QoE-Lab: Towards evaluating Quality of Experience for Future Internet Conditions

Muhammad Amir Mehmood, Andreas Wundsam, Steve Uhlig,
Dan Levin, Nadi Sarrar, and Anja Feldmann
{amir | andi | steve | dan | nadi | anja}@net.t-labs.tu-berlin.de

Deutsche Telekom Laboratories, TU Berlin, Berlin, Germany

Abstract. Multimedia applications have recently been challenging existing mobile access networks and are raising the bar for next generation mobile networks (NGMN), both in terms of network traffic as well as in the expectations of end users. At the same time, the network and server landscape sees changes due to the advent of virtualization and split-architecture networks like OpenFlow. In this heterogeneous environment, quantitative measurement and prediction of the user's Quality Of Experience (QoE) require testbeds capable of studying these effects combined as well as in isolation, in a controlled and reproducible manner. In this paper, we present the design and architecture of *QoE-Lab*, a multi-purpose heterogeneous testbed that supports a variety of networking conditions to evaluate QoE in future Internet scenarios. QoE-Lab includes 1) NGMN networks, 2) access/backbone network emulation, and 3) virtualization. It provides services like traffic generation, topology emulation and high-precision cross-layer monitoring. The experiments are provisioned, orchestrated and analyzed by a tool called ExpAuto. We analyze the experimental results for relationships between network performance and user perception. We report initial results from our studies, qualitatively indicating that each of the discussed components has a significant impact on the QoE.

Key words: QoE, NGMNs, OpenFlow, virtualization, testbed

1 Introduction

Internet usage patterns have seen changes recently. Multimedia-enabled mobile devices, such as recent smart phones and Internet tablets, are putting unexpectedly strong demands on the networks, in terms of traffic volume, throughput, and latency. The evolution of smart phone and broadband wireless technologies is expected to be responsible for a significant growth of mobile data traffic [2, 3]. Recent traffic studies have already identified a change in application distribution and traffic characteristics caused by user demands and new services [21].

These mobile platforms use multiple wireless access technologies, such as WiFi, and different flavors of 3G UTMS and LTE. Despite the diversity in the network medium, users are expecting ubiquitous connectivity and free roaming between different access technologies without impact on their QoE. At the same time, the wired Internet is evolving as well. Especially at the edge, server virtualization inside data centers greatly improves flexibility, consolidation, and reliability, and brings down operational costs. Split architecture approaches like OpenFlow [22] are starting to be adopted in production networks to provide a more direct and flexible control of network resources.

This heterogeneous combination of technologies creates an entirely unknown environment for data traffic whose combined effects have not yet been widely studied. The data packets of the typical future Internet user might be generated by a smart phone roaming through different wireless access technologies, potentially migrating between different links, traversing the wired backbone through many access and backbone routers (potentially virtualized) and terminating at a virtual machine inside a data center. In spite of this, the user will still expect a seamless, high quality of experience with no interruption to their multimedia service, e.g., while watching an on-demand HD video on their Internet tablets.

According to the ITU-T P.10/G.100 [9], QoE is defined as the overall acceptability of an application or service as perceived subjectively by the end-user. This includes many factors such as end devices, network, service infrastructure, user expectation and the environment in which the user is communicating [28]. The recommendations from the ITU-T for evaluating QoE with subjective tests are described in ITU-T Rec. P.800 [16]. For the evaluation of VoIP quality, the ITU-T recommends the PESQ model (ITU-T Rec. P.862) [17] and the E-model (ITU-T Rec. G.107) [15]. However, recent QoE studies have shown the limitations of these models to capture new transmission effects for next generation mobile networks [20, 23]. Therefore, it is important to include new networking paradigms like seamless mobility where network handovers and service adaptation, e.g., codec changeover and bitrate switching, are prevalent and where virtualized resources are subject to different background traffic properties. Current QoE studies lack consideration for these diverse networking conditions for VoIP, video, and web applications.

The main impetus for our research is to broaden the specific conditions used in today's QoE experimentation to understand the user perception requirements for the future Internet. To this end, we present an integrated testbed called "QoE-Lab" which provides the ability to evaluate scenarios for the future Internet by combining all these new networking entities under different traffic properties with high-precision monitoring at different layers. It exposes applications to complex real networking conditions to gain insight about the user experience. We employ both subjective and objective methods to validate and improve current quality perception models for both voice and video applications.

The main contribution of our research is a heterogeneous testbed that enables evaluation of scenarios and the correlation of user perceived QoE with the networking conditions. It enriches the modular BERLIN [19] testbed framework with support for mobile next-generation wireless networks. QoE-Lab adds several QoE-specific services to BERLIN, including multimedia streaming for VoIP, video and generation of controlled background traffic with Internet backbone/access properties. It also improves the monitoring and instrumentation capabilities by providing high precision monitoring points both at the network, TCP stack, and application level. Among the effects that can be studied are network handovers between different wireless access technologies, the impact of dynamic migrations and resource contention in virtualized scenarios. All these effects can be studied combined as well as in isolation, with repeatable controlled background traffic patterns.

To orchestrate these components and provide repeatable experimentation we developed a software suite, called ExpAuto that handles the setup, orchestration, monitoring, and analysis of the experimental data. To the best of our knowledge, this is the first testbed to address the following diverse goals together for quality perception studies: (i) different background traffic properties which are typical of access and backbone networks, (ii) time-varying channel transmission characteristics which are typical conditions of NGMNs, and (iii) including virtualized networking components in the backbone and edge networks. We believe that studies conducted on this testbed will provide new insights into the design choices for mobility management as well as service adaptation according to the user experience for future Internet scenarios.

We structure the rest of the paper as follows. In Section 2, we explain the key components of the testbed. We discuss the testbed services in Section 3. The experimentation control plane, ExpAuto, which manages experiments is explained in Section 4. We present some illustrative results in Section 5. We discuss related work in Section 6 and summarize our work in Section 7.

2 QoE-Lab Architecture

Designing a testbed with diverse requirements is a challenging task. One critical task is to select suitable components that provide the right level of control at the software and hardware level. The main QoE-Lab components are shown in Figure 1. We use commodity hardware and open-source software in order to ease the reusability of components developed in the community and to be able to contribute to it and share our experience.

QoE-Lab is built upon the modular testbed architecture BERLIN [19]. Its layered structure enables the flexibility and experiment life cycle management required for QoE experiments in order to understand the relationship between user perception and network performance in future Internet scenarios.

2.1 BERLIN **Experimental hardware**

BERLIN's hardware includes 30+ commodity rack servers with 2-8 cores, routers from Cisco and Juniper, and switches from Cisco, HP, NEC, and Quanta. Special purpose NetFPGA cards are available at a subset of servers. BERLIN features a hybrid, customizable physical topology. One part of the testbed is organized in a *router-centric* fashion for experimentation with commercial routers and current routing protocols. The other part is *switch-centric*, with devices fully meshed onto a manageable switching fabric with 200+ ports. This part is mainly used for clean-slate experiments.

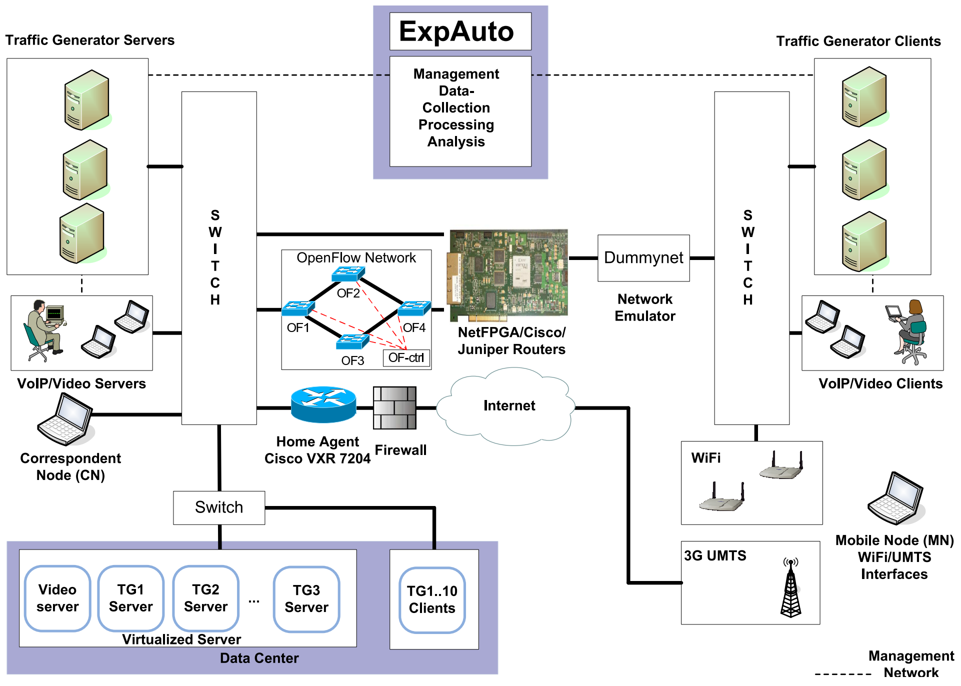


Fig. 1. QoE-Lab overview.

2.2 Integration of Heterogeneous Wireless access

For our QoE study, we add state of the art wireless access technologies having similar characteristics of production-grade NGMNs. The main components of the NGMN part of the testbed consist of the Mobile Node (MN), the Correspondent Node (CN) and the Mobile IPv4 Home Agent (HA). The MN acts as the VoIP/video client and the CN as the VoIP/video server. All communication between the CN and the MN is managed by the HA. MN has the ability to connect to WiFi, 3G UMTS/HSDPA at the same time and maintain calls while roaming between these technologies. Integrating new wireless technologies require only minor changes in the testbed. With this setup, subjective quality tests for multimedia applications related to mobile conditions can be performed with users in online as well as offline manner.

2.3 Labtool management system

The Labtool [19] software management system provides a foundation for the QoE-Lab, sandwitched between QoE-Lab and the foundation infrastructure hardware. It serves as the primary interface through which users of the QoE-Lab reserve and simultaneously interact with their experimental hardware and perform system-level device configuration. The Labtool is experiment-centric, in that it organizes all of its functionality around the management, configuration, and repeatable deployment of experimental topologies and device configurations. The software architecture of the Labtool utilizes a three-layer client, server, and database structure, and is built to be extendable and scriptable with a client API in Ruby. Labtool provides the following functionalities:

Experiment life cycle management: The Labtool maintains an experiment-level view of all the actions it performs. This means that devices, the physical and virtual links connecting them, and their individual configurations are kept in the underlying database schema. This allows for easier hibernation, migration, and restoration of any particular experimental setup.

Physical topology versioning: The Labtool keeps track of all custom cabling changes over time and across experiments. Versioned cabling enables QoE-Lab administrators to alter and reliably restore topology changes.

Boot-mode configuration with imager system: An experiment performed on one set of devices should be repeatable on another set of suitably similar devices. To this end, the Labtool allows experimental device configurations for a given device to be redeployed onto any sufficiently similar device. The Labtool provides a collection of hardware-agnostic base operating system images which facilitate quick deployment of experimental environments.

2.4 Virtualized resources

Virtualization is one of key enabler that drives the evolution of data centers – the server side of the Internet. Thanks to initiatives like OpenFlow [22] it is also rapidly moving into the network domain, both in Enterprise networks and in ISP networks. This development brings many advantages for the operator, including more flexibility and agility in network management and potential for consolidation and reduced operational expenditures. However, these new possibilities can also create new challenges for the user experience, e.g., when flow setup times vary in OpenFlow, hosts are dynamically migrated, or traffic spikes in different virtual network influence each other due to insufficient isolation. To comprehensively study user experience scenarios in the evolving future Internet it is thus necessary to include a broad landscape of virtualization solutions. To precisely quantify the impact of Virtualization, a dedicated testbed is useful where cross-traffic patterns can be controlled and virtualized and non-virtualized approaches can be compared. Shared testbeds that rely on virtualization, e.g., Planetlab [25], are not ideal for such studies, as they do not allow precise control over the environment. QoE-Lab leverages the virtualization support built into BERLIN and provides solutions built onto XEN [10], VMWare ESXi [1] and KVM [18]. It also contains OpenFlow enabled hardware switches from three different vendors that can be flexibly integrated into the experiment setup.

3 Services

Based on the hardware architecture, QoE-Lab enriches BERLIN with important services for QoE studies, including controlled background traffic generation, topology emulation, and high-precision cross-layer monitoring. Next we discuss each service in detail.

3.1 Controlled generation of traffic with Internet characteristics

One of our main design objectives for the testbed is to study the impact of changing trends in traffic properties on user perception. We therefore need nodes which are capable of generating traffic which has similar characteristics to what can be observed in the Internet. To feed the network with background traffic, we rely on multiple PCs. We select Harpoon [29] as our network traffic generator because of its ability to reproduce flow-level behavior consistent with the Internet traffic characteristics. The two main parameters used for customizing Harpoon are the *flow-size distribution* and the connection *inter-arrival* time distribution. We note that TCP traffic makes up most of the traffic by bytes and most flows in the Internet rely on closed-loop feedback. We reflect this TCP traffic component in our traffic generation. One of the distinguishing features of Harpoon is its ability to use the underlying system's native TCP implementation, improving realism. While generating traffic which relies on TCP's feedback mechanism, it is also crucial to be able to change the *packet size distribution* for different experiments. At the time of execution, desired packets size distributions can be specified. Harpoon clients send Web requests for certain file sizes drawn from a pre-fed file size distribution to the Harpoon servers which subsequently responds by sending these objects. The superposition of these connections leads to the bursty behavior of traffic as seen in the Internet.

For traffic generation we use Intel Core2 Duo 2.20GHz servers with 2GB of RAM running Linux. Each server has two dual port Intel 82546 Gigabit Ethernet controllers. Each experimental machine has at least three network interfaces. One is exclusively used for controlling and managing the experiments while the other ones are used for traffic generation. To create different network conditions we rely on different traffic load levels by changing the number of parallel Harpoon sessions. Note, increasing the offered load can lead to different link utilizations. To determine the necessary number of Harpoon sessions, we run the experiments without link capacity limitations. A Harpoon session is equivalent to flows generated by an Internet user. Our testbed allows us to generate more than 10Gbps of peak traffic load.

3.2 Topology Emulation

The network topology we use for *backbone* network is the classical *dumbbell* one as shown in Figure 1. All network interfaces are one Gigabit Ethernet cards. The configurable network bottleneck is located between the NetFPGA router and the Dummynet network emulator. We use Dummynet [27] to add different round trip time (RTT) distributions and to configure different access bandwidth policies for emulating DSL clients with different access bandwidths. In addition, it is also used for creating conditions with packet loss impairments. We prefer Dummynet over NISTNet [13] for its better performance and reliability.

3.3 High-precision cross-layer monitoring

The reliability of experimental results highly depends on the accuracy and degree of the monitoring events during the experiments. Typical flow-level logging tools, such as *netflow*, do not provide precise enough information, e.g., about packet losses within a flow. Thus, we rely on custom high-precision monitoring in our testbed at various networking layers.

Router Buffers: Commercial router vendors such as Cisco and Juniper do not provide fine grained statistics about their router buffer occupancy. To circumvent this limitation, we use a NetFPGA [4] board as a router. A NetFPGA card is a special purpose network card with four one Gigabit Ethernet ports. As shown in Figure 1, two ports are used to interpose it between the Dummynet and the switch and a third port is used for VoIP or video traffic. Fourth port is used for connecting it to the host for capturing buffer statistics. It enables gathering highly accurate buffer statistics such as storing, removing and dropping packets at $8ns$ time granularity. In addition, buffer sizes and link capacities can also be controlled in NetFPGA devices.

Protocol Stack: We monitor the internal behavior of the TCP stack at Harpoon/Video servers using the *tcphook* [31] Linux kernel module, which is based on the In-kernel Protocol Sniffer (IPS). It provides a hook from user space into the kernel TCP implementation. All important TCP protocol stack status information, such as the TCP congestion window size, estimated round trip times and slow-start threshold – *ssthresh*, to be recorded. This set of information is vital for establishing cause and effect relationships between network performance and QoE.

Data capturing: We capture packet level traces at all the experimental machines. By comparing these captured traces, we are able to pinpoint missing packets along with transport layer information, e.g., TCP sequence numbers as well as timing information about when the drop occurred. In addition, we can observe all generated flows from the ingress and egress ports traces. This data enables us to understand per-flow loss process.

Application: Going a step further, we also provide a monitoring for the jitter buffer within VoIP application. This view enables us to study packets that are received but affected by jitter and therefore not usable by the application. These effects cannot be observed by monitoring the networking layer only. We also record the VoIP speech signals for validation of speech quality prediction models for NGMN conditions, such as network handovers, codec switchover and bitrate switching.

4 Experimentation Control Plane

The experimentation control plane ExpAuto of the QoE-Lab testbed is designed to run experiments in an automated manner for different networking conditions using the previously mentioned hardware components, and to analyze experimental data captured at different layers. It provides a unified user interface to run experiments parameterized by command line arguments, and is executed from a central computer with ssh access to all the experimental devices through a management network. ExpAuto is implemented as a collection of bash scripts, whereas the analyzer consists of perl, python, and awk components. An overview of ExpAuto is given by Figure 2. Its main features are:

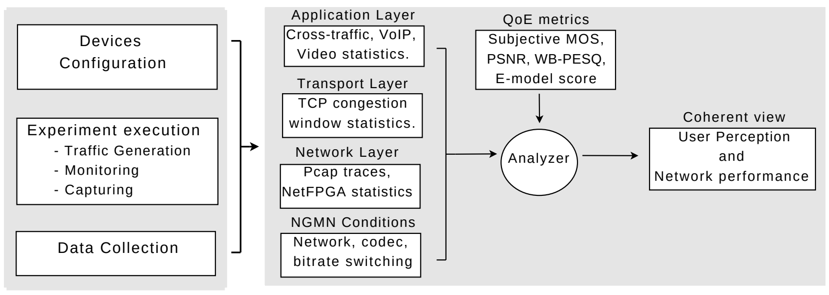

Devices Configuration
Experiment execution
- Traffic Generation
- Monitoring
- Capturing
Data Collection
Application Layer
Cross-traffic, VoIP, Video statistics.
Transport Layer
TCP congestion window statistics.
Network Layer
Pcap traces, NetFPGA statistics
NGMN Conditions
Network, codec, bitrate switching
QoE metrics
Subjective MOS, PSNR, WB-PESQ, E-model score
Analyzer
Coherent view
User Perception and Network performance

Fig. 2. ExpAuto architecture.

(i) configuration for experimental scenarios, (ii) measurements and trace collections, and (iii) automated analysis. We next describe each functionality in detail.

Configuration for experimental scenarios: To be able to create different combinations of networking scenarios with traffic load and burstiness, e.g., client-server where download traffic exceeds upload traffic or P2P where upload traffic is comparable to download traffic, we have pre-configured such scenarios in ExpAuto. This means that the number of Harpoon processes are adjusted accordingly in both directions to match the traffic load. After receiving parameters through the command line interface, relevant configuration files for the selected scenarios are deployed on the experimental devices for traffic generation and network emulation. Dummynet nodes are configured to provide specific round-trip time distributions and packet delays. A separate management network is used for this control information. A separate process is used for controlling the VoIP/video application parameters.

Measurements and trace collections: After starting *tcpdump*, *tcphook* and NetFPGA capturing processes on the experimental devices according to the experiments requirements, the experiments are executed for the desired duration. Once the experiment runs are finished, all processes are stopped and locally collected experimental data is transferred to a data storage server for further storage and post-processing.

Automation and Analysis: ExpAuto orchestrates and collects data from several high-precision monitoring points in the devices across different layers, as described in Section 3.3. To obtain a consistent view of the experiment across multiple layers, it merges different network statistics from the NetFPGA buffer and packet-level traces. These statistics allow us to precisely reconstruct the behavior of individual flows at the transport-layer and correlate it to events that take place within the testbed hardware, e.g., inside router buffers. After experiments, ExpAuto's analyzer post-processes the raw data and provide graphs using Gnu-R. These graphs visualize traffic properties, such as throughput at different timescales, RTT measured distributions and congestion window distributions, etc. Due to its modular design new analyzers can easily be added.

5 Experiments

In this section, we present results from the experimental studies carried out on the QoE-Lab testbed. Due to space limitations, we do not present full quantitative results here — our aim is to qualitatively assert that each of the testbed component has a noticeable impact on the Quality of Experience (QoE) as perceived by the user. We start by showing the impact of different load regimes on audio and video QoE. We then illustrate the impact of common virtualization events such as virtual machine *migration* and host *overload* on video QoE. As split-architecture approaches like OpenFlow are currently gaining importance, we evaluate the impact of a prototype OpenFlow setup on video QoE under load. In the end, we provide a glimpse of the impact of network handovers between WiFi and 3G UMTS/HSDPA technologies on video QoE.

5.1 Methodology

To illustrate the usability of our QoE-Lab, we present how various different networking components and networking conditions impact the multimedia quality of experience. To this aim, we use two types of network traffic in our QoE-Lab. For background traffic, we choose the Harpoon traffic generator for generating web-like workloads. In our experiments, Harpoon is configured with flow sizes generated from Pareto distribution with $alpha = 1.2$ and $shape = 1500$ (bytes). The inter-connection time has exponential distribution with mean $\mu = 1$ s. These two parameters entail that the generated traffic is bursty in a manner similar to Internet traffic. For multimedia traffic we capture a 55 second sequence from a real IPTV stream with video in standard definition (SD) resolution, encoded in H.264 and audio encoded using the mp2 codec, the standard format for high-quality portable video today. We process this IPTV video stream so that it can be easily replayed by using *tcpreplay* [7] from any server in the Internet. We capture the multimedia stream by using *tcpdump* [6] at the receiver and extract audio/video quality metrics by using T-V-Model [26] – a parametric based IPTV quality prediction model. The T-V-Model extracts information about which packets are lost and based on the importance of the packet, it reports audio/video quality, i.e., Mean Opinion Score (MOS) for every 16 seconds interval where MOS value of 4.5 is the maximum and 1.0 is the minimum, i.e., unacceptable quality. We subjectively validate the MOS results by playing the video stream at the receiver in *VLC*. We also capture Harpoon TCP traffic at the sender and receiver along with the TCP congestion window statistics at the servers and buffer statistics from the NetFPGA router. This monitoring enables us to understand the interactions between different networking layers and to establish relations between network performance and QoE. Each experiment lasts for two minutes. After starting the background traffic, we wait for 60 seconds to let it stabilize. We start the video streaming after this initial period and report the results for the second minute during which background traffic and video traffic compete for the limited bandwidth. During all the experiments, the bottleneck link capacity and the buffer size at the NetFPGA router are set to

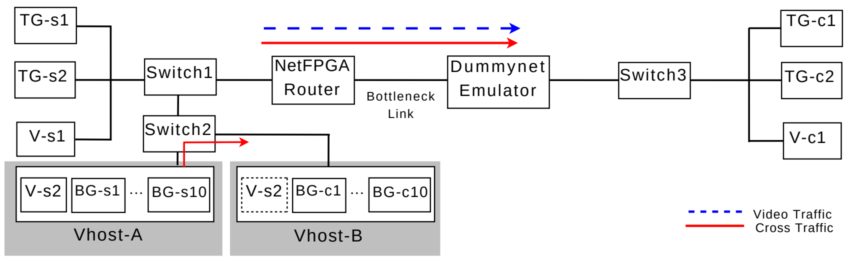


Fig. 3. Setup for video QoE experiments with varying load and video server migration.

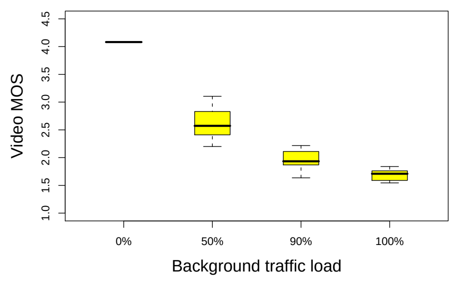
242Mbps and 128 packets respectively. To emulate WAN conditions we introduce a delay of 155ms to every ACK packet of the background traffic going in the direction from the clients to the servers. For every parameter setting, we repeat the experiment multiple times to get quality estimations.

5.2 Impact of background traffic on audio/video QoE

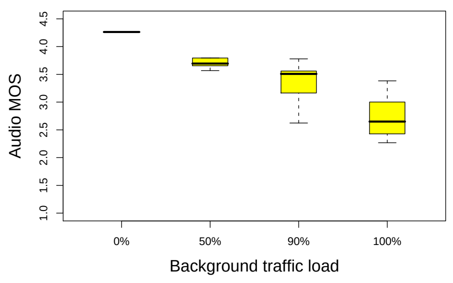
Here we aim to understand the impact of background traffic with different levels of burstiness on the Quality of Experience for audio and video applications. For this purpose, we use the servers TG-s1/2 and clients TG-c1/2 for background traffic generation as shown in Figure 3. Similarly we use video server V-s1 and video client V-c1 for multimedia traffic. During these experiments no traffic is generated from Vhost-A or Vhost-B. The results for audio and video QoE are shown in Figure 4. As a baseline, we first measure audio and video quality without background traffic, resulting in a MOS of 4.08. Then, we repeat the experiments by adding background traffic, leading to a link utilization of 50%, 90%, and 99% on the bottleneck link. It becomes apparent that video QoE is much more sensitive to network traffic load when compared to audio QoE. The main reason is that bursty packet losses within the video stream affect multiple frames resulting in poor visual quality, whereas in audio, losses are more easily concealed and recovered. Note that we have reported results of only few experiments with different background traffic properties due to space limitations.

5.3 Impact of virtual server migration and overload

Service providers such as Google, Amazon and Microsoft all use large-scale data centers for delivering content to the clients, including on-demand videos. To optimize utilization, reduce costs and ease management, these data centers often consolidate several *Virtual Machines* (VMs) onto a single host that can be managed in a *cloud-computing* fashion. Some companies, e.g., Amazon, even rent out virtual machines to 3rd parties. For studying the impact of this virtualized environment, we use two virtualized servers Vhost-A and Vhost-B running XEN 3.3 with Ubuntu Linux 8.04 (2.6.24) as a Dom0. One virtual machine V-s2 runs on Vhost-A as shown in Figure 3. We now measure MOS results for different scenarios and present the results in Figure 6(a).



(a) Video Results



(b) Audio Results

Fig. 4. Impact of background traffic on video and audio QoE.

In a multi-tenant setup where different entities share a single physical host, *performance isolation* between the different VMs is a critical factor. We investigate how the QoE provided by an on-demand video VM is impacted by background traffic on collocated VMs. We thus overload the active virtualization host (Vhost-A) by starting 10 additional virtual machines BG-s1 to BG-s10 and another 10 virtual machines BG-c1 to BG-c10 on second virtualization host (Vhost-B). One process of harpoon is started with 500 sessions on each BG VM to completely overload the link connecting Vhost-A and switch2. While video transmission was going on from V-s2 on Vhost-A to V-c1, 910Mb/s of background traffic from Vhost-A to Vhost-B was sharing the link in parallel. Even in this heavily overloaded setup, the QoE remains good with a median video MOS value of 3.8 shown as overload (OL) in Figure 6(a). This indicates the queue scheduling of XEN achieves good fairness for outgoing traffic. Complementary experiments show that fairness for *incoming* traffic can be much more problematic.

Virtualization also provides new management primitives that can affect QoE. VMs can be *provisioned* on-demand and after some time, once their utility is no more required, they can be destroyed to free the resources on the host. Virtualization solutions also provide the ability to *migrate* running virtual machines between running hosts, for load balancing or maintenance purposes. This migrations can either be performed in *offline* fashion, i.e., by freezing the VM on the source host, transferring its state to the target host, and then restoring it on the target host, or as *online* migration that minimizes downtime. In this mode, the source VM state is repeatedly and incrementally copied to the target host by the Hypervisor, while the VM keeps on running. When the source VM is finally frozen by the Hypervisor, only a small delta has to be transferred. We investigate how an ongoing migration of a VM acting as a video-on-demand server affects the QoE of the clients, both in online and offline fashion. We again use our two virtualized servers, Vhost-A and B, and a video serving VM starting out at Vhost-A. At the start of the experiment, video is streamed from V-s2 on Vhost-A and during the video session, the virtual machine V-s2 is migrated from Vhost-A to Vhost-B (see Figure 3). The comparison of video quality re-

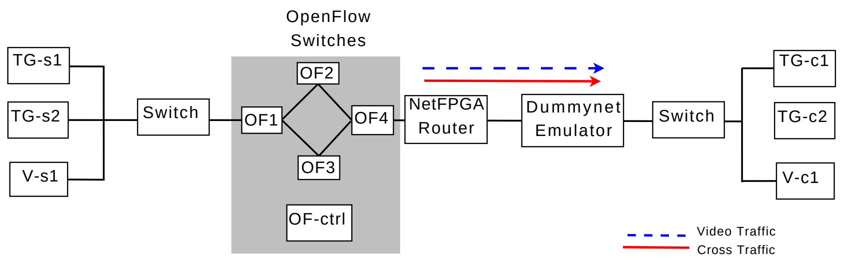


Fig. 5. Setup for video QoE experiments with OpenFlow in the path.

sults indicate that offline migration (OM) has more impact on video quality as compared to live migration (LM) as shown in Figure 6(a). We observe that during migration, $1 - 2$ second of video is lost resulting in an abrupt video quality degradation.

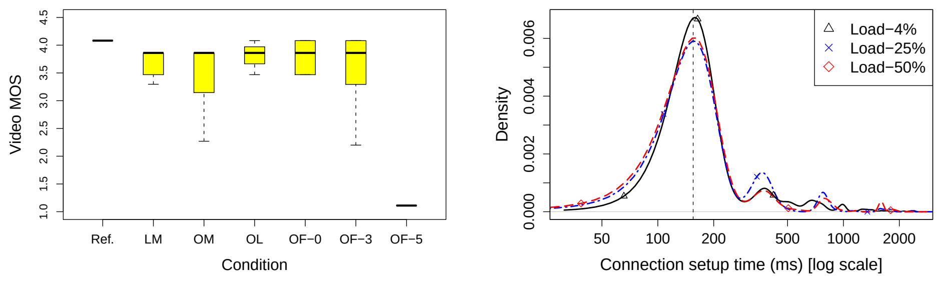
5.4 Impact of a prototype OpenFlow setup on video quality

Recently, OpenFlow [22] has emerged as a disruptive approach to enable evolvable, software defined networks. Despite its prototype status, it is already deployed in production networks in campuses and enterprises. It adds many degrees of freedom to the network management, particularly the *granularity* of flows and whether flows are installed in a *reactive* or *proactive* fashion. We investigate the potential impact of these decisions on QoE by using a simple reactive, fine-grained controller[1].

As shown in Figure 5, we interpose a four node OpenFlow network consisting of prototype switches from two different vendors between the NetFPGA router and the switch. The switches are running OpenFlow 1.0 on prototype firmware, and have flow table sizes of around 2000 entries. They are managed by a dedicated NOX [14] controller running the `routing` module. This module implements shortest-path routing based on Layer 2 destination addresses, with a fine grained flow model, i.e., each individual TCP connection results in two flows. Flow rules are installed reactively. This is the common setup in use in many production OpenFlow networks today.

We are interested in the impact of this setup on the video quality in the presence of background traffic. We use the same methodology for background and video traffic as explained in Section 5.2. The results are shown in the right part of Figure 6(a). First, we send only one video flow without background traffic and we receive highest video quality MOS of 4.08 with little variation, marked as OF-0 in the Figure. Then we add background traffic of low average throughput, with 3Mbps (OF-3) and 5Mbps (OF-5). OF-3 achieves the same

[1] OpenFlow is an open protocol that enables a commodity PC (the *controller*) to exercise flexible and dynamic control over the data traffic passing through Ethernet switches. To this end, the controller defines *Flow Rules* that specify the actions to be taken for matching packets. Flow rules can match on a 11-tuple of Layer-1 to Layer-4 properties, and can be installed *reactively* or *proactively*.

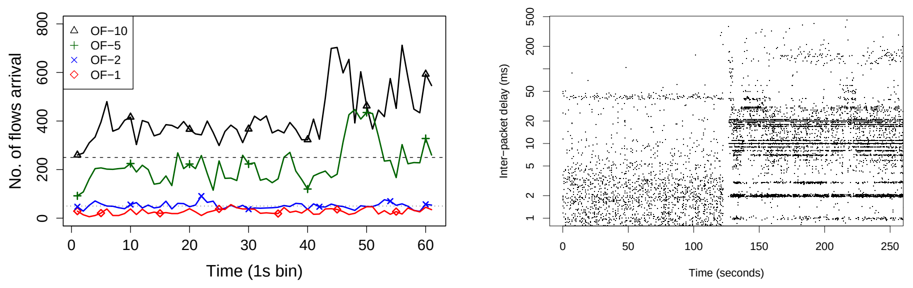


(a) Virtualization networking conditions (b) Connection setup time for different load

Fig. 6. Impact of virtualization on video QoE. LM:Live migration; OM:Offline migration; OL:Overload; OF-X, where X is background traffic load in Mbps

median MOS as OF-0, but shows more variation due to the burstiness of the background traffic. Surprisingly, for OF-5, the 2Mbps increase in background traffic results in a severe degradation of video quality, with a MOS of 1.1. This huge impact on video quality needs further investigation. It appears that in the case of an OpenFlow controlled network, quality is not a direct function of the network throughput as the load is significantly smaller as compared to the load in the experiments of Section 5.2. We start our investigation by looking at the connection setup time, i.e., the time difference between SYN and SYN-ACK packets. As in this setup, the switch needs to contact the *controller* for a decision at every new flow arrival, connection setup time can be a critical factor. For our network we have introduced a RTT of 155ms as explained in Section 5.1. The flow setup time for traffic loads of 4%, 25% and 50% in case of OpenFlow is shown in Figure 6(b). At 25% load, the flow setup time has multiple modes beyond the configured RTT of 155ms. We further look into the flow arrival rate per 1s time bin for different traffic rates going through OpenFlow. The results are shown in Figure 7(a). This figure shows the bursty arrival of flows at the OpenFlow switches. We notice that the flow arrival rate surpasses 250 flows/s for OF-5 and OF-10. Further investigation shows drastic drops in QoE for flow rates beyond 250/s. While the exact cause for this behavior is still under investigation, preliminary results indicate that such high flow rates can overload the switch CPU, and inhibit timely cleanup of expired flow entries from the flow tables. This causes the flow table to overflow and causes random evictions of the video flow, which explains the drastic deterioration in quality.

We conclude that with today's hardware, the frequently used approach of a purely reactive controller with a fine-grained *flow model* can be highly problematic for QoE when faced with realistic, bursty background traffic. Further investigation with different controller logic, reactiveness patterns and flow definitions is clearly necessary, as well as continued improvement of the prototype hardware.



(a) Flow arrival rate/sec for 1-10 Mbps traffic (b) Handover from WiFi to HSDPA

Fig. 7. Impact of NGMN networking conditions on QoE.

5.5 Impact of NGMN conditions

Another important feature of QoE-Lab is the availability of heterogeneous wireless access to assess multimedia quality in next generation mobile network (NGMN) conditions such as network handovers, codec changeover, bit rate adaptation during ongoing calls. The main components of a NGMN are the Mobile Node (MN), Correspondent Node (CN) and Home Agent (HA) as shown in Figure 1.

The CN and MN are laptops running the Ubuntu Linux operating system. The HA is configured on a *Cisco 7204 VXR* router with *IOS 12.1*. The MN has two wireless interfaces, WiFi and 3G HSDPA, and a one Gbps Ethernet interface. The CN is connected to the HA through one Gbps Ethernet. The HA is dual-homed, i.e., connected to both QoE-Lab and the Internet. We connect the HA to the Internet because 3G HSDPA access is available through commercial operators only. The degradation in multimedia streams from CN to MN is realized by (i) cross-traffic interactions, (ii) Dummynet, or (iii) *netem* [8] in case of public Internet.

The WiFi access point is configured as a standard IEEE 802.11g router on OpenWrt [5], set to 54 Mbps. The UMTS/HSDPA connection as provided by a large European service provider operates at 7.2 Mbps in the download direction. For the mobility support, we rely on a “make-before-break” policy using *lmip*, a closed-source implementation of a Mobile IPv4 client. We use *lmip* at the MN, which allows the MN to perform network handovers during on-going sessions. Note that our setup allows us to switch between on-going multimedia sessions from controlled settings in our experimental setup to the real Internet. Figure 7(b) shows the inter-packet delay for video transmission. This call is started while the user, i.e., MN is attached to the WiFi network and after 125s, MN is switched to 3G HSDPA network. These results show that HSDPA network conditions are more aggressive as compared to WiFi. The main reason is due to the fact that HSDPA is a shared downlink and gives preference to users having good channel conditions in allocating resources. The detailed results of NGMN studies conducted on QoE-Lab are available at (i) Audio-visual QoE [12, 23], (ii) Mobile video QoE [24].

6 Related Work

Future Internet studies mostly rely on simulators, emulators and real testbeds. For QoE studies, testbeds are the most popular choices especially when subjects are involved. Our effort builds upon the previous effort of Mobisense testbed [30], which was primarily designed to assess the user perceived quality of mobility in NGMNs using VoIP. We extended this testbed to include QoE evaluation for both VoIP and video in the presence of controlled background traffic, virtualized resources and different NGMN conditions such as network handovers, codec changeovers and bitrate switchovers. WAIL [29] and Stanford testbed [11] were built to study the impact of traffic properties and router buffer sizes. However, the focus of these testbeds was not QoE evaluation.

7 Summary and Outlook

This paper presents the design and architecture of QoE-Lab, a modular testbed for the evaluation of Quality of Experience (QoE) of different applications under the heterogeneous networking conditions expected in the future Internet. It features controlled background traffic generation, network emulation, seamless mobility between different access technologies, various service adaptation mechanisms, as well as an interconnection between experimental setup and the real Internet, and includes productive as well as emerging virtualization technologies, including OpenFlow. We augment this testbed with a tool, ExpAuto, which is used for creating different test scenarios, allocating resources, monitoring and collecting data at various networking layers. The qualitative results reported in this paper hint at the significant impact of each of these elements for QoE. More extensive quantitative studies are ongoing. In the future, we plan to add new wireless access technologies such as Femto cells and LTE for QoE studies.

References

1. VMware vSphere HypervisorTM (ESXi). `http://goo.gl/Eyhqe`.
2. Cisco.com: Cisco Visual Networking Index: Global Mobile Data Traffic Forecast Update, 2009-2014. `http://goo.gl/xxLT`.
3. Mobile TV Market: Analysis and Opportunities 2010-2015. `http://goo.gl/XBUWm`.
4. NetFPGA. `http://netfpga.org/`.
5. OpenWrt Wireless Freedom. `http://openwrt.org/`.
6. TCPDUMP. `http://www.tcpdump.org/`.
7. Tcpreplay. `http://tcpreplay.synfin.net/`.
8. The Linux Foundation, Network Emulation. `http://goo.gl/YvN9`.
9. Definition of Quality of Experience. In *ITU-T: Appendix I to P.10/G.100*, 2007.
10. P. Barham, B. Dragovic, K. Fraser, S. Hand, T. Harris, A. Ho, R. Neugebauer, I. Pratt, and A. Warfield. XEN and the art of virtualization. In *Proceedings of ACM SOSP*, 2003.
11. N. Beheshti, Y. Ganjali, M. Ghobadi, N. McKeown, and G. Salmon. Experimental study of router buffer sizing. In *Proceedings of ACM IMC*, 2008.
12. B. Belmudez, S. Möller, B. Lewcio, A. Raake, and A. Mehmood. Audio and video channel impact on perceived audio-visual quality in different interactive contexts. In *Proceedings of IEEE MMSP*, October 2009.

13. M. Carson and D. Santay. NISTNet: A Linux-based network emulation tool. In *Proceedings of ACM CCR*, volume 33, 2003.
14. N. Gude, T. Koponen, J. Pettit, B. Pfaff, M. Casado, N. McKeown, and S. Shenker. NOX: towards an operating system for networks. In *Proceedings of ACM CCR*, volume 38, July 2008.
15. ITU-T Rec. G.107. *The E-model, a Computational Model for Use in Transmission Planning*. ITU, 2009.
16. ITU-T Rec. P.800. *Methods of Subjective Determination of Transmission Quality*. ITU, 1996.
17. ITU-T Rec. P.862. *Perceptual Evaluation of Speech Quality(PESQ): An Objective Method for End-to-End Speech Quality Assessment of Narrow-Band Telephone Networks and Speech Codecs*. ITU, 2001.
18. A. Kivity, Y. Kamay, D. Laor, U. Lublin, and A. Liguori. KVM: the Linux Virtual Machine Monitor. In *Proceedings of the Linux Symposium*, 2007.
19. D. Levin, A. Wundsam, A. Mehmood, and A. Feldmann. BERLIN: The Berlin Experimental Router Laboratory for Innovative Networking. In *Proceedings of TRIDENTCOM*, 2010.
20. B. Lewcio, M. Wältermann, S. Möller, and P. Vidales. E-model supported switching between narrowband and wideband speech quality. In *Proceedings of QoMEX*, San Diego, CA, United States, 2009.
21. G. Maier, F. Schneider, and A. Feldmann. A first look at mobile hand-held device traffic. In *Proceedings of PAM*, 2010.
22. N. McKeown, T. Anderson, H. Balakrishnan, G. Parulkar, L. Peterson, J. Rexford, S. Shenker, and J. Turner. OpenFlow: Enabling innovation in campus networks. In *Proceedings of ACM CCR*, volume 38, 2008.
23. M. A. Mehmood, B. Lewcio, P. Vidales, and S. Möller. Understanding Signal-Based Speech Quality Prediction in Future Mobile Communications. In *Proceedings of IEEE International Conference on Communications, ICC*, 2010.
24. M. A. Mehmood, C. Sengul, N. Sarrar, and A. Feldmann. Understanding Cross-Layer Effects on Quality of Experience for Video over NGMN. In *Proceedings of IEEE International Conference on Communications, ICC*, 2011.
25. T. T. of Princeton University. PlanetLab. http://www.planet-lab.org/.
26. A. Raake, M.-N. Garcia, S. Möller, J. Berger, F. Kling, P. List, J. Johann, and C. Heidemann. T-V-Model: Parameter-based Prediction of IPTV Quality. In *Proceedings of the IEEE ICASSP '08*, 2008.
27. L. Rizzo. Dummynet: A Simple Approach to the Evaluation of Network Protocols. In *Proceedings of ACM CCR*, volume 27, 1997.
28. D. Soldani. Bridging QoE and QoS for Mobile Broadband Networks. In *ETSI workshop on QoS, QoE and User Experience focusing on speech, multimedia conference tools.*, 2010.
29. J. Sommers and P. Barford. Self-configuring network traffic generation. In *Proceedings of ACM IMC*, 2004.
30. P. Vidales, N. Kirschnick, F. Steuer, B. Lewcio, M. Wältermann, and S. Möller. Mobisense Testbed: Merging User Perception and Network Performance. In *Proceedings of TRIDENTCOM*, 2008.
31. J. Yoo, T. Huhn, and J. Kim. Active capture of wireless traces: overcome the lack in protocol analysis. In *Proceedings of WiNTECH*, 2008.