APP and PHY in Harmony: 
Demonstrating Scalable Video Streaming Supported by Flexible Physical Layer Control 

Denny Stohr, Matthias Schulz, Matthias Hollick and Wolfgang Effelsberg

Abstract—Wireless, high-bandwidth data transfer gains increasing significance with a wider adoption of high-quality video streaming on mobile devices. Technologies enabling such data transfer (e.g., 802.11 or LTE) can adapt dynamically to changing conditions, which allows data transfers in a reliable way. However, applications have little or no influence on how those underlying systems execute their adaptations. Yet, in applications such as video streaming, this could provide important clues on what requirements are relevant for data delivery. In this demonstration, we present a system that allows application-defined underlay control to use such potentials in a mobile scalable video based streaming scenario by combining Android smartphones with software-defined radios (SDRs) for advanced physical layer control.

I. INTRODUCTION

Video streaming on mobile devices is increasingly adopted by users and is said to be the main traffic source on mobile networks in the future. As predicted by Cisco, the share of video traffic consumed on mobile devices is going to rise up to 70% until 2018 [2]. Nevertheless, mobile video streaming is challenging given the diverse wireless network connections, limited energy resources and processing power on mobile devices. Therefore, it is crucial for an efficient and robust playback process to leverage the full potential of the underlying networking infrastructure.

In this work (which is based on [3]), we demonstrate how to optimize the transmission of application layer data by allowing an application designer to modify normally inaccessible physical layer parameters. As an example application, we use scalable video streaming on Android smartphones and allow to choose transmission requirements on a flow basis. Thereby, we abstract from physical layer details while preserving flexibility. To gain full control over the physical layer, we use software-defined radios (SDRs) that we connect to smartphones as networking interfaces. The impact of changing physical layer parameters is observable in real-time during video playback.

The remainder of this abstract is structured as follows: First, we give a system overview in Section II. Next, we describe the implementation of the demonstrator in Section III. In Section IV, we depict how users interact with our application, followed by the demonstration requirements in Section V.

II. SYSTEM OVERVIEW

In this section, we provide an overview of the underlying system for the proposed demonstration. A more detailed description of the system can be found in [3].

The system is composed of two main hardware components: Android smartphones and the Wireless Open-Access Research Platform (WARP) SDRs [1], as depicted in Figure 1. The WARP board is used to replace the regular wireless network interface of the Android devices. By combining those components, we intend to leverage Android as a platform for rapid application development and WARP boards to allow flexible real-time physical layer implementations on Field Programmable Gate Arrays (FPGAs).

The Android smartphones are connected to WARP boards using USB-to-Ethernet adapters and USB-OTG cables. Internet protocol (IP) traffic of predefined address ranges is tunneled over the Ethernet interface to the connected WARP devices. For tunneling network layer packets coming out of the Linux networking stack, we use the Android virtual private network (VPN) service API. The tunneled frames itself are equipped with a physical layer control header, that is interpreted by the WARP before transmission over the wireless interface.

For controlling physical layer parameters on a per packet basis, we implemented a flow-based architecture. This allows the

978-1-4799-8461-9/15/$31.00 ©2015 IEEE
specification of transfer parameters for different ports and protocols. Multiple applications can use our flow-control service efficiently in parallel and define physical layer parameters as rules in a flow table. Applications unaware of the configurable underlay can still use regular sockets in combination with best effort wireless transmission rules. Hence, our solution is completely transparent for existing applications that can coexist with new applications that make use of the increased flexibility by modifying physical layer parameters.

Based on the rules specified in a flow table, a service translates application requirements into physical layer parameters, which are stored in an additional frame header. This header is interpreted by the WARP board and translated into specified modulation schemes or transmit powers on a per-frame basis. An overview of the system components involved in the process is depicted in Figure 2.

In order to evaluate and demonstrate the system, we developed a lightweight scalable video codec (SVC). The codec is based on a compression algorithm similar to the one used in JPEG. To encode, we extract layers where each represents a separate degree of detail of the input video. Higher layers rely on lower layers for proper playback. This is illustrated in Figure 3. While the base layer contains only the DC components of $8 \times 8$ pixel blocks in the frequency domain, first and second layers add additional high frequency components to enhance video quality. Missing intermediary frequencies (first layer is missing but second layer is available) decrease image quality. Hence, lower layers require robust transmissions, while higher layers rely on an increased transmission rate due to more coefficients per block.

To keep our codec simple, we do not use inter-frame dependencies, as implemented in state of the art video codecs such as H.264 or H.265. Thus, we sacrifice smaller file sizes for reduced complexity and robustness.

III. IMPLEMENTATION

As the application was build on Android, we leveraged the available software development concepts of the Android Open Source Projects (AOSPs). Most notably, we designed the application for controlling two different services that could also be used independently of our software.

First, the WarpVPNService is used to handle the data processing between the WARP board and the Android device. When started, network layer packets from applications running on the smartphone are captured with a TUN device. Access to the device is available without root privileges by implementing a VPN service. If desired, the service can also be bypassed by applications that do not require physical layer control, by modifying the VPN’s subnet accordingly. The service is used to filter frames and attach WARP configuration headers according to flow table rules. During runtime, applications can send instructions to configure those flow filters using the Messenger interface so that only one service is required to map application requirements to physical layer parameters.

In our demonstration scenario, application data is generated by a second service, called MediaStreamingService. Its main task is to encode either a video stream coming from one of Android’s cameras or a video file that was supplied by the user. Both video sources are accessed through a common OpenCV2 interface. To support a large number of video formats, we rely on FFmpeg3. For performance reasons, we implemented our SVC in native C/C++ code using the Android NDK4. Using the Messenger-interface, SVC layers can be activated and deactivated during runtime.

IV. DEMONSTRATION

The purpose of the demonstration is to show the potential of physical layer control for a scalable video streaming application. Users of our demonstration will be able to manually configure how each of the SVC layers is transmitted and observe the influence of their changes on receiving devices. The transmitter tries to optimize the video quality on all receiving devices while keeping the occupation of the wireless channel low by maximizing the transmit rate and minimizing the transmit power. Even though receivers move around, they

1Big Buck Bunny Video: https://peach.blender.org/
2OpenCV: http://opencv.org/
3FFmpeg: https://www.ffmpeg.org/
should always get the best possible video quality but at least the base layer for continuous playback.

After starting the application, the user lands on the mediaplayer view, that has multiple menu buttons on the right side (see Figure 4). The user can either connect to a video transmitter to receive a video stream, or initialize the transmission of a new video stream. In any case, the mediaplayer shows the video playback in real time.

Figure 4a illustrates the stream configuration in case of a transmission. The user can choose either a camera or a video file as streaming source. Depending on the available hardware, the user selects to either utilize the increased flexibility offered by the WARP or setup an access point to which streaming clients can connect (The latter option does not allow to adjust physical layer parameters). Then, the user defines which SVC layers to generate and transmit and selects predefined physical layer requirements, such as robustness for the base layer.

To configure the mapping of requirements to physical layer settings, the user opens the settings view (see Figure 4b). Using sliders, the user selects the desired physical layer parameters (currently transmit rate and power). The configured values are directly updated in the flow tables of our WarpVPNService so that changes have a direct influence on the video qualities of the receiving devices. To load a predefined mapping configuration, users can also select from a list of preset profiles.

During the demonstration, users will be able to move around and see how changing the position of the receiving or sending device influences the video quality. Especially changes due to varying amounts of received layers as well as their visual impairments due to transfer errors at a given time will be observable in real-time. Different statistics can be plotted on the statistics view (see Figure 4c). Thus, the demonstration shows how the enhanced physical layer control influences the video transfer in the described scenario. Participants may use their own Android devices to send and receive scalable video streams in a limited fashion as they may only use their build in WiFi chipsets without physical layer control.

V. DEMONSTRATION SETUP

The demonstration will be presented using five Nexus 5 Android smartphones being connected to WARP boards. No additional equipment needs to be provided in the demonstration venue.

ACKNOWLEDGMENT

This work has been funded by the German Research Foundation (DFG) as part of the projects C01, B01, and project group “Demonstrator” in the Collaborative Research Center (SFB) 1053 “MAKI—Multi-Mechanism-Adaptation for the Future Internet” and by LOEWE CASED.

The authors would like to thank Stefan Wilk and Benedikt Rudolph for their work on the previous publication which is the foundation for this demonstration.

5Supported Codecs: https://ffmpeg.org/general.html

REFERENCES