Scalable Video Streaming with Utilization of Multiple Radio Interfaces: A Customized Method for Signaling and Bandwidth Estimation*

Chayan Sarkar, Stephan Rein, Adam Wolisz
Technische Universitat Berlin, Telecommunication Networks Group
Email: {sarkar, rein, wolisz}@tkn.tu-berlin.de

Abstract **Scalable video coding allows to stream the multimedia content in hierarchically structured ows selected by the client due to his quality needs. In this paper we aim to improve the mobile user's media experience by a utilization of multiple wireless interfaces of the user's terminal over which the ows are streamed. Thereby we mitigate the factors of varying cell load, fading effects, and interference, which result in uctuations in the link quality. We rst introduce appropriate server and client entities and provide a signaling method for switching codec layers to the most suitable wireless interface. We verify the performance of the operation of the signaling mechanism using WLAN and UMTS as examples. In addition we develop a tool for dynamic bandwidth estimation that mixes probe packets in between the video stream to induce less traf c. We verify for the case of WLAN that the bandwidth estimator gives more accurate results than using probe packets only. Using both the signaling mechanisms and WLAN bandwidth estimation we use an example selection rule and measure the achieved PSNR gains in the received video.**

I. Introduction

Multimedia content distribution takes the signi cant part of the IP traf c. As predicted by the well known Cisco study [1], Internet video will account for 62 percent of the consumer Internet traf c by the end of 2015. And more and more will be delivered by means of wireless communication. Unfortunately enough wireless communication suffers from uctuating link quality due to the composition of three factors: varying cell load, fading effects, and interference with other wireless networks. As a result the media packets are delayed and lost, resulting in drops and uctuations of media delivery quality. These effects can only partially be mitigated by buffering, de nitely not in case of real-time services like video conferencing or live broadcasting.

In this paper we aim to improve the stability of user experience for streamed media by exploiting the fact that most of contemporary mobile devices are equipped with multiple wireless interfaces utilizing diverse communication technologies (e.g. UMTS, GPRS, WLAN). The diversity of used frequency bands and system solutions assures a rather uncorrelated uctuation of quality of the individual links available for the mobile device. We advocate the simultaneous usage of multiple wireless links (multiple radio interfaces) for transmission of a single stream of content.

*This work has been partially supported by the FP7 COAST (FP7-ICT-248036) project, funded by the European Community.

We feature the usage of Scalable Video Coding (SVC) [7] extensions to the H.264 AVC standard, which allows for encoding the video into multiple hierarchical layers, each of them providing additional quality. Individual SVC layers will be streamed in separate ows which can be delivered to the mobile device using different wireless interfaces and combined there. For that obviously three partial problems have to be addressed. (i) Flexible mapping of the individual layers to different ows, and being transported via selected wireless technologies. (ii) The available network resources have to be estimated, and (iii) a proper dynamic mapping policy of the above mapping has to be developed. In this paper we present a complete solution to the problems (i)-(iii) including design and prototyping using a new approach for bandwidth estimation in WLAN, thus being a very attractive technology for video streaming. By a simplistic ow assignment policy we demonstrate the operation of the whole system.

The paper is organized as follows. In the next subsection we review related work. The proposed system architecture is discussed in Section II. The signaling mechanism to enable dynamic layer mapping is elaborated in detail in Section III. The new tool for the WLAN bandwidth estimation is described in Section IV. In Section V, we evaluate our system and present the results, and Section VI concludes the paper.

A. Related Work

Related literature on SVC streaming (extension to H.264 AVC) using multipath is rather limited. In [4] a method is described for scalable video adaptation due to changing available bandwidth in heterogeneous networks. Streaming over two networks simultaneously is not considered, and the measured available bandwidth is not veri ed with the actually available one. In [2] deterministic packet scheduling algorithms are derived. TCP is selected as transport protocol and each packet is scheduled depending on the timing information, which makes the packet scheduling computationally expensive. The algorithms are evaluated via simulations against the rate control algorithms de ned in the Datagram Congestion Control Protocol (DCCP) standard. In [6] a scheme for scalable video transmission over multiple wireless networks is detailed. The work does neither provide any signaling method for multipath streaming nor considers the available bandwidth to schedule the packets. The focus there is on service provision to a group

of users who are connected via a multi-homed access point. Streaming in individual multi-homed devices is not considered.

II. Concept for network aware video delivery

In general, a SVC video streaming session has two parts - session setup and video data transmission. For the session setup the RTSP protocol can be used. After receiving the request for a video, the server describes the video using the Session Description Protocol (SDP) as plain text in an RTSP message. The description contains the number of ows available in the stream, the number of codec layers along with the codec information available in the video, and the required bit rate for each ow in the stream. Then they set up each ow in the video by determining a set of port numbers (from each side). When the client sends a play request, the server starts sending video data packets using the RTP protocol. For each ow, a separate RTP session is created. Each RTP session is assigned an unique Synchronization Source (SSRC) identi er and all packets of an RTP session are sent to a particular port (determined during ow setup). The RTP protocol is coupled with the RTCP protocol which monitors the RTP session. For RTP sessions, two consecutive ports are used for data transfer, where the even port is used for the RTP packets and the odd port is used for RTCP packets. In this work, the RTSP protocol uses TCP, and RTP uses UDP as transport layer protocol.

For the network aware video delivery we introduce two entities, the *server entity* and the *client entity*. The entities are placed between the streaming server / client player and the supporting media delivery protocols (RTP/RTCP) properly enhanced for SVC delivery [8]. The *client entity* informs the *server entity* about its available interfaces and routes the video ows to the required interface of the client due to the estimation performed by the *bandwidth measurement entities*. We assume N wireless interfaces to be available. Figure 1 illustrates the information ow between the entities. The *con-*

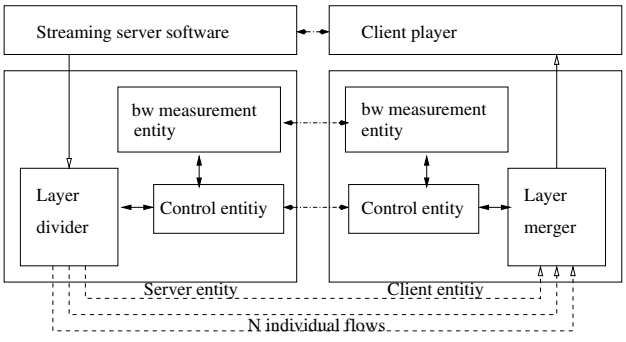

Fig. 1. Flow of information between server and client entity.

trol entities exchange signaling messages at different phases of a streaming session and establish a mutual agreement on the ow distribution policy. According to this policy, the *layer divider* of the *server entity* distributes ows of a streaming session among the interfaces of the client by changing the destination address in the packet's header. The *server control entity* collects relevant information about the video during the session setup, such as the number of ows, included codec layers, and bit rate requirement per ow. It needs to maintain state information about each scalable stream. For extraction of coding information we used the works in [3], [8] as a reference. Before the packets are routed they are queued by a high performance FIFO queue. At the client entity the packets are similarly en-queued. The *layer merger* merges packets received via multiple interfaces by adjusting the packet header and forwards them to the client player application.

III. Signaling for layer switching

We introduce a signaling method to enable the ow switching to the required interface. The general format of a control message is shown in Figure 2. The types of control messages

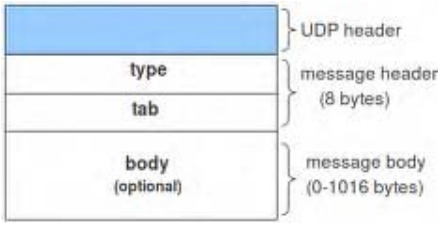

Fig. 2. Layer switching control message format.

used to accomplish a speci c task are listed in Table I. There

Message type	Purpose of the message
CTRL-RTP	Register IP address of an interface and RTP port number
CTRL-RTCP	Register RTCP port number
CTRL-SELECT	Distribute ows among multiple interfaces
CTRL-REMOVE	Remove ow distribution entry
CTRL-UNREGISTER	Unregister an interface

TABLE I
List of control messages for flow switching.

are two parts of a control message, the message header and message body. The message header has a xed size of 8 bytes. The message body is optional. The length of the message body can vary from 0-1016 bytes. The message header contains two elds - *type* and *tab*. The *type* eld signals the purpose of the message, and related information is given in the *tab* eld and the message body. Thereby the action of the recipient after receiving the message is determined. The introduced entities work together to provide a network-aware video streaming session and exchange speci ed control messages as follows.

(1) *Overcoming the NAT restriction:* As the video ows (port numbers, IP address) are negotiated between the server and the client during the session setup via RTSP (over TCP/IP), the video data transmission (using UDP) does not work as the interfaces are hidden due to the Network Address Translation (NAT). To solve this CTRL-RTP and CTRL-RTCP

messages are sent to initiate a video data transmission. As the port number generally serves to distinguish between an RTP and a RTCP packet, two different initiation messages are sent for two ports.

(2) *Registering multiple interfaces:* If the client has multiple interfaces available to receive a video stream, the *server entity* needs to be informed about the IP addresses of all interfaces using the CTRL-RTP. The *server entity* maintains a database about the registered interfaces of different clients. After receiving a CTRL-RTP message, it inserts a new entry in the database with the source IP address and source port number of the message. It also assigns a *registration-id* to the entry and replies to the client with that id. Any future communication related to a registered interface is referred using this id. The CTRL-RTCP message is used to register the RTCP port. The *tab* eld of the message header contains the *registration-id* of the interface, which is known from the reply of the CTRL-RTP message.

(3) *Streaming setup and parameter gathering for video streaming:* When a streaming session starts, initially a set of RTSP request/reply messages are exchanged between the client and the server. The server provides all required information about the video stream at this stage. The text-based Session Description Protocol (SDP) is used to describe a stream. The *packet-interceptor* of the *client entity* collects all information about the available codec layers and ows of a video from this message. After receiving the description, the client's player and the streaming server software set up the ows by a mutual agreement (agree upon the source and destination port for each ow).

(4) *Distributing ows among multiple layers:* When all ows are set up, the *client entity* makes the decision about receiving ows via its interfaces. The ow receiving policy is informed to the *server entity* using the CTRL-SELECT message. A customized message body called the *select message* is used for this purpose. It contains the SSRC value for the *synchronization source* of the ow it is associated with. In addition the message body has for each interface a pair of two elds, the *id* eld for the interface registration identi er and the *op* eld to signal if the ow is routed via that interface or not. For each ow in the stream, a separate message body is created. All select messages are sent together within a single control message. The *tab* eld contains the count of the select messages. The *server entity* stores the select messages in a table called *ow-table*. When a data packet arrives, the packet's destination is decided by a table look-up.

(5) *Ongoing streaming session:* Now as the routing of ows is decided, the client player sends a play request and the streaming server starts sending the video data. When the data packets are passing through the *server entity* the packets are routed accordingly due to the CTRL-SELECT message sent earlier by the client. This is achieved by changing the packet's header with respect to the registered address of the client interface. At the client, the *client entity* receives packets via multiple interfaces and adjusts the packet's header according to the agreement between the video player application on the client side and the streaming server software.

(6) *Content adaptation:* If there is a signi cant change in the available bandwidth, the ow routing is modi ed. The new ow routing is signaled to the *server entity* using CTRL-SELECT messages.

(7) *Concluding streaming session:* When a streaming session is over, the *client entity* sends the CTRL-REMOVE message to clear all entries from the *ow-table*. In the message body, the *SSRC* of each ow is mentioned. The *tab* eld contains the count of ows associated with the past streaming session. The client may start a new streaming session, as the interfaces are still registered.

(8) *Unregister interfaces:* If the *client entity* does not request for any more streaming sessions, the interfaces can be unregistered by sending the CTRL-UNREGISTER message. After receiving that message, the *server entity* removes the interface entry from the database. The *tab* eld contains the *registration-id* of the interface to be unregistered.

IV. In-stream bandwidth estimation

A. Measurement Principle

For the development of a WLAN bandwidth estimation tool suitable for video streaming we have selected the tool *WBest* [5] as a starting point. The underlying principle rst estimates the effective capacity over a ow path using a *packet-pair*. Two packets are sent back-to-back from the sender to the receiver. The packet *dispersion rate* of a packet-pair generally re ects the capacity of the link which has the lowest capacity along the ow path. As it is here assumed that the last hop is wireless and has the lowest capacity, it re ects the capacity of the wireless link. 30 such packet-pairs are sent, and the capacity of the wireless link is calculated as the median of all dispersion rates. In the second part of the principle, a *packet train* of length 30 is sent to estimate the available bandwidth at the rate of the ow path capacity. The mean of all dispersion rates is taken as the available train rate. If the capacity of the ow path is CP and the available train rate is AT, then the available bandwidth is calculated as

$$AB = CP(2 - \frac{CP}{AT}). \quad (1)$$

As the bandwidth estimation is done based on probe packet dispersion, it increases traf c in the last hop. If these probe packets create congestion, even if only for a short period of time, the video traf c will be affected for that period. The congestion will also affect the probing traf c, which will lead to inaccurate bandwidth estimation.

We introduce the customized bandwidth estimation tool *EStream* to avoid congestion effects and to provide accurate in-stream measurements. EStream intrudes probe packets into the stream and utilizes the video packets as probes. It inserts a probe packet immediately after the current data packet into the stream. The general video data pattern is shown in Figure 3.a. The modi ed stream with packet-pair probes is shown in Figure 3.b. At the receiver, if both packets of the pair are received correctly, they are treated as a packet-pair

Fig. 3. Mixing of probing packets in between the video traf c. The pure video traf c (a) is mixed in probing packets for the *packet-pair* technique (b) and the *packet-train* technique (c).

and the dispersion rate for this pair is calculated before the data packet is forwarded to the client program. Two packets belonging to the same pair are identi ed by the sequence number, as the probe packet uses the same sequence number as the data packet. Instead of a packet train, EStream sends 30 packet-pairs (using probe packets of equal size than the video packets) each packet in a pair separated by a gap due to the required train sending rate, see Figure 3.c. The train rate is calculated as the mean of the 30 dispersion rates. Then the available bandwidth is calculated using equation 1. The introduced probe packets contain relevant video data to make the transmission more robust against packet loss.

B. Signaling for bandwidth estimation

Before starting and during a streaming session, the available WLAN bandwidth is measured. A list of the control messages for the bandwidth estimation signaling is given in Table II. The body of the respective control messages contains

Message type	Purpose of the message
PACKET-PAIR	Request packet-pair
PACKET-TRAIN	Request single long packet train
PACKET-SPARSE	Request multiple small packet trains

TABLE II
LIST OF BANDWIDTH ESTIMATION CONTROL MESSAGES.

two elds, *probe-count* and *probe-value*. The PACKET-PAIR control message contains the *registration-id* of the interface for which the packet-pairs shall be sent in the *tab* eld. The *probe-count* contains the number of packet-pairs to be sent. The *probe-value* of the message body is not used. The PACKET-TRAIN message requests packet trains. The *probe-count* signals the length of the packet train. The *probe-value* determines the sending rate of the packet train. During the video data reception, the bandwidth-estimator measures available bandwidth of the wireless interface in a periodic manner. To request multiple packet-pairs (as a packet train), the PACKET-SPARSE control message is used. The *probe-count* signals the number of trains to be sent and the *probe-value* the sending rate of the packet train.

V. EVALUATION

A. Implementation and measurement setup

The *server entity* and the *client entity* are implemented in C and tested in Ubuntu-10.04 with Linux kernel version 2.6.32. For user space packet handling, the *libnet lter_queue* library of the Linux kernel is used. It requires a kernel that includes the *nfnetlink_queue* subsystem. The signaling between the two entities is realized by socket programming. Videos are encoded using the JSVM (version 9.19.7) software. The Darwin Streaming Server-5.5.5 (DSS) software is used. A customized VLC player extended by a H.264 scalable video decoder plugin is used as video player. The measurement testbed is shown in Figure 4. The DSS is installed on a desktop

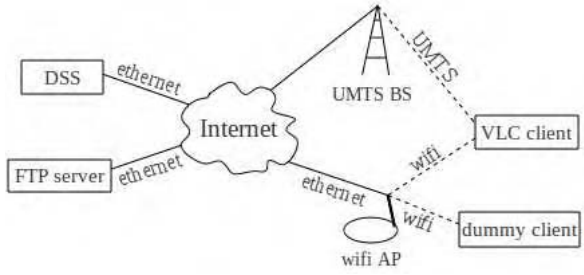

Fig. 4. Measurement testbed. The VLC client is streamed scalable video from the streaming server, while the load in the WLAN is varied.

computer and connected to the TU Berlin network via a 100 Mbps Ethernet LAN. A laptop is used as FTP server, which listens on a port to send UDP packets of a requested size at a constant rate. The FTP server is similarly connected via Ethernet LAN. The WLAN base station is connected via a 10 Mbps Ethernet LAN to the TU Berlin network. A laptop is used as a dummy client to create load in the WLAN. The VLC player (installed on a laptop with Broadcom WiFi card) is used to request and receive RTSP video streams from the Darwin streaming server. An external USB UMTS stick (O2 network operator with assured bit rate of 2 Mbps) is used to access the UMTS network.

B. Measurements and results

The accuracy of the EStream tool is veri ed as follows. The VLC client and the dummy client are connected with the access point. To avoid interference from other wireless users, all experiments are conducted inside a lab either after 10 pm or before 6 am. Four different loads are imposed by the dummy client by requesting traf c from the FTP server at a constant rate, 0.5 Mbps, 1.25 Mbps, 2.4 Mbps, and 3 Mbps. We measure a WLAN capacity of 4.02 Mbps by requesting packets with variable size from the FTP server at a high rate. During these four loads a video is requested from the streaming server and available bandwidth is measured. During the video streaming, the available bandwidth is measured periodically after every 10 seconds. The 95 % con dence

intervals with 50 measurement samples are plotted in Figure 5. For comparison the WBest tool is used for all given cases

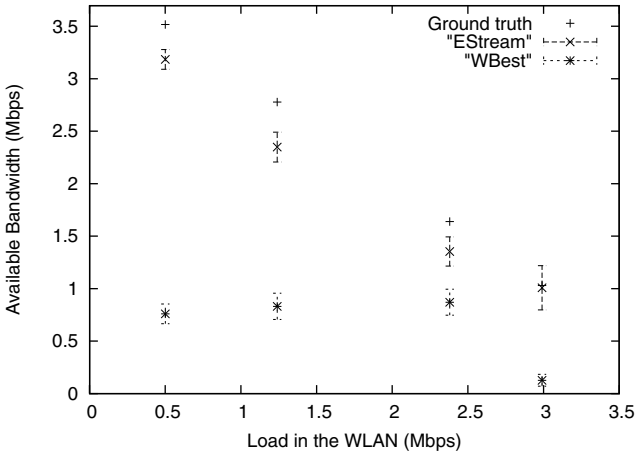

Fig. 5. Performance of WBest versus EStream. The bandwidth is estimated by both tools for different loads in the WLAN.

via separate measurement series. The measurements indicate that the WBest tool gives a maximum underestimation error of 2.7 Mbps, while the EStream tool estimates the available bandwidth with a maximum underestimation error of only 0.3 Mbps. With more load in the WLAN the EStream estimates even become more precise.

To verify that our system also reacts to changes in available bandwidth we compare the video PSNR quality achieved by utilization of WLAN and UMTS with the case of using WLAN only. As a test video we have used the *Paris* sequence (available at http://media.xiph.org/video/derf). It has a total of 1065 frames at a rate of 30 frames per second (fps). We encode the video in two spatial layers, the base layer in the Quarter Common Intermediate Format (QCIF) with a frame size of 176x144 pixels and one enhancement layer in the CIF format with 352x288 pixels. Each of these layers is streamed in a separate ow, where the base and enhancement layer require at most 0.2 and 0.7 Mbps, respectively. Results for the described setups are given in Figure 6. At frame number 105 a load of 3.5 Mbps is induced, and for WLAN only and both interfaces (WLAN. UMTS) the quality drops shortly after. For usage of both interfaces, however, the quality is restored again from frame number 170 on due to the switch of the enhancement layer to UMTS. At frame number 730 the WLAN dummy load is released, and the enhancement layer is switched back to WLAN without any change in the PSNR quality.

VI. Conclusion and Future Work

We have introduced a signaling method to enable network aware scalable video streaming by utilization of multiple wireless interfaces, thereby mitigating the effects causing uctuating available bandwidth. For realization of the signaling we introduce entities at server and client side, which do not require any change in the streaming server software or client video player. For the bandwidth estimation we have developed

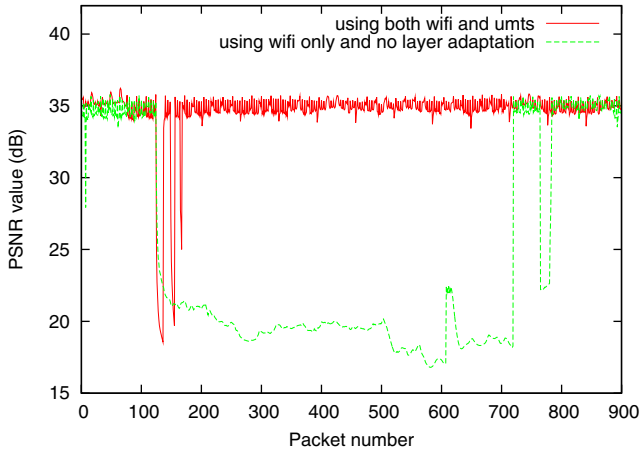

Fig. 6. Measurements for an example decision policy where the enhancement layer is switched to UMTS and back to WLAN due to estimated WLAN bandwidth.

a tool to induce less probing traf c by mixing probe packets in between the video stream. The evaluation measurements indicate that the bandwidth estimation is more precise than with the use of the general packet dispersion techniques. Our measurements verify that the video quality is not affected by the probing packets. For an example selection rule a codec layer is switched properly to an alternative interface upon changes is the available bandwidth, resulting in an improved user experience. In our future work we will verify our system for more than two codec layers and the suitable decision policies, work in an estimation method for available UMTS bandwidth, regard not only the last hop but the complete path from the server to the client, and take the mobility of the user into account to improve user connectivity.

Acknowledgment

We are very grateful to Karsten Gruneberg, Fraunhofer HHI, for provision of the SVC codec implementation, tools for the PSNR evaluation, and extensive technical support.

References

[1] Cisco visual networking index: Forecast and methodology 2009-2014. Technical report, 2010.

[2] Cheng-Hsin Hsu et al. Rate Control and Stream Adaptation for Scalable Video Streaming over Multiple Access Networks. In *18th International Packet Video Workshop*, pages 33 40, Dec. 2010.

[3] Ingo Ko er et al. An H.264/SVC-based adaptation proxy on a WiFi router. In *18th International Workshop on Network and Operating Systems Support for Digital Audio and Video*, May 2008.

[4] Pik Jian et al. Design and Implementation of Adaptive Scalable Streaming System over Heterogeneous Network. In *IEEE International Conference on Signal and Image Processing Applications*, page 84, Nov. 2009.

[5] M. Li, M.Claypool, and R.Kinicki. WBest: a Bandwidth Estimation Tool for IEEE 802.11 Wireless Networks. In *IEEE Conference on Local Computer Networks*, Oct. 2008.

[6] J. Nightingale, Qi Wang, and C. Grecos. Optimised Transmission of H.264 Scalable Video Streams over Multiple Paths in Mobile Networks. In *IEEE Tran. on Cons. Elec.*, volume 56, pages 2161 2169, Nov. 2010.

[7] H. Schwarz, D. Marpe, and T. Wiegand. Overview of the scalable video coding extension of the h.264/avc standard. In *IEEE Transactions on Circuits and Systems for Video Technology*, volume 17, Sept. 2007.

[8] S. Wenger, Y.K. Wang, T. Schierl, and A. Eleftheriadis. RTP Payload Format for Scalable Video Coding. *Internet draft*, Feb. 2011.