kHz samples per seconds and fill the pre-encode raw audio buffer in a continuous stream. Audio encoder compressed the raw audio stream into the length of 20 MS frame which represents a bit rate 510 kbits per seconds. Encoded frame or multiples of frames put together inside the payload of RTP packet then buffered in pre-secure buffer for encryption. Camera generate the raw samples for the visual scene spatially (rectangular grid of pixels) and temporally (frames per

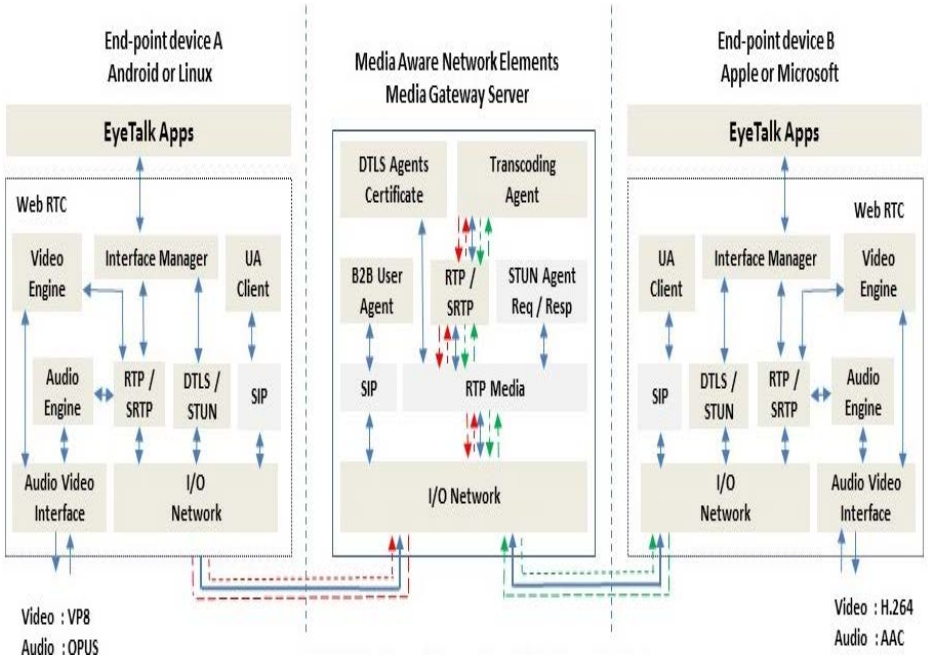

Fig. 2: Architecture of cooprative media streaming system

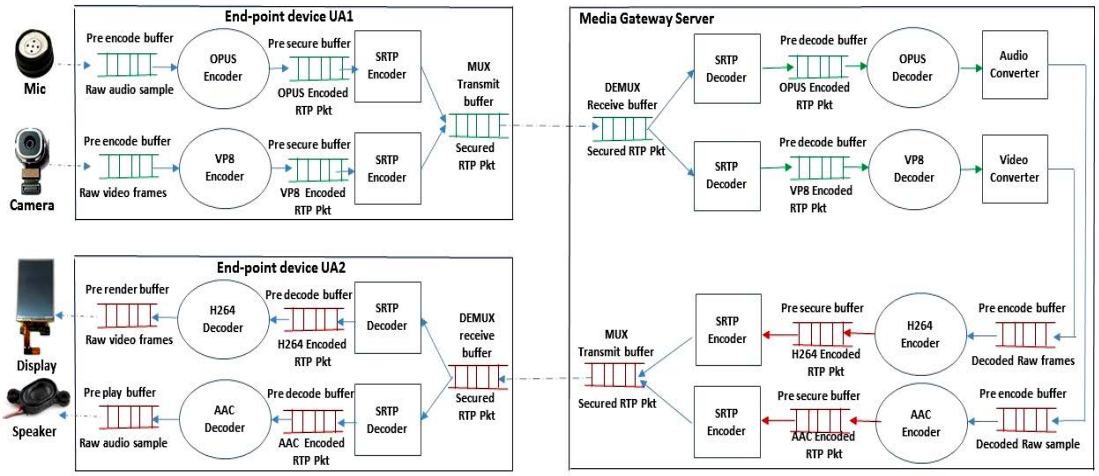

Fig. 3: Transcoding data flow for video VP8 to H.264 and audio OPUS to AAC

time) for visual graphic adaptor (320×240)@20 frames/seconds which represents a bit rate of 36,864 kbits sec^{-1}. Video encoder compressed the raw video by determining the prediction residual then transforming and entropy coding. Thus, it reduces the stream length and create the Network Abstraction Layer (NAL) unit. Encoded video NAL unit or multiples of NAL units put together inside the payload of RTP packet then buffered in pre-secure buffer for encryption. SRTP encoder encrypt and create the SRTP packet then transmit through network interface (Fig. 3).

Our algorithm use the flag to decide whether the ongoing live call must have the transcoding. During the time of call establishment doTrans flag has set to true or false depends on the SIP session description parameters. The transcoding has not required and the received live stream of SRTP packet has by passed thru the allocated channel without transcoding when the do trans flag has set to false. The received live stream of SRTP packet is passed to the input buffer of the transcoding agent when the doTrans flag has set to true. The transcoder continuously stream the SRTP payload from the DeMux

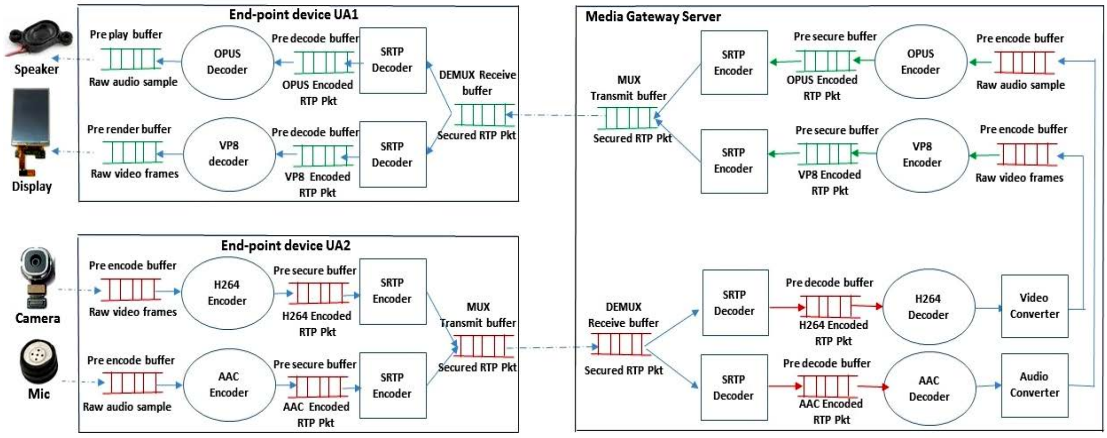

Fig. 4: Transcoding data flow for video VP8 to H.264 and audio AAC to OPUS

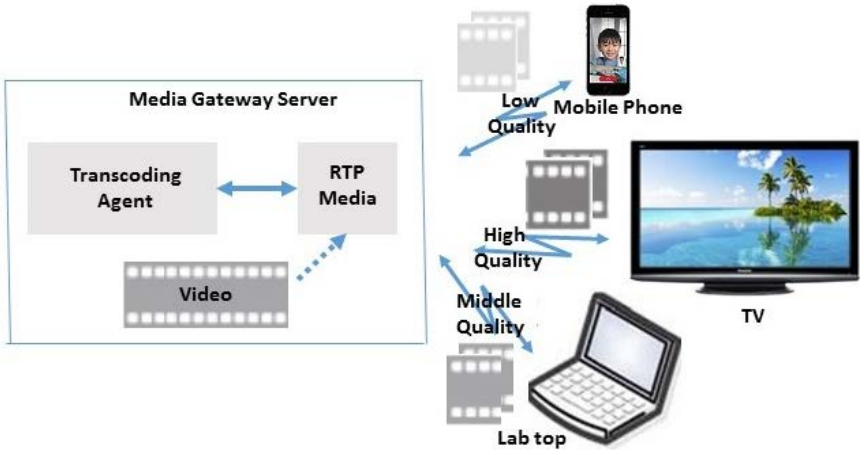

Fig. 5: Xgate Adaptive transconding server

receive buffer and subsequently transmit the transcoded SRTP payload thru mux transmit buffer. The conversion of video format either from VP8 to H.264 and convert the audio format either from OPUS to AAC is described in Fig. 4.

The conversion of video format either from H.264 to VP8 and convert the audio format either from AAC to OPUS as described in Fig. 5. The transcoding in media gateway server is a three-step process, i.e., Decode, parsing and encoding. First, Decrypted and compressed (encoded) RTP packet buffered at pre-decode buffer must be uncompressed by decoder using the same codec that was used at the time of encode. The output of decoder will produce the raw audio sample or video frame. Second, the audio decoder output is processed by audio convert parser to change the audio sample into the one recognized in other audio compression (encode) format and placed in the pre-encode audio buffer. The video decoder output is processed by video convert parser to change the frame into the one recognized in other video compression (encode) format and placed in pre-encode video buffer. Third, the audio parser output of raw audio sample is compressed (encode) with the expected audio codec. The video parser output of raw video frame is compressed (encode) with the expected video codec. The compressed audio and video packet has transmitted to destination through muxer.

Adaptive media transmission: The RTP media module of the media gateway acts as a server to the end user. Hence, an RTP media is built to stream the SRTP media payload to and fro for the sake of end user devices through UDP sockets. The outgoing SRTP media payload obtained either from the output of the transcoder or the caching system without transcoding. The size of the transmit and receive buffers can be small given that the transcoder

processes the video/audio data in a streamlined fashion. The speed of the process, i.e., the transcoding bit-rate is defined as the number of bits the transcoder generates per second. Given that the transcoding bit-rate is larger than the minimum of the origination link and the destination link bandwidths, the transcoding process does not significantly increase the end-to end delay. Given the real time transcoding capability, the Transcoding agent dynamically transcode video objects to different variants to satisfy the end users in heterogeneous environments. Each variant is a version. If version x can be obtained from transcoding version y, we call version y a transcodable version for x. Conversely, version x is the transcoded version of y. In video transcoding, a higher bit-rate version can be transcoded to a lower bit-rate version. For example, if a video at bit-rate of 64 kbps can be transcoded from the same video at bit-rate of 128 kbps, the 128 kbps version is a transcodable version for the one at 64 kbps. Consequently, the 64 kbps version is a transcoded version from the one at 128 kbps.

Note that the variation in versions delivered to the client device may not be transparent to the end user. The end user specifically asks for a certain version of an object based on the awareness of one's connection and display device. Alternatively, a client agent software informs the client's connection and device capability to the transcoding agent. The transcoding agent then chooses a version for the session.

RESULTS AND DISCUSSION

Experiments

Methodology and configuration: To evaluate the efficacy of exGate application several questions need to be answered. First, the relation between the growth of number end-point device accessing a media gateway Access Point (AP) running with exGate and the overload of the exGate needs to be estimated. To support the idea of using exGate for transcoding in viability of performing it at the mobile devices should be demonstrated. Finally, an improvement in power consumption, achieved due to video customization, needs evaluation.

To conduct the experiments, a smartphone with 1.2 Ghz CPU and 1 GBRAM running Android 4.0.3 version was used as the end user device. Two desktop PCs with 3.1 GHz quad-core CPU and 8 GB RAM each, represented the Session Border Controller (SBC) server and media gateway access point (i.e., exGate server). Both desktops used Linux OS while the transcoding server ran exGate in a Linux machine.

Measurement of media server overload: For quantitative assessment of possible media server overload, several VMs ran at the desktop representing offloading server. All VMs performed transcoding of the same video file simultaneously. Depending on the number of active VMs, the change in time required to finish transcoding was measured. During the experiment a 30 sec long video recoded with H.264/MPEG-4 AVC video codec at 1.8 Mbps bitrate was transcoded. A resolution of the video was changed from1280×720-480×272 pixels. Figure 6 shows the results of the experiment for arrange of clients varying from 2-10.

It can be seen from the plot that when the number of devices becomes more than two, the time needed to finish transcoding is getting longer than the time length of the video file itself. Although, transcoding time depends on the performance of the media server, even for most powerful ones there exists the number of clients, so that the time required for transcoding will be greater than the length of the video. In this case, noticeable delays during video viewing will occur affecting the quality of media streaming service.

Table 1: Summary values of energy, average power and time consumption for offloading, streaming and viewing

Video parameters (Mbps)	Time (sec)	Avg. power consumption (mW)	Energy consumption (mAh)
480×272.2	123.84	2311.81	21.49
960×545.2	243.48	2206.07	40.33
240×186.2	53.16	2586.28	10.32
480×272.4	149.07	2119.70	23.72
480×272.1	105.79	2353.33	18.69

Transcoding by mobile device: A straightforward solution for relieving media servers can be to localize transcoding of a video to the client side that is to move it from a media server to the mobile device requesting the video. Thus, media server do not need to do transcode anymore Summary values of energy, average power and time consumption for offloading, streaming and viewing and can send the requested video to the client right away. However, computational power of the mobile device is much lower compared to the one of the media server what makes transcoding last significantly longer (Table 1).

Besides, transcoding process consumes a great amount of energy. For example, to transcode the 30 sec speech video packet to the resolution of 480×272 pixels took about 20 mAh. Extending the results for 1 h-long speech video packet, apparently, the capacity of the mobile device battery will not be enough to finish such transcoding. Values of energy, average power and time consumption for video streaming only.

Table 2: Values of energy, average power and time consumption for video streaming only

Video parameters (Mbps)	Time (sec)	Avg. power consumption (mW)	Energy consumption (mAh)
480×272.2	30.00	1523.14	3.75
960×545.2	70.94	2036.18	9.72
240×186.2	30.00	1429.71	3.39
480×272.4	30.00	1511.93	3.41
480×272.1	30.00	1529.93	3.87

Table 4: Offloading server transcoding

Video parameters (Mbps)	Time (sec)	Avg. power consumption (mW)	Energy consumption (mAh)
480×272.2	17.04	10.76.11	1.47
960×545.2	34.07	1148.08	2.54
240×186.2	16.67	991.28	1.50
480×272.4	23.31	1201.35	2.22
480×272.1	16.49	1268.12	1.39

Transcoding by media gateway server: An alternative to the mobile device transcoding which also keeps the process localized at client side is to perform it on the media gateway AP server. In this case, total energy consumed by mobile device can be separated into two parts: the energy spent to migrate transcode then stream the packet and the media packet viewing energy. Experimental results, showing energy consumption for each part as well as total energy consumption of exGate server are given in Table 2 and 3.

Table 4 and shows that total energy consumption in case of offloading is substantially less than the one spent to perform only transcoding on the mobile device. Also the energy consumed during streaming or offloading is relatively small compared to the energy spent for playing back. Since, playback energy consumption decreases considerably over worsening of video resolution, it can be inferred that the use of the offloading server as a transcoder is more energy efficient than viewing video without customizing its parameters.

CONCLUSION

In this study we investigated the impact of video transcoding by a mobile device based on the battery power usage. By analyzing the transcoding implications of inserting a software transcoder agent in a media gateway device. we derived the solution that the software transcoder agent should transcode the video packet to reduce the transcoding burden of mobile device and the battery usage. Experimental results have shown that the proposed solution will effectively reduce the battery power usage in mobile device.

REFERENCES

Ahlehagh, H. and S. Dey, 2013. Adaptive bit rate capable video caching and scheduling. Proceedings of the 2013 IEEE Wireless Communications and Networking Conference (WCNC), April 7-10, 2013, IEEE, San Diego, California, ISBN:978-1-4673-5938-2, pp: 1357-1362.

Curran, K. and S. Annesley, 2005. Transcoding media for bandwidth constrained mobile devices. Int. J. Network Manage., 15: 75-88.

Goode, B., 2002. Voice over Internet Protocol (VoIP). Proc. IEEE., 90: 1495-1517.

Han, B., P. Hui, V.S.A. Kumar, M.V. Marathe, J. Shao and A. Srinivasan, 2012. Mobile data offloading through opportunistic communications and social participation. IEEE Trans. Mobile Comput., 11: 821-834.

Handley, M. and V. Jacobson, 2015. SDP: Session Description Protocol. IETF, Network Working Group, University in Glasgow, Scotland. https://tools.ietf.org/html/rfc2327.

Holmberg, C. S. Hakansson, G. Eriksson, 2015. Web real-time communication use cases and requirements. Internet Eng. Task Force, 2015: 1-29.

Huang, Z., C. Mei, L.E. Li and T. Woo, 2011. Cloud Stream: Delivering high-quality streaming videos through a cloud-based SVC proxy. Proceedings 2011 IEEE Conference on INFOCOM, April 10-15, 2011, IEEE, Shanghai, China, ISBN: 978-1-4244-9919-9, pp: 201-205.

Jin, T. and C. Yu, 2011. Quick detection of stealthy sip flooding attacks in VoIP networks. Proceedings of the IEEE International Conference on Communications, June 5-9, 2011, Kyoto, pp: 1-5.

Liu, L., X. Cao, Y. Cheng, L. Du and W. Song *et al*., 2014. Energy-efficient capacity optimization in wireless networks. Proceedings of the 2014 IEEE International Conference on INFOCOM, April 27-May 2, 2014, IEEE, Chicago, Illinois, ISBN:978-1-4799-3361-7, pp: 1384-1392.

Namboodiri, V. and L. Gao, 2010. Energy-efficient VoIP over wireless LANs. IEEE. Trans. Mobile Comput., 9: 566-581.

Repantis, T., Y. Drougas and V. Kalogeraki, 2009. Adaptive component composition and load balancing for distributed stream processing applications. P2P. Networking Appl., 2: 60-74.

Rho, S., J. Cho and E. Hwang, 2005. Adaptive Multimedia Content Delivery in Ubiquitous Environments. In: Web Information Systems Engineering WISE 2005 Workshops, Mike, D., G. Yuanbo, J. Woochun, K. Roland and K. Shonali *et al*. (Eds.). Springer, Berlin, Germany, ISBN:978-3-540-30018-2, pp: 43-52.

Rosenberg, J., H. Schulzrinne, G. Camarillo, A. Johnston and J. Peterson *et al*., 2002. SIP: Session initiation protocol. RFC 3261, Internet Engineering Task Force.

Shen, B., S.J. Lee and S. Basu, 2004. Caching strategies in transcoding-enabled proxy systems for streaming media distribution networks. IEEE. Trans. Multimedia, 6: 375-386.

Tsao, S.L. and C.H. Huang, 2011. A survey of energy efficient MAC protocols for IEEE 802.11 WLAN. Comput. Commun., 34: 54-67.

Xiang, L., X. Ge, C.X. Wang, F.Y. Li and F. Reichert, 2013. Energy efficiency evaluation of cellular networks based on spatial distributions of traffic load and power consumption. IEEE. Trans. Wirel. Commun., 12: 961-973.

Yang, J., H. Hu, H. Xi and L. Hanzo, 2011. Online buffer fullness estimation aided adaptive media playout for video streaming. IEEE. Trans. Multimedia, 13: 1141-1153.

Zhang, W., Y. Wen, Z. Chen and A. Khisti, 2013. QoE-driven cache management for HTTP adaptive bit rate streaming over wireless networks. IEEE. Trans. Multimedia, 15: 1431-1445.