Multi-interface Streaming of Scalable Video

Tiia Sutinen, Jouni Knuutinen, and Markus Luoto

VTT Technical Research Centre of Finland, Espoo, Finland
{tiia.sutinen,jouni.knuutinen,markus.luoto}@vtt.fi

Abstract. Scalable Video Coding (SVC) creates new possibilities for video service delivery in heterogeneous multi-access networks as it allows splitting the video into independently streamable layers. Today's terminal devices often have multiple network interfaces, and with multihoming and mobility capabilities the devices are able to use the interfaces also simultaneously for connecting to networked services. Thus for streaming SVC-encoded video, an interesting new use case is the ability to receive the same video stream over multiple network connections. Such multi-interface streaming capability would benefit situations where, for example, no single access network within the user's reach is capable of supporting the whole video stream or when the service provider wants to make only the base quality stream available to all users (e.g. via DVB-H) but offers a quality enhancement for paying customers via some other connection (e.g. WLAN). In this paper, we present a prototype implementation for realizing multi-interface streaming for SVC.

Keywords: Scalable Video Coding, multihoming, prototype.

1 Introduction

Mobile terminals available today are typically equipped with several network interfaces and are capable of utilizing them also simultaneously to maximize the user experienced quality of Internet services. The network access options of the mobile terminals are primarily heterogeneous, that is, they are based on different technologies (e.g. WLAN, WiMAX, HSPA and DVB-H), and change in time as the user moves from place to place. The terminal can rely on multi-access capability and mobility solutions such as Mobile IP (MIP) and Host Identity Protocol (HIP) to stay connected to the Internet.

This kind of a heterogeneous network environment is however very challenging especially for Quality of Service (QoS) sensitive multimedia services. This is because in a heterogeneous network environment, continuous QoS support cannot be assumed on the network-level, as the access networks involved in the communications are based on different technologies and may be managed by different parties (e.g. operators). Thus, application-level adaptation (e.g. reducing video stream bitrate via coarser encoding to meet better the available network bandwidth) is a traditional solution for supporting multimedia services in heterogeneous networks, although it causes varying Quality of Experience (QoE) for the user.

A fairly recent standard for H.264 Scalable Video Coding (SVC) [1] provides not only means for easy adaptation of video streams to networks and terminals but it also enables new provisioning schemes for video services. In SVC, the video content is encoded into layers that can be used to reconstruct the video with a desired bit rate, resolution, and/or frame rate just by selecting a certain subset of the layers for decoding. Due to the fact that the upcoming RTP payload format for SVC standard will support also multi-session transmission (MST) of SVC-encoded video [2], SVC can be used to enable new communication scenarios for video transmission. In specific, MST allows parts of an SVC stream to be transported as their own RTP sessions. That is, one can for example transmit the base quality layer in one RTP session and the enhancement quality layers in another RTP session, or send all the layers in their own RTP sessions. The use of the MST mode is signalled to the client by the means of the Session Description Protocol (SDP) extended with dependency grouping [3].

The ability to stream SVC layers separately enables in principle one to route the layers easily via distinct network paths in end-to-end transmission. For heterogeneous multi-access network environments, this means that a user can harness the available network resources in the video stream delivery and receive the SVC-encoded video stream using one or several access networks depending on their capabilities by the means of IP multihoming. To direct flows to go through several network interfaces, one can either assign routes to the hosts (or networks) at the other end of the flow in the routing table or use a more advanced mobility management solution supporting IP multihoming (e.g. [4]). For simplicity, the solution presented in this paper relies on simple routing table adjustments in assigning the SVC layers to different networks. However, the same mechanisms could be utilized with a more advanced multihoming mechanism in the future.

In this paper, we present our prototype implementation for multi-interface streaming of SVC. Specifically, we focus on the case where the mobile terminal needs to use more than one access network in receiving an SVC stream to provide maximum QoE for the video user. We also take into consideration the different nature of network links, that is, our demonstrator supports both bidirectional and unidirectional communication links (e.g. WLAN/HSPA and DVB-H).

2 System Architecture

Our solution for multi-interface streaming of SVC utilizes the MST feature of SVC transport and relies on the presence of multiple network interfaces in the client and the server in assigning the routing for the SVC subflows. In this section, we present the overall system architecture and the prototype implementation of the system is presented in Section 3.

In order for the multi-interface streaming scheme proposed here to work, SVC streams must be split into separate RTP streams. The basic functioning is as follows: The server binds to several IP addresses on the host on which it runs (possibly using virtual interfaces, if needed). Each layer is then served from a different address. So, if we have a stream with a base layer and one enhancement layer, and two interfaces if_1 and if_2, we can map the base layer to if_1 and the enhancement layer to if_2. The principle of mapping SVC layers to server interfaces is depicted in Figure 1. The mapping itself is obtained by the client via the `m` field of the SDP information extended with layered decoding dependency information [3].

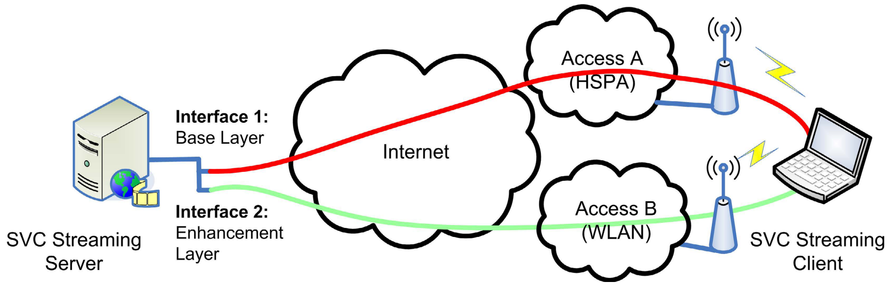
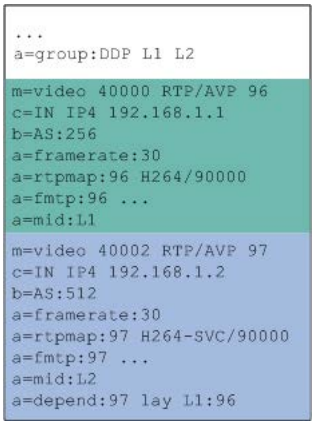


Fig. 1. Assigning the SVC layers to server interfaces and routing the streams via different networks

Having bound the available layers to different network interfaces of the server host, it becomes possible for the client to connect to them via different access networks, for instance, HSPA and WLAN, and mix the two separate RTP streams into a single SVC stream which is fed to the decoder. This naturally requires synchronizing the SVC substreams based on, for example, the RTP timestamps. In addition, the client side must also implement a suitable buffering scheme to deal with potential jitter due to the differences in the access networks' performance. In practice, this means that a somewhat larger buffer is likely to be necessary. Note, however, that the server side can help the client cope with different network delays by throttling the streaming of different layers accordingly.

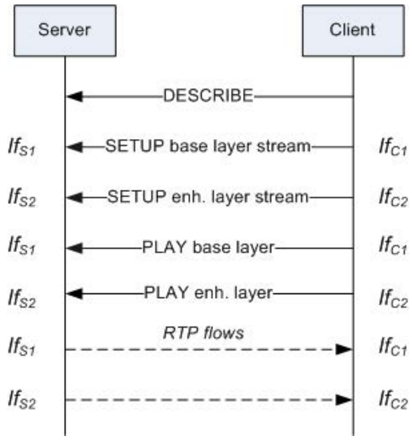
The client also needs to be able to cope with changes in network availability. Host mobility and differences in networks' coverage may cause temporary unavailability of a network access. An SVC player can adapt to situations where the connection carrying the enhancement layer is lost, but the minimum requirement is that at least the base layer stream is protected through a mobility management mechanism (e.g. MIP or HIP) to ensure continuous playout of the video. In our prototype implementation, we make the assumption that the base layer is continuously available for simplicity, and the more advanced mobility management issues are left for future work.

The session descriptor for a given SVC stream indicates at which network address each SVC layer can be found. In practice, the most likely situation is that all these addresses will correspond to virtual interfaces on the server, although this need not be true. Therefore, the simplest way to achieve the desired mapping of SVC layers to access networks is by altering the client's routing tables. The modifications needed are such that the route to each host address representing a given layer of the SVC stream is forced to go over the desired interface.



```
...
a=group:DDP L1 L2
m=video 40000 RTP/AVP 96
c=IN IP4 192.168.1.1
b=AS:256
a=framerate:30
a=rtpmap:96 H264/90000
a=fmtp:96 ...
a=mid:L1
m=video 40002 RTP/AVP 97
c=IN IP4 192.168.1.2
b=AS:512
a=framerate:30
a=rtpmap:97 H264-SVC/90000
a=fmtp:97 ...
a=mid:L2
a=depend:97 lay L1:96
```

(a) SDP message example



(b) RTSP signalling sequence (server responses omitted for clarity)

Fig. 2. Client-server signalling in multi-interface streaming case

The client uses RTSP for session signalling. To initiate a streaming session, the client first requests a session descriptor from the server via the RTSP DESCRIBE command. The session descriptor, which is encapsulated in the SDP format [3], contains information about the requested SVC stream, including the locations of the SVC layers as well as their characteristics and dependencies. An example of the session description is given in Figure 2(a). The client can use the default interface for transmitting the command.

The client utilizes the information contained in the session description in selecting the interfaces through which it wants to route each of the SVC sub-streams (i.e. layers). Once the routing table entries have been set, the client contacts the server via RTSP as shown in Figure 2(b). The communication to initiate the base layer and enhancement layer streams is carried out through their respective network interfaces.

Optimal routing table configuration at the client side requires also context information of the available access links and their characteristics. Access network availability information can be collected from network interface cards (NICs) and routing table entries but also mechanisms like the IEEE 802.21 Media Independent Handover Services [5] are expected to be used for this purpose once they become more common.

The above description applies to bidirectional communication links. We however consider also a hybrid communications scenario where the base layer is received over a unidirectional DVB-H link and the enhancement layers via a bidirectional WLAN or HSPA link following the procedures described above. In the case of DVB-H, the base layer stream setup is somewhat different, as DVB-H is a one-way communication link and it uses IP multicast to transmit the stream. Thus, no end-to-end signalling can take place in the session setup, but the client orders the base layer substream by joining the corresponding multicast group.

3 Implementation

The demonstrator implementation has two intercommunicating modules: an SVC streaming server and a client. The server is based on the Darwin Streaming Server (DSS) which implements RTSP streaming. The video is encoded using the JSVM reference encoder into a stream with two layers applying the quality scalability of the SVC standard. The encoded video stream is then encapsulated into MPEG-4 Part 1 format. The resulting file has two tracks, one for each layer, which allows the server to divide the SVC stream into two separate RTP sessions. For sending the base layer, we use the H264 RTP payload format and for the enhancement layer the H264-SVC format as suggested in [2]. The client software is a modified version of the MPlayer multimedia player with integrated OpenSVCDecoder decoding tools for SVC support.

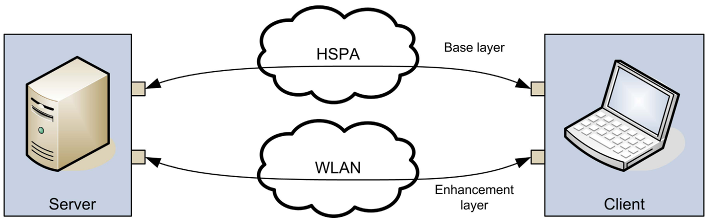
The client device has two network interfaces (e.g. HSPA or DVB-H and WLAN), which are used concurrently to receive the video stream. In the demonstrator, the video stream can be delivered by the means of unicast or hybrid (i.e. combined multicast and unicast) transmission. The hybrid scheme is considered to support broadcast networks like DVB-H that use multicast to carry IP streams. Different means need to be used to setup the multi-interface video streaming in the two cases as the multicast connection lacks a return channel.

The two demonstrator setups are shown in Figure 3. In the unicast case, we currently configure the client routing table manually. In our configuration, the client initiates the streaming by sending an RTSP DESCRIBE request to the server through the network interface that is used for receiving the base layer. The server responds with a description of the stream (SDP). The client sets up the multi-interface streaming through sending two SETUP messages: one for the base layer and another for the enhancement layer. The messages are sent using the client network interfaces assigned for the base and the enhancement layers, respectively. After completing the setup, the playback is started for each track by issuing a PLAY message through the corresponding client interface. Alternatively, a single aggregated PLAY message can be sent through the base layer interface to start the streaming for both tracks.

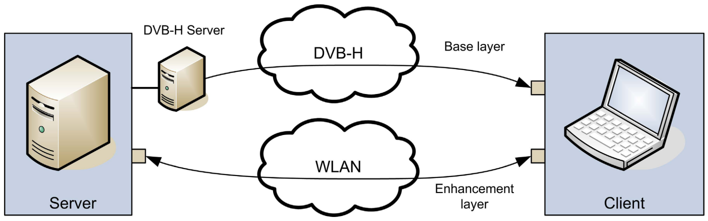
In the hybrid case, the client uses a multicast interface, for example DVB-H, for receiving the base layer and another interface with a return channel for the enhancement layer. The playback of the base layer is started by joining the multicast session. The playback of the enhancement layer is initiated and started in the same way as in the unicast case.

During the streaming, the playback of the enhancement layer is synchronized to the base layer in the client and it does not have to be started at the same time with the playback of the base layer. The two substreams are combined in the client and forwarded to the decoder as one continuous stream. To demonstrate dynamic dropping and adding of the enhancement layer, the client can execute a layer switch in the middle of the streaming by sending PAUSE and PLAY RTSP messages to the enhancement layer interface of the server.

Finally, the use of two network interfaces in the client terminal is demonstrated using network traffic monitors.



(a) Unicast communication scenario.



(b) Hybrid communication scenario.

Fig. 3. Multi-interface streaming demonstrator setup

4 Conclusions

In this paper, we presented an overall system design and a prototype implementation of multi-interface streaming of scalable video. We showed how scalable video coding and the multi-session capability of RTP allow - with simple routing adjustments - for utilizing multiple network interfaces in the stream reception simultaneously to maximize the QoE of the video service. As future work, we plan to test the prototype in various networking scenarios to verify its operation. In addition, we will investigate and develop more advanced mechanisms for synchronizing the SVC substreams in the client side as well as for intelligent interface selection and dynamic routing for the client.

References

1. Schwarz, H., Marpe, D., Wiegand, T.: Overview of the Scalable Video Coding extension of the H.264/AVC standard. IEEE Transactions on Circuits and Systems for Video Technology 17(9), 1103–1120 (2007)
2. Wenger, S., Wang, Y.-K., Schierl, T., Eleftheriadis, A.: RTP payload format for SVC video (draft-ietf-avt-rtp-svc-21.txt). IETF Draft (April 2010)
3. Schierl, T., Wenger, S.: Signaling media decoding dependency in the Session Description Protocol (SDP). IETF Request for Comments: 5583 (July 2009)
4. Pierrel, S., Jokela, P., Melén, J., Slavov, K.: A policy system for simultaneous multiaccess with Host Identity Protocol. In: 1st IEEE Workshop on Autonomic Communications and Network Management, Munich, Germany (May 2007)
5. IEEE Standards group. Media Independent Handover Services. IEEE Standard 802.21 (2009)