Poster: Realistic testing of RTC applications under mobile networks

Gianluca Perna
Politecnico di Torino
gianluca.perna@polito.it

Martino Trevisan
Politecnico di Torino
martino.trevisan@polito.it

Danilo Giordano
danilo.giordano@polito.it
Politecnico di Torino

ABSTRACT

The increasing usage of Real-Time Communication (RTC) applications for leisure and remote working calls for realistic and reproducible techniques to test them. They are used under very different network conditions: from high-speed broadband networks, to noisy wireless links. As such, it is of paramount importance to assess the impact of the network on users' Quality of Experience (QoE), especially when it comes to the application's mechanisms such as video quality adjustment or transmission of redundant data. In this work, we pose the basis for a system in which a target RTC application is tested in an emulated mobile environment. To this end, we leverage ERRANT, a data-driven emulator which includes 32 distinct profiles modeling mobile network performance in different conditions. As a use case, we opt for Cisco Webex, a popular RTC application. We show how variable network conditions impact the packet loss, and, in turn, trigger video quality adjustments, impairing the users' QoE.

CCS CONCEPTS

• **Networks → Network services**; **Network performance evaluation**.

KEYWORDS

Mobile Networks; Real Time Control Applications.

1 INTRODUCTION

In the last years, Real-Time Communication (RTC) applications have become extremely popular, and dozens of platforms are now competing for the market. They allow individuals to communicate with friends and relatives and support enterprises for remote working, especially during the COVID-19 pandemic [1]. However, for the proper functioning, the network is required to carry packets with reasonably low delay, avoiding losses which would cause degradation in the users' Quality of Experience (QoE). Applications are known to adapt their sending rate to reduce the possibility of events affecting users' QoE [3], but it is hard to measure the real benefits of such approaches on real networks. This is particularly true for mobile networks, whose study is complicated by their scale and the diversity of deployments and technologies.

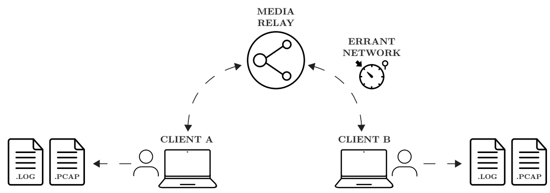

Figure 1: Testbed setup and RTC communication schema.

In this work, we lay the basis for realistic and reproducible experimentation of RTC applications under mobile networks. We instrument a testbed in which ordinary equipments make audio and video calls, while using a network emulator to impose realistic constraints on the network conditions. To this end, we use ERRANT [4], an open-source tool that emulates mobile networks with a high level of realism, following a data-driven approach. Important to our analysis, ERRANT models both the typical behavior of the network and its *variability*. As a case study, we target Cisco Webex Teams[1], which allows multi-party meetings with audio, video and screen sharing. In our setup, two clients communicate with audio and video, while, in the background, ERRANT imposes network conditions typical of 3G and 4G Radio Access Technologies (RATs) on one of the two clients.

Using the Webex client-side logs, we gather QoE-related metrics, like jitter and packet loss. Our initial results, built on 10 hours of calls, allow dissecting the impact of the network on the users' QoE. The variability emulated by ERRANT allows not only to statically characterize the meetings metrics under different conditions but allows the study of the application strategies to react to network impairments and degradation. This work poses the basis for the development of better adaptation strategies, providing a reproducible methodology to study the impact of new application choices, i.e., new approaches to send redundant data or bitrate adaptation.

2 EXPERIMENTAL SETUP

Our goal is to design a realistic, reproducible yet simple testbed in which two or more equipments participate on a call under controllable network conditions. We build on ERRANT, a state-of-the-art network emulator, which imposes realistic network conditions based on a large-scale measurement campaign under operational mobile networks. ERRANT can reproduce the variability of conditions intrinsically rooted in mobile networks due to different operators, RATs (i.e., 3G or 4G), signal quality (e.g., bad due to weak

[1] https://www.webex.com/team-collaboration.html

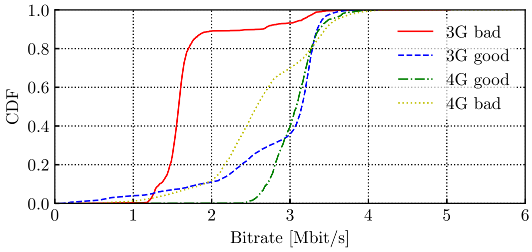

Figure 2: Webex bitrate under different emulated networks.

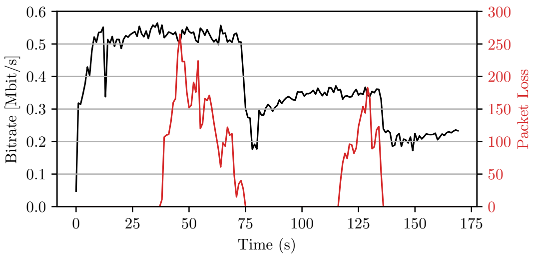

Figure 3: Bitrate and losses on an emulated 3G-bad network.

signal). ERRANT comes with 32 network profiles describing the typical network conditions observed in different European operators, under 3G and 4G. We exploit its capability of sampling new emulation parameters periodically, mimicking e.g., a user moving around. This is particularly useful to assess the impact of variations in network conditions, which in turn trigger the application mechanisms that adapt the transmission characteristics.

We focus on the Webex Teams, a popular RTC application which allows calls between multiple participants with audio, video and screen sharing media. It uses the Selective Forwarding Unit (SFU) approach, in which participants send/receive multimedia content to/from a centralized server. The media streams are encapsulated using the RTP protocol [2]. Webex implements different approaches to react to poor network conditions. In particular, it adjusts the video resolution to avoid the sending rate to exceed the network capacity. Our goal is to create a reproducible environment to evaluate how the application reacts to variations of network conditions, allowing, in turn, the design and measurement of novel approaches.

We sketch the architecture of our testbed in Figure 1, showing two clients communicating via Webex, while ERRANT periodically updates the download, upload and delay network parameters. We use profiles for a single network operator, 3, as measured from Sweden. We make calls using profiles emulating 3G and 4G RATs and different signal qualities. The calls last approximately 1 hour, and ERRANT varies network conditions every 60 seconds. In parallel, we record packet traces on both devices to later analyze network data and collect the Webex logs which indicate, among others, QoE-related statistics such as jitter and video resolution. In total, we collect data for 10 hours of calls, using 4 different network profiles.

3 INITIAL RESULTS

We now show the initial results we obtain out of our preliminary experimental campaign. The figures we provide are obtained by analyzing the packet traces as well as the application logs we collect at each experiment.

We first show in Figure 2 the Webex sending bitrate, under different emulated scenarios, focusing on the equipment where ERRANT runs – i.e., on which we emulate a mobile network. In general, Webex supports HD videos, if the network offers enough capacity. The 3G and 4G good scenarios meet this requirement, and the blue and green lines in Figure 2 show that the bitrate is typically around 3 Mbit/s. When emulating bad signal qualities, Webex reduces the resolution, and, especially with bad 3G (red line), the video often has low resolution, and, in turn, low bitrate. This result shows how, with ERRANT, we can understand the typical call conditions under mobile networks.

We now show the temporal evolution of a call. We are interested in analyzing how Webex adjusts the video quality upon a deterioration of network conditions. Indeed, when the application detects an impairment – e.g., large packet losses, it may decide to downgrade the video quality and, in turn, the sending bitrate. In Figure 3, we show the temporal evolution of bitrate and packet loss during three minutes of call taken under an emulated 3G Bad network. In this example, ERRANT imposed, by chance, two deteriorations of network conditions, approximately at seconds 30 and 115. As a consequence, the sending bitrate of the application started exceeding the network upload capacity and caused a severe packet loss (red line in the figure). Webex reacted by reducing the video quality, which passed from an HQ video to MQ and finally to a further lower quality. This example clearly shows how, with ERRANT, we can trigger and observe the application mechanisms to react to bad network conditions. Interestingly, the application reacts with a sizeable delay, allowing up to 30 seconds of packet loss, and, in turn, poor user's QoE, before reducing the sending bitrate.

4 NEXT STEPS

This work is only a first step towards a thorough system to emulate the behaviour of RTC applications under operational mobile networks. Here, we showed that, using ERRANT, it is possible to trigger a range of application mechanisms designed to react to bad network conditions. Our long-term goal is twofold. First, we aim at fully automating the experiments, so that large experimental campaigns can be instrumented with minimal manual intervention. Second, and more important, we are targeting open-source RTC clients (e.g., Jitsi Meet), on which we can tune and expand the network functionalities, and, then, assess the impact on a realistic environment. Finally, we aim at including 5G network profiles in ERRANT, to test how RTC applications adapt their quality with drastic network changes – e.g., from a high-speed 5G network, down to a bad 3G one.

REFERENCES

[1] Anja Feldmann, Oliver Gasser, Franziska Lichtblau, Enric Pujol, Ingmar Poese, Christoph Dietzel, Daniel Wagner, Matthias Wichtlhuber, Juan Tapidor, Narseo Vallina-Rodriguez, et al. 2020. The Lockdown Effect: Implications of the COVID-19 Pandemic on Internet Traffic. In *Internet Measurement Conference (IMC '20)*.

[2] Ron Frederick, Stephen L. Casner, Van Jacobson, and Henning Schulzrinne. 1996. RTP: A Transport Protocol for Real-Time Applications. RFC 1889. https://doi.org/10.17487/RFC1889

[3] Tobias Hoßfeld, Michael Seufert, Christian Sieber, and Thomas Zinner. 2014. Assessing effect sizes of influence factors towards a QoE model for HTTP adaptive streaming. In *2014 sixth international workshop on quality of multimedia experience (qomex)*. IEEE, 111–116.

[4] Martino Trevisan, Ali Safari Khatouni, and Danilo Giordano. 2020. ERRANT: Realistic Emulation of Radio Access Networks. *Computer Networks* (2020), 107289.