Julius-Maximilians-Universität Würzburg
Institut für Informatik
Lehrstuhl für Kommunikationsnetze
Prof. Dr. P. Tran-Gia

Quality of Experience Management in Virtual Future Networks

Daniel Schlosser

Würzburger Beiträge zur
Leistungsbewertung Verteilter Systeme

Bericht 01/12

Würzburger Beiträge zur Leistungsbewertung Verteilter Systeme

Herausgeber

Prof. Dr. P. Tran-Gia
Universität Würzburg
Institut für Informatik
Lehrstuhl für Kommunikationsnetze
Am Hubland
D-97074 Würzburg

Tel.: +49-931-31-86630
Fax.: +49-931-31-86632
email: trangia@informatik.uni-wuerzburg.de

Satz

Reproduktionsfähige Vorlage vom Autor.
Gesetzt in LaTeX Computer Modern 9pt.

ISSN 1432-8801

Quality of Experience Management in Virtual Future Networks

Dissertation zur Erlangung des
naturwissenschaftlichen Doktorgrades
der Julius–Maximilians–Universität Würzburg

vorgelegt von

Daniel Schlosser

aus

Fulda

Würzburg 2012

Eingereicht am: 04.10.2011
bei der Fakultät für Mathematik und Informatik
1. Gutachter: Prof. Dr.-Ing. P. Tran-Gia
2. Gutachter: Prof. Dr. Markus Fiedler
Tag der mündlichen Prüfung: 07.02.2012

Danksagung

Die letzten sechs Jahre am Lehrstuhl für Informatik 3 waren eine lange, manchmal „stressige“, aber in den meisten Fällen sehr schöne Zeit. Während dieser Zeit haben mich viele Studenten, Kollegen, Freunde und Lehrer begleitet und unterstützt, bei denen ich mich bedanken möchte.

Zuerst gebührt mein Dank meinem Betreuer und Doktorvater Herr Prof. Dr.-Ing. Phuoc Tran-Gia. Er hat mich nicht nur am Lehrstuhl für Informatik 3 aufgenommen und auf viele spannende „Journeys to the East“ begleitet, sondern es mir auch ermöglicht an vielen interessanten Forschungs- und Industrieprojekten mitzuwirken. Neben seinem Rat und dem stets wertvollen Meinungsaustausch war er immer ein Vorbild im Umgang mit Studenten und Mitarbeitern.

Bei Herrn Prof. Dr. Markus Fiedler möchte ich mich für die Begutachtung meiner Arbeit und die vielen hilfreichen Kommentare und Anmerkungen bedanken. Herrn Prof. Dr. Andreas Hotho und Herrn Prof. Dr. Reiner Kolla danke ich dafür, dass sie sich für meine Arbeit interessiert und zusammen mit Herrn Prof. Dr.-Ing. Tran-Gia die Prüfungskommission für meine Disputation gebildet haben. Prof. Dr. Kurt Tutschku und Dr. Tobias Hoßfeld habe ich es zu verdanken, dass ich den Weg vom Hiwi zum Mitarbeiter am Lehrstuhl gefunden habe. Darüber hinaus haben sie mir gezeigt, wie wichtig es ist, Ergebnisse nicht nur zu erzeugen, sondern sie auch zu verkaufen. Bei Herrn Prof. Dr. Nguyen Huu Thanh möchte ich mich für die gemeinsame Arbeit im Projekt Ecodane und die Einsicht bedanken, dass man auch in einem Berliner Schnellrestaurant kurz vor Mitternacht noch produktive Meetings haben kann.

Spannend wurde meine Arbeit am Lehrstuhl durch Projekte und die vielen Partner aus der Industrie und Wissenschaft, die neue Fragestellungen aufgewor-

fen haben und gerne bereit waren, ein Thema von einer anderen Perspektive aus zu betrachten. Besonders erwähnen möchte ich an dieser Stelle Frau Anja Moog-Lölkes, die bewiesen hat, dass offene und ehrliche Kommunikation mit Industriepartnern möglich und produktiv ist. Diese vielen Projekte wären nicht so erfolgreich verlaufen ohne die kostbaren Empfehlungen und Richtlinien zum Projektmanagement von Herrn Prof. Dr. Harald Wehnes. Aber auch die stetige und unermüdliche Arbeit im Sekretariat von Frau Gisela Alt hat viel zum Erfolg meiner Dissertation beigetragen. Ohne ihren Einsatz hätte ich viel mehr Arbeit in Akten, Bestellscheine und Rechnungen der Hardwareverwaltung stecken müssen und hätte keine Zeit mehr gefunden, Papiere zu schreiben. Daher gebührt ihr ein herzliches Dankeschön.

Die Basis einer wissenschaftlichen Arbeit entsteht nicht nur bei Diskussionen mit Kollegen im Fachgebiet, sondern noch viel häufiger bei Besprechungen im Etagensozialraum nach Dienstbeginn und vor der Arbeit. Ich danke daher allen Kollegen, mit denen ich so wundervoll über Projektkürzel, -anträge und mehr oder minder sinnvolle Ideen für die Netze der Zukunft debattieren konnte: Dr. Klaus Heck, Dr. Jens Milbrandt, Dr. Rüdiger Martin, Dr. Andreas Mäder, Dr. Alexander Klein, Matthias Hartmann, David Hock, Dominik Klein, Frank Lehrieder und Christian Schwartz.

Besonderen Dank schulde ich Herrn Dr. Rastin Pries neben der fachlichen Diskussion für die vielen Anmerkungen und Korrekturen zu dieser Arbeit. Daneben möchte ich den Kollegen Dr. Simon Öchnser und Florian Wamser aus dem LSV-LS3-Vorwärts und meinem Sparringspartner Matthias Hirth danken, dass sie geholfen haben, das Projekt „mens sana in corpore sano“ nicht scheitern zu lassen. Einen regelmäßigen Ausgleich zur wissenschaftlichen Arbeit am Schreibtisch habe ich im Judo gefunden, zu dem mich Herr Prof. Dr. Michael Menth gebracht hat. Neben so manchem Wurf hat er mich bei seiner hingabevollen Arbeit am Lehrstuhl vor allem gelehrt, wie wichtig es ist als Teamleiter immer für jeden in der Gruppe ansprechbar zu sein, auch dann wenn man selbst gerade zu mehr als 200% ausgelastet ist.

Der IT-Betrieb eines Lehrstuhls ist eine komplizierte Angelegenheit, die

man nur mit einem echten A-Team stemmen kann. Ich danke allen vollzeitbeschäftigten Mitgliedern des Administrationsteams, namentlich Dr. Robert Henjes, Dr. Michael Duelli und Michael Jarschel für die hervorragende Zusammenarbeit genauso wie den A-Team Hiwis: Steffen Gebert, Inanc Gürültücu, Christopher Metter und Nicholas Gray.

Wie man viel Benzin und manches Meeting sparen kann, wenn man gemeinsam zur Arbeit fährt, durfte ich mit Thomas Zinner erleben. Besonders danken möchte ich Herrn Dr. Andreas Binzenhöfer für seine geduldigen Hilfestellungen bei meinen ersten Publikationsversuchen und unserer Matlabflüsterin Dr. Barbara Staehle. Ohne ihren Beistand hätten meine Bilder nie in so wundervollen Farben entstehen können. Auch ihrem Mann Dr. Dirk Staehle bin ich zu Dank verpflichtet, da er immer bereit war, ein Thema kontrovers zu diskutieren und auch dann noch ausharrte, wenn die meisten Kollegen sich schon wieder dem Tagesgeschäft widmen wollten.

Außer den vielen Kollegen und Professoren möchte ich meinen Diplomanden und Bachelorstudenten danken. So oft sind sie und besonders Valentin Burger über ihre Leistungsgrenzen hinausgewachsen, um die hochgesteckten Erwartungen gerade noch vor Projektende zu übertreffen. Es macht mich stolz zu sehen, wie viele von ihnen inzwischen selbst promovieren und/oder gute Stellen in der Industrie gefunden haben. Darüber hinaus möchte ich mich bei Marieta Stoykova, Sebastian Goll, Christian Sieber und Kathrin Borchert bedanken. Ohne ihren tatkräftigen Einsatz wäre das Intranet des EuroNGI/FGI/NF Network of Excellence nie in der jetzigen Form entstanden.

Zum Schluss möchte ich mich noch ganz herzlich bei meiner Familie bedanken für die moralische und finanzielle Unterstützung während der ganzen Studienzeit. Ein besonderes Dankeschön möchte ich meiner Partnerin Claudia Hofmann aussprechen. Sie stand mir immer zur Seite, wie lange die Arbeitsnächte waren, wie weit die Reise mich führte oder wie unausgeglichen ich von der Arbeit zurückkam. Claudia, du bist immer mein zu Hause.

Contents

1 Introduction

We're living on the edge.
Steven Tyler

In the past scientists and engineers created communication networks for a special purpose. They built networks for telephony, television, and data transmissions between computers. Even if they integrated other services into these networks, they constructed them using the same communication methods, e.g. for implementing the fax service a standard phone call is used transmitting bits as different tones, or separated them strictly, e.g. data transport in cellular networks. These networks are only accessible for the operator, which provides the services and charges for them. The Internet was initially one of these special purpose networks. It was constructed to robustly connect computers with a packet-switched network, which is able to survive disasters, i.e. outages of one or even more network nodes, cf. [20].

In the last decades the Internet grew in terms of networks operated by different providers, interconnections of these networks, transmission speeds, and attached end points, e.g. universities, enterprises, and customers. With this increase, new markets were built and new business models evolved. For instance, we watch TV shows on Youtube, phone using Voice over IP (VoIP) services like Skype, chat and twitter instead of writing SMS, and use email instead of sending letters or facsimiles. But the Internet was never designed for these kind of services. It was invented as a best-effort network in which all packets are equal. For the initial services, e.g. email, ftp, irc this philosophy worked perfectly, since the offered resources of the network always exceeded the demand of the users and none of

the services required a reliable real-time transmission. However, for the many services, e.g. video-conferencing, VoIP, video-on-demand, best-effort may work in many cases, but cannot guarantee as satisfactory quality in any case.

Currently, we notice that some services with high resource requirements do not work as expected anymore. For instance, a German Internet Service Provider (ISP) had difficulties with delivering Youtube videos without stalling, cf. [21]. Another problem the large ISPs currently face is that service providers like Google or Akamai only pay the ISP they are directly connected to. The large transit ISPs with a lot of small ISPs attached have to forward the traffic from the service providers through their network. However, they cannot identify the transit traffic that requires a high Quality of Service (QoS) and therefore, they have to provide high quality for all transit traffic, for which they are not directly reimbursed. Current approaches to address these problems are built based on deep packet inspection , i.e. routers scan the content of each packet at the beginning of a flow, cf. [22, p. 169] and violate net-neutrality as well as the privacy of the end-customers, cf. [23].

A promising solution to overcome all these problems is *network virtualization*. Like server virtualization, network virtualization separates functionality and implementation. This *abstraction* allows to use any kind of physical networks, i.e. fixed networks, wireless mesh networks, and even cellular networks, without knowing the way they are built. Furthermore, the abstraction allows to partition resources and share them to different concurrent virtual devices. Consequently, we are able to run different concurrent virtual networks on the same physical infrastructure.

Whenever virtual networks are run concurrently, they might interfere with each other. In the worst case, a failure or overload in one network causes another concurrent network to discontinue the service. We, therefore, need *isolation*, which means to ensure that interference never happens, e.g. by assigning different wave lengths to different virtual networks in DWDM, or at least that extensive resource usage of one network is only slightly noticeable in concurrent networks.

Network virtualization providing abstraction and isolation is the basis for the consolidation of network resources, while supporting functional differentiation for various services. Standardized interfaces, which compose a virtualized *control plane* for the complete life-cycle of a virtual network, are able to create, operate, and tear-down a virtual network independent of the control plane of the physical implementation. In order to use resources more efficiently, virtual networks can be tailored to fit to the service. This includes fast changes to the network size and reach. Furthermore, the differentiation of services allows to apply a divide-and-conquer approach to current network management strategies, which reduces the complexity of this task.

For computer scientists, network virtualization provides a plethora of new and interesting research areas. First of all, we need methods to virtualize network devices and connections that guarantee the isolation of concurrent networks. Interfaces to access and control virtual components need to be defined and standardized. Methodologies for management and monitoring have to be adapted to virtual networks, as many of them rely upon assumptions about the actual implementation.

In this monograph, we focus on virtual networks built for delivery of a special service. In this case, the virtual network is planned, dimensioned, and operated to satisfy the user demands in terms of perceived service quality. The main purpose is to find use cases for virtual networks and discuss what is needed to provide them with a high Quality of Experience (QoE) to the end-customer. We, therefore, investigate current virtualization techniques and analyze how to assess the QoE of the user. We characterize the requirements of a service from a technical point of view and research how we are able to monitor both, the technical aspects and the users' satisfaction of a service.

1.1 Scientific Contribution

In this monograph we study Quality of Experience management in virtual future networks. Hence, we consider network virtualization and network management,

which comprises of network monitoring, i.e. measurements of relevant aspects, and network control, which means changing settings in the network or the application services. The contribution of this monograph is three-fold.

The first contribution is in the area of virtual network architectures and isolation of virtual network. We discuss the general aspects of network virtualization. We describe use cases, which cannot be implemented using current technology for two main reasons: Either there are significant shortcomings in the current technical implementation, or the mostly manual operation of nowadays networks changes too slow or is to inflexible to implement these use cases. These use cases clearly demonstrate the potential of virtual networks in terms of business models and improved service quality. Based on these use cases, we derive basic building blocks of virtual networks and propose a virtual network architecture, which defines functional roles and interaction points. Besides interfaces to contract resources based on Service Level Agreements (SLAs), we recommend to integrate interfaces for exchanging monitoring data. These interfaces allow to establish a common view between the different roles, which consider different aspects of the network. Without these interfaces, the roles are unable to compare their quality measurements and are not able to resolve problems caused by interactions between the different layers. From this point, we extend the discussion of the architecture design towards the possibilities, which current virtualization techniques offer. We characterize different hardware virtualization solutions according to their performance overhead and their isolation quality. Our results demonstrate that many current solutions reveal isolation deficits, which affect the operational use in data centers as well as the credibility of research results originated from virtualized experimental facilities.

The second contribution is situated in the area of network monitoring. We consider the assessment of the user-perceived QoE for two business-critical applications, namely VoIP and Microsoft (MS) Office applications delivered as Software as a Service (SaaS). VoIP has come into its own and many enterprises and telephone operator now rely on it, since classical telephone systems have already been replaced by VoIP systems with varying success regarding speech quality. .

Office applications in the cloud are currently gaining momentum for enterprises. The Gartner group estimated the revenue of SaaS application for 2010 with 9.2 billions and expects this market to grow to about 15 billion in 2012, while office applications are responsible for a share of 25% within this market, c.f. [24]. Focussing on Office as a SaaS and VoIP, we study how the QoE can be assessed based on accessible network metrics and how the QoE can be improved by adapting application layer settings. Furthermore, we investigate how sampling can be applied to reduce the measurement effort, of packets that need to be examined, down to 10%. Additionally, we consider the precision of a measurement tool integrated into routers. Such in-network tools probe the connection between two routers and provide measures for all kinds of relevant QoS parameters characterizing the network connection. Hence, the precision of such tools is of particular interest, as they might provide reasonable results, in case the virtual router is able to roam, whereas physical measurement components can not be moved accordingly. In order to extend the meaning of local measurements, we combine these results in a recursive manner and discuss how an end-to-end perspective for a complete network can be calculated. Together, these techniques allow an operator to further reduce his own measurement effort, by integrating measurements provided for network parts from other parties and only validating these results with random control samples.

The third aspect of this contribution is traffic characterization and application layer control. In this area, we focus again on the example of MS Office delivered as SaaS. This example is of specific interest, as Office is considered as business-critical application in most enterprises. Furthermore, the scenario we consider allows for an application layer control, which measurably affects the network layer. Hence, changes on the application layer should not be triggered without knowing the current status of the network, or may be harmful to the QoE and the network otherwise. Therefore, we investigate how the traffic profile in this scenario changes depending on the application and the user. We consider MS Word, MS Excel, as well as MS Powerpoint. Furthermore, we analyze the difference between light, normal, and power users. Based on this basic traffic characteri-

zation, we extend our study towards the costs of QoE enhancing options for the SaaS scenario in terms of bandwidth usage. Combining these findings with the results regarding the QoE monitoring aspect, we are able to derive best practice guidelines, how to trigger these options in a virtual network environment in order to optimize the QoE and the network load simultaneously.

1.2 Outline of Thesis

We structure the previously discussed contributions in a top-down approach: In Chapter 2, we provide use cases for virtual networks that cannot be built using current technology, but are made possible by network virtualization. Either they are too costly to implement or the implementation fails due to insolvable technical or organizational problems. Based on these examples, we derive basic building blocks for network virtualization and propose a virtual network architecture. This architecture defines five different functional roles and interaction points between these roles. We discuss how network monitoring can be integrated into this architecture, which problems arise due to the virtualization, and how to exchange monitoring data between adjacent roles. In the following, we compare our proposal to related work and provide an overview of performance measurements focussing on virtualization. Additionally, we consider five different solutions for hardware virtualization and investigate their performance in terms of virtualization overhead and quality of isolation. We conclude this chapter with lessons learned. The architecture and findings are based on the results of our publications [8, 9].

In Chapter 3, we focus on the challenges of the interaction point between the user centric role on top of the architecture and the network operator role beneath. After an introduction to the background, we provide an overview of different QoE assessment techniques and discuss related work. Next, we focus on a QoE-QoS-Mapping for the Skype VoIP codec SILK. We derive a precise model for a worst-case assessment in this scenario and explain how sampling can be used to reduce the necessary monitoring effort. Afterwards, we analyze the delivery of MS Office as SaaS and discuss how the QoE is affected by different QoS pa-

rameters. Additionally, we study how application layer options, provided by the considered SaaS solution, can be used to optimize the user-perceived QoE. We conclude this chapter with lessons learned. The content of this chapter summarizes the findings of our publications [10, 12, 14].

Chapter 4 is directed towards the dimensioning of networks and QoS measurements. We first introduce the technical background and discuss related work. Next, we reconsider MS Office as a SaaS use case and investigate the different influence factors in order to provide a traffic characterization. We analyze different applications and user types. Additionally, we investigate the network layer costs of the QoE enhancing options in terms of required bandwidth. We exemplarily study the accuracy of an in-network measurement tool and demonstrate how measurements of network partitions can be combined to an end-to-end QoS prediction. Finally, we summarize our finding as lessons learned. The results in this chapter combine findings of our work published in [12, 13, 16].

In Chapter 5, we summarize the main findings of this work and draw conclusions for QoE management in future virtual networks.

2 Virtual Networks in the Future Internet

It's the end of the world as we know it, but I feel fine.

Bill Berry

The Internet of today provides access to many services, e.g., voice over IP, video streaming, content delivery for fast file transmission, and email. However, the protocol structure of the Internet is inflexible. Hence, it is hard to introduce new services that cannot be supported by networks built for TCP/IP transmissions. Furthermore, we currently notice trends, which influence the future of the Internet. First, an increasing number of users connect to the Internet using a wireless link, which currently cannot provide the same Quality of Service (QoS) as a fixed network. Second, the Internet is changing from an everywhere network to a 'real-time' network. It is no longer sufficient to just transmit the information. The information should be available everywhere in real-time, independent of its type. In this context real-time means, that the delivery of the information has a user-specific and context-specific deadline, after which the information is less useful. Hence, the Quality of Experience decreases if the information is not delivered until the deadline. The customer does currently not necessarily differentiate between short twitter messages and HD video clips. However, the Internet architecture is still bound to its best-effort basis and not able to satisfy the technical requirements of high quality real-times transmission, e.g. on-demand provisioning of connections providing a well-defined QoS.

One of the promising solutions to satisfy the demands and provide high quality data delivery services is network virtualization, since it allows efficient set up, operation, and tear down of concurrent network offering QoS. This technique is used to consolidate networks on a functional level and differentiate them on a service level, which allows to construct networks fitting to the needs of the service they provide and reducing management costs. In the future Internet, a multitude of virtual networks coexist and complement each other. These coexisting networks allow specialization but require isolation of functionalities in order to provide dependable networks without interference between concurrent physical resource occupancies. They allow different network technologies to be integrated on a common control plane, e.g. for mobile and fixed networks. Furthermore, they support network resource scalability, i.e., the ability to drastically increase or decrease the network resources, to reduce the time and overhead required to introduce new services, or to change the reach of existing networks. Each network should be able to run its own specialized protocols that may fundamentally differ from today's Internet Protocol (IP) stack.

In order to make network virtualization a successful technology, several steps have to be taken. First of all, use cases need to be defined, which cannot be implemented using current Internet technology due to excessive costs or missing functionality. The architecture, i.e. building blocks and functional relationships, is based on the necessities described in these use cases. Furthermore, the architecture can be validated against the use cases and we can evaluate different architectural options regarding these use cases to identify essential functional units and revise parts, which cannot fulfill these requirements.

Another important aspect in network virtualization is the virtual router. A virtual router needs to perform on a carrier grade level, i.e. the performance and the stability are critical for the integration into the Internet's physical infrastructure. As a first step in this direction, we analyze the performance of currently available hardware virtualization solutions for commodity server hardware.

The remainder of this chapter is organized as follows. In Section 2.1 we define the use cases, which build the basis for the virtual network architecture and are

used to evaluate the architecture. Section 2.2 describes the basic building blocks of virtual networks, constitutes different functional roles needed to operate virtual networks, and characterizes the virtual network monitoring and management architecture. Related work are presented in Section 2.3. The performance of nowadays hardware virtualization platforms with respect to forwarding capacity and isolation between different virtual nodes is considered in Section 2.4. Finally, Section 2.5 concludes this chapter with lessons learned.

2.1 Use Cases for Virtual Networks

Integrating new features into existing networks needs a lot of effort, since installing new components is on the one hand expensive and on the other hand error-prone. In the following, we present six use cases that offer new business models for cost-efficient network provisioning or enable new services. These use cases cannot be or are uneconomic to implemented with current technologies.

2.1.1 Special-Purpose Networks

In the early days of Internet, it was intended to connect computers with long term resilience over fixed data network links, i.e., the data transmission should automatically recover from outages. But recently we observe a convergence to an all-IP network. This means that all kinds of devices are connected to the Internet using all kind of different network technologies, like passive optical networks, cellular networks, mesh networks or even sensor networks. Additionally, all kind of services transmit their data using the Internet. The problem of this converged network is that in most cases neither the network knows the requirements of the service, nor does the service know the status of the network. Hence, it is challenging to guarantee a high QoE to the user or adapt the application layer to the conditions within the network.

Another feature that is missing in today's communication networks is the ability to exchange the protocol stack of the network. Different services have various

requirements to the protocol stack. For example, an IP-TV service might prefer a broadcast network with hop-by-hop acknowledgments. A network for brokers and other financial services might like to set up a protocol stack that only attaches end devices, which have been authenticated and authorized before even attaching to the network. A network supporting anonymity would be interested to camouflage the origin of a message even on the lower layer of the networks. However, with IP networks, these requirements are impossible or very hard to fulfill. Current IP network routers and end-systems are optimized for the IP stack only and cannot be reprogrammed for other network stacks, since many IP systems draw there performance from algorithms implemented in integrated circuits and not from fast reprogrammable network processors.

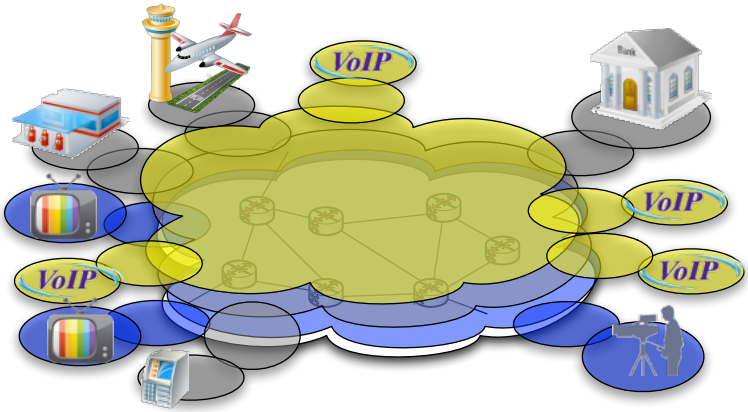


Figure 2.1: Parallel special-purpose virtual networks for different applications.

Virtual networks, which allow to program virtual routers down to abstracted access of the virtualized data link layer, can solve all of these problems. If they are build on programmable network elements, any operator is able to optimize his network dependent only on the supported services. An example for coexisting special-purpose networks is given in Figure 2.1. It is reasonable that during the

creation of a virtual network, the service running on top of the network declares its needs towards the networks capabilities. Conversely, the network guarantees these requirements or reject the request. This feedback of the network enables the service to adapt to the network quality of service and provide the best quality of experience possible under the given conditions. Techniques like functional composition [25] can be used to generate protocol stacks according to service needs. Strict isolation of coexisting networks means that networks running in parallel on the same physical substrate are neither able to see the data of the other networks nor can influence them. Together with on-demand creation, extension, and tear-down of virtual networks, this allows building reliable and more secure network services.

2.1.2 Service Component Mobility

Mobile networks are nowadays wide-spread and cover most of our everyday life. This allows the customer to connect everywhere to mobile phone and data services. However, coverage with high bandwidth and short network delay are still missing in some areas, since a couple of technical challenges, e.g. backhaul network management, still remain. In remote locations, users might have difficulties to access the services they use on a regular basis.

Future virtualized networks integrate the control planes for networks and for virtual server resources. This allows not only the user to be mobile, but also the place of service delivery can be moved, in order to guarantee short access latencies and to reduce the network resource consumption. Services components, which encapsulate all essential parts of the application service, provide virtualized containers that can be migrated to any location in the network. Examples for these service components are, e.g., a transcoder to optimize video streams for network transmission, a translation software, user related web pages and multi media caches, or performance-enhancing proxies, which enable multicast transmissions over different networks. In case of these examples, the service components moves from one place of the network to another, following the movement

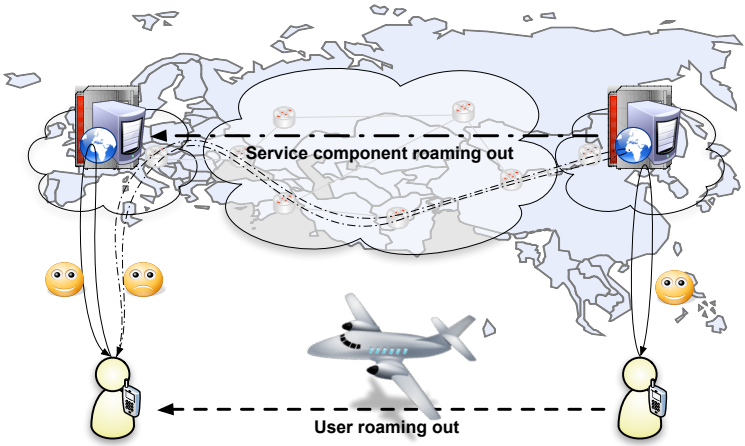


Figure 2.2: Service component mobility.

of the user and optimizing the Quality of Experience.

The live voice translation service can then roam with the user to foreign countries, e.g. from China to Germany and therefore reduce the time the user has to wait for the service to respond. Without the service component mobility, the significant network delay between China and Germany would reduce the usability of the service. Especially for situations with real-time requirements, e.g. during a voice conversation, a network transmission delay of half a second is close to intolerable, cf. [26]. In this case, the live translation service component roams with the user from China to Germany and provides the translation in real-time to users, cf. Figure 2.2.

Another scenario in which service component mobility makes sense from an operators point of view is content caching or distributed content delivery networks. In this case, the operator would locate the content cache close to a sufficiently large user group. The local cache reduces the costs for wide area network link resources and provides better delivery times for flash-crowds.

From a technical point of view, this use case is challenging. Application layer functionality has to be provided at the routers or close to the network edges to migrate service components to wherever a user could possibly be. Furthermore, management tools like cloud management systems need to be adapted to migrate service components between different network operators and different hardware virtualization solutions. Finally, movable service components introduce additional security requirements. If malicious software is able to exploit this migration infrastructure, new thread potentials arise. Strict isolation of different networks is therefore crucial to exclude the interaction of mobile service components with networks and services of other operators.

2.1.3 Autonomous Network Management

Networks automatically adapting to the current load and optimizing their performance are often discussed in literature. However, practical network management is still a challenging task. Many factors have to be considered and in many scenarios it is hard to precisely foresee how control mechanisms on different network layers interact. Additionally, the mixture of different services and applications within the same network add further complexity to network management, as some applications like Skype [27] try to overcome network shortages by sending multiple copies of a packet to suppress TCP traffic. Thus, this mechanism is affecting other applications negatively. Another example is rerouting on OSI Layer 2.5 and Layer 3. If both layers act simultaneously on the same link, error flapping network configurations might be the consequence. Escalation strategies are not always easy to define. Hence, network management is currently mostly done on a reactive manner and on the basis of human interaction.

Virtual networks separating different applications with well-known traffic profiles can be used to simplify the monitoring within each layer. As virtualization introduces additional layers of abstraction, the control can only affect components of the same layer. Together, these facts reduce the complexity of network management. On the other hand, the strict abstraction layers prevent any form of direct interaction between layers. Therefore, the problems with interacting controls of different layers still exist. To solve this problem, interfaces for exchanging control information between different network layers need to be defined. These interfaces allow for a divide-and-conquer approach in network management that is not possible in current networks.

In Figure 2.3, we present the control loops and interaction points of a three-layer virtual network. The lowest layer in this example takes care of the physical infrastructure and virtualization of resources. The monitoring of the lowest layer does therefore focus on the health of the physical infrastructure, the performance of the virtualization and the quality of the isolation between networks. The operator of the lowest layer might also take care of resilience feature, e.g. dedicated link protection. He might also consolidate the used virtual resources on an optimal number of physical nodes and shut down other nodes and links to save energy.

The middle layers are providing some kind of data transport services. Hence, the focus of the monitoring in these layers is on Quality of Service parameters, like loss, jitter, delay, and bandwidth for the regional reach of their virtual resources. Based on the monitoring, the routing algorithms in the control plane might change the forwarding within the data plane. If the problems identified by the monitoring cannot be solved on the same layer or are not reasonable based on the current network configuration, the interfaces to the adjacent virtualization layers are used. Claiming SLA breaches between partners on different layers as well as requesting more or less resources are handled over these interfaces. Furthermore, essential control information is exchanged. If for some reason, a problem in one layer is discovered, the adjacent layers should be notified. Such a notification can provide more information about the problem and should define

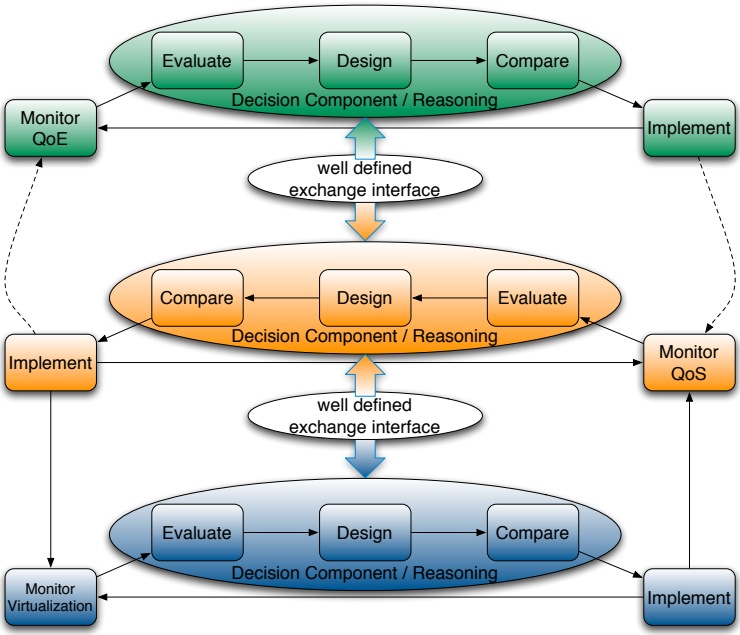


Figure 2.3: Interaction points of management cycles in different virtualization layers.

escalation strategies, i.e. if the layer the problem is located in can handle the problem in a short time scale on its own or if adjacent layers should react. An example would be the switch to a backup path on layer two, which might take some milliseconds but should not trigger rerouting on layer three. On the other hand, it can be used to request temporarily support, if the problem cannot be solved instantaneously. If we consider an IP-TV broadcast service, the customers of this service might increase rapidly due to some special event. Neither the provider of the application service nor the provider of the network could foresee this flash-crowd

and the request of new virtualized resources takes some minutes to negotiate. In this case, the network provider could notify the IP-TV broadcast provider to reduce the bandwidth of the streamed video, so that he can connect new users, even before the additional virtual resources are delivered.

The highest layer is providing an application service, e.g. multimedia streaming, VoIP conferences, or a virtual desktop infrastructure. The operator of this layer focuses mainly on the QoE of the customers. Application-specific parameters, e.g. video frame loss rate or the number of lost voice samples, and customer response analysis are in the interest of the uppermost layer. Depending on the business model, it might be important to map these customer centric metrics into Quality of Service parameters within the network. If this mapping is not provided in the highest layer, it has to be done in a layer beneath, in order to connect the customer-oriented view of the top layers to the network-oriented view in the middle layers.

The interfaces between adjacent layers are also used to share parts of the monitoring data, filtered according to the SLA contracts, in order to provide a common view of the virtualized network. Depending on the measurement technique and the abstraction of the network layer, the view on the same data flow might differ significantly. An example of nowadays network management is that the network provider is monitoring loss on a connection via SNMP requests of its routers, while the application provider estimates the loss by inspecting the TCP headers of the transmitted packets with tools like Tstat, cf. [28]. Although the exact values both receive are different, these values are highly correlated. Exchanging the measurement data creates a common view of the traffic, which enable both to identify problems, optimize the performance of the own measurements in terms of accuracy and monitoring overhead, and correctly assess situations in which the SLAs could not be guaranteed, as it will be discussed in Chapter 4.

2.1.4 Service Broker in Access Networks

With smart-phones, tablet PCs, televisions, and set-top-boxes the vision of a 'smart home' and the Internet of Things is affordable. These days, increasingly many devices are able to connect to the home network. Thus, the home network is interesting for many different service providers, which try to place their products there. For instance, video-on-demand and Pay TV providers try to place their devices in the end-customers network and their applications on the end-customers own devices. Additionally, the customers network is often multi-homed, i.e., it is not only connected by classical telephony and broadband access, but also using wireless broadband services like 3GPP and LTE networks.

The users are in a comfortable position. They can choose from a variety of applications and network service providers. However, most users do not want to have many individual contracts for each service, but prefer all-in-one packages, which provide the required hardware and software and offers a single point of contact if customer service is needed.

This situation calls for a mediating player, who is trusted by the service providers and the customer. We call this mediating player the *service broker*, as it acts as a broker who offers individualized packages to the customer. Preferably, this player takes also care of providing the required software and additional hardware if needed. However, many service providers contributing to the package need exclusive control over the hard- and software required for the service. Hence, the service broker selectively assigns exclusive controllership over some components to the corresponding service provider.

The selective controllership has not only an impact in the home network but also throughout the transport networks, since services are configured end-to-end, between service providers and user equipments. Hence, the attachment of the customer to the provider network is initiated by the service broker. This new role may also be a virtual one: those operators with an unimpeachable footprint in their home country may act as their own service broker while they may contract with other service brokers in foreign countries.

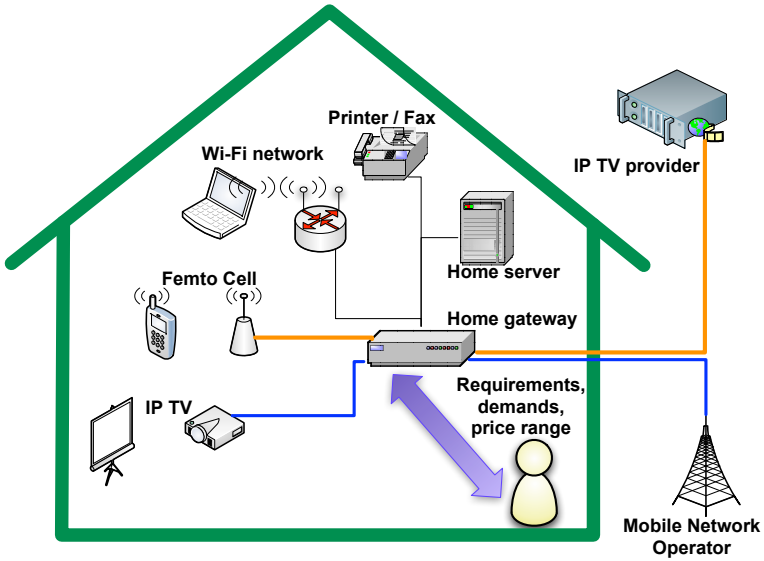


Figure 2.4: Service Broker providing access to different operators delivering services on different end devices.

In order to assemble the best solution for the user, the service broker asks the customer for its personal preferences and select those appropriate service providers that best match the users' demands. Figure 2.4 serves as an illustration for such a scenario. Next, it configures the home network to provide the required resources and assigns these resources to the virtual network of the service provider. For instance, a Pay TV provider is connected to the set-top-box and the mobile network operator is attached to the femtocell access point.

A large network and application service provider, e.g. a global telecom company, might want to act as a service provider itself, bundling its own services for special prices. In other cases, the service provider might be a company which is focusing on "packaging of services of many others". Such a service broker may

bundle e.g. the DSL access of an Fixed Network Operator (FNO) and the cellular service of a mobile network operator (MNO) with the video-on-demand and IP-TV services of video broadcast provider and offer it as package for a certain customer target group – or it may additionally bundle Internet services such as web access and email.

As promising future step, the *service broker* might be extended to the operator level. In this case, the service broker would also negotiate and contract network and computational resources on a global scale, like platform as a service works for current cloud data centers. This way, a Pay TV provider may contract a resource pool, e.g. optical wavelength or TDM frames and balance their usage according to the individual traffic demands of its customer base. Additionally, the service broker of the service provider and the service broker of the customers could interact to utilize spare resources, e.g. bandwidth and forwarding capacity of an optical modem, of the customers in order to support the service and charge some refund for the users.

However, this use case requires a network that provides its resources in a unified manner, regardless of their physical nature, i.e., copper, fiber, or radio. Furthermore, it is necessary that these resources can be partitioned and assigned to different operators, which then have exclusive access to these resources.

2.1.5 Green Networks

Saving energy by better utilizing resources is a trend in today's data centers. But this resource consolidation does not only help to reduce operational expenditures (OPEX), it can also decrease the carbon footprint. The basic technology that enables this consolidation in data centers is hardware virtualization. By migrating virtual servers between physical server machines, it is possible to guarantee high availability and performance while reducing the energy consumption.

In todays networks, we currently face two major problems, which can be solved when adapting to network virtualization. The first problem is that most existing network hardware is not able to run at reduced performance and with

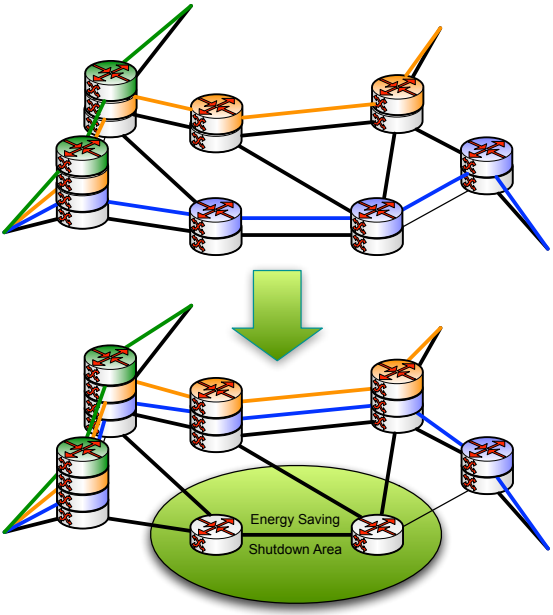


Figure 2.5: Consolidation of virtual routers to optimize the carbon footprint of the network.

lower energy costs. A typical router or switch consumes nearly the same energy in idle mode than under full load. The only way to reduce the energy consumption is to shut down unused interfaces or the complete system. The second major problem is that we are currently not able to migrate network nodes. In order to migrate a traffic demand from one link to another, technologies like GMPLS can be used to signal a new data path and re-route the data to this new path without disruption of the service. However, the new path has to end at the same router or otherwise the network topology changes. Hence, the operator has to reconfigure

its network and a seamless migration is more complicated. Furthermore, traffic demands are sometimes hard to forecast in today's network. In the aggregation networks, traffic demands are easy to estimate, but changes of big players on a global level, e.g. content distribution network operators like Akamai, can change the traffic matrix completely.

Virtual networks solve some of these problems and mitigate others. By introducing an abstraction layer between the physical hardware and the implementation of the virtual router, the operator is able to run software-defined routers on different hardware platforms. Furthermore, it enables the migration of these routers between different nodes, cf. Figure 2.5. Hence, it becomes possible to move routers and links from one physical entity to another, without changing or breaking the network topology. This feature can now be used to operate *green networks*, i.e., a substrate provider can consolidate network resources within his area of responsibility. Other parts of his substrate network, which are not used during low load periods, e.g. the night time, can be shut down. Moreover, this leads to the point where the price of resources is affected by the power management schemes of the substrate operator. If we consider that a part of the substrate network that cannot be shut down due to existing contracts, the owner of the hardware might try to offer unused capacity of the devices at a lower price.

2.1.6 Beta Slice

The term *Beta Slice* denotes a single isolated network that supports the development and roll out of new services. Nowadays, new network services are usually developed in small dedicated testbeds. These testbeds are rather expensive and in most cases scalability cannot be investigated. Simulation might address scalability questions in some cases, but in many cases a detailed simulation takes too much time to compute before providing reliable quantitative results. In a future network, this problem can be solved if it is possible to run different networks in parallel whose resources and reach can be easily adapted, cf. Figure 2.6.

For the *Beta Slice* use case, we consider a service provider that has invented

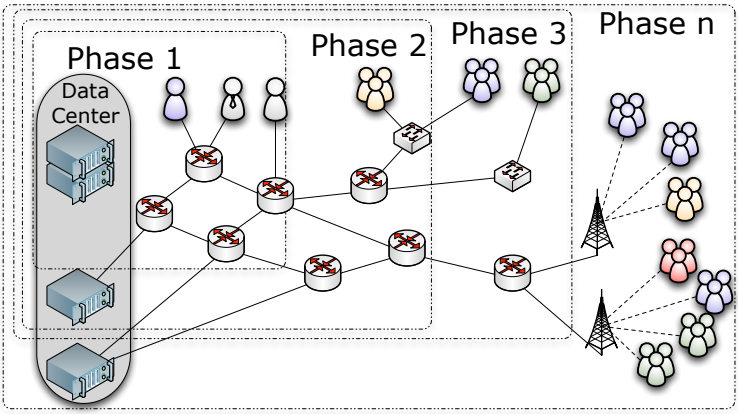


Figure 2.6: Beta slice iteratively extending its size and reach.

a new service, e.g. a new network protocol or a new application service which needs special network features. Instead of creating a specialized testbed, the service provider sets up a network with a small number of hops and a limited geographical reach. The users accessing this network is restricted to a small number, which the service provider can individually select. With this setup, the service provider is able to perform its initial testing and basic functionality tests. If this evaluation phase is successful, the number of network hosts, the network reach, and the number of participating users can easily be increased. With this larger environment, other tests can be performed, which evaluate other functions or consider more inhomogeneous network characteristics, i.e., wired networks and wirelessly connected end hosts. Progressively expanding the network and testing the functionality of the service enables development strategies, which are common in software development but cannot be implemented economically for network protocols these days. Furthermore, this use case allows to test the scalability of a new service in a real world environment and not only in a simulated environment. Another benefit of this development methodology is that the time to market could be decreased significantly.

2.2 Virtual Network Structure

From the use cases described in the previous sections, we can derive some basic building blocks of virtual networks for all use cases. Thereafter, we present functional roles, which interact and create virtual networks. Each role also makes sense from an economical perspective. Finally, we discuss the interworking of these roles and point out interfaces that need to be defined in order to automatically create, operate, and tear-down virtual networks.

2.2.1 Basic Building Blocks of Virtual Networks

The corner stone of network virtualization is the *virtual router* concept. A virtual router basically abstracts the data transmission from the implementation in the physical world and decouples the functionality of the forwarding device from the hardware platform. From the use cases, we see that the spectrum of different abstraction levels is wide. Depending on the knowledge and the requirements of the network operator as well as the trust relationship between the operator and the substrate owner, the access to the virtualized resources might be given on a different layer. If we consider an operator that is willing to contract parts or the complete transmission media, the virtual router needs to provide interfaces, which model the physical access to the hardware very closely. In this case, functionalities like media access schemes and signal transmission is provided by the software of the virtual router. A concrete example for this would be a cellular network operator that has its own physical substrate nodes. In this case, the operator knows how to tune the parameters of its network to optimize it for the offered services. The opposite extreme is a network operator that just wants to have a reliable connection between two or more network end points. In this case, the operator does not want to set up or control the lowest three OSI layers of the network. All the operator wants is to provide a reliable connection between two or more points of presence and therefore expects a single central virtual router, where only the forwarding decision needs to be specified by the software logic.

Link virtualization is simpler to accomplish. Every appropriate form of multi-

plexing can be used to partition the transmission media into separate channels for different virtual networks. However, the requirements of the operator might limit the options of multiplexing techniques. Considering the two previous examples, the requirements of the cellular network operator would only allow for spatial multiplexing, as this operator would be granted with access to the complete functionality of the antenna. In contrast, the operator only requesting a bit pipe could even be satisfied with a multiplexing technique based on packet scheduling. However, all virtual routers accessing the same transmission media need to support the used multiplexing technique and have to be configured accordingly.

At this point, we can set up virtual routers on physical network nodes and create virtual links between them. For larger networks the manual setup of these resources is time consuming and error-prone. Therefore, a *control plane* to set up, tear-down, and connect virtual resources is needed. This control plane needs to know all offered and contracted resources as well as the current status of these resources. Based on this information it is possible to react on resource requests and configure virtual routers and links as needed.

For large networks, it cannot be expected that all physical components are owned by the same company. Hence, we need to provide and store data about which virtual resources of different owners can be connected. The peering of two virtual resources of different owners is even more complex, as both sides do not only have to offer a physical connection and some common protocols, but need also to agree on the virtualization technique and on common parameters to fully specify the way data is exchanged between the peering points. Different virtual resource providers do most probably not only differ in the price of the offered resources, but also in the peering points, the capability of virtual routers, and capacity of virtual resources. Hence, we need an entity that has an overview of the virtual resources available on the market and is able to build global virtual networks based on the points where the network service should be delivered.

The next step is to use the virtual resources and to offer network services. The entity doing this needs to define how data has to be transmitted using the virtual resources. The algorithms the virtual routers execute have to be provided

and installed on the virtual routers. Protocols need to be stacked according to the offered service and delivered to all relevant devices. The Quality of Service a virtual network provides should be monitored. This allows to prove that the network operates within the guaranteed parameters. Failures that depend on virtual resources not working as expected have to be detected and transmitted to the provider of this resource.

Most of the presented use cases are application driven. We, therefore, need an application and entities offering and requesting this application service. The requests, offers, and the requirements of the application service define the delivery points of the virtual network service and the parameters required for the service. In the following, we provide five functional roles that request, provide, implement, or negotiate these building blocks and explain the business models of each.

2.2.2 Functional Roles that Build Virtual Networks

Basically, we distinguish five functional roles. 1) The *Physical Infrastructure Provider* (PIP) owns and operates the hardware and offers virtualized resources. 2) *Virtual Network Providers* (VNPs) gather these virtual resources and construct virtual networks. 3) A *Virtual Network Operator* (VNO) requests networks with special requirements, e.g. setting up a *Service Level Agreement* (SLA), and brings them to life, i.e., it installs the network nodes, defines the protocol stack, and controls the network. At the edges of the network 4) *End-Customers* (EC) and 5) *Application Service Providers* (ASP) request and offer services, which are delivered in high quality by the virtual networks. Figure 2.7 provides an overview of the stacking of the functional roles in a real world environment.

We call all forms of hardware and transport media, which are used to establish the connection between two end hosts, the physical substrate of the network. The entities that own and operate this substrate are called physical infrastructure providers (PIPs).

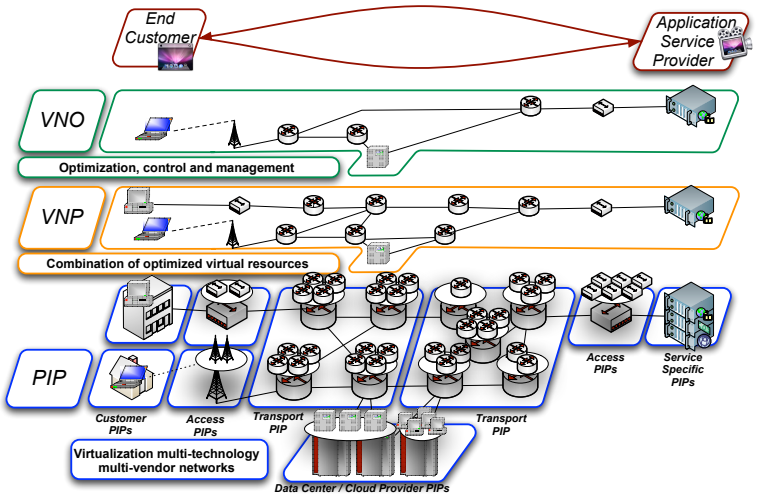


Figure 2.7: Stacking of functional roles

A PIP creates virtual resources on its hardware and offers these resources to its customers. The virtualization techniques have to support strict isolation of virtual resources as explained earlier. Isolation means that overload on one virtual resource does not effect other virtual resources on the same physical resource. The parameters and properties of the virtual nodes and virtual connections, which the PIP offers to a customer, are described by SLAs. Hence, the PIP and its customer have to negotiate a contract including SLAs with the properties of the virtual network the PIP provides. As the PIP has direct access to the physical substrate, it is able to measure the utilization of its physical resources. Furthermore, if the virtualization technology is able to support seamless migration of nodes and connections without affecting the topology and SLAs for the resources, the PIP is able to move virtual resources. This enables the PIP in low load situations to move many virtual nodes on the same physical node and switch off the spare hardware resources in order to save energy costs. Additionally, the PIP can take advantage

of the knowledge of all the SLAs contracted on virtual resources hosted by this hardware. Combining this knowledge with precise utilization measurements, the PIP is able to overbook its physical resources. Another important function of the PIP is to protect network links with resilience features. Only the PIP knows where exactly in the physical network the virtual resources are located and it is able to move the virtual resources on his physical substrate, as long as the topology and SLAs are kept. It is the only one to implement a virtual connection that relies on two disjoint paths in the real world.

The second functional role is the virtual network provider (VNP). The main function of the VNP is to gather resources from a single or different PIPs and combine them to create a virtual network.

The VNP provides interfaces between the PIPs and the VNO to install software and protocol stacks on the virtual nodes. The VNP might want to reserve a pool of virtual resources in order to react quickly on its customer's wishes. On a global market, it is reasonable that there are some VNPs, which only contract certain PIPs and vice versa. Therefore, it is possible for a VNP to request parts of the network he offers from another VNP. The VNP providing a part of the network of another VNP acts as a subcontractor.

The third functional role is the virtual network operator (VNO). The VNO requests a virtual network from a VNP. The VNO uses the interfaces provided by the VNP to install software and network stacks on the virtual nodes. With this setup, the VNO operates and optimizes the network according to its purpose, e.g. a special broadcast network, which is optimized for a certain service like IP-TV or a content distribution network. The network could also support other features like extended security to provide secure banking or cooperate networks. The VNO controls how, which kind of data is transported in the network. He takes care of the operation and maintenance of the virtual network nodes. Furthermore, he keeps a close watch on how the network reacts on network service degradations. If the network does not transfer data as expected, the VNO needs to locate the problem and to act accordingly. If the problems are caused by some part of the networks, which does not operate within the requested parameters,

the VNO has to report the SLA breach to the VNP. Otherwise, the VNO adapts the operation of the network. This could cover changing settings of the deployed virtual nodes or requesting new resources.

Finally, we consider the role of the application service provider (ASP) and the end customer (EC), which can be a single user, a household or an enterprise. The ASP provides the service to be delivered and the target group, i.e., it specifies a service coverage area. Furthermore, as the ASP is the one that knows the service in most details, it is in its responsibility to provide the VNO with QoS requirements, which can guarantee a high Quality of Experience (QoE) perceived by the EC. The EC knows the desired services and decides, which price and quality is acceptable for the service. It has to be noted that even the EC uses special hardware that supports virtualization and acts as a PIP for the EC equipment. This guarantees that the QoE and QoS requirements are kept up to the local end point of the data transmission. Furthermore, the service and requirements can be adapted to the equipment and the network status of the EC.

2.2.3 Interaction of the Functional Roles

In the previous section, we described the functional roles and their tasks. Some of these tasks can be performed within the scope of the corresponding role. Other tasks need the interaction of different roles. Focussing on the *Service Component Mobility* (SCM) use case, cf. Section 2.1.2, we discuss interfaces needed between the roles. These interfaces need to be standardized in order to enable global interworking of the functional roles.

In the SCM use case, an ASP wants to offer an application service to the end-customers. He knows the implementation details of the service and is therefore able to specify QoS parameters a network needs to yield a high QoE. Furthermore, the ASP has to specify the policies under which the service is offered and the reach of the network, i.e., in which area and technology the access to the service should be possible. This information is exchanged between the ASP and the VNO using a network service request interface. This interface has to be stan-

dardized as well as the service description language, in which the parameters are expressed. In response to the request, the ASP receives offers for network services along with the corresponding pricing schemes. In the case of the SCM, the ASP can decide to which regional areas it wants to offer the service. Furthermore, the ASP has to define the required end-to-end delay beside other QoS parameters and if he wants to hand over the responsibility of hosting the service to the VNO. In the case of a video service, the ASP can also determine, if the service has to be delivered using a special codec or if the VNO could adapt the codec to the situation of the network, as long as the service satisfies some QoE constraints. These constraints have to be defined on a technical level, the ASP and the VNO are able to measure, e.g. PSNR, VQM, (C)MPQM, SSIM, NQM, cf. [29].

The VNO receives the network service request from the ASP and decides, how to implement a network with the requested functionality. This means that the VNO generates a virtual topology and decides, which network protocol is used. With these new requirements, the VNO needs to contact a VNP using a network request interface. In reply to this request the VNO receives proposals of virtual networks along with charging criteria, which it communicates with added value back to the ASP. In the SCM use case, the VNO requests a virtual network connecting the communication end-points and the resources for the translation service. He considers the influence of the translation service onto the end-to-end delay and jitter defined by the ASP.

The VNP investigates which PIPs are able to provide virtual resources suiting the requested parameters. Then the VNP starts negotiating with the PIPs, which resources are available at which price and how they can be used. If the VNP finds one or more possible solutions, it might want to reserve this resource for the time needed to offer the networks to the VNO and the consequent decision process. In case the VNO is not satisfied, the VNP has to search for new resources and the negotiation process starts again. In case the VNO accepts an offered network, the VNP needs to book the resources for the network and to release reservations which are not used for the virtual network. In the next step, the VNP aggregates the resources provided by the different PIPs and exposes them to the VNO using

a unified interface. This means that the VNO can set up, control, and operate all virtual resources using this interface. Particularly, this interface unifies all virtual network nodes, e.g. virtual routers, so that the VNO can install any network stack on the devices. If there are many different interfaces for virtual resources running on different hardware, this unification is practically very hard, because of the great diversity. Hence, the standardization of interfaces for accessing virtual resources has to be considered, which might take some years to take place.

2.2.4 Monitoring within Virtual Networks

The business model of many use cases for virtual networks is to provide the service more cost-efficient and/or with a better quality. In order to achieve this promise, the traffic management of virtual networks needs to be highly automated. Most tasks which create, adjust, and tear-down a virtual network need to be executed without human interaction to shorten the reaction time.

An integrated solution, as described in Section 2.1.3, incorporates monitoring, analysis, and adjustments in control loops for each functional role. Monitoring data coming from the network is considered by a decision component. If adjustments are needed to provide proper operation, the control plane directly executes these changes. The functionality of the decision component depends heavily on the service the virtual network has to provide, as well as the implementation of the control plane depends on the used hardware platform. Therefore, we focus on the monitoring architecture and present details here.

The five different roles have different viewpoints and therefore focus on different aspects of the network. The infrastructure used, however, is very similar. Each role sets up measurement points at each relevant network device, whether it is a physical entity or a virtual device. The measurement data produced is gathered by *Monitoring Agents* (MA) positioned near the measurement points. These agents filter and pre-process the measurement data before the data is made available over a *middleware* architecture and stored in a *Traffic Monitoring and Management DataBase* (TMMDB). Both the middleware and the monitoring data base

might be implemented in a centralized or distributed way, according to the needs of the corresponding system. The middleware's task is to correlate measurement data and to analyze trends to forecast the future load. To provide data to the *Decision Component* (DC), virtual network customers and virtual network providers specialist *Data Exchange Agents* (DEA) are used. These agents implement multi-client capabilities. This means that depending on the contracting party, which is attached to the agent, different access levels and details are provided. It is reasonable to differentiate the exchanged measurement data on the contracts by the functional roles and their trust relationship. Furthermore, the agent is able to translate the internal data representation, which is used within the middleware, to an external exchange format. Finally, the middleware supports the attachment of *graphical user interface*(GUI) components to visualize the status of the network to the administrative team of the company taking the functional role. Figure 2.8 depicts the connection between the basic components.

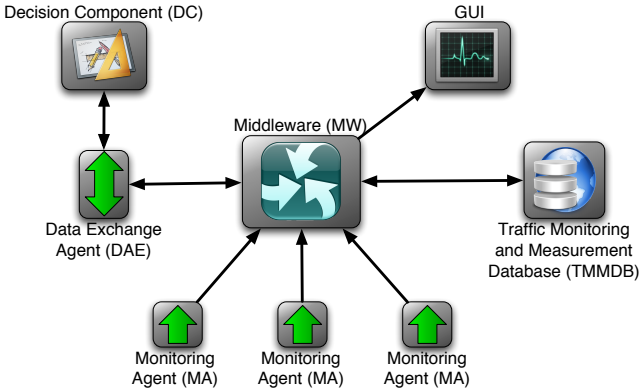


Figure 2.8: Components of a virtual network monitoring solution.

The physical infrastructure provider (PIP) monitors the health of his own hardware components, the virtualization, and the isolation of the virtual networks.

First, it needs to make sure that all transport media work like intended. An example for this is the loss of signal strength, which might indicate a problem with the transmission media, a signal amplifier or another technical defect. Also cable cuts due to physical link damage is something the PIP has to consider and repair. The second monitoring task is everything related to the virtualization. The status of the hypervisor providing the virtualization, its version number, and failure messages need to be tracked and gathered in the central database. Furthermore, the PIP has to take care of the virtual resources contracted by the VNP and the isolation. The SLAs related to all the provided resources define the way these resources should work and the PIP is interested in measuring if everything works as expected or if it has to adjust settings. Especially, the quality of virtual network isolation depends on the techniques and settings used for the hypervisor. Different virtual networks which are multiplexed on the same physical transmission media cannot be isolated completely. However, the PIP needs to monitor that his hypervisor setup can achieve the quality the PIP contracted in the SLAs with the VNP. It has to be noted, that the PIP is a role that does not run a virtual network service. To access its own monitoring infrastructure via out-of-band management, we assume the company taking the role of the PIP to be able to operate a small virtual network. Hence, the PIP company will in most cases also implement the functionality of the VNO for their own monitoring network. Another interesting option for the PIP is to monitor the overall utilization of its resources. If we consider a PIP that has contracted most of its resources, but only a small percentage is used, the PIP might want to overbook its virtual resources to optimize its profit. In this case, the PIP needs precise information of its resource utilization on a small time scale, i.e. in the order of 100 ms to some seconds, in order to minimize the risk of overloading a physical resource and breaking SLAs with the VNPs contracting its resources. However, nearly all of the measurements of the PIP can be done passively and there is no need to access the components and the traffic of virtual networks hosted.

The monitoring of the VNP is very special. The functional role of the VNP assembles network from the resources offered by the PIPs and provides them to

the VNOs. However, the VNP does neither have access to the virtual resources provided for a network nor can it look into the virtual networks. The former is only possible to the PIP who provides the resources. The latter is based on the fact that the VNO can orchestrate his own protocol stack for the network without disclosing its algorithms to the VNP. The VNP is, therefore, not monitoring any technical parts of any virtual network. But the VNP supervises its own resource pool, i.e. preallocated virtual resources not contracted by a VNO, to be able to provision networks quickly and monitors the markets where the PIPs offer the virtual resources in order to gather the resources cost-efficiently. However, we can imagine the company implementing the VNP role to engage a department taking the role of a VNO, which request resources from PIPs and performs network measurements on a random basis to verify that the PIP keeps its QoS-SLAs

The VNOs monitoring task is the most challenging one. On the one hand, the VNO has to verify that all PIPs provide their resources as contracted. This means that the VNO does not only have to consider the overall performance, but also has to set up a precise monitoring for each part belonging to a single PIP. The VNO has to verify the performance of the resources contracted by each PIP to prove a SLA violation. The VNO can only access the virtual network and its components from within the hypervisor. It, therefore, needs special hypervisor functions to access a precise real-time clock. Otherwise, the VNO cannot perform any measurements besides rough mean value estimation, which is simply not enough for high quality network services like the ones presented in the use case section, cf. Section 2.1.

On the other hand, the VNO has to monitor the overall end-to-end performance of its network. Therefore, it has to measure the performance of the used protocols and algorithms as well as the resource usage within the network. Active measurements as well as measurements based on piggy-backing information to the transmitted data can be performed by a VNO as it controls the complete protocol stack. The communication between the VNO and the adjacent roles is divided into two interfaces. All monitoring information is transmitted to the adjacent roles via data-exchange agents. Other information, e.g. requests of new

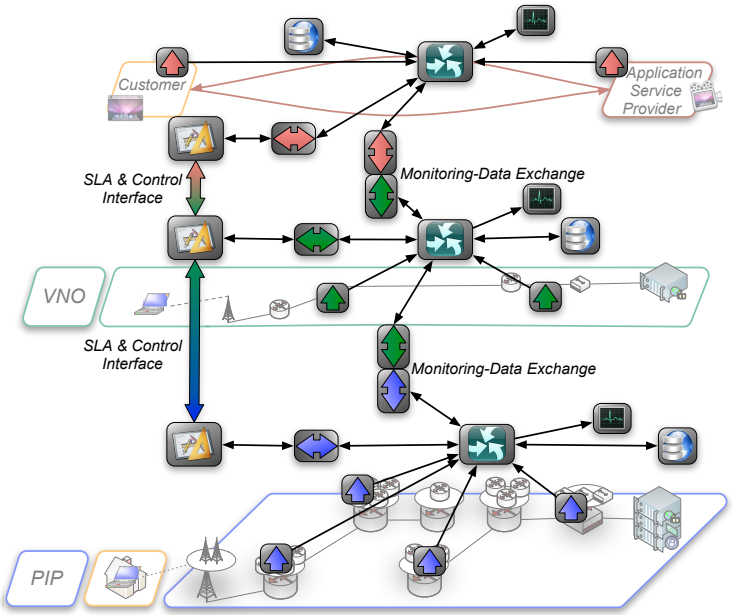


Figure 2.9: Monitoring Framework of a virtual network with one PIP, one VNO and a single ASP customer connection.

resources, control information for network nodes, and SLA claims, are delivered over the interface connecting the decision components for the corresponding customer-provider-relationship.

The ASP is interested in the Quality of Experience the end-customer perceives. Hence, the ASP monitors application layer data. For instance, the ASP monitors lost frames for a video transmission service. This data is available at the end points. The ASP only tracks the data transmission, in case the VNO is only offering plain data transmission services. Furthermore, the ASP implements features in his software and hardware that makes it possible to receive user feedback on the current quality of experience. We assume that in most cases the VNO takes

care of the transmission of the monitoring data of the ASP, i.e., the ASPs uses in-band signalling within the virtual network offering the application service to transmit its monitoring data. However, in some cases, it might make sense for the ASP to run a second virtual network from another VNO to transfer the monitoring data and create an out-of-band management network. A summary of the monitoring architecture and the interconnections between the roles is provided in Figure 2.9.

2.3 Related Work

In this section, we discuss related work. We first focus on publications that presents use cases for future Internet scenarios as well as contributions characterizing virtual network architectures. Later on we consider work that has been performed in the area of network monitoring. For monitoring virtual networks the solutions are similar to non virtualized networks. However, creating exact time stamps in virtualized routers and standardized communication between the virtualization layers is needed. Finally, we include references to papers describing how virtual routers can be implemented and discussing performance measurements of future networking concepts.

A very well written survey on network virtualization is presented by Chowdhury et al. [30, 31]. Besides of historical aspects, the authors describe a general network virtualization environment, give an overview on network virtualization projects and provide an overview of possible research directions.

Marikar et al. [32] propose a use case for self organization in future networks. In their view, networks need to become partly self-conscious to be able to handle the increasing complexity, as underlying transmission technologies become more and more heterogeneous and the services running on top of it become more demanding. They introduce a hierarchical layer model in which every layer is responsible for monitoring, decision making, and execution of these decisions. Higher layers control lower layers. We have pursued a similar approach based on virtualization in our architecture. Feamster et al. [33] introduce the concept of the

Infrastructure Provider, who slices its physical hardware into virtual resources and rents these to service providers. We adapted this idea of the infrastructure provider but introduced more roles into this architecture to enable a more flexible provisioning of virtual networks and services.

PeerMart is a distributed marketplace for network bandwidth introduced by Stiller et al. [34]. We further abstract this concept to the Virtual Network Provider, who can broker entire virtual networks with a wide range of parameters. Kroeker [35] suggests that virtualization will play an important role on mobile devices in the future. Hence, our architecture explicitly includes end devices as physical resources, since we want to guarantee QoE and QoS up to the end user. Anthias et al. [36] propose an Application Oriented Network (AON), which runs on top of a very basic transport network that essentially serves as a bit pipe and enables application aware services. We can accommodate this concept in our architecture as a virtual network. Niebert et al. [37] identify four steps necessary to instantiate virtual networks. We addressed these steps in our architecture in the following way: 'Resource discovery' is handled by our VNP as it is constantly informed by the PIPs about available resources. 'Resource description' works via a standardized interface the PIPs expose and is communicated via a resource description language. The 'resource provisioning' step is the central task of the VNP. As required by Niebert, he 'identifies, allocates, configures and aggregates resources into virtual networks' [37, p. 516]. Shiomoto et al. [38] developed a heuristic algorithm for computing virtual network topologies based on underlying optical networks. This can be used by the PIPs to accommodate resource requests on optical links. Banniza (edt.) [39] describes a visionary future network and derives many use cases and discusses two of them in more detail. These two use cases are namely 'Community-Oriented Applications' and 'Internet of Things or One Thousand Network Devices'. The document describes how these use cases can be implemented using the virtual network architecture of the project and details on how regulatory aspects affect the implementation of the use cases. Bauke (edt.) [40] published their concept for network virtualization. They introduce their hierarchical role model consisting of 'Infrastructure Provider',

'Virtual Network Provider', and 'Virtual Network Operator'. Our functional role model is inspired by their work. However, our approach is derived from the use cases and is service driven. Therefore, we introduce two roles on top: the 'Application Service Provider' and 'the End-Customer'. In addition, we included the customer edge, service edge, and data centers as enabler for end-to-end virtualization and consider the implementation of the transport network. Interfaces and signaling of virtual networks are described in [41].

Subramanian [42] wrote a book covering nearly all aspects of 'classical' network management, e.g. standards, models, languages, SNMP, RMON, Broadband Network Management, and many other tools and applications. A service-oriented approach is discussed by Hanemann et al. [43], which allows for monitoring of multi-domain monitoring. Many of the ideas presented in this work can be used to create measurements, which horizontally extend the measurements on the PIP level. However, they are not able to be used for vertical exchange of monitoring data, e.g. between the VNO and the ASP. Later publications on network monitoring often focus on specialized problems. For instance, Iliofotou et al. [44] describe a way to identify *unwanted applications* by studying traffic dispersion graphs. Although network virtualization with strict isolation would restrict the spreading of unwanted application to single networks, a PIP might be interested to identify a scenario in which some network is not behaving as intended. The proposed solution can be used by the PIP as no information from within the virtual network is needed to create the graphs and identify anomalies. A scaling monitoring infrastructure is described by Repantis et al. [45], which might be very useful in use cases like the *beta slice*, cf. Section 2.1.6.

We differentiate between two fundamental different approaches towards building virtual routers. The first is based on the idea of separating the data plane and the control plane. Virtualization is achieved by partitioning the rule set each control plane entity is allowed to write to the data plane. Secondly, virtualization can be achieved by demultiplexing traffic on a single host to several instances of virtual guest systems.

The most popular solution for separating the data plane and the control plane is

Openflow [46]. This technology is used by the 'Global Environment for Network Innovations (GENI)' [47] project to integrate their testbed into their production network. While the Stanford OpenFlow Group [48] describes the data plane and specifies a protocol to connect the data plane to the control plane, Gude et al. [49] discuss an implementation of the control plane. First performance measurements provided by Naous et al. [50] show that the forwarding performance is quite high, as long as the data plane has been preconfigured. Sherwood et al. [51] discuss how virtualization can be achieved in this context. The authors describe a transparent proxy called Flowvisor, which verifies all commands exchanged between the control plane and the data plane. This allows different parallel networks on the same hardware substrate.

The second virtualization approach, which uses hardware virtualization, has the advantage that commodity hardware can be used and the virtual routers are isolated by a hypervisor. Hence, misconfiguration of a single virtual router does only affect the network the virtual router belongs to. An overview of different hardware virtualization techniques and mechanisms as well as their history and motivation is provided by van Doorn [52]. Several publications consider *resource access fairness* when multiple guest systems run on a single host. A comparison of hypervisors with the target of server virtualization is provided by Camargos [53]. Ongaro et al. [54] studies virtualization using the Xen hypervisor. The authors show that while the processor resources are fairly shared among guests, the scheduling of I/O resources is treated as a secondary concern and decreases with each additional guest system. Furthermore, they discuss the influence of eleven different scheduling schemes for Xen is analyzed. Another performance analysis of Xen for network virtualization is presented by Anhalt and Primet [55]. Their finding suggest, that Xen is well-suited for network virtualization considering TCP throughput and CPU utilization of virtual guest systems. However, they did only consider up to two interfaces in a single server. Root causes for the delay in virtual routers implemented with hardware virtualization is presented by Whiteaker et al. [56]. The authors show that virtualization causes packet forwarding delays that cannot be measured within a virtual machine. They suggest

to provide virtual guests with an option to access the time stamps of the hypervisor in real-time. We support the idea in principle but decided to integrate the real-time access to a hardware clock on the host machine. The hypervisor might be able to provide a high quality time stamp for the time the packet enters the host system, but the demultiplexing process and forwarding delay to the virtual guest cannot be characterized this way. Tripathi et al. [57, 58] describe how to enhance the performance of network virtualization with Xen. The authors discuss how to enhance the multiplexing and demultipexing of packets in the Xen Dom0. Unfortunately, the implementation did not work on our hardware platform and could therefore not be included in the measurements we provide in the next section.

Anwer et al. [59] use a NetFPGA-based prototype to achieve network I/O fairness in virtual machines by applying flexible rate limiting mechanisms directly to virtual network interfaces. Fairness issues in software virtual routers using *Click* are considered by Egi et al. [60]. The impact of several guests with and without additional memory intensive tasks on packet forwarding is measured. The authors provide more details and some performance enhancements for the Click router in [61].

As mentioned before, the performance and fairness heavily depend on the used hypervisor. Hence, several comparisons of mostly two hypervisors have been conducted. We focus on the references considering network I/O performance which altogether consider packet forwarding performance with different packet lengths from 64 to 1500 bytes. VMware (edt.) [62] published comparison of VMware ESX Server 3.0.1 and open-source Xen 3.0.3 with up to two guests using the Netperf packet generator. They show that Xens heavily lags behind VMware ESX in terms of network packet handling performance. This study was repeated with VMware ESX Server 3.0.1 and open-source Xen Enterprise 3.2.3 by Xen Inc. (edt.) [63], which showed that a specialized Xen Enterprise lags only by a few percent. Bredel et al. [64] consider multiplexing of two data flows with different scheduling mechanisms on a Cisco router, a virtual machine host, in two VM guests using Xen, and a NetFPGA packet generator. They show the dependency of the software routers on the packet length. Measurements of OpenVZ

and User Mode Linux (UML) were conducted by Bhanage et al. [65] on wireless nodes of the Orbit testbed. UDP throughput and FTP performance is measured, but it is also stated that the hardware is not the best choice for virtualization. Bhatia et al. [66] present the network virtualization platform *Trellis* and packet-forwarding rates relative to other virtualization technologies and native kernel forwarding performance are compared. The considered virtualization technologies comprise Xen, Click, OpenVZ, NetNS, and the Trellis platform. The Linux kernel module *pktgen* [67] is used for packet generation. In their results, Xen has the lowest performance followed by OpenVZ.

In contrast to the related work our virtual network architecture is derived from use cases and is service driven. We focus on complete virtualization on an end-to-end basis, in order to guarantee a high QoE. Our role model, the interface descriptions, and the monitoring concept is built to support automated provisioning and autonomous management from the service to the customer, in which we differ from related work.

Our performance analysis of today's hardware virtualization solution considers the five most used implementations on the same hardware platform. Therefore, our work provides a broad performance overview, which cannot be found in related work. Additionally, we focus on the forwarding performance with high traffic rates using up to six interfaces. Our study reveals that CPU and memory intensive tasks, mainly studied in related work, affect current virtualization solutions only slightly. Additionally, our investigation reveals yet unidentified isolation problems, which are likely to affect the provisioning of virtual machines in professional environments and need to be monitored in scientific testbeds in order to guarantee credible results.

2.4 Performance of Current Isolation Mechanisms

From the previous sections, we know that there are many requirements for a virtual router, e.g. providing interfaces to different hardware abstraction levels, being able to be migrated throughout the network, etc. The most interesting de-

mand from a performance analysis perspective is isolation of parallel networks and the achievable throughput. As discussed in the previous section there are different approaches how to build virtual routers. But technologies like OpenFlow do not yet support other protocol stacks than IP with acceptable performance, cf. [6]. Hence, in this section we focus on the option of using software routers on commodity hardware, as it is a promising option that is already available.

In order to run several software router on the same server we apply hardware virtualization. The term *hardware virtualization* means abstracting functionality from physical components of a server. This principle is used for sharing computer hardware by multiple logical instances – the *virtual guest systems* – and originated already in the late 1960s. Recently, the importance of virtualization has drastically increased due to its availability on commodity hardware, which allows multiple *virtual guests* to share the resources of a physical machine. The resource access is scheduled and controlled by the *Virtual Machine Monitor* (VMM), also called *hypervisor*. Different virtualization platforms have been implemented and are widely used in professional environments, e.g. data centers and research facilities like G-Lab [68], to increase efficiency and reliability of the offered resources. The resources that may be used by a virtual guest system as well as lower performance bounds regarding these resources are at least in professional environments determined in SLAs between providers and customers. The implementation of the VMM influences how well the resource allocation complies with these SLAs and to which extent different guests interfere with each other. Hence, a provider has to know the performance limitations, the impact factors, and the key performance indicators of the used VMM to ensure isolation and SLAs.

There are two basic scenarios in which hardware virtualization is used. In the *dedicated* scenario, only a single virtual guest runs on a physical host. By means of hardware virtualization, the guest system is made independent of the underlying physical hardware, e.g., to support migration in case of maintenance or hardware failures. Hardware virtualization is also used to consolidate resources. Therefore, the physical hosts are shared among multiple virtual guest systems in a *shared* scenario. While the performance of a virtualized system can be directly

compared to the bare host system in the dedicated scenario, the performance of a virtual guest may also be affected by interference from other virtual guests on the same physical host in a shared scenario.

The performance cost of virtualization and *isolation*, i.e., the prevention of virtual guest interference, has been studied for CPU, memory, and hard disk usage. Nowadays, most commodity servers have built-in hardware support for virtualization, enabling fast context switching in the CPU and memory resource restrictions. However, the *input/output* (I/O) system, especially network throughput, is still a crucial factor if we consider using the VMM for building virtual routers.

Therefore, we focus on the network throughput of virtualized systems as the performance metric. We apply this metric on a non-virtualized Linux and on a virtualized version of this system using several popular hardware virtualization platforms, i.e., OpenVZ, KVM, Xen v4, VirtualBox, VMware ESXi. Furthermore, we install a second virtual guest and analyze the effects when the second virtual guest is idle, running some CPU and memory intensive tasks, or is transmitting packets.

2.4.1 Virtualization Concepts

The recent popularity of virtualization on commodity hardware created a plethora of virtualization platforms with different complexity and environment-dependent applicability. An overview of virtualization platforms is given in [69].

Several ways to implement hardware virtualization have evolved, which differ in the required adaptations of the guest system. For instance, a VMM may run on bare hardware (called *Type 1*, *native*, or *bare metal*) or on top of an underlying/existing *operating system* (OS) (called *Type 2* or *hosted*). Also *hybrids* between Type 1 and 2 are possible. Finally, OS-level virtualization is a special form, where the guests are run in a container of the native OS.

Native VMMs and hosted VMMs might provide *full virtualization*, i.e., the guest system does not have to be modified. This means that the virtualized guest operating system is working as if it would run directly on the hardware. When-

ever the guest wants to access a hardware resource, which is only a virtualized resource, the VMM has to interrupt the guest and 'emulate' this hardware access. This effects the performance of the guest system. To alleviate these problems, the guest system can be modified, e.g. with special drivers, to redirect resource accesses to function calls of the VMM. This method is called *para-virtualization* and may improve the performance of the guest. OS level virtualization solves this resource access problem by using the same modified kernel in the guest containers as in the hosting OS. This technique has the advantage that the overhead for virtualization is quite low compared to the other solutions. However, the OS and the kernel of the guest are predetermined.

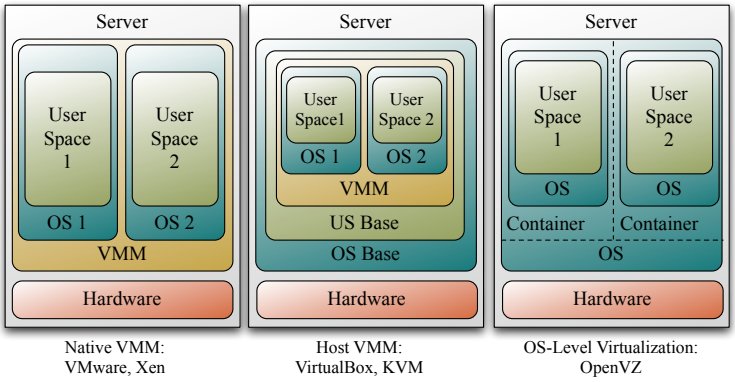


Figure 2.10: Virtualization concepts of the considered VMMs.

Figure 2.10 depicts how the considered virtualization platforms encapsulate virtual guests and indirect access to resources for applications, which run in user space. In the following, we consider five popular hardware virtualization platforms which can be categorized as

- *native virtualization*, represented by VMware ESXi [70] and Xen v4 [71],
- *host virtualization*, represented by VirtualBox [72] and KVM [73], and
- *operating system-level virtualization*, represented by OpenVZ [74].

Fair access to the physical hardware is important and known to work for resources that can be split into quotas, e.g. memory, disk space, or processor cycles per second. However, access to any of these separately restricted resources involves the (I/O) of data which typically shares a single bottleneck, e.g. an internal bus. Especially, network I/O fairness is a crucial issue which involves another shared physical resource, the *network interface card* (NIC). Hence, we consider the packet forwarding throughput of the aforementioned virtualization platforms as a performance metric.

In the next section we characterize the hardware platform used for the performance test and explain the exact software set up.

2.4.2 Measurement Setup

For our measurements, we use seven Sun Fire X4150 x64 servers from the G-Lab testbed [68]. Each server is equipped with 2×Intel Xeon L5420 Quad Core CPU (2×6 MB L2 Cache, 2.5 GHz, 1333 MT/s FSB) and 8×2 GB RAM. The *unit under test* (UUT) is equipped with eight 1 Gigabit ports on two NICs, while all other servers are equipped with only four 1 Gigabit ports. The network controller chip sets identified themselves as Intel 80003ES2LAN and Intel 82571EB. The UUT is connected on each NIC to three other servers each with a single direct connection using Cat.5e TP cable or better. The setup and interconnection of the servers is illustrated in Figure 2.11.

Each machine is running Debian 5.0.4 (“lenny”) with Linux kernel 2.6.26 compiled for the amd64 (x86_64) architecture. Only for the test of Xen we need to adjust the base system with Debian 5.0.5 (“lenny”) with Linux kernel 2.6.31 for amd64. Since VMware ESXi installs directly on the host machine, the choice

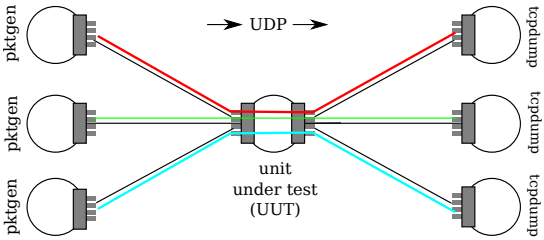


Figure 2.11: The hardware setup: UUT connected to measurement environment.

of a custom operating system is not applicable in this case. We investigate the following versions of the different VMMs with a Debian "lenny" as a guest system and the given kernel installed:

- KVM 0.9.1 (kvm-72) [73]: Guest running Linux 2.6.26 for amd64
- OpenVZ 3.0.22 [74]: Guest running same OS kernel as host
- VirtualBox 3.2.8 [72]: Guest running Linux 2.6.26 for amd64
- VMware ESXi 4.1.0 [70]: Guest running Linux 2.6.26 for amd64
- Xen development version of July 16th, 2010 [71]: Guest running Linux 2.6.26 for i386 (i686; x86) with *physical addressing extensions* (PAE)

For the KVM and VirtualBox machines, virtual NICs are using the "virtio" para-virtualized network driver. For OpenVZ and Xen, para-virtualization is achieved implicitly through the OS-level or para-virtualization approach taken by OpenVZ and Xen, respectively. For VMware ESXi, both the "E1000" full-virtualized network card emulation and the "VMXNET 3" para-virtualized network driver (using the VMware Tools on the guest) are considered.

2.4.3 Measurement Metrics and Methodology

In all scenarios, we investigate the packet forwarding throughput performance of the different virtualization solutions on the UUT. In each experiment, network traffic is generated using the packet generator integrated into the Linux kernel (`pktgen` [67]). It sends out UDP packets which are forwarded by the UUT to another destination server where the traffic is recorded by `tcpdump` [75] and discarded afterwards. After stopping packet generation, the packet generator reports how many packets it was able to send. We calculate the mean throughput from this value and the measurement of the time in which the traffic has been sent. We call this fraction the offered throughput: $t_o = \frac{packets\ sent}{send\ duration}$. At the same time, we calculate the measured throughput from the packets forwarded by the router and the corresponding duration in which the packets are received: $t_m = \frac{packets\ received}{receive\ duration}$. For our performance evaluation, we set up two different scenarios, a dedicated host scenario and a shared host scenario, which we describe in the next sections.

2.4.4 Dedicated Host Scenario – Impact of Virtualization

In this scenario, we measure the packet forwarding performance of the *raw* Linux system, i.e. without any virtualization, and compare it with a setup where the same system is installed inside the tested hypervisor. We consider up to three traffic sources. It has to be noted that each traffic stream through the UUT does not interfere with any other stream outside the server, i.e., each traffic stream enters the server over a different network interface and also leaves the system via another distinct network interface. Each network interface is directly bridged to a corresponding virtual interface of the guest system.We consider packets with an Ethernet frame size of 64, 500, 1000, and 1500 byte for each data stream. The complete logical size “on the wire”, i.e., bits used on layer 1, is 20 byte larger, due to the layer 1 preamble of 7 bytes, the start of frame delimiter, and the inter frame gap of 12 octets, as defined in the 802.3 standard [76].

For each packet size, a target bandwidth of 10, 100, 200, 400, 600, 800, 900, and 1000 Mbit/s we conduct nine runs for statistical evidence. The selected values allow us to see the overall response of the system as well as the forwarding performance for different offered throughputs. This enables us to analyse in which area the UUT reveals the highest forwarding rates and how overload effects the system. Nine runs per setting are chosen, since initial test runs showed us that this number yields small confidence intervals in most cases and reduces the measurement effort down to about two months of pure measurement time. Since `pktgen` does not allow for setting a target bandwidth, but instead manages different loads of traffic by waiting a certain amount of time between sending two consecutive packets, this delay has to be calculated. For target bandwidth b [Mbit/s] and target packet size s [byte], the resulting delay [ns] is $8000 \times (s + 24)/b$. This formula can only be applied for large packet sizes. However, the kernel is not able to generate small packets fast enough, i.e. the inter-frame time measured at the sender is larger than the value calculated by the formula. Hence, we additionally adapted the inter packet delay to 1 and 0 ns for each nine tests, in order to maximize the generated traffic and send as many packets as possible.

2.4.5 Shared Host Scenario – Quality of Isolation

The consolidation of multiple virtual systems on a single physical host may have an impact on the network performance of the virtualized guests. Thus, we add a second virtual guest to the VMM, which runs another Debian "lenny" and repeat all measurements. Initially, the second guest does not run any processes generating additional load (*no load*). In order to investigate if CPU intensive processes in another guest systems have an influence on the performance of our UUT, we generate variable amounts of CPU and virtual memory load with the `stress` tool (version 0.18.9) [77] and considere two more cases. In the first case, called *light load*, `stress` is running with parameter `-cpu 1`, corresponding to a single CPU worker thread. In the second case, called *heavy load*, `stress` is running with parameters `-cpu 8 -vm 4`, corresponding to 8 CPU worker threads, and

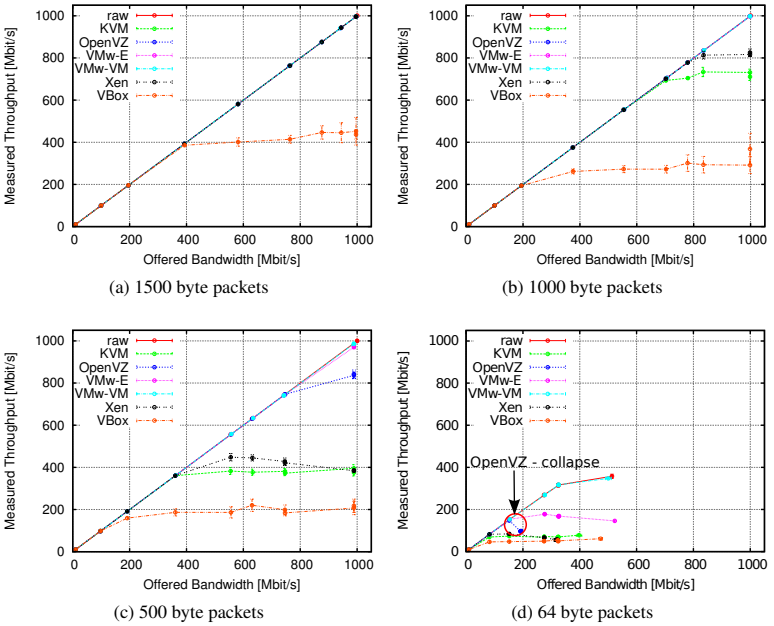
4 virtual memory threads, which allocate, write to, and release memory. In case of full virtualization (KVM, VirtualBox, VMware, and Xen), each virtual guest is assigned a single CPU out of the 8 physical CPU cores installed on the server.

Besides CPU and memory load, we consider the influence of a second guest systems network load, even if the guests do not share physical network interfaces. We focus in this scenario on Ethernet frame sizes of 64 byte, as the previous scenarios reveal this packet size to be most critical. Hence, we reconfigure the UUT to have only two virtual interfaces, which are bridged to different physical interfaces. We adjust the second guest system to have also two virtual interfaces, which are bridged to two distinct physical interfaces. We run the same tests as before, but this time we also send traffic through the second guest. We adjust the traffic in such a way that it is 50% and 95% of the maximum forwarding rate, which we measure in the test with the second guest being idle. For this test, we also measure the throughput of the second guest to investigate if the second guest system can still forward the offered data rate.

Both tools, `pktgen` and `stress` have been analyzed previously. We compared the packet reported by pktgen and packets received by the sender in various load situations. For the stress tool, we verified its correctness against the linux proc file system, i.e. against the performance measures provided by the linux kernel. This analysis confirmed that both tools perform their tasks in a reliable way.

2.4.6 Performance Costs of Virtualization

Depending on the used VMM, the network throughput performance of the virtual guests differs significantly. In Figure 2.12, we plot the bandwidth the traffic generator produced ("offered throughput") against the throughput the central test system was able to forward ("measured throughput") averaged over all nine repetitions of a test. The error bars present the 95% confidence intervals (CI) for the mean values. Since we only have only nine measurements, we calculate the confidence interval based on the student-t distribution: $CI_{0.95} = \overline{\mu} \pm t_{0.025,n-1} \frac{\sigma}{\sqrt{n}}$, where $\overline{\mu}$ is the mean value of the measured results, σ is the standard deviation of



(a) 1500 byte packets

(b) 1000 byte packets

(c) 500 byte packets

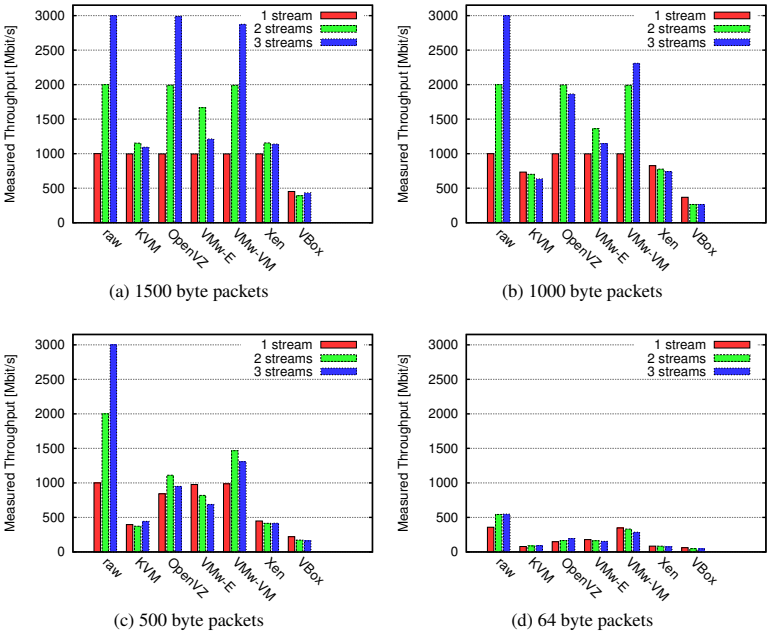
(d) 64 byte packets

Figure 2.12: Offered throughput vs throughput of VMMs for different packet sizes.

the measured results, and n is the number of samples, cf. [78, p. 157]. As the traffic generators are not able to produce exactly the same offered throughput in all cases, we also plot confidence intervals for this variable. However in most cases both error-bars are hardly visible, since the measured results do not vary much. In the following, we use abbreviations to shorten the labels in the figures: raw for the system without virtualization, VMw-E for VMware with the full-virtualized network card E1000, VMw-VM for VMware with the para-virtualized network VMXNET3, and VBox for VirtualBox. The graphs for the 1500 byte packets reveal no difference between the system without virtualization and most of the

VMMs. Only the VirtualBox system is not able to forward all packets. For smaller packet sizes, i.e., 1000 bytes and 500 bytes, the difference between the different VMMs is clearly visible. The fact that the offered throughput for all packet sizes above 500 bytes increases up to the maximum of the link, i.e., 1 Gbit/s, shows that the VMMs are able to fetch the packets from the network interface. However, in all cases in which the measured bandwidth is lower than the offered throughput, a significant number of packets are lost in the *unit under test* (UUT). Looking at Figure 2.12d, two additional effects can be noticed for the smallest used packet sizes, i.e., 64 bytes. Not all graphs extend to the same length on the x-axis. This means that the traffic generator was not able to offload the same amount of traffic to the UUT in all cases. This behavior is caused by *layer-2 flow control*. Whenever the frame buffer of the network card is fully occupied, the network card sends a notification to the sending interfaces to throttle the packet rate. The maximum throughput of the UUT is never affected by this behavior. Hence, we keep it activated, as it provides another option to the UUT to react on traffic rates it cannot handle. Please note that in all cases in which the layer-2 flow control is actively reducing the number of sent packets, the test system already drops packets. This behavior, therefore, never influences the maximum throughput of the system. The second observable effect is the behavior of OpenVZ when forwarding 64 byte packets. The OpenVZ system achieves a maximum throughput at a given input rate and collapses afterwards. We also notice this effect in other scenarios with 64 byte and 500 byte packet sizes. But in these cases, the throughput rate does not only collapse but varies heavily.

Next, we consider how the throughput of the UUT scales if we increase the number of traffic generators. Figure 2.13 depicts the maximum achieved throughput with one, two, and three traffic sources. Note that as we see, e.g. in Figure 2.12c for Xen, the maximum achieved throughput is not necessarily recorded at the highest offered throughput. In most cases we see that the UUT achieves the highest measured throughput for some offered throughput values and decreases for higher offered throughput rates. This behavior is often perceived with overloaded systems, as the additional load introduces additional overhead that reduces



(a) 1500 byte packets

(b) 1000 byte packets

(c) 500 byte packets

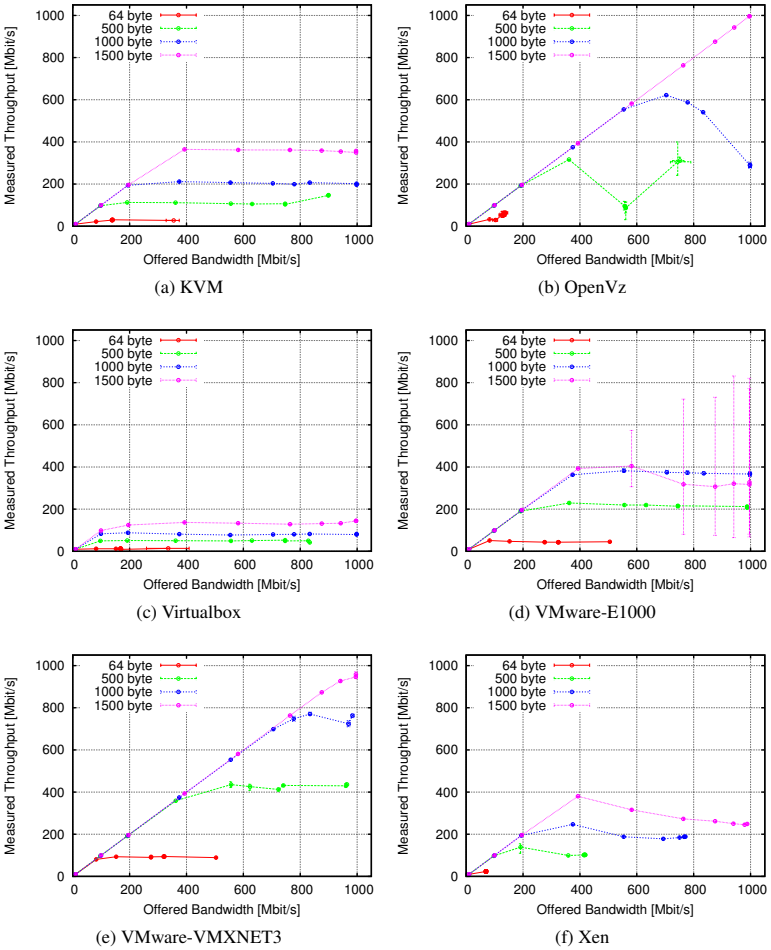
(d) 64 byte packets

Figure 2.13: Maximum accumulated throughput of VMMs.

the overall performance. The bar plot for the UUT without any VMM reveals that in all cases, except the one sending the 64 byte packets, the performance of the raw system scales linearly, i.e., we measure a throughput of 1 Gbit/s, 2 Gbit/s, and 3 Gbit/s. In the case of 64 byte packets, Figure 2.13d reveals a limitation of the raw system, as the accumulated throughput for two and three traffic generators is the same. In this case, the raw system uses layer-2 flow control to throttle down the traffic generators. But the raw system is able to forward all received packets, in contrast to all VMMs which accept up to 600 Mbit/s input rates but drop packets down to the depicted maximum accumulated throughput. VirtualBox and

KVM reveal a internal forwarding capacity limit, which is already reached using 1500 byte packets on a single interface. In Figure 2.13, we clearly identify that the forwarded bandwidth is not increasing, even if we connect more interfaces.

More details of the throughput performance are presented in Figure 2.14. In its subfigures, we plot the mean measured throughput as well as the minimum and maximum measured throughput of all tests for the different VMMs. This means that whenever there is a noticeable confidence level plotted, the three parallel streams have a different throughput measured at the receiving devices. The measured throughput for KVM, cf. Figure 2.14a and Virtualbox, cf. Figure 2.14c, shows that both systems have a stable maximum throughput rate. This means that the measured throughput reaches a certain threshold but does not increase any further. But the measured throughput does also not decrease, if more traffic is offered to the UUT. The VMMs just drop packets which exceed the maximum achievable throughput. However, the maximum, mean, and minimum bandwidth forwarded through the VMMs is not diverging, i.e., the measured throughput of all parallel streams is equal. In case of OpenVZ, cf. Figure 2.14b, we observe a different behavior. For packet sizes of 1000 byte the maximum throughput is reached for an offered throughput of about 700 Mbit/s. If more packets of 1000 byte size are offered the received throughput dramatically decreases down to about 300 Mbit/s. For 500 byte packets the behavior is even worse. In this case, the maximum received throughput is achieved at an offered rate of about 380 Mbit/s. For higher offered packet rates, the received throughput of the system is changing between minimum rates about 50 Mbit/s and rates up to 400 Mbit/s for individual flows. Furthermore, the throughput of the parallel flows is no longer the same, but varies considerably. Only if we use VMware without the para-virtualized driver, cf. Figure 2.14d, we measure a high variation in the throughput of the parallel flows while offering 1500 byte packet streams. For offered throughput higher than 500 Mbit/s the difference between the parallel streams is increasing. In the worst case we measured about 100 Mbit/s for the lowest forwarding bandwidth of the three streams and about 800 Mbit/s for the highest forwarding bandwidth. In Figure 2.14e the measured bandwidth for VMware with the para-virtualized



(a) KVM (b) OpenVz

(c) Virtualbox (d) VMware-E1000

(e) VMware-VMXNET3 (f) Xen

Figure 2.14: Differences in throughput of three streams for different VMMs.

network driver VMXNet3 is depicted. This configuration provides high throughput rates and does not have the problem of different forwarding rates for different parallels streams. However, even in this configuration the VMM fetches more packets from the network card than it can handle, which results in high packet loss rates. The results for the Xen VMM are shown in Figure 2.14f. In the considered scenarios Xen achieves a forwarding bandwidth that does not vary much between the parallel streams. However, the maximum forwarding bandwidth is not achieved for the maximum offered traffic. Instead, the maximum forwarding bandwidth is reached for a lower offered traffic rate and decreases for higher rates.

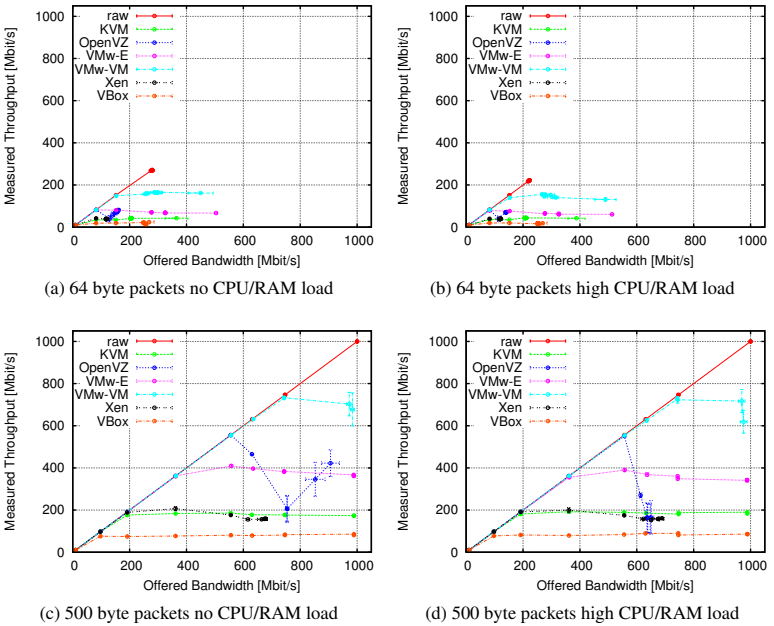
The results presented in Figure 2.13 and Figure 2.14 show that most VMMs have a throughput limitation, which depends on the packet size of the offered stream. All VMMs except OpenVZ reveal this limitation even for 1500 byte packet sizes. Only OpenVZ is able to forward packets of this size with the same performance as the raw system. However, if we look at the results for the other packet sizes, we clearly see the impact of virtualization even for OpenVZ. The varying throughput rates for smaller packet sizes make it hard to foresee the forwarding behavior of OpenVZ. In the same range, i.e., 500 byte packets and below, VMware with the para-virtualized driver outperforms all other solutions.

In conclusion, we see that the performance costs of virtualization increases, the more small packets are sent and the higher the offered throughput on the UUT is. Even OpenVZ, which only provides OS-level virtualization, has a noticeable influence on the systems' performance in these cases. The main problem identified is that the VMMs try to fetch as many packets of the network card as possible and therefore prevent layer-2 flow control. Therefore, the data link layer cannot limit incoming data rate and a high number of packets are lost in the VMM.

2.4.7 Isolation of Virtual Resources

In the following, we study how two virtual guests are interfering with each other. Therefore, we first conduct tests with a second idle virtual guest Linux. The in-

fluence of a second idle guest compared to only one guest running is mostly negligible. Thus, we focus on the influence of a second guest running CPU-intensive and memory-intensive tasks or forwarding small packets.



(a) 64 byte packets no CPU/RAM load

(b) 64 byte packets high CPU/RAM load

(c) 500 byte packets no CPU/RAM load

(d) 500 byte packets high CPU/RAM load

Figure 2.15: Maximum accumulated throughput of VMMs.

The impact of a second virtual guest, which is performing CPU-intensive and memory-intensive tasks, is also small in most cases. In Figure 2.15 we compare the impact of a second virtual guest system running CPU-intensive and memory-intensive tasks, as described in Section 2.4.5, while forwarding two packet streams over the first guest system. Figure 2.15a shows the mean forwarding capacity per stream while the second guest is idle. The plots for the

measured throughput develop as observed from the previous section. All VMMs receive more packets from the network interfaces than they can forward and a high number of packets are lost. Comparing these results to the scenario in which the second guest is running CPU-intensive and memory-intensive tasks, cf. Figure 2.15b, there is hardly any difference visible. If we consider forwarding two streams of 500 byte packets with an idle second guest, cf. Figure 2.15c, against a second guest running CPU and memory intensive tasks, cf. Figure 2.15d, some small changes can be seen. These differences apply to VMware with the para-virtualized network driver and OpenVZ. For VMware the plot might indicate a lower throughput performance for high offered loads. However, the confidence intervals overlap. Thus, this result is not statistical significant and requires further investigation. In case of OpenVZ, Figure 2.15d shows that the variation of the forwarded bandwidth is smaller while the measured mean throughput is decreasing rapidly.

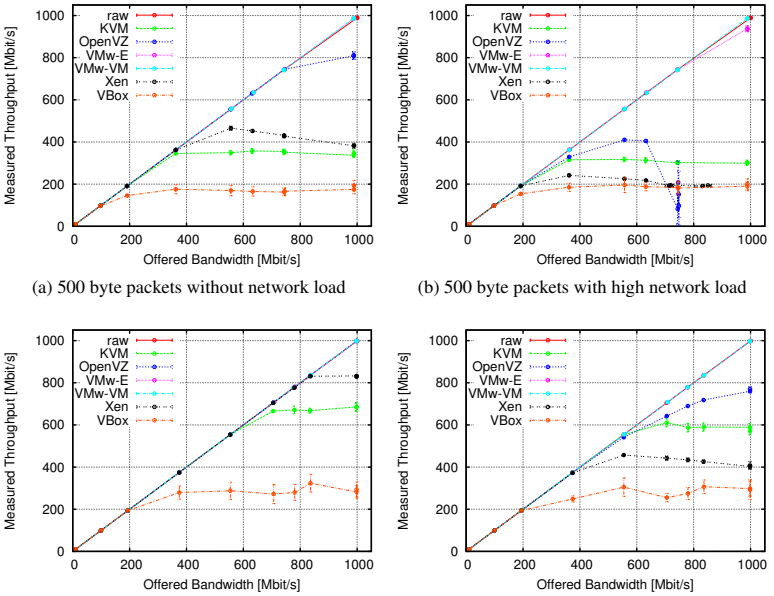
Table 2.1: Averaged relative throughput changes considering a single traffic source and a second guest running CPU/memory-intensive processes, cf. Section 2.4.5.

packet size	KVM		OpenVZ		VM-E1000	
	light	heavy	light	heavy	light	heavy
64	4.2%	-0.4%	1.0%	-18.2%	-2.7%	-4.5%
500	5.5%	4.0%	1.5%	-11.4%	-4.4%	-5.3%
1000	5.1%	5.6%	0.0%	0.0%	0.0%	0.0%
1500	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%

packet size	VM-VMXNET3		Xen		VirtualBox	
	light	heavy	light	heavy	light	heavy
64	-0.5%	-0.6%	6.8%	7.9%	2.1%	2.0%
500	0.0%	-0.1%	3.7%	3.4%	-2.0%	-4.0%
1000	0.0%	0.0%	0.8%	1.7%	3.6%	-4.8%
1500	0.0%	0.0%	0.0%	0.0%	3.4%	2%

We provide the difference in maximum achieved throughput in Table 2.1 for CPU-intensive (*light*) and CPU-intensive and memory-intensive (*heavy*) loaded second guest. It has to be noted that positive values in Table 2.1 correspond to

an increase of throughput under these conditions. For example, Xen achieves in any considered scenario a higher maximum forwarded bandwidth, if a second virtual guest is running CPU or CPU and memory intensive tasks. Only OpenVZ is negatively affected by more than 10% relative forwarding performance considering small packets. This effect might be caused by the fact that OpenVZ, in the original implementation, does not foresee the option to restrict the execution of processes to predefined CPUs. Thus, the `stress` processes are run on all CPUs compared to the other scenarios, where the `stress` processes are strictly bound to the CPU of the second guest. It has to be noted that both OS containers in OpenVZ are started with the "CPUUNIT" parameter, which means equal sharing of CPU resources.



(a) 500 byte packets without network load

(b) 500 byte packets with high network load

(c) 1000 byte packets without network load

(d) 1000 byte packets with high network load

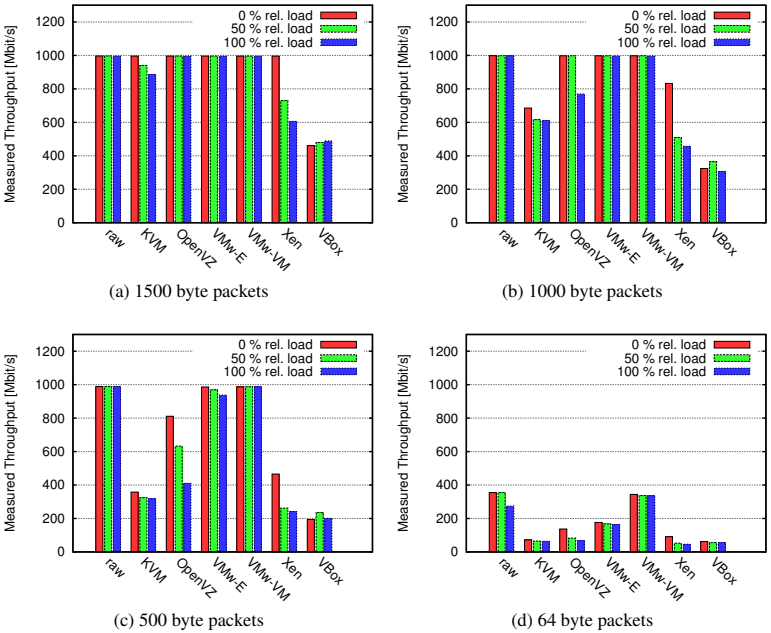
Figure 2.16: Throughput while transmitting 64 byte packets over a second VM.

The impact of a second guest is clearly recognizable, whenever it is handling small packets, as depicted by Figure 2.16. In the subfigures we contrast the throughput performance of the VMMs forwarding 500 byte packets with an idle second guest, cf. Figure 2.16a, and a second guest forwarding the maximal 64 byte packet rate we measured in the dedicated host scenario, cf. Figure 2.16b. It is clear to see that the measured throughputs using Xen and OpenVZ as VMMs are notably decreasing. In the scenario with a high network load on the second guest, OpenVZ is no longer able to fetch all packets from the network interface. We can see this problem in Figure 2.16b by looking at the offered throughput. In our tests, it was not possible to offload more than about 750 Mbit/s to the UUT. Although the measured throughput for OpenVZ is not reaching the maximum rate, it is again heavily varying. These effects are also visible for larger packet sizes, cf. Figure 2.16c and Figure 2.16d. Again Xen and OpenVZ are affected most, but with this larger packet sizes, OpenVZ does not show the fluctuation of the measured throughput.

In Figure 2.17 we present the maximum measured throughput for all scenarios in which the second guest forwards 64 byte packets. Even if the UUT is only handling 1500 byte packets KVM and Xen are not able to achieve their maximum throughput. However, if we consider the results for streams of 1000 byte and 500 byte packets, also OpenVZ and VMware without the para-virtualized network driver are affected, whereas the raw system and the VMware guest with VMXNET3 driver still handle all packets.

For 64 byte packets it seems as if the VMware system with the para-virtualized VMXNET3 driver outperforms the raw system. But this is not the case. Due to the fact that we send packets through the second guest relative to the performance we measured before, the raw system handles 270 Mbit/s of 64 byte packets in the background and forwards all packets. The VMware system is only forwarding about 200 Mbit/s in the second guest and is buying the higher throughput of the test system by a loss rate of about 20%.

The offered throughput and the measured throughput for the second guest in this experiment are provided for both load scenarios in Table 2.2 and Table 2.3,



(a) 1500 byte packets

(b) 1000 byte packets

(c) 500 byte packets

(d) 64 byte packets

Figure 2.17: Maximum throughput considering a second guest transmitting small packets relative to the maximum throughput in the dedicated scenario, cf. Figure 2.13d.

respectively. These measurements reveal that the second guest is also negatively affected. If both guests are forwarding 64 byte packets, loss rates up to 50% can be experienced. The most severe impact can again be seen for OpenVZ. It seems as if the well-known problem of the Linux kernel to handle small packets, cf. [79], has an even larger impact when using OS-level virtualization. This effect has to be considered when planning experiments on experimentation facilities using this kind of virtualization, as it might cause a high variance in the results of experiments, depending on the processes in other guest systems.

Table 2.2: Averaged offered throughput (in) and throughput measured (out) for 2nd guest in background at 50% relative load, cf. Figure 2.13d.

packet size	raw [Mbit/s]		KVM [Mbit/s]		OpenVZ [Mbit/s]		VM-E1000 [Mbit/s]	
	in	out	in	out	in	out	in	out
1500	153.1	153.1	37.1	37.1	72.1	72.1	71.8	71.8
1000	153.0	153.0	37.1	37.1	72.1	72.1	71.8	71.8
500	153.0	153.0	37.1	37.1	72.1	68.1	71.8	71.8
64	153.0	153.0	37.1	37.0	72.1	62.5	71.8	71.8

packet size	raw [Mbit/s]		VM-VMXNET3 [Mbit/s]		Xen [Mbit/s]		VirtualBox [Mbit/s]	
	in	out	in	out	in	out	in	out
1500	71.8	71.8	152.2	151.9	37.1	36.8	37.1	36.7
1000	71.8	71.8	152.0	151.8	37.1	36.6	37.1	36.0
500	71.8	71.8	152.0	151.7	37.1	36.1	37.1	36.2
64	71.8	71.8	152.1	151.8	37.1	35.4	37.1	36.8

Table 2.3: Averaged offered throughput (in) and throughput measured (out) for 2nd guest in background at 95% relative load, cf. Figure 2.13d.

packet size	raw [Mbit/s]		KVM [Mbit/s]		OpenVZ [Mbit/s]		VM-E1000 [Mbit/s]	
	in	out	in	out	in	out	in	out
1500	275.8	270.4	72.1	63.2	127.0	127.0	126.6	126.6
1000	275.6	270.4	72.1	61.3	127.0	102.7	126.7	126.6
500	275.9	271.4	72.1	60.8	125.9	80.1	126.6	126.6
64	274.9	269.3	72.1	64.6	124.4	62.7	126.6	126.6

packet size	raw [Mbit/s]		VM-VMXNET3 [Mbit/s]		Xen [Mbit/s]		VirtualBox [Mbit/s]	
	in	out	in	out	in	out	in	out
1500	275.8	270.4	274.4	268.3	72.0	56.2	72.1	42.1
1000	275.6	270.4	274.3	267.6	72.1	50.9	72.1	41.5
500	275.9	271.4	274.3	267.5	72.1	48.9	72.1	40.2
64	274.9	269.3	274.4	268.4	72.0	47.2	72.1	42.4

2.5 Lessons Learned

In this chapter, we characterized six use cases for future virtual networks. These use cases demonstrate that the current convergence to an all-IP-network adds complexity to the Internet that is very challenging to handle. In order to build an architecture that is able to create networks which can guarantee a high Quality of Experience for the customer in the future, virtual networks need to partition the traffic and the functionality into blocks of reasonable complexity that can be monitored and managed automatically. Furthermore, network virtualization enables new functionality in the network, e.g. green networking and the beta

slice, that cannot be build with today's technology. The creation of the use-cases in many iterative steps clearly revealed problems of today's Internet and options to improve this in future networks.

From the use cases we derived basic building blocks and described five functional roles that build, operate, and use virtual networks. Trying to implement the use cases with the role modell, we learned that it is crucial to separate different networks completely on an end-to-end basis. Therefore, we integrated the end customer in the role model. The interfaces between these roles need to be defined by hardware vendors in order to speed up standardization and shorten the time to market. We focused on a monitoring architecture for virtual networks. The unique characteristic of this architecture is that besides the physical infrastructure provider, no other role is able to access the physical substrate. Hence, interfaces for real-time clock access and monitoring data exchange have been defined. The end-customer being a PIP of its own network adds new aspects. The end-customer PIP does not earn money with its infrastructure, i.e. is not working profit-oriented. Hence, it operates on a completely different reasoning basis. As the end-customer PIP is not paid for its resources, it also is not punished for SLA violation in its network. Hence, a end-customer PIP might violate the SLAs of a virtual network or unexpectedly cancel a virtual network in favor of another.

Finally, we investigated currently available hardware virtualization platforms. We compared current concepts, including KVM, OpenVZ, VMware, Xen, and VirtualBox to a raw host system and revealed that the impact of virtualization is noticeable for most virtualization platforms even when only considering the throughput on a single network interface. When increasing the number of interfaces and traffic sources, each VMM revealed a throughput bottleneck, which depends on the size of the packets being transmitted but is significantly lower than the forwarding capacity of the raw system. The impact of a second guest, which is idle or running only CPU and memory intensive tasks, is rather small. However, a second guest that forwards small packets even at comparably low rates is able to influence the performance of a virtualized system severely.

Our findings show that virtualized systems and the traffic they handle need to

be monitored. Network monitoring and exact knowledge of the used systems enable the diagnosis of performance bottlenecks and definition of relocation strategies. Our work is important to studies carried out in research facilities, in which many scientists share the resources of a testbed. In this case, it is crucial to record all influences which might be introduced by another test running on the same testbed, in order to provide credible scientific results. Most of today's scientific testbeds do not provide this option.

3 Mapping User-Perceived QoE to Network QoS

I can't get no satisfaction.
Mick Jagger

In the future Internet, virtual networks enable the creation of concurrent networks, each designed for a special purposes. Functional roles optimize the network on different layers to guarantee a high QoE to the end customer. However, between adjacent roles the monitoring focus differs significantly. Thus, it is necessary to translate between the different monitoring views in order to enable efficient communication and create a common view on the network management.

In this chapter we focus on the interface between the ASP and the VNO. The ASP is mainly interested in the QoE of the customer and thus focuses on it. If the ASP uses a self-developed application service, e.g. Skype using their own voice-codec silk, it focuses on service-specific parameters. However, in the case the ASP uses a third party application service or library, e.g. a VoIP company using the iLBC-library, it will monitor network parameters and try to assess the QoE based on these measurements. In both cases, the ASP will only monitor at the end-points of the transmission, as it is interested in the end-to-end quality and does not necessarily have access to any intermediate network node. In contrast, the VNO is operating the transport network and, thus, only measures the QoS parameters of the network. In order to dimension the network and guarantee a high QoE perceived by the user, an application-specific mapping between the QoE of the user and the QoS in the network is needed.

Using a self-implemented service, the ASP needs such a QoE-QoS-mapping to translate between the measurements of the ASP and the VNO and to enable efficient communication between the roles. The values monitored from one role and the ones computed from the measurements of the other role are compared using the interfaces described in Section 2.2.4. This information exchange allows to identify different management views on the same network scenario and helps to concurrently optimize the network for all roles. In the case, the ASP also focuses on monitoring network parameters, the information exchange does not need a mapping. However, the ASP is in need of a QoE-QoS mapping that allows itself to infer the user-perceived quality.

The process of defining an QoE-QoS-mapping can be simplified, if a standardized tools was available, which enables the computation of the user-perceived quality. Such a tool can be used to generate an application-specific mapping. If no standardized tool is available, it is necessary to define a measure that is highly correlated to the user-perceived quality. In the following chapter we apply both methods and show that each can be used to design a QoE-QoS-mapping. We will use a standardized tool to assess the audio quality of speech transmissions in Section 3.2. For using MS Office as Software-as-a-Service (SaaS) such a tool is not yet available. In Section 3.3 and Section 3.4, we will define a metric for assessing the user-perceived QoE in the studied scenario.

In the remainder of this chapter we first introduce in Section 3.1 the terms QoE, QoS, standardized tools for assessing the QoE of different applications, and consider related work. In Section 3.2 we examine a QoE-QoS-mapping for a SILK based VoIP service based on automated user perception assessment using a standardized QoE-tool. The codec named SILK is developed by Skype Limited and has been submitted for standardization within the IETF in July 2009. For analysing the QoE-QoS relation for Microsoft Office applications provided by a SaaS approach, we choose the completion time of different task as a measure of the QoE. In Section 3.3, we analyse the influence of network QoS parameters on the user-perceived quality. We extend our measurements in Section 3.4 to investigate how different application layer settings can optimize the user-perceived qual-

ity under different network scenarios. Finally, Section 3.5 summarizes lessons learned within this chapter.

3.1 Background and Related Work

The two main network management objectives in this chapter are Quality of Service (QoS) and Quality of Experience (QoE). QoS is interesting for the VNO, which implements the data transport within the virtual networks. In contrast, the ASP is concerned about the QoE of the user only. On the ASP level it does not matter how the data is transferred, it is only necessary that data is transferred sufficiently well. What 'sufficient' actually means depends on the service itself and its implementation. Consequently, the information about the services' requirements needs to be exchanged. Another important factor is the user itself. Its opinion is influenced by many different aspects, e.g. prejudices, previous experience, and character traits. One option to better describe what sufficient means on a technical level is to specify all necessary parameters by a pre-defined template characterizing a group of services. The VNO extracts the information it requires for the specific service from the template. If a template is not available for the service, the ASP has to describe its services requirements in terms of QoS.

3.1.1 Quality-Metrics for Virtual Networks

The term QoS was originally intended to describe the users perception of a service, but was afterwards more and more often used in a technical context, cf. [80]. Nowadays, if we speak of QoS, we think of the well defined technical service description used in traffic engineering. The QoS of a connection or a network describes the properties of the data transfer. This includes the available bandwidth, the latency, i.e. end-to-end delay, the jitter, the packet loss ratio, and also the availability of the service, cf. [42, p. 351]. The term QoS is also used together with traffic management techniques, which allow to guarantee a predefined QoS, as it characterizes a data transmission service.

From the VNOs perspective, using QoS to describe a network makes sense. Depending on the parameters and the guarantees within the SLA, QoS can more easily be defined, contracted, and monitored as QoE. In contrast, from the ASP or user perspective, QoS has to be taken with a pinch of salt. It is for instance possible, that two completely different QoS parameter sets allow for the same effective TCP-transmission bandwidth. However, for another service, e.g. VoIP transmission using UDP, one of these parameters set could provide a good quality while the other only allows for a poor service quality. Hence from this perspective it is more reasonable to describe the service quality more user centric.

In the last decade, the term Quality of Experience (often abbreviated with QoE but also QoX is common) was introduced in order to quantify the users satisfaction with a service. In the past, QoE has also been studied in other science areas like psychology. There, QoE is a multidimensional construct that considers emotional, motivational, and cognitive aspects of experience. The core dimensions of this construct include activation or arousal (e.g., feeling vigorous), affect (e.g., feeling happy), and concentration, cf. [81]. In this work, we will focus on the users perception of a network enabled service and use technical measures to assess the QoE. However, if we cannot assess the QoE this way but have to ask user groups about there perception of a service, it has to be considered that the users perception might be influenced by the content and the presentation of the service. This means, that the aforementioned psychological effects might influence the result of the survey. For instance, if we study a video transmission service, the results might be affected by the video, although the users truly believe that they are judging the satisfaction with the service without any prejudice.

With the increasing interest of service providers in the assessment of QoE to improve their services, many studies with real user groups have been carried out during the recent years. One of the problems is how to characterize the outcome of such an experiment. The ITU-T describes in recommendation P.800 [82] a simple but efficient solution. Instead of describing the user perceptions in general, each user is requested to characterize its perception on a discrete numeric scale between five, i.e. excellent quality or imperceptible impairment, and one, i.e.

bad quality or very annoying impairment. Based on many individual results the mean value of the users perception is calculated and denoted by the term *Mean Opinion Score* (MOS). Hence, it is possible to accumulate the satisfaction of many different users perceiving the same service quality and translate this to a statistical measure.

For practical reasons, it is not possible to conduct a user survey whenever a new service implementation is constructed. Even a single technical parameter change can imply a survey with thousands of users consulted in order to gain reliable results. Hence, a lot of effort has been put into creating and standardizing tools, which allow to assess the MOS of a given service under certain conditions. The different approaches used for these tools can be split into three basic categories. *Full Reference* (FR) metrics compare the input of a service with the output perceived at the user and calculate the MOS based on this information, cf. [83]. For instance, a tool comparing a wave file before and after the transmission of a VoIP service would be an example for a full reference metric. Other tools described by the term *Reduced Reference* (RR) metrics calculate the MOS only by investigating application or network layer parameters, cf. [83]. An example for such a metric would be a tool, that is able to assess the MOS of a VoIP service by analyzing the transmitted and lost wave form samples or only network packets, without reconstructing the transmitted speech. Finally, there are tools that are able to assess the quality of a service by only looking at the delivered output. We call these tools *No Reference* (NR) metrics, cf. [83], as they do not need any reference material. Staying with the VoIP example, a no reference metric would only need access to the output wave file to predict the MOS.

Table 3.1 provides an overview of available QoE assessment tools for audio and video transmissions and the corresponding publications describing them. In the following we will focus on reduced reference metrics, as they are the ones, which can be used to translate between the view of the ASP, i.e. the QoE of the user, and the view of the VNO, i.e. the QoS provided by the network. We present an overview of publications addressing each topic and provide tables summarizing the considered QoS parameters and reference the proposed mappings.

Table 3.1: Overview on Available QoE Metrics and Measurement Concepts

Full Reference Metric	Reduced Reference Metric	No Reference Metric
PSQM (audio) [84] MNB (audio) [85] ESMBSD (audio) [86] PESQ (audio) [87] SSIM (video) [88] [89] [90] VQM (video) [91] TSSDM (video) [92] [93] PSNR, RMSE, MPQM, MQM, PVQA, PEVQ, PVQM (video) [29]	E-Model (audio) [94] [95] [96] ICPIF (audio) [97] PESQ (audio) [87] IQX (audio) [98] [99] The Effects of Jitter on the Perceptual Quality of Video [100]	PSQA (video) [101] Motion-based video models (video) [102]

Web Browsing and FTP

FTP and Web Browsing are classical services provided by the Internet and therefore are widely researched. Beuran et al. [103] show that the transfer time performance of FTP, which they think of reflecting best the user-perceived quality, in dependence on the packet loss in the network. They demonstrate that for increasing packet loss the transfer time is increasing exponentially, as FTP is TCP based. Furthermore they give a short introduction to the influence of network delay. For surfing holds the same as FTP, as the main performance metric for the user would be the delivery of the web site, which is also done over TCP. However, the authors explain, that the size of a file has also an influence. Transmitting only short files, the TCP connection is not in stable state and thus the transfer of one big file is more efficient than transferring the same amount of data in many short files. This is especially interesting, as current Web 2.0 pages usually consists of tens or hundreds of small files, which all need to be downloaded separately.

Khirman and Henriksen [104] study how bandwidth and delay influence the QoE perceived by a user. In order to characterize the QoE they consider the can-

cellation rate as a measure of the users satisfaction. They assume that if a user is not satisfied with the download speed of the web site, it most probably cancels the transfer. From their study they conclude that the service response time, which is composed of round trip time and server response time, is negligible in the investigated range between 50 ms and 500 ms. After this they also study the influence of delivery bandwidth and derive the cancellation rate CR from the delivery bandwidth b [kbit/s] as follows:

$$CR = -0.017 \log_e(b[kbit/s]) + 0.13. \tag{3.1}$$

Finally they take a look at the object delivery time, as this depends on all network parameters, which influence the data transfer. They show that the cancellation rate increases rapidly to a delivery time of one second and slowly linearly afterwards. Let D be the delivery time of a web object in seconds and again CR the cancellation rate. Then CR is given by

$$CR = 6 \exp^{-0.6D+0.014}, \text{for D < 1 sec} \tag{3.2}$$

$$CR = \frac{0.6\% * D}{1s} + 1.4\%, \text{for D} \geq 1 \text{ sec.} \tag{3.3}$$

There is also a ITU-T recommendation [105] on how to estimate the end-to-end performance for web browsing. In this recommendation the authors distinguish three user expectation classes, i.e. the user expects the response of the browser within six, fifteen or sixty seconds. The study presented in their publication is based on a user experiment with 6 phases. First the user loads the web site of a search machine, then enters the request, waits for the results, selects one of the results and finally waits for the selected page to be delivered. From their measurements they compute for each expectation class a matrix with different weighting factors. Multiplying this matrix with the duration of the test periods results in a weighted session time S [sec]. S can be used to estimate the MOS of

a session with the users expected maximum time (max) [sec] by:

$$\text{MOS} = \frac{4}{\ln((0.011\text{max} + 0.47)/\text{max})}(\ln(S) - \ln(0.01\text{max} + 0.47) + 5. \quad (3.4)$$

The authors also present a formula which considers the session time of one web page S_1, i.e. the time passing between entering the URL and the page is downloaded and shown completely. The considered user expectation of the maximal response time is between three and fifty seconds:

$$\text{MOS} = \frac{4}{\ln((0.003\text{max} + 0.12)/max)}(\ln(S_1) - \ln(0.003max + 0.12)) + 5. \quad (3.5)$$

We can conclude that the interesting parameter for user perceived QoE is the download time of the web page or the content. However, this is not one of the classical QoS parameters. Therefore, formula 3.1 uses the bandwidth as an approximation of a web page download time. However, if web pages vary in size, the precision of this measure is reduced. Against, if we know the size of a web page, we can infer the tcp-based download times from delay, loss, and bandwidth of the connection. Hence, we adjusted the QoS parameters needed for the formulas in Table 3.2 accordingly.

Table 3.2: Existing QoE metrics for web traffic and the influence factors considered in the QoE mapping formula

Reference	Parameter				formula
	delay	packet loss	jitter	bandwidth	
Khirman et al. [104]				x	(3.1)
Khirman et al. [104]	x	x		x	(3.2), (3.3)
ITU-T G.1030 [105]	x	x		x	(3.4),(3.5)

Voice Applications

Voice Applications often imply an interactive communication that is more demanding in terms of requirements for the network than simple data transmission services. The quality of a voice application can be judged for a complete session or for parts of it. While the former provides a good overview, the latter corresponds better to the perception of a user during the call. Hence, we will discuss both approaches in this section.

One of the widely known models to estimate the QoE perceived by a user is the E-model [95]. It is based on the assumption that psychological factors are additive, which means that is is possible to calculate the quality of a voice transmission by judging the quality of the encoding and then subtracting several impairments which negatively influence the quality:

$$R = R_0 - I_s - I_d - I_e + A. \tag{3.6}$$

In this formula the 'perfect' quality is represented by R_0. This, however, includes background noise at the sender and the receiver as well as circuit noise. I_s summarizes impairments simultaneous to the voice signal like overdriven loudness. I_d are the delay impairments of voice signal considering absolute delay and echos. I_e specifies the effects of special equipment which describe e.g. the influence of the codec. Finally the advantage factor A is based on the assumption that user expectations differs depending on the utilized equipment and his quality expectations in this setup. The advantage factor therefore is 10 for GSM and only 5 for DECT telephones as the customer expects a better quality of the DECT phone compared to a 2G mobile phone. Finally the R value can be transferred to a MOS value using a linear transformation. The advantage and disadvantage of this approach is based on the formula used to compute R. If the E-model is used to describe VoIP only Equation 3.6 reduces to:

$$R = 94 - I_d - I_{e-eff} + A, \tag{3.7}$$

where I_d is caused by delay and I_{e-eff} represents the impairment of the codec and the packet loss.

Beuran et al. [103] explain that the dependency of the QoE on the delay is well-known and therefore focus on the influence of jitter and loss. They present measurement results for loss ratios between 0% and 10% and jitter values up to 75ms. The considered application did an ITU-T.G711 encoding and used a de-jitter buffer of 80 ms. In this setting the authors show that the influence of loss depends on the jitter. For high jitter values, e.g. 50 ms jitter and more, the influence of a packet loss of up to 10% does not influence the PESQ score, which is used to describe the QoE. The authors show that even for small jitter values, i.e. 25 ms and below, the influence of loss is only small and based on their figures it seems that its influence is linear. In contrast, the results for the jitter analysis reveal an exponential decay. However, the authors do not provide any formula which allows to estimate the QoE based on measurements of jitter and loss.

Hoßfeld et al. [99] hypothesize an exponential dependency between the QoE and QoS. This is motivated by the fact that it is a basic feature of human perception that small changes on a high service level are experienced more severe than the same changes but expecting a lower service level. In order to prove this relationship the author investigate PESQ values of speech transmission with two different codecs, namely G.711 and iLBC, over disturbed networks emulated in their testbed. On the one hand they do not study the interdependence of the considered parameters but focus on the sensitivity of the QoE with respect to a single parameter. On the other hand this study considers a wide range of packet loss, i.e. 0% to 40%. For jitter the paper considers additionally the case of packet reordering. This means that the jitter introduced in their testbed is large enough that the packets are not received in the order they were sent. Their intensive measurement study supports their hypothesis of an exponential dependency between the considered QoS Parameters p and the QoE in terms of MOS:

$$\text{QoE(p)} = a \exp -bp + c, \tag{3.8}$$

where a, b, c are free parameters.The paper presents various solutions for the free model parameters a,b,c in order to fit the equation to their measurement results.

Table 3.3 provides an overview of the QoS parameters considered in the pa-

Table 3.3: Existing QoE metrics for voice traffic and the influence factors considered in the QoE mapping formula

Reference	Parameter				formula
	delay	packet loss	jitter	bandwidth	
E-model [95]	x	x	x		(3.6),(3.7)
Beuran et al. [103]		x	x		
Hoßfeld et al. [99]		x	x		(3.8)

pers. It is clear to see that jitter and packet loss are considered to be the main influence factor. Although all paper state that delay is also an important factor, most of them do not analyze it. Since the influence of delay cannot be assessed with a full reference metric, it has to be studied with human test persons, which is much more time consuming. Furthermore, in many cases the delay is mostly caused by the physical restrictions of the transmission medium and can hardly be optimized within the network.

Video Streaming

To judge the quality of a video with a full reference metric is not an easy task. Due to many different tools considering different features of the video, the scientific community did not agree on a full reference model yet. The discussion, which model fits the QoE perceived by the user best, is still ongoing. Therefore it is not easy to create a reduced reference model, even with an automated testbed, as there is no algorithm that assesses the perceived quality based on a transmitted video and the reference. However, there are some publications that provide a reduced reference model even if only for certain applications and network scenarios.

One reference for mapping QoS to the QoE of a transmitted video is the *Media Delivery Index* (MDI) described by Welch et al. [106]. The MDI defines two values which are used to describe the quality, i.e. *Delay Factor* (DF) and Media

Loss Rate (MLR). DF is calculated from

$$\Delta = bytes_{received} - bytes_{drained} \tag{3.9}$$

at each packet arrival, which calculates the amount of application layer data lost between two consecutive packets that arrive at the receiver. For a fixed time the DF, which describes the time needed to fill or drain a the data of a application layer video buffer, can be derived by:

$$\text{DF} = \frac{\max\Delta[bytes] - \min\Delta[bytes]}{\text{media rate [bytes/ms]}}. \tag{3.10}$$

The MLR is just the number of packets lost in an observation interval:

$$\text{MLR} = \frac{\text{Packets}_{expected} - \text{Packets}_{received}}{\text{duration of the interval}}. \tag{3.11}$$

Note that packets, which are still in transmission are neglected, as many video codecs are considered not to reorder out-of-time frames, but skip them instead. Although these values provide detailed information on the distortion of the video, the QoE cannot directly be derived. In oder to estimate the QoE it is necessary to have another function, which takes care of the features of the codec, the size of the jitter buffer etc.

Khan et al. [107] mainly focus on a cross layer optimization but also provide a reduced reference model, in which they use the *Peak Signal-to-Noise Ratio* (PSNR) as an input to calculate the quality of a video stream transmitted:

$$\text{MOS} = \max(\min(\frac{3.5}{20}(f(R,p) - 20) + 1, 4.5), 1), \tag{3.12}$$

$$f(R,p) = 10log_{10}\frac{255^2}{\frac{a}{e^{Rb^{-1}}-1} + \beta p}, \tag{3.13}$$

where R is the encoding bit rate, p is the loss probability, and a,b and β are free parameters to adapt the model to the considered system. Equation 3.13 calculates the achieved PSNR and Equation 3.12 maps the PSNR to MOS scale.

Another model is defined by the ITU-T [108], which provides quality assessment for video-telephony applications. The quality of the video and the voice stream are judged independently from each other and afterwards combined. Although an interactive service is studied the video quality V_q is found to be independent from delay and jitter, as long as overall delay including the transmission and the jitter buffer delay is less than one second.

$$V_q = 1 + I_{coding} \exp^{-p_{plv} D_{PplV}^{-1}}, \tag{3.14}$$

where I_{coding} the is the distortion of the video caused by the encoding and the frame rate, p_{plv} is the packet loss probability, and D_{PplV} is the robustness of the encoding against packet loss. The paper presents a formula to calculate the D_{PplV} based on encoding parameters, i.e. the video frame rate and the video bit rate, and five adjustment factors, which are provided in a table in the appendix for two specific codecs with predefined video resolution.

A different approach is chosen by Naegele-Jackson [109]. This study does not rely on a full reference model. Instead, the author conducted a survey, where she used different encoding schemes and different transfer technologies on the network and the physical layers. Although she did not present a reduced reference model, she presents a lot of results, which could be used to adjust another model to a specific encoding and transmission scheme.

Table 3.4: Existing QoE metrics for video traffic and the influence factors considered in the QoE mapping formula

Reference	Parameter				formula
	delay	packet loss	jitter	bandwidth	
Welch et al. [106]		x	x		(3.10),(3.11)
Khan et al. [107]		x			(3.12),(3.13)
ITU-T G.1070 [108]	x	x	x		(3.14)

As we can conclude from Table 3.4 most reduced reference metrics considering video consider packet loss. If no jitter buffer is used or the buffer is not sufficiently dimensioned, this is also perceived as application layer loss. Hence, jitter is also a relevant influence factor. Only one modell considers delay as an important influence factor. Practical experience with television broadcast over different technologies showed that depending on the context and the content, delay might have a severe influence. If we consider a soccer transmission and the delay of some recipients is noticeably larger than other recipients, this will have a severe influence on the QoE. Especially, if the first group is still receiving the play in the midfield while the others already celebrate the goal of their team. However, such situations can hardly be emulated in a experimental environment and are therefore hard to integrate in universal reference metrics.

Software-as-a-Service and Terminal Services

Nowadays, most software products and especially office applications are run locally on the client computer. Cloud computing is about to change this in the near future. Instead of executing software on a local personal computer, the application is hosted on a server somewhere in the Cloud, i.e. a data center connected to the Internet. Therefore, is is no longer necessary to buy powerful end-systems and software licenses, but use the application as a cloud based service on any end device and pay only for the usage. Hence, this paradigm is called *Software-as-a-Service* (SaaS).

SaaS offers many advantages like lower hardware and administration cost, less energy consumption, and a more efficient use of resources. Especially small and mid-size organizations can achieve a much higher degree of security and reliability if all their applications are hosted, managed and maintained by an Application Service Provider who can spread the support and maintenance costs among a large number of customers [110]. A SaaS architecture also enables the simple integration of home office workers into the corporate environment, as the data is always kept on the SaaS servers and does not need to be transferred between different work places. The downside is that the response of the application to the

input of the user is no longer direct but has to be transmitted over the network.

The technical implementation of this transmission depends on the concrete systems. Often web-based solutions using Asynchronous JavaScript and XML (AJAX) or HTML 5 are used to deliver SaaS applications to the browser of the user. Other solutions implement rich clients using Adobe Flex or Microsoft Silverlight. The disadvantage of these solutions is that their implementation is based on the https protocol, which adds a lot of overhead and was initially not designed for live data transfer. Other solutions for delivering SaaS are PC over IP (PCoIP) by vmware [111] and Terminal Services. PCoIP has the advantage that it can deliver any kind of graphical output rendered in the cloud to the client system. However, specialized hardware is needed for PCoIP, which makes the implementation expensive and is orthogonal the idea of delivery to any end device. In contrast, Terminal Servers are an established solution that is able to deliver complete desktops as well as single applications. Furthermore, there are already implementations of the client software for any kind of end device. Hence, we will focus on SaaS based on Terminal Servers as a delivery platform in this chapter.

Other Applications

Many other publications discuss the influence of network QoS on the user-perceived QoE for a variety of different applications. For example [112] shows that the influence of delay is critical to the quality of highly interactive multiplayer games. Jarschel et al. [7] analyze how different kind of games are affected under changing QoS for Cloud Gaming. What is common to all those publications is that none of these provide a concrete formula to estimate the QoE of the user based on QoS parameters measurable in the network.

3.1.2 Related Work

In the remainder of this chapter we will focus on two different services. The first one is a VoIP service. The second one is MS Office provided as Software-as-a-Service. Thus, we consider the following publications as related work.

QoE of VoIP services

VoIP is an interesting topic and many publications consider the QoE of VoIP transmissions. An exhaustive overview is, therefore, out of scope of this work, but we summarize some closely related examples.

Deri [113] describes his work on an open source software named *ntop*, which monitors VoIP traffic. This software, which has been further developed over the last years, is able to detect VoIP flows and export the flow characteristics, e.g. source and destination IP and port, flow length in time, packets and bytes, using the netflow/IPFIX protocol. However, it does not consider the QoE of the monitored flows.

Other publications compare different VoIP applications considering different network characteristics. For instance, Chiang et al. [114] present a comparison between MSN and Skype with respect to the available bandwidth, packet loss, cross traffic, and different network scenarios. They conclude, that MSN sends smaller packets with a higher rate and higher variance at almost fixed size, while Skype sends larger packets at a lower rate with less variance of the inter-sent time and higher variance of the packet sizes. Additionally both programs differ in their way they react on bandwidth bottlenecks. While MSN reduces the throughput when facing a bandwidth bottleneck, Skype increases the throughput. Finally, Skype is reported to better handle connection problems due to NAT devices in the network. Another special feature of this publication is that the results of real user surveys are provided. Barbosa et al. [115] compare Skype to Google Talk. Besides the effects of network QoS parameters on the QoE perceived by the user, this paper focuses on the strategies these applications use to cope with degraded network conditions, e.g., loss and low available bandwidth. However,

both publications do not provide a model mapping the QoS of the network and the QoE perceived by the user. In order to assess the perceived quality, the authors use the PESQ [116] tool, which is characterized by Rix [117]. The PESQ tool compares two wave file representing the input signal and the received output of a voice transmission. Based on these two files, the PESQ tool is able to assess the user-perceived quality in terms of MOS. Pennox [118] analyzes the accuracy of the PESQ tool and shows that the calculated MOS values are not necessarily precise. He reveals that the residual error of the PESQ tool is only below 0.25 if the result of the PESQ tool is close to 3. The used codec changes the precision of the PESQ tool and the absolute value of residual error increases with higher packet loss values. Considering these results, an additional gap between the targeted lower MOS values and the results of the PESQ tools may be added to assure a high QoE.

Chen et al. [119] provide a reduced reference model. It is based on the assumption that the call duration and the QoE are positively correlated. In their paper, they develop a model that can estimate the mean QoE a user perceives under a given network QoS. It is claimed that the network latency between the communication partners might influence the QoE of a user. However, this impairment does very seldomly decrease the QoE so strongly that the user would terminate the call. Furthermore, the authors show that in 46% of all cases, in which the user hang up due to disturbances in the network, this was caused by a low bit rate of the transmission. In 53% of all cases the hang up was caused by jitter, and only in 1% of all cases it was caused by the network latency. However, the model does not contain quantitative information about the range of qualities different users may perceive.

Sat and Wah [120,121] summarize coding schemes for VoIP transmissions and special features a peer-to-peer VoIP network should implement. They discuss in detail how latency affects the QoE of VoIP conferences with two or more partners. They conclude that there are currently no models which can correctly map these latency issues to the user-perceived QoE.

The earlier mentioned hypothesis of an exponential interdependency between QoS and QoE is postulated by Hoßfeld et al. [99]. The authors study the impairment of loss and jitter in the network and prove that there is an exponential mapping between random loss and QoE.

Our work in Section 3.3 is inspired by the last three publications, but extends their work in many ways. First of all, we focus on the wideband codec SILK, which is used since Skype version 4.0 and was revealed to the public as an IETF draft in March 2010. Moreover, we consider uniform and bulk loss as well as different speech durations to provide a model which can be used for QoE-monitoring. It considers not only the medium MOS but also the range in which different implementations of the same loss percentage are perceived by the user.

Application delivery via SaaS and Terminal Services

The research on SaaS and terminal service based application delivery has mainly focused on typical traffic characteristics and mostly neglected the user-perceived quality.

Deboosere et al. [122] compare the performance of different Thin Client solutions when faced with slow motion (i.e. using a text editor and browsing) and high motion (watching a video, playing a 3D game) tasks. The results illustrate that all tested protocols are well-suited for low motion applications. The study showed that in high-motion scenarios, in contrast, not all clients are able to send the graphical data fast enough to the user. Citrix is one example for terminal services that is suitable for this task and outperforms all other protocols in terms of bandwidth and used server CPU, if the Speedscreen Multimedia Acceleration is activated, cf. [123]. We discuss Speedscreen in more detail in Section 3.4.1.

Tolia et al. [124] measure the response time of text editing, presentation creating, and image processing applications accessed via VNC. For this purpose, all response times are classified to be between “crisp”, if smaller than 150 ms and “unusable”, if over 5 seconds. Other steps are not considered in this work. Based on this simple QoE measure, it is shown that the performance of highly interactive applications is more sensitive to network delays.

However, none of these studies deals with the problems, which is of key interest to SaaS providers: 'How can the QoE of an end user be monitored?' and 'How can the QoE of a user be improved, if the QoS cannot be changed?' To some extend Citrix Inc. and other companies selling network solutions are already advising their costumers on how to optimize their working setup. One example for this is provided by Juniper Networks [125]. They describe the optimal settings for the Juniper WX/WXC platform as well as the Citrix client and server for low bandwidth conditions. A Citrix white paper [126] addresses the problem of using the MetaFrame presentation server in a wireless WAN. To reduce the perceived user latency and login time and to ensure the most efficient bandwidth utilization, it is proposed to optimize the packet sizes, as well as to turn Speedscreen on and to use caching and compression. Another publication by Citrix [127] discusses various issues to be considered when setting up a Citrix network environment for maximizing the available bandwidth. The Speedscreen feature is also promoted to enhance the user experience on very small mobile devices, e.g. the Nokia Communicator 9210 [128]. However, all previous works only roughly explain *why* a certain parameter value should be chosen, while no details are given on the qualitative influence or the QoE perceived by the user.

3.2 Quality-Mapping for SILK VoIP Services

In the last few years, VoIP has come into its own. It is used by end consumers, at small and medium enterprises, universities, and even in large companies on a professional level. With the deployment in professional environments, it is necessary to guarantee the same or even better quality than the end user is accustomed to from classical telephone services. However, transmitting speech in packet-switched networks suffers from far more influence factors as compared to circuit-switched telecommunication networks. Due to this fact, VoIP transmissions need to be monitored carefully in order to achieve a satisfactory quality for the user. Monitoring on an end-to-end basis with a full reference metric is rarely possible, since the send and the perceived voice needs to be available.

However, a reduced reference model assesses the QoE based on the networks QoS and enables QoE monitoring based on network measurements. Many studies have analyzed the influence of degraded QoS in packet-switched networks on the user-perceived quality. To enable this research, tools like the PESQ [116] have been developed, which implements a full reference metric that reliably calculate the mean opinion score a large group of users would rate, comparing the speech quality at the sender and the degradation captured at the receiver side.

There are many publications, e.g. [99,114,115], which look at different codecs and analyze the influence of certain QoS parameters on the user-perceived QoE in terms of the MOS. The models that are derived from these investigations consider mostly the average MOS. However, different implementations of the same QoS may lead to completely different MOS values. This is due to the fact that losing a single part of a transmission, as it is perceived if several consecutive packets are lost, is rated completely different to a constant noise, which roughly describes the influence of random packet loss, although the loss percentage is equal in both cases. Hence, results on the average MOS are not sufficient, if we have to guarantee a certain QoE level for 95% of the users, which is an exemplary requirement for a professional environment. In this case, more detailed models are necessary.

In the following, we address the QoE monitoring problem described above for virtual networks supporting SILK-encoded VoIP transmissions. We focus on the SILK codec, as it is the standard codec used by Skype. It has been published in March 2010. This codec has broad application range, as it is developed to support different bit rates and is optimized for scenarios with transmission delay variation and packet loss rates, which frequently occur in today's Internet. These features are valuable in the Internet, but can also be used with virtualized networks. The robustness against predefined QoS degradations can be used to tightly dimension and provision the virtual networks. In an overload situation the virtual network automatically requests new resources to support the increased bandwidth demands. Regarding the target of the architecture characterized in Chapter 2, this might be done within a few seconds or some minutes in the worst case. However,

during the provisioning duration of new resources the SILK codec still guarantees high QoE.

The problem with the SILK codec is that it is a proprietary codec and does not allow the ASP to monitor application-specific parameters, like the number of lost waveform samples per time unit. Hence, it is necessary for the ASP to infer the user-perceived QoE from network parameter it is able to measure on an end-to-end basis. The mapping function discussed in the following assesses the distribution of the users' QoE by analyzing the application layer loss. Using this mapping we can infer that under given network conditions a predefined percentage of all users perceive a good voice transmission quality. We achieve this by deriving a function translating network QoS to a MOS value distribution describing a worst case assessment of the perceived QoE. Hence, the service quality of a predefined percentage of the users can be guaranteed. Our worst case assessment is based on a detailed analysis of the QoS influences on the QoE perceived by the user. We consider the impact of different loss patterns. Furthermore, we apply these pattern to samples of different transmission durations and study how the QoE is affected. Combining these results, we infer how the QoE perceived by the user can be safely assessed. Finally, we consider sampling techniques, which allow to reduce the measurement effort.

3.2.1 Automated QoE Assessment

The quality of a voice transmission is affected by many different parameters. In order to identify all influence factors, we analyze the way the voice signal is transmitted from speaker to listener, depicted in Figure 3.1. The first factor is the quality of the microphone and the analog-digital-converter the sender utilizes. A cheap microphone or digital-analog-converter introduce noise to the system, before the voice stream is actually transmitted. In our work, we focus on the influence of the network QoS on transmissions using a given codec. Therefore, we try to eliminate these impairments and use samples of studio audio productions with CD-quality for our evaluation. This is reasonable, as a VoIP service

provider would use high quality VoIP phones to achieve the best input quality possible, avoiding signal quality loss during digitizing the speech.

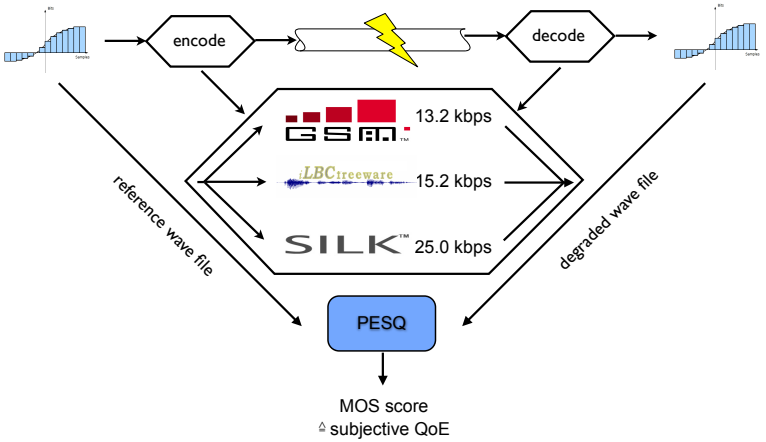


Figure 3.1: VoIP quality evaluation setup

The next influence factor is the codec, which is used to convert the digitized voice into a binary stream. For our evaluation we compare the quality of the well-known GSM codec and the former Skype codec iLBC with the Skype SILK codec. Skype calls SILK an 'ultra wideband audio codec', c.f. [129], which means that this codec should achieve a high QoE and should be less affected by network degradations.

After the voice sample has been encoded, the stream is cut into frames, which are transmitted over the network. Different network QoS parameters, e.g. packet loss, jitter, delay, and bandwidth restrictions, affect the quality perceived by the user. The two main influence factors according to [119] are packet loss and jitter. Although these factors are completely different on the network level, their effects are the same. The reason for this is that all modern VoIP systems integrate

a so-called play-out or jitter buffer. The purpose of this buffer is very simple. Whenever a packet arrives at the destination, the information in the packet is extracted and stored in the jitter buffer. Initially, the replay of the voice stream is delayed for a short time, i.e. a value below 300 ms for most applications and most network scenarios, c.f. [130]. Hence, the buffer enables smooth replay of packet streams, which suffer from a small amount of jitter, as long as all frames reach the buffer before they are accessed by the replay function. However, if the replay algorithm accesses a buffer place, for which the frame has not yet arrived due to higher jitter, the algorithm considers this frame to be lost. Therefore, jitter and loss can be considered to have similar consequences from the application layer perspective. From the monitoring perspective, we can conclude that it is necessary to check whether a frame arrives in time at the destination to cover jitter and packet loss.

Other influences of the network are bandwidth bottlenecks and latency. Bandwidth bottlenecks, which reduce the available bandwidth below the transmission rate of the sender also results in packet loss for the transmission. SILK can be adapted to this scenario by reducing the target bit rate of the encoding process as soon as packet loss is detected. However, a reduced target bandwidth also affects the perceived quality of the user. As we focus on a specialized virtual network for VoIP transmission, we assume that the network is able to allocate new resources in a reasonable short amount of time and we do not need to reduce the target bit rate at the sender.

The influence of latency is not yet researched in detail. We know that high latencies negatively affect the conversation quality. Especially for multi-user voice conferences, where different users experience different latencies, the conversation quality is severely affected, since crosstalk is hard to avoid in this situation, cf. [120]. These influences are still not completely clear. Additionally, there are currently no tools to map these influences to MOS values. Therefore, we focus on loss and jitter and leave bandwidth restrictions and latency to future work.

In order to investigate the influence of each part of the transmission, we analyze the influence of the network QoS on the QoE in a simulated environment. We encode the original wave-file with the different codecs using their default parameters for sampling and encoding bit rate. The resulting bitstreams are saved and packet loss is simulated by removing the information from selected frames. We decode the bitstream generated in the previous step and compare the resulting wave files with the original ones using the PESQ tool [116]. This way, we have full control over the network layer. We can exactly determine, which and how many fames are lost on the network. Another advantage of this method is that we do not have to resynchronize the sent and received wave-file, since we know exactly the beginning and the end of the transmission.

The PESQ tool, which we use to calculate the MOS value of each disturbed and undisturbed voice transmission, only achieves valid results within the specified input voice duration range between 5 and 30 seconds. In order to examine how an equivalent amount of loss is perceived over different time durations, we use voice excerpts of 5 seconds, 10 seconds, and 20 seconds length. Each sample contains a short silence period at the beginning and the end of the file. This is also a requirement of the PESQ tool as explained in the PESQ application guide [131].

3.2.2 Mapping QoS and QoE for VoIP Codecs

To gain an overview of the statistical characteristics of the results measured under the same network conditions, we use a modified box plot as exemplarily shown in Figure 3.2. All measured values lie within the interval given by the minimum and the maximum, i.e. by the lower and upper triangle, respectively. The mass of the distribution, i.e. 50 % of all observed values, lies within the box given by the 25 % and 75 % quantile.

The length of this box, i.e. the inter-quartile range (IQR), is thus a measure for the statistical dispersion of the observed test values. The positions of the mean and the median in the box, furthermore, characterize the skewness of the distribution: If both are roughly on the same position in the middle of the box, the

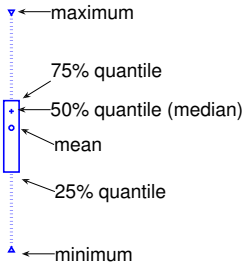


Figure 3.2: Illustration scheme of a distribution, i.e. the modified box plot.

mass of the distribution lying above the mean value is equal to the mass lying below the mean value. In contrast, if the median is not equal to the mean as in Figure 3.2, this is an indicator for a asymmetric distribution or statistical outliers, i.e. a small number of extreme test durations which are more than 2.5 times the IQR away from the median significantly influence the mean value. In the example, the median is, furthermore, very close to the 75% quantile, indicating, that many results, i.e. approximately 25%, are located within a small interval. Note, that in some plots we do not provide the mean value, as we focus on the distribution of results within different areas.

We start determining the user-perceived QoE from measured network QoS by applying the PESQ tool to samples for which we simulated the transmission. The resulting QoE values are based on the MOS, which assigns numbers from one for unacceptable quality, over three for acceptable quality, to five for very good quality. We perform tests using PESQ on the influence of random and bulk packet loss on the voice codec SILK as well as its predecessors iLBC and GSM. It has to be noted that comparing two binary identical files with PESQ leads to a maximal score of 4.5.

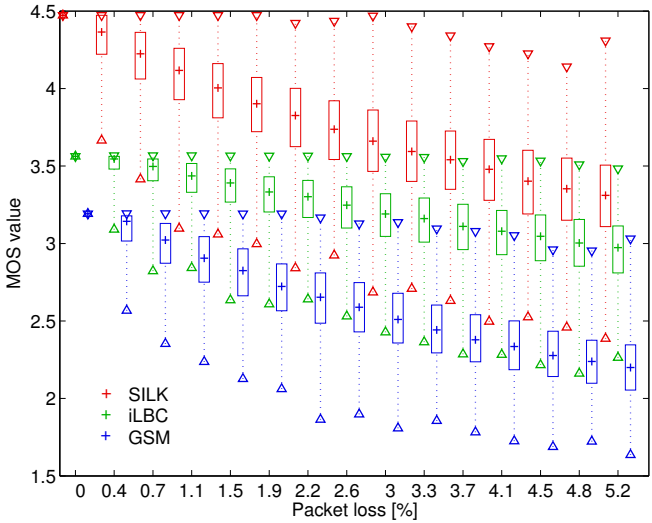


Figure 3.3: QoE of GSM, iLBC, and SILK for increasing random loss

Figure 3.3 illustrates the test results for uniform distributed packet loss. For each loss value, we considered 2000 different random loss patterns with exactly the same number of lost frames, i.e. we discarded the exactly same number of randomly chosen packets. In Figure 3.3 the boxes depict the inter-quartile range of the MOS for each value of loss and the triangles show the maximum and minimum values. As a modern voice codec, which is optimized for higher bandwidths than iLBC and GSM, SILK achieves the highest PESQ-based MOS value of 4.5, when not exposed to any loss. In the same scenario, i.e. no lost packets, iLBC and GSM score 0.9 and 1.3 points MOS lower, respectively. With increasing loss, the achieved MOS values decrease for all codecs. However, at 5.2% packet loss, 75% of the SILK connections still have an acceptable quality, i.e., the 25% quantile is above a MOS value of 3.0. Each of these connections are

rated with a higher MOS score than 75% of the iLBC and all GSM connections. Although the range between the connection with the highest and the connection with the lowest MOS score increases rapidly, the inter-quartile range stays relatively small with a range of about 0.35 MOS values. This shows that uniformly distributed loss affects many transmissions in a similar way. For Figure 3.3 we only show to the results of the five second voice duration tests. Note that the other experiments for longer voice periods reveal slightly different quantile values, but mainly the same influences. Hence, we do not present these results here.

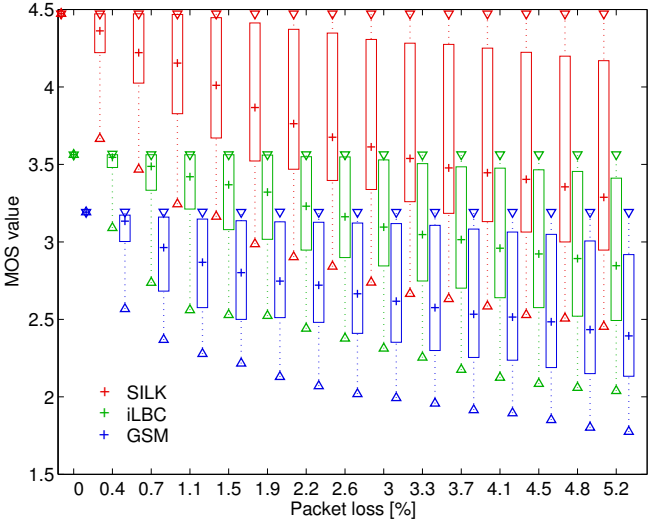


Figure 3.4: QoE of GSM, iLBC, and SILK for increasing bulk loss

In contrast to uniformly distributed loss, where lost frames are more or less equally distributed over the complete transmission, we now consider bulk loss. In order to capture all effects of a bulk with a predefined length n, we remove n consecutive frames at each possible position of the transmission. This means,

if we consider a transmission of 400 frames and a bulk length n=20, i.e. 5% bulk loss, we remove frame 1 to frame 20 for the first evaluation. For the second evaluation we discard the frames 2 up to frame 21 and so on until we delete frames 380 to 400 for the last evaluation. The resulting MOS values for the five second transmissions are shown in Figure 3.4. While the main trend is the same as in the uniform loss case, we notice two main differences. First, for all considered loss values there are some transmissions, which seem to be undistorted, i.e., the maximum MOS value in all cases is reached. This value is 4.5 for SILK, about 3.5 for iLBC, and about 3.2 for GSM. This phenomenon can be explained by the silence periods at the beginning and the end of the transmissions, which are needed by the PESQ tool. If the lost frames fall within these periods, their loss does not degrade the quality of the transmission and the MOS value is unchanged. Secondly, we recognize that the inter-quartile ranges increase in size. This means that the QoE differs heavily, for these cases.

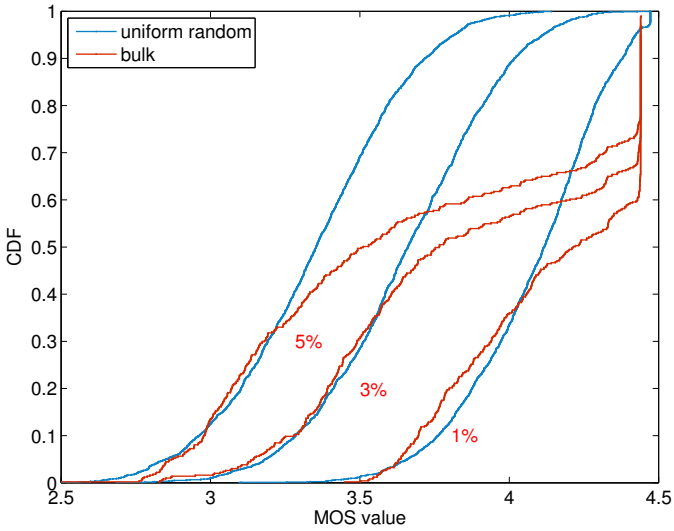


Figure 3.5: QoE comparison between random and bulk loss for SILK Codec

It is reasonable that SILK provides better quality in both scenarios. The interesting point is that SILK can provide MOS values above 3 for most of the customers, even if 2% of all packets are lost. This feature makes it interesting for QoE based network management. It allows us to careful overbook, cf. [132], the virtual network providing the VoIP service without risking users to perceive a bad quality. The other codecs provide only fair quality in an undisturbed network. Therefore, iLBC and GSM transmissions are less attractive for QoE monitoring, since even a small network degradation may lead to unacceptable quality using these codecs.

In order to define a worst case assessment, which covers random and bulk loss, we need to compare these influences in more detail. Figure 3.5 shows a comparison of the Cumulative Distribution Functions (CDFs) derived from the SILK PESQ measurements for 1%, 3%, and 5% of uniform and bulk loss. The plots of the bulk loss show that bursty loss pattern lead to better user experience in most cases, i.e. the range in which the curve of the bulk loss is situated beneath the corresponding uniform loss curve. In the other cases, the gap between the plots is quite small. Only in cases with low packet loss, i.e., the plots for 1% packet loss, it is noticeable. However, in cases with low packet loss, the user-perceived quality is relatively high, i.e. 3.5 and above. In the critical areas between a MOS of 3.5 and 2.5, a model describing the influence of uniform loss is sufficiently precise. Hence, we will use the influence of uniform loss for our monitoring solution, as it is easier to model and provides sufficient precision.

For the performance comparison of the different codecs and for deriving a suitable model we used the results, which we generated using five seconds of voice transmission. The cause for this is presented in Figure 3.6. It provides a comparison of the MOS value CDFs for 5, 10, and 20 seconds of voice transmission given five percent bulk loss. It has to be noted that due to different voice transmission durations, different numbers of frames are consecutively removed at each point of the encoded file. Due to the higher frame number of a longer voice transmission, the CDFs of longer files contain more values, as we consider every possible starting point for the loss period.

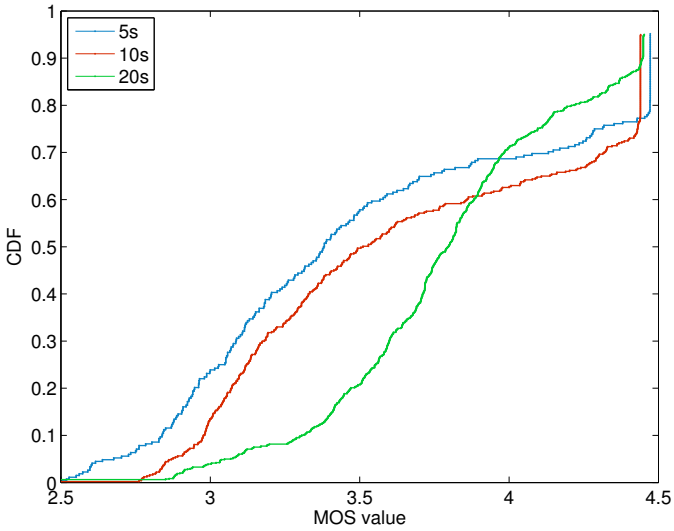


Figure 3.6: Comparison of bulk loss for different voice transmission durations

For the lower 70% of all results, the plot describing the results of the 5 second voice duration shows the lowest MOS values. Only for the better 30% of the resulting MOS values, the results from the 20 seconds file predict lower QoE values. This is due to the fact that we have to consider the silence periods at the beginning and the end of the files, which are needed by the PESQ tool to work properly, c.f. [131]. In a file containing 20 seconds voice, the relative part of these silence periods is smaller than in a 5 second file, which explains that the relative percentage of perfect transmissions is larger for shorter voice durations. However, the MOS values in this area are all above 3.9 and therefore not critical for our monitoring solution. Thus, we are able to use the model for the influence on a five second voice file for our monitoring strategy.

3.2.3 Modeling the QoE of a SILK Transmission

In the previous section, we concluded that we can use the MOS distribution for a certain level of random loss as a worst case assessment of the user-perceived quality for a QoS to QoE mapping. We now take a closer look on how to analytically describe these distributions.

Figure 3.7 depicts the CDFs of our measured MOS distribution for increasing random loss in bright colors. The dark thin curves show the results, which we achieve with a normal distribution, for which we adapted the mean value and the standard deviation to fit the corresponding measurement results. The graphs mostly overlap, as the mean squared relative error between the measured results and the approximation is below $1.3 \cdot 10^{-3}$ for all fits.

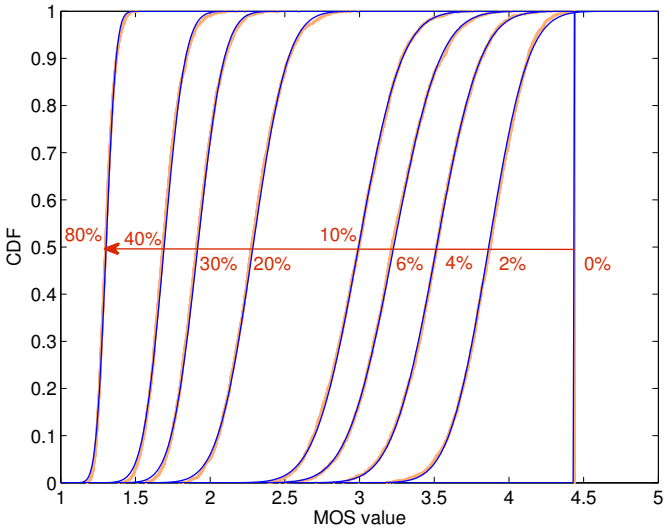


Figure 3.7: Estimating the QoE of random loss with normal distributions parameterized with Equation 3.15 and Equation 3.17

In order to use these estimations for monitoring, we need a model which is able to map the measured loss to the corresponding mean values and standard deviations of the fitted normal distributions. We consider linear, quadratic, exponential, and radical functions as potential fittings. Table 3.5 presents the Mean Squared Errors (MSEs) between the considered approximations and mean values. It is clear to see that the exponential fit outperforms the others. However, in many cases also a radical or quadratic fit can be used to assess the quality.

For modeling the standard deviation, we applied the same models. Again, the fit with an exponential function matches well. However, a quadratic fit is even better for modeling the standard deviation. Note that we could not find any good fit with a radical function. Table 3.6 denotes the mean squared error for the best approximations of the standard deviation. We conclude from the low MSE values that we can model the worst case assessment of the QoE with normal distributions using the exponential function fit presented in Equation 3.15 for the mean value. We can use an exponential or quadratic function fit given in Equation 3.16 and 3.17 for the standard deviation. In all formulas x denotes the application layer packet loss rate including packets dropped due to uncompensated jitter.

$$\mathrm{E}[x] = 2.415\exp(-0.05332x) + 1.328, \tag{3.15}$$

$$\sigma_x = 0.268\exp(-0.01957x) - 0.009182, \tag{3.16}$$

$$\sigma_x = 2.652 \cdot 10^{-5}x^2 - 0.004668x + 0.2563. \tag{3.17}$$

For virtual networks providing a SILK based VoIP service, these results can be used in the following way. If we know the loss probability within our virtual network we can use Equation 3.15 and Equation 3.17 to parameterize a normal distribution. This distribution is a worst case assessment of the QoE perceived by the users. This means, we can infer from this distribution the relative amount of users, e.g., perceiving a service quality lower than acceptable. If we don't

know the exact loss value, but know the upper packet loss limit, e.g. from the SLA definitions, we can conclude which amount of users might experience bad quality in the worst case. Finally, we could also use the worst case assessment to dimension the virtual network or careful overbook it, assuming we can predict the number of customers and their call inter-arrival rate as well as the distribution of the call duration.

Table 3.5: MSE of different functions fitted to measured mean value

function	MSE	normalized MSE
exponential	0.0016	0.0038
radical	0.0126	0.0296
quadratic	0.0159	0.0376
linear	0.0853	0.2011

Table 3.6: MSE of different functions fitted to measured standard deviation

function	MSE	normalized MSE
quadratic	$5.9841 \cdot 10^{-5}$	0.0165
exponential	$7.1722 \cdot 10^{-5}$	0.0198
linear	$2.1937 \cdot 10^{-4}$	0.0606

3.2.4 Reducing the Monitoring Effort Using Sampling

In the previous sections, we proposed a model to assess the QoE of a SILK call. This involves high monitoring effort as we have to monitor each packet stream and look for packets that will not arrive at the destination in time. To reduce the needed effort, sampling can be used. We focus on classical n out of N sampling, which means that we only analyze n random packets out of a group of N transmitted packets.

This sampling method can be understood by looking at the classical urn model. We consider a urn with N balls. If we assume a loss probability of p, this means that $M = p \cdot N$ balls out of these N balls are red. The others $N - M$ are considered to be green. Whenever we draw a red ball from this urn, we consider the packet to be lost. If we draw a green ball, we assume that the packet arrived at the destination in time and without any transmission error. Dependent on the number of samples n, different outcomes are possible. If we draw $n = 2$ balls, we could draw two green, a green and a red, or two red balls. We would interpret the outcome of this results as 0% loss, 50%, and 100% loss respectively. Considering the proposed MOS assessment, this yields:

$$\begin{aligned} P(MOS = x) =& \sum_{i=0}^{n} P(V = i) P(MOS = x | V = i) \\ =& \delta(x - \mu_{n,0}) h(0|N, M, n) \\ &+ \sum_{i=1}^{n} h(i|N, M, n) N(\mu_{n,i}, \sigma_{n,i})(x), \end{aligned} \tag{3.18}$$

where V ist the number of packets considered as lost, δ is the Dirac Impulse, h is the hypergeometric distribution, and N is the QoE assessment based on a normal distribution $\Phi(\mu, \sigma)$, cf. Section 3.2.3.

Considering N=500 overall and M=25 lost packets, the resulting CDFs are shown in Figure 3.8. The brightness increases with a higher sampling resolution, i.e. larger number of sampled packets n, and the red graph denotes the target function $\Phi(\mu, \sigma)$ of the model we want to approximate. If we consider only a small fraction of samples, i.e., $\frac{n}{N} < 1\%$, we can clearly distinguish two areas. In the part right of the target function, representing at least 80% of all sampling outcomes, we did not draw any of the red balls and believe in perfect quality, i.e. overestimates the actual MOS values. In all other cases, we draw some red balls. However, as the sample fraction is small, a single red ball in the sample lets us overestimate the loss probability, which is represented by the part of the curve left of the red target function. For sample fractions $\frac{n}{N} > 10\%$ it is more likely to

approximate the real loss value by drawing n balls. Hence, the QoE estimation is closer to the target function $\Phi(\mu, \sigma)$ and might be used in practice.

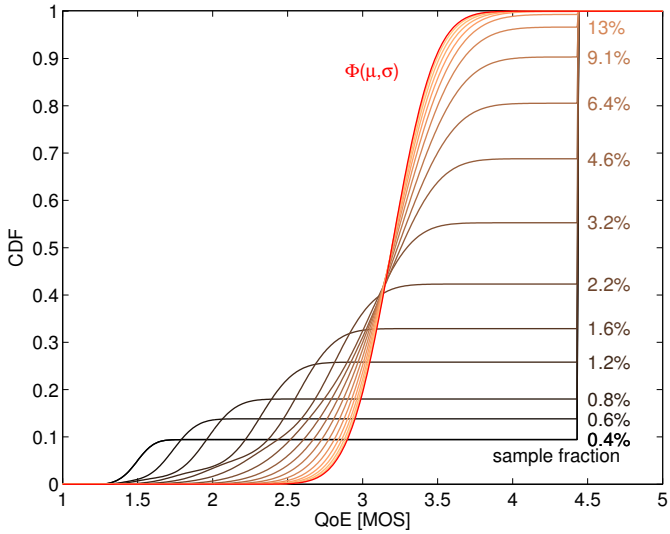


Figure 3.8: Comparison of QoE estimations for an increasing number of samples

To analyze the precision of the n out of N sampling, we consider the difference of the quantiles of the original distribution and the results of Equation 3.18. Figure 3.9 depicts these differences for $p = 5\%$ loss. We see that the absolute error, i.e. the maximum difference between the MOS distribution perceived by the user and MOS assessment based on the loss approximation we inferred from our random sample, is decreasing for an increasing number of samples. If we want to know the relative number of random samples, that provide us with a maximum MOS approximation error, for a given sample fraction and a given loss probability, we need to look at the quantiles of the error distribution. For instance, if we want to know the maximum underestimation of the MOS value, which we will not reach in 99% of all outcomes of the random sampling, we compute the

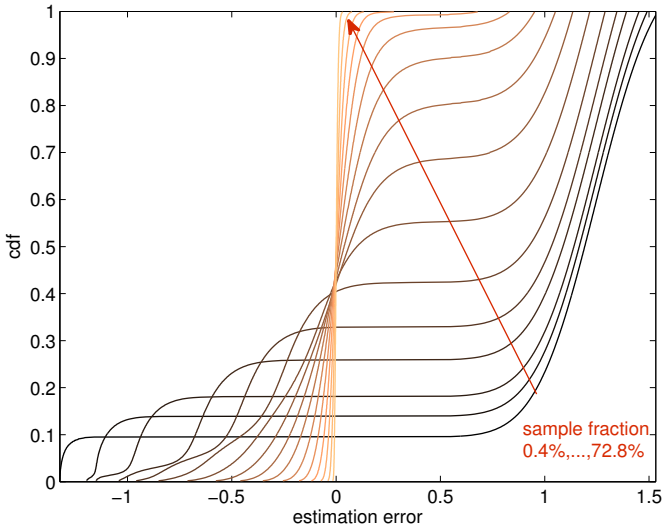


Figure 3.9: CDF of the error caused by sampling

1% quantile of the error distribution, parameterized with the real loss probability and sample fraction we consider. To provide a rough impression of such a error assessment, we present the 1%, 5%, 25%, 50%, 75%, 95%, and 99% quantiles of the error distribution at different sample fractions for 5% loss in Table 3.7. Due to the structure of Equation 3.18, results for other sampling fractions or loss probability can only be calculated numerically.

Table 3.7: Quantiles of the estimation error for different sampling rates

Fraction	1%	5%	25%	50%	75%	95%	99%
0.4%	-1.3234	-1.3111	1.0173	1.1918	1.3361	1.4852	1.5296
0.8%	-1.1156	-0.9734	0.9125	1.1399	1.2853	1.4122	1.4436
1.6%	-0.8796	-0.6854	-0.3643	1.0319	1.2011	1.3156	1.3398
3.2%	-0.6126	-0.4654	-0.1583	0.1005	1.0539	1.1871	1.2103
6.4%	-0.3796	-0.2824	-0.1053	0.0385	0.2593	1.0067	1.0432
12.9%	-0.2076	-0.1534	-0.0613	0.0126	0.0943	0.2993	0.7993
25.8%	-0.0966	-0.0722	-0.0302	0.0038	0.0413	0.1013	0.1495
51.5%	-0.0326	-0.0245	-0.0106	0.0010	0.0143	0.0365	0.0545
72.8%	-0.0127	-0.0096	-0.0041	0.0006	0.0060	0.0153	0.0227

3.3 QoE of MS Office delivered as SaaS

Enterprise applications, nowadays, become more and more complex. Hence, installing, maintaining and updating software on a local PC basis is becoming error-prone and expensive. The solution many companies currently working on is software-as-a-service, i.e. the software is run somewhere in a cloud-computing data center and is delivered to the user over an Internet connection. Thus, the interaction of the user and the software is no longer direct, but happens via a network connection. This distribution model has the big advantage that the company offering the software also takes care of updates and maintenance. Furthermore, the customer does not need to buy the software but is able to lease the software on a pay per use basis. Hence, software-as-a-service delivery platforms increase their share within the enterprise application software market rapidly, cf. [24].

As explained in Section 3.1.1 terminal services are one of the most promising solutions to implement the SaaS concept. While most terminal service architectures had initially been designed for local area networks, today they are used for SaaS applications in wide area networks or over leased lines. In such environments typical network characteristics like packet loss, jitter, or delay play a decisive role. Naturally, the QoS offered by the network directly influences the QoE: In the presence of network disturbances remote applications will no longer react directly to user input like key strokes or mouse movements. Again, it is non-trivial to translate measured QoS parameters into an indicator for the QoE perceived by the end user.

We, therefore, study and quantify the influence of typical network parameters on the QoE achieved in Terminal Server environment implemented via a Citrix MetaFrame environment. In particular, we use a controlled testbed to vary different network characteristics and analyze how these changes in the QoS affect the service quality on application layer.

To the best of our knowledge, there exists currently no standardized tool to assess the QoE of MS Office users in a SaaS environment. To be able to quantify the QoE we measure the time required for specific tasks in dependence on

the network conditions. As we consider the usage of SaaS over virtual networks in a professional environment, it is reasonable to use the duration a user needs to process a task compared to local delivery as a measure for the QoE. A task duration which does not differ from local execution can be identified with a perfect service, as there is no noticeable difference for the user. A prolonged task completion time reduces the work efficiency of the user and therefore correlates to reduced QoE. A user survey related to this problem has been performed by M. Weiß while working on his diploma thesis [133] and was published in [14]. The results show, that this approach to measure the QoE corresponds to the users quality perception to a satisfying degree.

3.3.1 Measurement Setup and Method

In order to be able to capture the QoE of a terminal service user we set up a realistic testbed environment and define a simple metric which quantifies user experienced service quality by measuring the time it takes to complete a specific task under preset network conditions. To obtain stochastically significant results it is necessary to automate the measurement process for obtaining multiple measurement results under the same network conditions. The measurement infrastructure which we used for this purpose is described in the following.

Testbed Environment

In order to emulate a typical Terminal Server architecture used in productive environments, we set up a testbed as depicted in Figure 3.10. For the server side we use two 3.4 GHz Intel Xeon servers with 3.5 GB RAM each, running on Windows 2003 Server standard edition with Service Pack 1. The Windows Terminal Server (WTS) in Figure 3.10 hosts the entire Microsoft Office 2003 product family and runs Citrix Presentation Server 4.0 to make these applications available as SaaS. The second server is set up as a file server and stores the user data. The clients use version 9.237 of the ICA client to access the applications hosted on the server. ICA is short for “Independent Computing Architecture”. It uses TCP/IP, and is

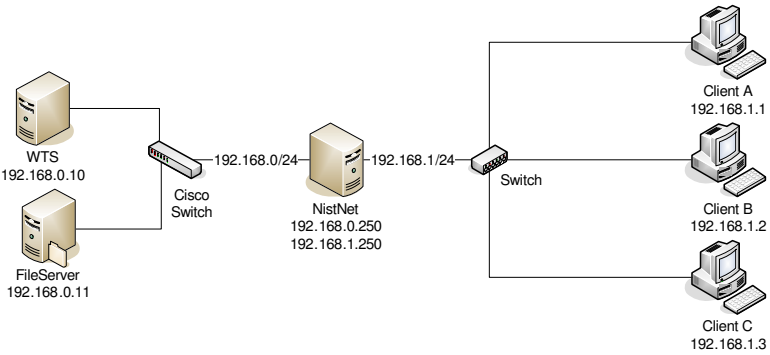


Figure 3.10: Overview of the measurement setup

the proprietary protocol used by Citrix products. On the ICA client, data compression and session reliability were enabled and the color depth was set to the default value of 16 bit. Besides, not further performance-enhancing options were used for these measurements. However, in Section 3.4 we take a look at these options to investigate how adapting the setup to different network conditions can improve the QoE perceived by the user.

To control delay, jitter, and packet loss of the end-to-end connection we put a Dual Pentium III 500 MHz computer with 512 MB RAM running OpenSuSE 10.0 and the network emulator NIST Net 2.1012.c [134] in the middle of the communication channel. On the client side we use Pentium IV 2.6 GHz machines with 1 GB RAM running Windows XP with Service Pack 2. All hosts are connected using 100 Mbit/s links. It has to be noted that we dimensioned the hosts and the network in such a way, that none of these components is a bottleneck and the performance of the applications is only affected by the emulated network conditions.

Organization of the Measurements

The service quality a user experiences under probabilistic network conditions is not deterministic but varies with the quality of the communication channel. In order to repeat measurements multiple times under the same conditions, we implement a control software, which automates the entire measurement process. It consists of a client that controls the measurements and several server components installed on all participating machines which receive simple instructions from the client. An automated emulation run is organized by the client according to the following pattern:

- Configure network parameters on the NIST Net machine
- Start user application, e.g. MS Word, on the terminal server
- Start recording packets
- Execute test
- Stop recording packets
- Collect results

Each emulation run is performed for the duration of an hour. Within this hour the network emulation settings on the NIST Net machine remain unchanged. The client starts the corresponding application and a test consisting of three typical MS Office tasks is performed several times (cf. Section 3.3.1). During the entire time we record all network traffic using WinDump [135] on the client as well as on the server. Please note, that we run our client on a standard PC instead of a thin-client system and our server only handles a single user instead of tenth of users. Therefore, we are able to record the transmission between server and client without interference to the client or terminal server and do not lose packets during the capture. Once the measurement is over, we collect all data and reset the testbed. The tests that are executed in the first five minutes and the last five minutes of the hour are discarded in oder to prevent incorrect data caused by a transient time which might occur at the network emulator.

Automation of User Tasks

Inherently, QoE is a subjective performance measure and is therefore difficult to quantify. We approach this problem by defining an objective QoE metric, cf. [14]: The time required to complete typical MS Office tasks on application layer. To make changes in the duration of a test measurable, we need to design short and well-defined tasks. We chose three popular MS Office products and created three typical subtasks for each of them. MS Word is mainly used for editing text, the main purpose of MS Excel is manipulating and representing data and most MS PowerPoint users create multimedia presentations. We sketch the user tasks mimicking this behavior in Table 3.8.

Table 3.8: Outline of the automated user tasks

MS Word	MS Excel	MS PowerPoint
Typing: Enter text, misspell, delete and retype some words	**Typing:** Enter some values into cells of the spreadsheet	**Slide Design:** Click slide design button to choose a new slide design from list
Scrolling: Scroll several pages up and down using the scrollbar	**Selecting:** Select cells moving mouse from top-left to bottom-right	**Insert Picture:** Insert picture using the file option menu
Menu: Browse menu entries using the mouse	**Diagram:** Create a simple bar chart using diagram assistant	**Zoom:** Change zoom level of the current slide twice

In order to obtain repeatable and comparable measurement results, we need to generate exactly the same user behavior in each emulation run. Therefore the entire user input is done automatically using the open-source tool AutoHotkey [136]. It carries out keystrokes as well as mouse movements and clicks defined in a simple macro language. The tool additionally enables us to measure the duration of each task and to verify that the intermediate steps of each task are successfully completed. For this, AutoHotkey simply waits until some predefined pixels turn to a specific color. Under optimal network conditions, i.e. without any delay or packet loss, the duration of a test is deterministic and will therefore be used to normalize the QoE values in the following.

3.3.2 Analysis of the QoE Results

There is a vast number of network parameters,which all might have a different influence on the QoE of different MS Office applications. Evaluating the performance of all tasks described in Section 3.3.1 under all imaginable network conditions, would result in an infeasible amount of experiments to conduct and data to analyze. As a first step, we, therefore, identify the most critical QoS conditions to concentrate on. Initial experiments show that packet loss (PL) and network delay denoted by the round trip time (RTT) are the two network QoS parameters which mainly influence the QoE. In the following, we apply a design-of-experiments based approach, cf. [137], to identify which of the examined subtasks are particularly vulnerable to such degraded network conditions.

Identifying Critical Scenarios

We illustrate the change of the duration of the different considered MS Word tasks, when both parameters increase from a *low* value (-) to a *high* value (+) in Figure 3.11. In the SaaS scenario, we focus on, it is reasonable to consider PL(-) = 0 % and PL(+) = 2 % and RTT(-) = 0 ms and RTT(+) = 200 ms as typical low and high values of packet loss and network delay, respectively.

For all four possible combinations of delay and packet loss, we evaluate the performance for the chosen applications by repeating the corresponding tasks represented in Table 3.8 for the duration of an hour and measuring their durations. Each point shown in Figure 3.11 for a low or high value represents the mean task durations that were observed, when the parameter given on the x-axis is set to the indicated level and the other parameter is either at its low or high level. To make the results comparable, we show the normalized change in the test duration which is obtained by dividing the measured test lengths by the time it takes to complete the tasks under perfect network conditions. The error bars show the corresponding 95% confidence intervals of the mean values obtained.

Figure 3.11 shows that under the network QoS conditions we consider in this preliminary experiment, the round trip time has a more significant influence on

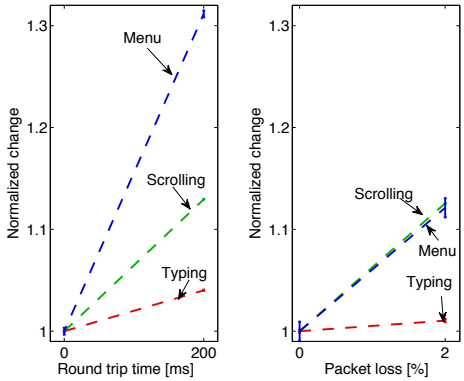


Figure 3.11: Influence of delay and packet loss on MS Word

the duration of the individual MS Word tasks than the packet loss. The Menu test, e.g., takes on average 30 % longer to finish if the delay increases, but only 12 % longer if packet loss is increased. The Typing test is least influenced by bad network conditions: The average test duration increases only slightly. Again packet loss has a smaller influence. The Scrolling test, in contrast, is influenced to a similar extent by both QoS parameters.

Examining the MS Excel tasks lead to very similar results: The Typing test is only slightly influenced by a degrading network quality, while the Diagram test and the Selecting test suffer more. We observe again, that both tests take longer to finish if the network delay is increased than if more packets are lost. Hence, we omit these results here. For instance, the changes for both network parameters were smaller than 1 %. When creating a diagram and selecting cells, a high RTT value leads to an increase of 46 % of the duration of the Diagramm test and to a 34 % longer duration of the Selecting test. In contrast, increasing the packet loss has a more significant impact on the Selecting test which on average takes 42 % longer to finish, while the Diagram test requires only 29 % more time.

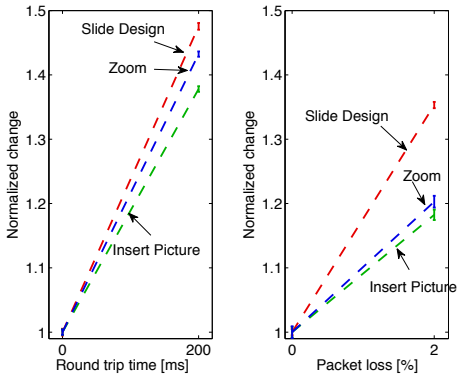


Figure 3.12: Influence of RTT and PL on PowerPoint

Figure 3.12 illustrates the normalized changes for the PowerPoint tasks under increasing delay and packet loss. A comparison with Figure 3.11 reveals that PowerPoint seems to be more sensitive to a decreased network quality than MS Word. One reason for this is that we defined no simple text editing, but rather complex and interactive tasks for PowerPoint, cf. Table 3.8. Text editing under PowerPoint would be affected likewise than under MS Word. Increasing the round trip time has a similar effect on the PowerPoint Zoom and Slide Design Tests as on the Menu test under MS Word. The tasks of inserting a picture into PowerPoint under a heavily delayed connection is even more severely influenced than the Word Scrolling test. In analogy to the MS Word results, packet loss influences the Zoom and Insert Picture tests less significantly than does the delay. This is not the case for the Slide Design test, which needs compared to the other two tests longer to complete if packet loss is increased than if the delay is increased.

According to the performance of the investigated tasks, MS Word is the most robust out of the three tested MS Office applications in terms of decreased network conditions. It has to be noted that we tested more complex tasks under MS PowerPoint and that not all tasks of a specific application are affected to the same extent. To obtain a first overview, we compared mean test durations together with

confidence intervals. The latter illustrated, that the single test durations are not deterministic and often vary strongly, even if the network conditions are stable on a long time scale. Mean values are thus suitable for a first comparison, but do not give any insight into the exact distribution of the measured values. In the next section we will, therefore, have a more detailed look on the different test durations obtained under varying network conditions.

Mapping Network Parameters to QoE Values

We investigate the effect of the QoS on the individual tasks defined in Table 3.8. Therefore, we closely examine the results of about 100 repetitions of each task. We, again, analyze *normalized test durations*, i.e. the observed test duration is divided by the time it takes the same test under optimal conditions to complete. This way we are able to compare the tasks of the different applications to each other. The analysis of the normalized Typing test duration depicted in Figure 3.13 shows that typing in MS Word documents suffers from an increasing delay, but only to a small extend, which confirms the results of Figure 3.11.

The normalized durations are plotted against an increasing delay on the x-axis. For each delay value, we show the results of measurements done for three different packet loss values. Within such an delay group, the packet loss increases from left to right and is visualized by different colors. Observe, that 0% packet loss leads to results that differ only on a time scale unnoticeable to human perception, i.e. beneath 1% of the task execution time. Against, for 2% packet loss the sizes of the IQR boxes become relatively large indicating heavily varying test durations. The minimal and maximal test durations show, furthermore, that a lossy connection may lead to different user experiences: When there is only 1% packet loss, the maximum test duration already increases significantly, whereas in most cases the tests completed much more quickly. Furthermore, note the near-to-linear increase of the fastest observed test, which only depends on the delay.

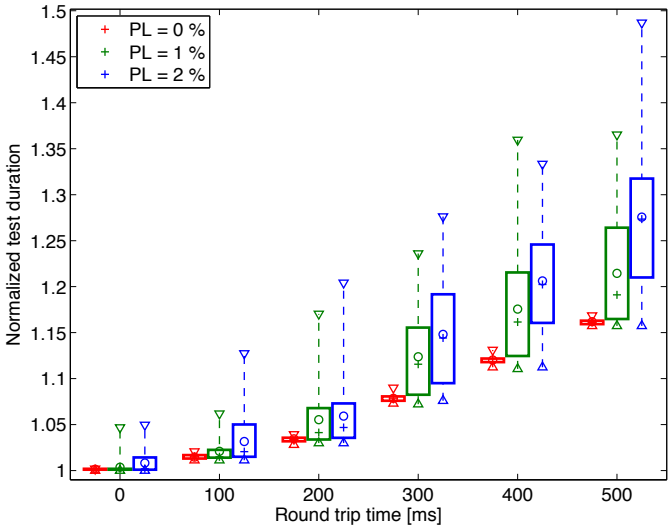


Figure 3.13: Influence of increasing delay on the Typing test (MS Word)

In Figure 3.14 we examine the behavior of the Scrolling test under increasing packet loss. Note that scrolling is affected in a very special way by lossy connections: while under all network conditions, some perfect test runs can be observed, the maximum test duration increases strongly as soon as packets are lost. This corresponds to some users encountering no problems while scrolling, whereas others may notice severe disturbances. This problem is further illustrated by small IQR boxes identical to the minimum going together with very high maxima, which represents a nearly homogeneous user experience except for a very limited number of users. Such situations, which make the test behavior and thus the user satisfaction hard to predict were, e.g. in this test observed for packet loss PL = 0% or 1% and delay RTT = 100 ms.

Typing values into an MS Excel spreadsheet is only very slightly influenced by increasing packet loss or round trip time. In Figure 3.15 we therefore concentrate on the tasks of creating a diagram and selecting cells in Excel. The results of the Diagram test under increasing delay are shown in Figure 3.15 and illus-

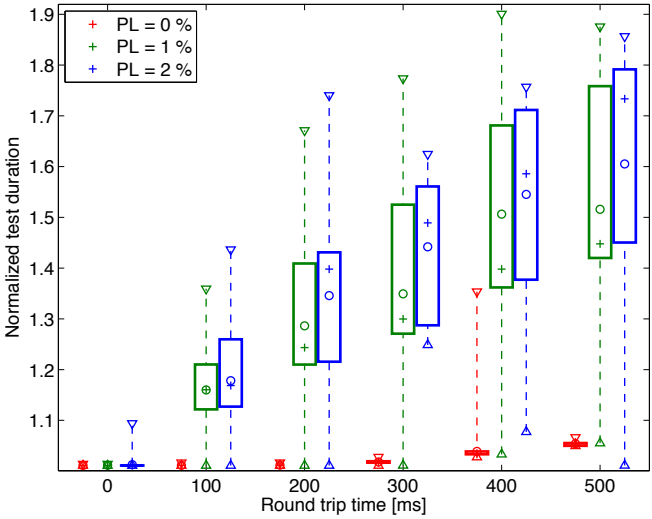


Figure 3.14: Influence of increasing delay on the Scrolling test (MS Word)

trate that the increase of the delay in combination with only a small packet loss probability of 0.5% already leads to a significantly higher and also more varying duration of the Diagram test as compared to the optimal conditions. The experiment furthermore reveals, that a packet loss of PL = 3% results in highly varying and significantly increased test durations. In the scenario with a delay and packet loss of (RTT = 500 ms, PL = 3%) we encountered too many Citrix connection failures to obtain enough values for statistically significant statements. The overall tendency of the test behavior again illustrates that for moderate network QoS conditions the minimum test duration is linearly increasing with the delay, while the individual test outcomes vary heavily, as soon as packets may be lost.

Selecting cells in MS Excel is influenced more heavily by packet loss than the examined MS Word tasks. Figure 3.16 illustrates that the measured test durations were up to 4.5 times longer in situations with very bad network conditions. Also note, that we were not able to obtain enough test values for the scenario

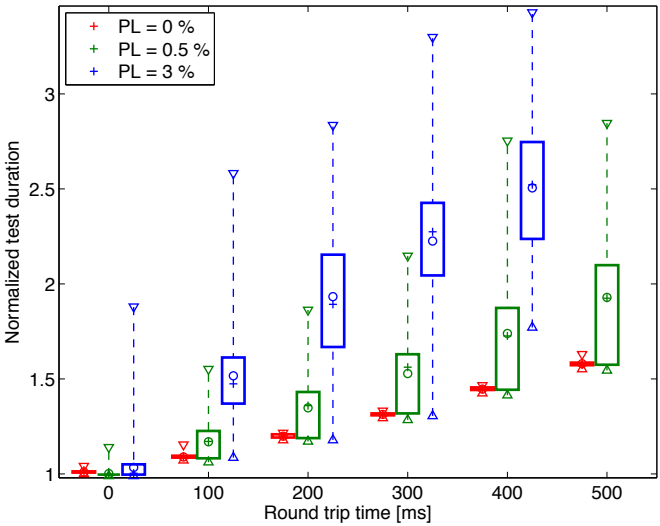


Figure 3.15: Influence of increasing delay on the Diagram test (MS Excel)

(RTT = 300 ms, PL = 2.5%) due to failures of the Citrix connection. The figure, furthermore, reveals that while the mean and the worst case durations of the Selecting test are both increasing with the packet loss, the minimum duration always stays very close to the optimum case. Again, a small number of users will encounter no problems when selecting cells, while others will find this task very annoying or even impossible to complete.

As we saw earlier, cf. Section 3.3.2: Figure 3.11 and Figure 3.12, that the examined MS PowerPoint tasks are more sensitive to increased packet loss and round trip times than the examined MS Word tasks. This fact can also be observed in Figure 3.17, where the depicted mean test durations visibly increase with the round trip time, from left to right over the entire figure, and packet loss within the three bars representing the measurement data for the same delay values. Observe

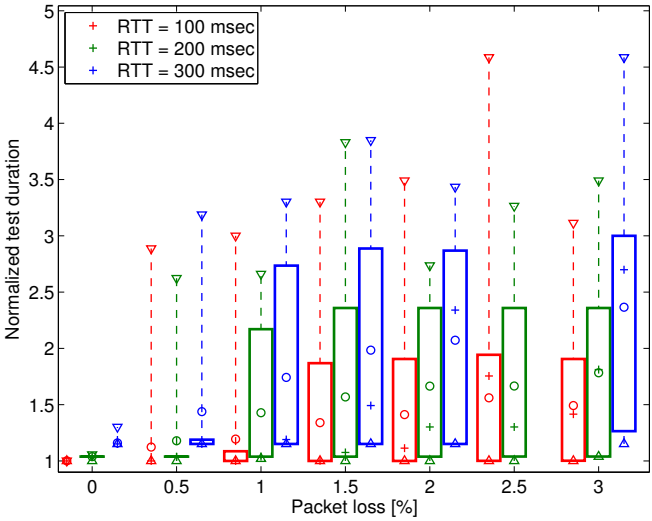


Figure 3.16: Influence of packet loss on the Marking test (MS Excel)

the strongly varying measured test durations, which indicate the sensitivity of this task to the network conditions. For degraded network conditions, the general user satisfaction with this task will thus be very hard to predict. Also note, that even without any delay (RTT = 0 ms), already a small packet loss can sometimes have a significant impact and cause the task of choosing a slide layout to take twice as long as under optimal conditions.

The analysis of the Zoom task represented in Figure 3.18 again reveals, that the influence of packet loss on the test duration is not dramatical as long as the delay is small, i.e. either 0 or 50 ms. As soon as the delay increases to 200 ms, the test duration is affected heavily. A closer look at the measurement results with delay fixed to RTT = 200 ms, unveils that the increase of the mean test duration (depicted by the circle) is mainly due to a small number of very long test dura-

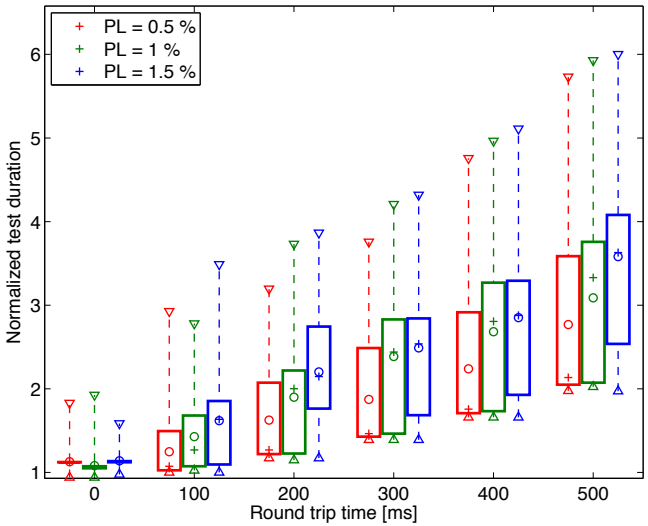


Figure 3.17: Influence of increasing delay on the Slide design (MS Powerpoint)

tions. In the (RTT = 200 ms, PL = 0.5%) scenario, e.g., the cross representing the median is very close to the lower edge of the box indicating the 25% quantile. Thus, the majority of tests need significantly less time to complete than the mean test duration, while a few tests require significantly more time.

The general insight which can be gained from the experiments shown in this section is that a delayed network connection always leads to an increase of the test duration. A lossy connection however, first of all leads to more varying test durations: while in some cases, the tests can still be completed in optimal time, the test duration increases dramatically in some other cases. Moreover, to which extent the absolute test durations increase differs strongly among the different applications and tasks.

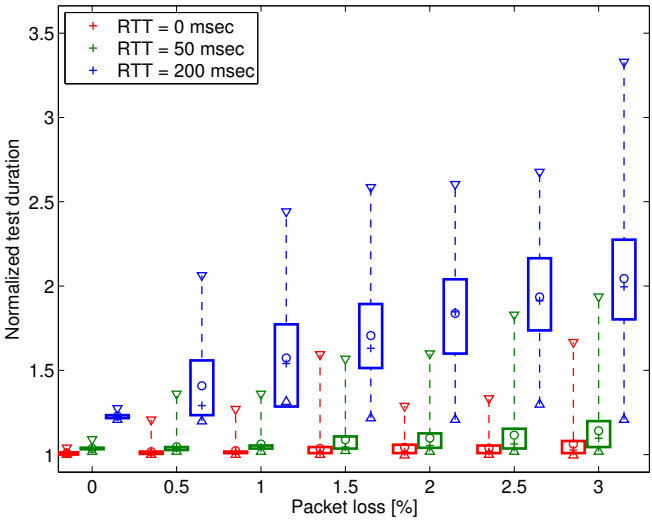


Figure 3.18: Influence of packet loss on the Zoom test (MS Powerpoint)

3.4 Network-Aware Applications

The Internet provides a data transfer service on a best-effort basis only. Although most of the time, all packets sent over the network are delivered to the receiver, there is no guarantee for it. Each router in the network is allowed to drop packets, if it is overloaded. Currently, there are only two wide-spread transport-layer protocols that handle this fact in a completely different way. The TCP-protocol monitors packet drops to achieve fair sharing of network resources by reducing the offered load, whenever packets are lost during the transmission. In contrast, the UDP-protocol hands over the responsibility for lost packets to the application.

For many applications, which do not rely on real-time traffic delivery, TCP provides reliable data transfer on a end-to-end basis. However, if real-time constraints are implied by the application, there are two basic options for the applica-

tion. The application can utilize UDP and ensure the reliability of the data transmission itself. For instance, the data transmission of Skype is based on UDP. If Skype encounters a bandwidth bottleneck, it increases the number of sent packets to toggle the congestion algorithm of TCP-connections on the same bottleneck link. However, as soon as this behavior does not free enough bandwidth, Skype changes the data transfer on the application layer, i.e. modifies the codec to consume less bandwidth. This option can also be implemented when transmitting data using TCP. An example for a TCP-based application changing the transmitted data is a video streaming service that reduces the video quality adapt to a smaller TCP-throughput.

In the SaaS delivery studied in the last section, the Citrix Terminal Server does not change the application layer data due to problems in the network by default. However, there are settings for the terminal server and the client, which can be changed in order to optimize the service according to the current network conditions. In our virtual network setup, the ASP can use these settings to offer his service at a cost-efficient price. As long as the ASP is able to handle short durations of resource shortages in the network by adapting the application layer, the ASP is able to choose VNOs which offer tight-dimensioned networks at a lower price.

In the following we investigate the different settings the Citrix ICA protocol offers to improve the user-perceived quality. In most cases, changing settings on application layer modifies the data transmission behavior. In this chapter we focus on the QoE aspects. A detailed analysis of the effect of these changes on network layer is presented in the next chapter.

3.4.1 SaaS Application Layer Settings

With the increasing popularity of terminal services, Citrix Inc. integrated options in their products to cope with WAN conditions.

For our measurements, we use the same measurement infrastructure, methodology, and automation of user tasks as described in Section 3.3.1.

In the following we study the most important options Citrix introduced for wide area network optimizations: Color Depth, Input Buffer settings, and the Speedscreen Latency Reduction option. They all have a different purpose and special features on which we detail in Section 4.2.

Color Depth

We first investigate the option to configure the color depth of the remote application. The color depth can be adjusted to 8 bit, 16 bit, and 24 bit. Changing this parameter will obviously alter the representation on the client screen. Our measurements revealed that besides a different color palette, there are no changes on application level that are statistically significant. The task completion times stay the same for all color depth settings. Only the colors on the screen changed. Even on the network level, there was no significant difference measured. The compression of visual data prior to the transmission is obviously able to level the differences in the color information. It is therefore reasonable to always use 24 bit color depth. Hence, we focus on the other features in the following.

User Input Buffer

As we have shown in [15], a Citrix client tries to maximize the responsiveness by sending all user input to the server as fast as possible. With the Input Buffer option it is possible to make the Citrix client collect user input information at the client and send the information only after some predefined intervals, which can be independently adjusted for mouse and keyboard input. By default these values are 50 ms for mouse input and 100 ms for keyboard activities.

Speedscreen Latency Reduction

The Citrix Speedscreen Latency Reduction (or shortly Speedscreen) option is promoted to be a very powerful tool for improving the QoE. This technique anticipates the server response at the client. Without Speedscreen, the client acts as a dumb terminal and merely displays the pixels sent by the server. Enabling

Speedscreen mainly changes two things. First, the client supports a higher degree of interaction by sending packets more frequently to the server. Second, a client with enabled Speedscreen option tries to 'anticipate' the servers response. For instance, if the users types a character on the keyboard, the client will render the character on the screen even though the servers response has not yet arrived. While this sounds great in theory, there are limitations to what is possible in practice. For example in MS Word the typed character is shown on the screen, but often in a different font. Only after the reception of the servers feedback, the font printed on the screen changed to the intended one. Our experience with human probands shows that the rendering of an incorrect font on the screen is acceptable for the user as long as the correct letter is shown. However, for our evaluation method using Autohotkey, this behaviour is critical. A character rendered with a different font changes the pixels on the screen that change their color. Further investigations revealed that the fonts are always rendered in the same way when using MS Wordpad. Hence, we adapted our tests to use MS Wordpad instead of MS Word for all measurements described in the following. The actions performed within each test remain unchanged.

Combining Speedscreen and Input Buffer

From our earlier descriptions it becomes clear, that the last two mechanisms have contradicting influences, which might neutralize each other: While the Input Buffer is supposed to decrease the QoE by delaying server responses, the Speedscreen Latency reduction should lead to an increase of the QoE. Therefore, we combine these two mechanisms and examine whether it is possible to improve the user-perceived quality or if this combination should be avoided.

3.4.2 Optimizing the Quality of Experience for SaaS

In this section we report on the most interesting results of the Citrix settings, which we observed in our testbed environment. In particular, we consider the Input Buffer, the Speedscreen Latency Reduction, and a combination of both and analyze the performance of these settings.

Input Buffer

The Input Buffer directly influences the frequency at which information is sent from the client to the terminal server. Therefore, the response should be delayed when using this option. Figure 3.19 compares the influence of enabling the Input Buffer with Citrix default settings on the QoE perceived by the user. We show the median of the test durations for increasing round trip times, while the error bars visualize the inter-quartile range of the test completion times. The two solid plots at the bottom depict the QoE measured for a Citrix client with disabled and enabled Input Buffer in a lossless network. The plot for the client with the activated Input Buffer reveals a slight increase of the test completion time. This is caused by the small delays occurring for each time a packet is delayed in the Input Buffer. These minimal delays accumulate to a measurable increase of the time it takes to perform the test tasks.

The QoE measurements in an environment, where the network randomly drops packets, reveal different effects. The upper two dashed plots in Figure 3.19 depict the measurements affected in a network with 1% packet loss. In this case, the plot for the activated Input Buffer is below the one for the task duration times of a Citrix client without the Input Buffer, i.e. the Input Buffer improves the QoE in case the network drops packets. Recall that under TCP, whenever a data packet is lost the receiver will have to wait for at least two more packets to arrive in order to trigger the fast retransmission mechanism. Due to the fact that the packets are send frequently a TCP time-out will not occur. The more packets are sent (no buffer) within a given period of time, the more packets can be lost causing the user to wait for retransmissions, thus slowing down his workflow. Hence, in the

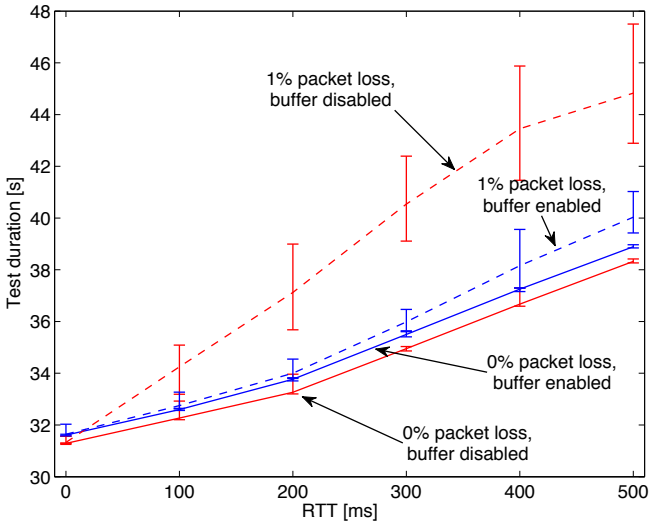


Figure 3.19: Influence of Input Buffer on the user-perceived QoE

presence of packet loss in the network it is better to gather user input and transmit less often.

Speedscreen Latency Reduction

As mentioned before the Speedscreen Latency Reduction is a technique, which is supposed to improves the responsiveness, by rendering of the applications response on the client before the servers response is available. Figure 3.20 depicts the influence of Speedscreen on the QoE for different network scenarios. The two curves at the bottom represent the task duration measured with and without the Speedscreen option for an increasing delay in a lossless network. Under perfect network conditions, i.e. no delay and no packet loss, the performance for both

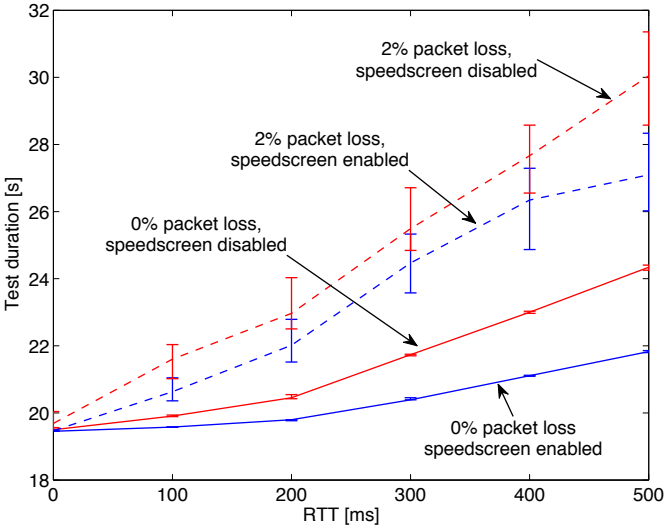


Figure 3.20: Influence of Speedscreen on the user-perceived QoE

Citrix client configurations is the same. This is caused by the fact that the servers response is directly available in this case. With increasing round-trip times the tests are completed faster with the Speedscreen enhanced Citrix client than for the client without Speedscreen. Thus, the Speedscreen option can improve the responsiveness of the application by anticipating the server response. However with round-trip times beyond 200 ms the test duration with Speedscreen also increases. This depicts the evident fact that even with Speedscreen enabled the client is not independent from the server. From time to time it has to wait for the server response and thus the test completion time increases. The upper two dashed curves in Figure 3.20 shows that the Speedscreen option also improves the task completion times in situations with packet loss as expected.

Combining the Input Buffer with Speedscreen

The Input Buffer and the Speedscreen option can increase the perceived QoE. Thus, the question arises whether combining them can further improve the QoE.

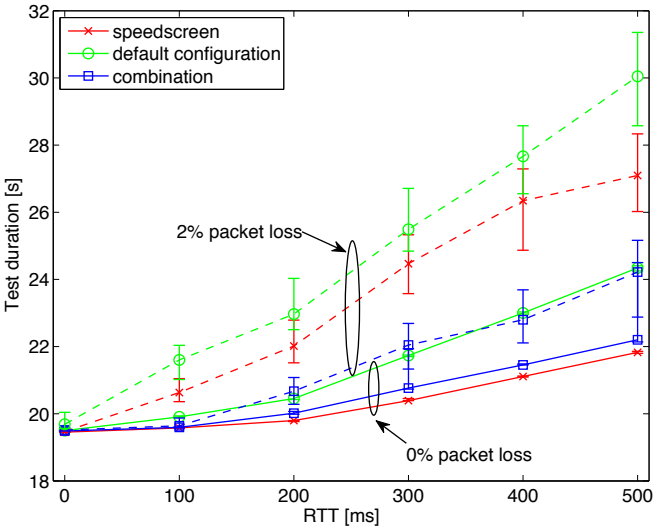


Figure 3.21: Influences of combing Speedscreen with Input Buffer on the QoE

If we take a look at the QoE metrics shown in Figure 3.21 we see how the combination of Input Buffer and Speedscreen affects the test duration. The solid lines depict the QoE measured in a lossless network. Observe that the Citrix client running with a combination of Speedscreen and Input Buffer provides better QoE than the unmodified client. However, it is a little bit slower than the Citrix client running with Speedscreen only. This result is reasonable, as on the one hand the Speedscreen accelerates the representation on the screen and therefore the user gets more direct feedback. On the other hand, even the Speedscreen enhanced

client is not independent from the server, and thus the additional delay introduced by enabling the Input Buffer slightly reduces the improvement of the QoE.

Under network conditions with packet loss the effects on the QoE are different. The dashed lines in Figure 3.21 depict the test durations measured in a network with 2% packet loss. In this case a combination of Speedscreen and Input Buffer clearly outperforms the other two options. This improvement is based on the user feedback enhancement of Speedscreen as well as on the fact that the Input Buffer reduces the mean time the client has to wait for the server response in a given time interval under loss conditions. Considering these results, we come to the conclusion that combining Speedscreen and Input Buffer can improve the responsiveness in congested networks.

3.5 Lessons Learned

In this chapter, we focus on the interface between the ASP and the VNO. The main challenge here is the is the translation between these two roles and their completely different view of the network. The ASP role is mainly interested in the QoE the users perceives. In contrast, the VNO focuses on the data transport network and therefore considers QoS. The aim of this chapter is, therefore, to define a mapping of these views for two different applications: a Skype SILK VoIP service and a MS Office service using the SaaS paradigm.

We provide a detailed analysis of the Skype SILK codec and compare it to its predecessors iLBC and GSM using an analysis based on the PESQ tool. We analyze bulk and uniform loss by applying error patterns directly to the encoded VoIP frames. From the results, we learn that SILK provides a better QoE in all considered cases. Furthermore, we reveal that for the SILK codec the impact of random loss is more severe and that distributions modeling these results can be used as a worst case assessment. Our study of different speech durations reveals that equivalent loss percentages lead to a stronger degradation of the perceived quality, if they are applied to shorter speech transmission. Thus, we fit models for short speech transmissions, in order not to overestimate the user-perceived

quality. We model these worst case assessments using a normal distribution and present formulas to derive all necessary parameters from the loss measured in the network. Finally, we demonstrate that sampling can be used to decrease the monitoring effort. We provide numbers for the precision of different sampling rates for some exemplarily chosen loss percentage and provide formulas to compute these results for other loss rates, so that the sampling rate can be chosen to fit the needed precision of the virtual networks monitoring system.

Measuring the quality of MS Office applications delivered as SaaS is a challenging task. As no full reference tools for automated QoE assessment are available for SaaS, we focus on a simple but effective measure for the QoE of the user. The normalized task completion time reflects the additional time a user has to spend on a task when using SaaS Office applications and is, thus, able to reflect the impairments of a SaaS user. Our studies reveal how far the different tasks are influenced by different network parameters.

Further studies of application parameters offered by the Citrix ICA protocol showed, that it is possible to adapt the service to the current QoS in the network. Our results can be used to optimize the user-perceived QoE in short durations of network overload and thus offer the service using tightly dimensioned cost-efficient virtual networks.

Besides this, our studies illustrate the importance of considering the entire distribution of measured QoE results instead of only concentrating on mean values, as the statistical characteristics of the distribution provide valuable insights as well and need to be considered when providing services on a carrier grade level.

4 Network Planning and QoS Monitoring

Don't pay the ferryman, until he gets you to the other side.

Chris de Burgh

Many advantages of network virtualization are based on the idea of separating different application classes into distinct parallel networks. These networks are tailored to fit the application and enable QoE monitoring. In order to build a network for a specific application service, it is necessary to understand the needs of the application. In Chapter 3, we study the influence of QoS on the QoE perceived by the user. In this context, we analyzed which network parameters influence the application, if it is already running. In this chapter, we focus on what is required to run the application in the virtual network, i.e. how to provision the network in terms of necessary resources, and monitoring tools, which are able to measure that the virtual network is performing as expected.

In the past, dimensioning a network or a end-to-end connection has often been synonymous to over-dimensioning. After monitoring an application for some time, the maximum peak bandwidth is determined and used to characterize the bandwidth requirement of the application. This method is a basic methodology that guarantees sufficient bandwidth. However, it is not very cost-effective, as these peaks might be rare and exceed normal bandwidth usage by far. In order to dimension networks more efficiently, it is necessary to understand the resource

requirements of an application in more detail. Creating a proper traffic characterization for an application is the first step towards resource efficient network planning, since it describes the different aspects of the applications communication behavior. Together with user profiles, which describe the frequency, the duration, and the way the application is utilized, it is possible to dimension the network to provide a high QoE and reserve only necessary resources.

We provide such a traffic characterization for MS Office provided as SaaS, for which we studied the QoE-QoS-mapping in Section 3.3. These applications are business-critical and therefore need to be provided in high quality. Our findings in Section 3.3.2 and Section 3.4.2 reveal that this quality requires a low loss percentage and short delays, i.e. a high priority connection, which makes bandwidth for these applications expensive. To provide the characterization needed for dimensioning, we analyze the bandwidth usage of different MS Office applications in both directions, i.e. from the client to the server and vice versa. Additionally, we characterize the effects of the QoE improving options investigated in Section 3.4. This allows us to understand the correlation of the QoE improvements and their costs on the network layer.

Operating a network with a predefined QoS level on an end-to-end basis, it is necessary to observe all network entities and monitor the data transmission. With virtual networks additional challenges occur. Besides technical problems, which we address in the next section, multi-domain scenarios imply additional complexity. In the multi-domain case, the virtual network operated by the VNO is composed of interconnected network parts from different PIPs. It is, therefore, not only necessary to precisely measure the network performance in the virtual network. Furthermore, the VNO needs to accumulate different PIP measurements provided over the monitoring interface, which we describe in Section 2.2.4. This way, the VNO is able to compare the data provided by the PIPs with its own measurements and identify problems resulting from combining the PIP networks.

We address these challenges by exemplarily examining the precision of a standardized network measurement system implemented directly on the router. Additionally, we discuss how measurements of network parts provided by the PIPs can

be accumulated to calculate values for the complete network. The VNO compares these results with its own measurement to identify problems in the end-to-end QoS and SLA breaches.

The remainder of this chapter is organized as follows. We introduce the technical background and discuss related work in Section 4.1. Afterwards in Section 4.2, we characterize the bandwidth consumption of Office applications in the SaaS scenario and study the network layer costs of the QoE-improving options introduced in Section 3.4. To monitor the QoS needed for a network supporting these demanding applications, we investigate the precision of a router based QoS measurement tool in Section 4.3 and provide formulas to accumulate these measurements into a end-to-end view in Section 4.4. Finally, we conclude this chapter in Section 4.5 with lessons learned.

4.1 Background and Related Work

In the following, we deal with two specific areas needed for network planning and QoS monitoring. First, we introduce the term traffic characterization and discuss related work providing traffic models. Second, we discuss the problems of QoS monitoring in virtual networks and propose extensions of existing technologies to allow QoS measurements within virtual networks.

Traffic Characterization

The first duty of measurements for traffic engineering is to provide a *Traffic Characterization* (TC), c.f. [138, p. 213]. In this context, TC means a statistical description of the traffic pattern on short, mid-term and long-term time scales. Hence, TC allows to determine the data volume traversing the network based on the number of end-points, to estimate the consumed bandwidth, and to forecast the required network resources in the future, c.f. [138, p. 213f]. A second step after traffic characterization is to derive a traffic model, which can be used for analytical planning and simulation.

In the following, we discuss traffic characterizations and models for different applications and provide details on exemplary contributions in the related work.

Due to the overwhelming dominance of web traffic in contemporary WANs, most source traffic models focus on this fraction of the utilized bandwidth. So does the stochastic source traffic model for HTTP traffic proposed by Cao et al. [139]. The authors analyze packet traces from two links connecting medium sized organizations to the Internet. A more general approach is proposed by Staehle et al. [140], who establish a source traffic model for realistic wireless simulations. For this purpose, they introduced a single user traffic model which considers Email, HTTP, FTP, and WAP traffic.

Dainotti et al. [141] propose a hidden Markov model to describe traffic sources at packet level. Besides SMTP and HTTP, they claim to also model instant messaging platforms with reasonable precision. Their analysis shows that this model is able to statistically match the results of real world traffic measurement and also produces synthetic packet series, which model the mutual and temporal dependencies. Hence, with their model they are able to do short-term traffic prediction from measurement data of real traffic sources.

A more general model for aggregated network traffic is provided by Li and Lim [142]. They propose to use a generalized Cauchy process to model self similarity features of the internet traffic. In contrast to a model employing fractional Gaussian noise, the generalized Cauchy process allows to independently choose the fractal dimension and the Hurst parameter.

Peer-to-peer applications produce a completely different traffic pattern, as they are edged-based services. Each peer application is acting as a client and a server at the same time. Therefore, these applications create a different kind of traffic. Several peer-to-peer applications are analyzed by Basher et al. [143]. They study the traffic pattern on an aggregated stream level as well as on an flow level and compare Bittorrent to Gnutella flows. To describe the differences, the authors compare distributions of flow sizes, flow inter-arrival times and flow durations.

A deep analysis of Skype traffic is presented by Bonfiglio et al. [144]. They study different voice and video codecs used by Skype and investigate how Skype

adapts to the network layer. Furthermore, they characterize the peer online times, the geographical distribution of peers as well as the signaling traffic caused by Skype. They provide a good characterization of Skypes network behavior and describe, which features have to be considered when planning a virtual Skype network.

Menth et al. [145] describe a rather general approach. The authors consider codecs with periodic packet transmission as well as codecs, which use silence suppression. The transmission behavior of the latter is dependent on the on-off-process, i.e. modeling the phases of speaking and listening in a conversation. Being able to model this behavior in voice transmissions improves the accuracy of the model in terms of fitting the autocorrelation function of the packet transmissions.

Another service, which is becoming more popular in the recent years, is video streaming. An approach towards the modeling of video traffic is presented by Lazaris et al. [146] for modeling multiplexed video-conference traffic with MPEG-4 encoding. The authors choose different distributions to fit the measured data and concluded that a Pearson type V distribution achieves the best results. However, none of the used distributions provided a high accuracy, since none is able to simulate the high autocorrelation in the frame structure. Hence, Lazaris et al. propose to investigate a discrete autoregressive model in future work based on the results of their measurements.

Dai et al. [147] approach the problem of modeling MPEG-4 and H.264 video traces using wavelets. In their work, they are able to show that besides successfully modeling intra-GOP (Group of Pictures) and inter-GOP correlation, their approach is also capable of fitting the temporal burstiness, the frame size distribution, and the autocorrelation of the network traffic. They claim that their method also possesses a lower complexity than other techniques.

Zink et al. [148] are concerned with the infrastructure of video-on-demand delivery and the impact of local popularity distribution. As an example, they investigate Youtube traffic in their campus network. After creating models for popularity and the distribution facility, they simulate a network with further en-

hancements, i.e. local caches and peer-to-peer distribution. They show that these enhancements can reduce the access times and occupy less network resources.

Another application field for virtual networks is the area of virtual online communities. One famous example, which even attracted companies to create virtual branch offices, is second life. Antonello et al. [149] analysed the traffic caused by this application and revealed that the bandwidth depends on the actions of the user, as well as the virtual area the user was situated in. Furthermore, they provide a synthetic traffic model, which can be used to dimension QoS networks for second life and other similar virtual reality applications.

The number of ASPs hosting office applications for remote users is growing, but the percentage of bandwidth consumed by this type of service remains negligible. Therefore, little work has been dedicated to analyzing the characteristics of traffic caused by SaaS. One of the few studies in this area has been performed by the Tolly Group [150] who evaluated the usability of Microsoft PowerPoint via WAN. They investigated the consumed bandwidth and completion time of a common PowerPoint operation executed on a machine running Citrix MetaFrame XP client software accessing a server hosting the corresponding Presentation Server.

Semoens et al. [151] follow a similar approach, which we use in our work, but analyze Novel clients as a SaaS implementation. They also study how local buffering of user input affects the responsiveness of the application. These results are extended towards an automated control, which adapts the buffer strategy to the round trip time of the network. They do not consider task completion times but instead focus on single interaction responses. The target is to save bandwidth while keeping the response times below 150 ms. From our results presented in Section 3.4.2, we support that this approach is able to reduce the necessary bandwidth. However, our investigations also show that this approach is affecting the user-perceived quality as well and should not be implemented without an acknowledgment from the user.

The work of Humar et al. [152] examines another feature of SaaS. They analyze the session duration and session inter-arrival times. A combination with our measurements, which we present in Section 4.2, can be used to dimension vir-

tual networks for SaaS, in cases in which the SaaS usage is not determined by periodical work processes but unsteady access to the application.

Quality of Service Monitoring

QoS Management and QoS Monitoring are well-researched topics. Basic architectures and implementations are described in books by Subramanian [42] or Räisänen [138].

In general, there are different options to be considered for measuring the QoS of a network service. A possible method is a passive measurement, which means that at some point of the network data is collected and analyzed. The big advantage of this methodology is that the system is only observed but practically not affected by the measurement. However, with a passive measurement many QoS parameters are hard to assess. Although bandwidth can be computed easily, the estimation of round trip times and packet loss is only possible with protocols in which corresponding messages are sent in both directions. For instance, TCP is a protocol that can be used to estimate loss and round trip time with a passive measurement, cf. [28]. However, parameters like jitter are only accessible with deep knowledge of the timing aspects of a protocol. Against this, active measurements, which introduce traffic to probe the network, are a rather easy method to estimate the QoS of a network service at the moment of the measurement. All QoS parameters, i.e. one-way delay, jitter, packet loss and bandwidth, can be quantified by an appropriate active measurement. The big disadvantage is that the measurement directly influences the system and it is therefore not possible to estimate the QoS of the system without the additional measurement data. In our case this 'disadvantage' can also be used to analyze the effects of a new network flow on the network, if the traffic characteristic of the flow is known and can be emulated by the measurement. Other measurement strategies combine passive measurements at different points of the network and exchange the data, cf. [17]. The advantage of these methods is that, although the exchanged data has only a small influence on the system, the same QoS parameters can be examined as with an active measurement. However, the exchange and correlation of corresponding data is

complex and the system is not easy to maintain. Hence, depending on the use case and the QoS parameters, which need to be measured, different monitoring tools are used. In the recent years many of these tools have been implemented and can be applied in practice. Due to this fact, the work in the area of QoS systems diverges to a number of solutions for special purposes in the last decade.

Yu et al. [153], for instance, discuss a method to select web-services in a service-oriented architecture to create complex services satisfying application-specific QoS requirements. In their contribution, the authors consider run-time, utility, price, and availability of atomic web-services. They use a broker architecture for the composition of the service and solve the computational problem using a heuristic approach.

Huang et al. [154] report on their work on an adaptive QoS management for wireless multimedia networks. They classify applications into different service classes and enforce QoS by using admission control and resource reservation. The evaluation is performed by means of analytical models and simulation and demonstrates that the system is able to guarantee all requirements specified.

Another important point in monitoring the QoS, which is often not discussed, is the precision of the used measurement methods and tools itself. Arlos and Fiedler [155] define a method to evaluate the precision of a measurement tool recording the arrival times of packets at a network interface. Furthermore, they evaluate different hardware and software solutions and present an overview on the precision of commonly used measurement tools, e.g. tcpdump. Fiedler and Arlos [156] also publish results on the accuracy of delay measurement tools, namely ping and J-OWAMP. A best practice guideline for performing accurate network measurements is provided by Arlos in [157]. He describes a framework that is capable of generating, measuring, analyzing, and visualizing traffic and discusses, which steps are necessary to achieve credible measurement results.

Network virtualization adds new complexity to the task of QoS monitoring. The fact that virtualization is hiding the technology, on which the resources are offered, introduces new options from a functional point of view, but also challenges from the measurement perspective. Many measurement tools are imple-

mented assuming some basic features of the data plane. For instance, bandwidth measurement tools relying on packet trains often assume that the packets are transmitted as they are generated. If the packets is fragmented and defragmented or are aggregated into jumbo frames this might cause unrealistic results and high relative measurement errors, cf. [158]. Another example is described by Arlos and Fiedler in [159]. They show that the packet size used for measuring the one-way-delay in 3G mobile network influences the outcome of the measurement. From their work, we know that in case of a 3G mobile network the packet size should be at least 250 byte. In case of a virtual network, it is not clear whether a link is implemented using a 3G mobile link, a 2.5G mobile link, WLAN, or fixed network. This means that for network virtualization, one option is to build measurement tools, which provide reliable results independent of the underlying technology. Otherwise, we need to trust in the measurements the PIP is able to perform on the physical substrate and the performance guarantees of the hypervisor. Santos et al. [160] have build such a physical layer monitoring solution for the OneLab2 experimental facility and report on their experiences. Their measurement system adds specialized hardware to each location of the distributed testbed and reduces the load by hash-based sampling of the measured packets. Additionally, the system implements multi-client capability by providing each owner of a virtual experimental network with only his measurement data. In a professional environment, we cannot assume the VNO to trust the PIPs, since both have conflicting economic interests. Hence, we need measurements each role can perform on its own, i.e. a VNO needs to be able to perform reliable QoS measurements within his virtual network.

Another challenge in virtual network measurement is caused by the fact that with many virtualization techniques the virtualization layer adds additional delay in the packet processing, which is hard to measure. The challenge is that only the time when a packet arrives at the physical interface can be exactly determined. This can be implemented in hardware logic, which records the time the packet is fully received. Afterwards, the packet will be handled by the hypervisor, which transfers the packet to the virtual interface of the guest. As soon as the guest is

scheduled for some processor cycles, the guest Operating System (OS) becomes aware of the packet. Whiteaker et al. [56] measured the duration between the host OS and the guest OS detecting the packet and reveal that this time span exceeds 100 ms in some cases. However, this delay was measured having direct access to the host system. If a virtualized guest wants to determine a time stamp, there are basically two options. The first option is to rely on the virtualized system clock, which might differ from the hardware-clock of the host and thus is not reliable. The second option is to access the hardware clock of the host system. This direct hardware access is either implemented in a way the task can be interrupted by other tasks or it is atomic. To be interruptible means that the precision of the time stamp cannot be guaranteed, since other tasks could interrupt the execution and the hardware access is delayed. If it is implemented to be atomic, i.e. not interruptible by other tasks, the time request is an computational expensive task, since no other task can be performed on the same processor and get read access to the clock during this access.

A possible solution for these problems might be to integrate some measurement functionalities into the hypervisor. This functionality needs to provide packet templates, which the hypervisor is able to identify. As soon as the hypervisor receives such a packet, it instantaneously performs all the tasks related to the measurements. The VNO needs to fill this template with further data that allows the packets to be transmitted within his virtual network and the protocol stack the VNO specified. This solution allows then to conduct measurements from any point to any other point within the virtual topology. The problem of using a device of the PIP to perform measurements for the VNO can be solved by integrating trust chains into the hypervisor and the hardware, e.g. as it is implemented in PCs for blue-ray playback.

Currently, virtual routers are developed in research and standardization is still some years away. Hence defining and testing such a system is practically not feasible yet. However, similar systems exist in our current routers. For instance, the IP SLA feature of Cisco routers implements a series of tests offering different active network measurements, which can be conducted between each two routers

with the same feature set. Hence, we analyze the precision of these IP SLA measurement in Section 4.3 to verify that such a router based measurement achieves sufficiently accurate results and is a candidate to be further developed to provide measurements in virtualized systems.

4.2 Traffic Analysis for Office provided as SaaS

We, again, consider Microsoft Office as SaaS as an example for our traffic charaterization. Recall that we focus on a concrete implementation using the Citrix Terminal Server approach, since it is a widely implemented solution applied by many companies. In order to characterize the network traffic we, first, study how a unmodified client-server-connection behaves. We investigate several user groups and different applications. Second, we address the QoE improvements investigated in Section 3.4 and analyze how these affect the network layer. Although these options are triggered by the ASP, the VNO needs to know the consequences on the network layer and agree with the ASP in which cases which option is triggered.

4.2.1 Measurement Setup

In order to measure and characterize the traffic of a SaaS MS Office application, we use the same measurement setup as described in Section 3.2.1.

Recall, this setup consists of a server farm which is accessed by up to three clients and a network emulator running NistNet [134], which enables us to alter delay, jitter, and packet loss of the end-to-end connection, cf. Figure 4.1. The Terminal Server as well as the corresponding file server are Intel Xenon 3.4 Ghz machines with 3.5 GB RAM running Windows 2003 Server in the standard edition with service pack 1. The clients are Pentium III 2.6 Ghz machines with 1 GB RAM running on Windows XP Professional with service pack 2. The Windows Terminal server hosts the entire Microsoft Office 2003 product family and runs Citrix Presentation Server 4.0 to make these applications available for remote

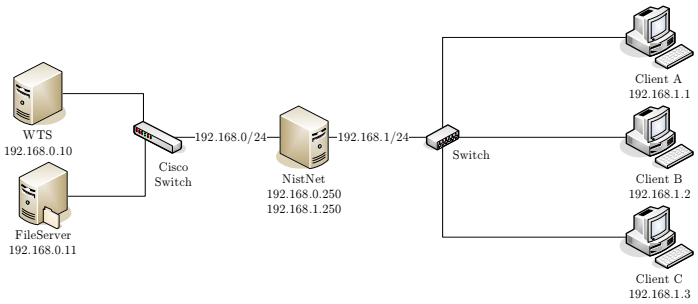


Figure 4.1: Overview of the measurement setup

users. The clients use version 9.237 of the ICA client to access the applications hosted on the server. This setup enables us to generate and measure SaaS caused traffic in a controlled environment.

Our emulation experiments are set up as follows. At first the client opens a Citrix session and starts the corresponding application. It then performs different tasks which are typical for MS Office users, cf. Section 3.3.1. To automate this procedure, we use AutoHotkey [136], which is able to carry out keystrokes as well as mouse movements and clicks according to a simple macro language. This way, we are, e.g., able to emulate different typing speeds in MS Word or the insertion of pictures into a MS PowerPoint slide. To capture the corresponding processes on network level, we record packet traces using Windump [135] on both the client and the server machines, which are then analyzed offline.

4.2.2 Measurement Results

The traffic generated by Office as SaaS on packet level depends on both the application and the behavior of the user. Intuitively, the activity of the user directly corresponds to the amount of consumed network bandwidth. To verify this assumption, we consider different MS Office applications and characterize the source traffic created by different types of users, where we distinguish between light, normal, and power users in the following.

Analysis of Typical MS Word Tasks

To obtain an initial qualitative understanding of MS Office as SaaS traffic, we emulated typical tasks performed by a MS Word user like typing, scrolling, or selecting menu entries. During the typing test, the user continuously types text and corrects some misspelled words. It then scrolls the document using the scroll bar and selects some entries from the menu bar. Figure 4.2 confirms that the different tasks require different amounts of bandwidth on network layer. Note that scrolling consumes the most bandwidth as large parts of the screen need to be refreshed.

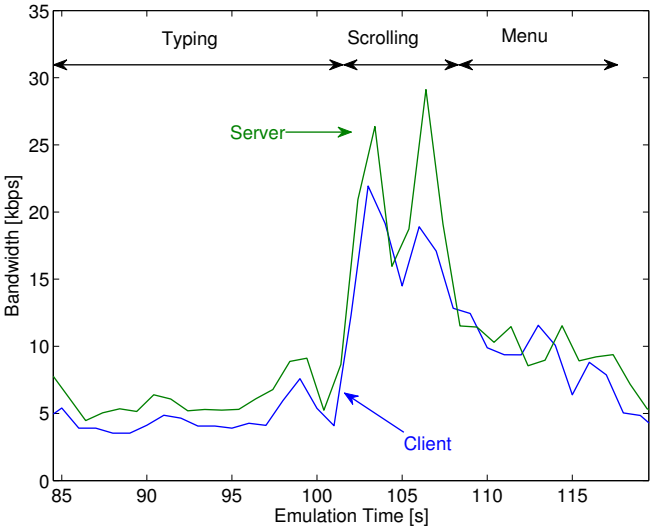


Figure 4.2: Consumed bandwidth

More interesting observations, however, can be made when looking at the sizes of the packets which are sent by the client during the emulation as shown

in Figure 4.3. Almost all packets sent by the client are either 40 byte TCP-acknowledgments or 46 byte TCP-push-acknowledgments. That is, the Citrix client encodes all user input using 6 byte payload and sets the TCP push flag to enforce the immediate delivery of the data. Instead of sending packets of larger size, the ICA protocol prefers to reduce the time between packets in times of higher bandwidth utilization. This can be seen in the zoomed area in Figure 4.3, which shows that the time between two sent packets is significantly smaller when scrolling than while typing. Thus, in the design of the ICA protocol, responsiveness seems to have played a more important role than overhead.

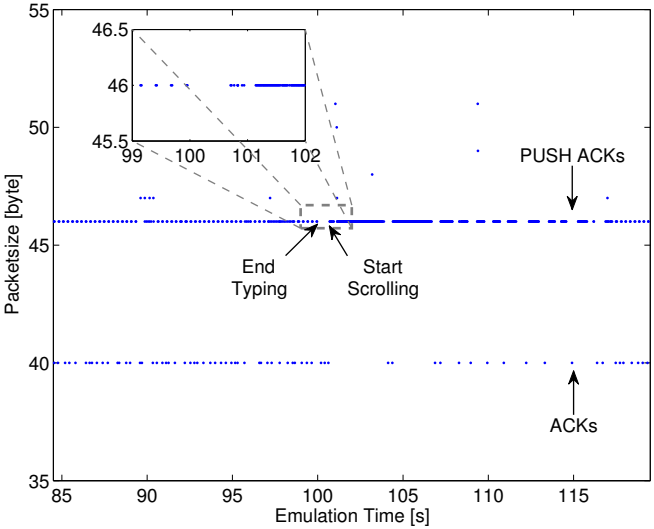


Figure 4.3: Packet sizes on client side

Impact of the User Behavior

Next, we analyze the traffic caused by different types of typing users. The results shown in Figure 4.4 and Figure 4.5 are obtained by emulating a user who types the beginning of Orwell's "War of the Worlds" for 10 minutes. We considered typing speeds of 100, 250, and 400 *characters per minute* (cpm) as to be typical for light, normal, and power users, respectively. For comparison purposes, we also include results obtained for 500 cpm.

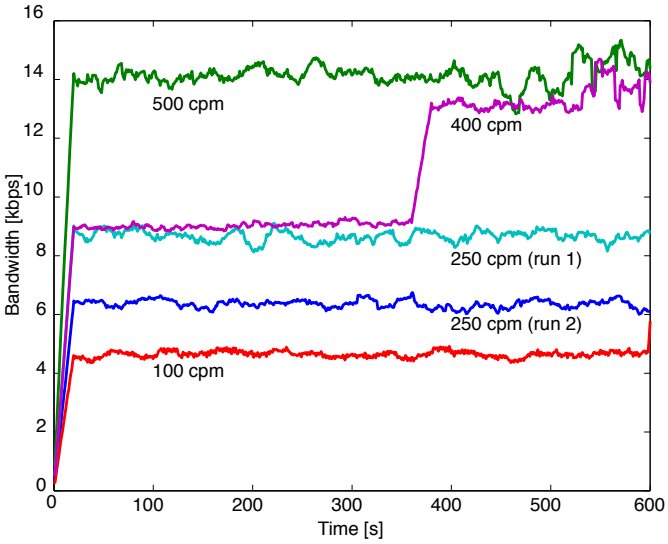


Figure 4.4: Consumed bandwidth on server side

When different types of users are emulated, the bandwidth consumed by the client varies between 2 and 7 kbps. In Figure 4.4 we show 20 s moving averages of the bandwidth consumed by the server for the same experiment. The abrupt rise in the curve representing the power user (400 cpm) in Figure 4.4 is illustrating our

observations during the measurements: In some cases, the consumed bandwidth suddenly grows by a significant amount, but remained constant before and after this event. As this increase is too fast for being TCP triggered, we assume that an internel QoE mechanism is responsible for it. This appears also to be the reason, why the obtained mean values for a normal user (250 cpm) differ by more than 2 kbps for two different experiments (run 1 and run 2). However, the total of our measurements show, that besides the mentioned increases, which do not occur on client side, a typing user consumes a nearly constant amount between 4 and 15 kbps of network bandwidth depending on the typing speed.

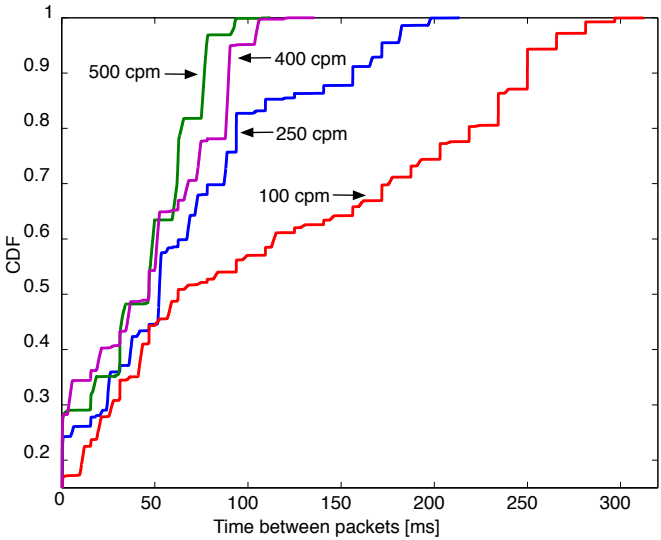


Figure 4.5: Interarrival times of client sent packets

The comparison of inter-arrival times of client sent packets in Figure 4.5 shows results similar to the observations described in the last section: The cumulative

density functions (CDFs) illustrate that the client increases the bandwidth by sending out packets faster in the case of a more active user. The server sent larger packets, while the size of most client packets is 46 byte. Hence, the Citrix client adapts to the user behavior by varying the time between two packets instead of the packet size.

Comparison of Different MS Office Applications

Figure 4.6 and Figure 4.7 visualize the impact of the application type on the network traffic. For this experiment we compare the emulations of users searching and selecting menu entries under MS Word, inserting text, pictures as well as looking at animations under MS PowerPoint and selecting fields in an MS Excel data sheet.

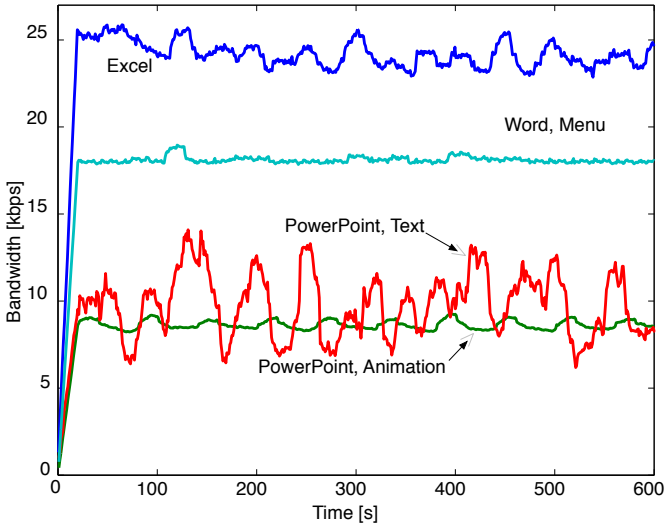


Figure 4.6: Bandwidth consumed on server side

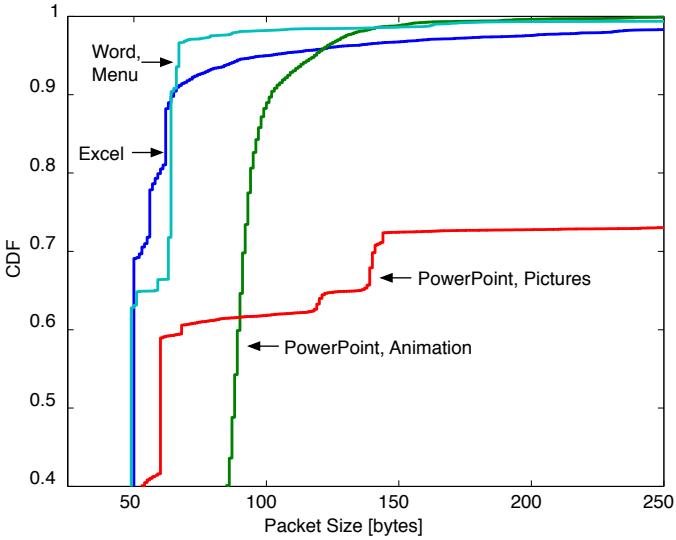


Figure 4.7: Server sent packet sizes

First, we compare the consumed server side bandwidth. Again, the bandwidth consumed by the client was only 2 kbps in the case where the animation was displayed. Note that in Figure 4.6 we do not show the bandwidth consumed for the insertion of pictures in a MS PowerPoint presentation, as the used files were roughly of size 2 MB. The amount of consumed bandwidth was thus significantly larger than for all other tasks. However, we are surprised that the compared 20 s moving averages for the MS Excel test and for MS Word menu operations are significantly higher than for displaying animations, which lie roughly in the range of inserting text in a presentation. The variations in the bandwidth of the latter test may arise from the necessary creation of new slides, a task which requires bandwidth intense mouse movements. In summary, we see that while the consumed bandwidth differs quite strongly between the applications, it is nearly constant for each single task.

As in the earlier described cases, the size of packets sent by the client is almost constant. The size of the server sent packets, however, differs significantly between the measurements, cf. Figure 4.7. Note that for the case of highly interactive mouse operations (MS Excel and MS Word menu), the size of more than 50% of all packets is only 40 byte, i.e. those packets are acknowledgments. Our measurements show that, similar to the typing tests analyzed earlier, the responsiveness seems to be improved by sending packets faster. This does however not hold for the case of animations and pictures under MS PowerPoint, where the bandwidth increase results from sending larger packets. Another interesting fact is illustrated by the CDF in Figure 4.7 representing the insertion of pictures in a presentation: While all inserted pictures were stored on the client, the large packet sizes observed during the experiment indicate that for presentation purposes the picture files still have to be transferred to the server.

Characterization of Server Side Traffic

Our last experiment is dedicated to a deeper investigation of the server sent packet sizes. For this purpose, we analyze traces collected during the insertion of pictures and the visualization of different animations under MS PowerPoint. A shorter and a longer animation are compared and the influence of the maximum transfer unit set at the server is investigated.

Figure 4.8 visualizes the autocorrelation of server sent packet sizes for a lag up to 800. Apart from two spikes in the case of Animation 1 and one spike in the case of Animation 2, the autocorrelation is fluctuating and not significant. This indicates that the traffic caused by animations is not bursty, but that quite often an acknowledgment packet seems to be transferred just after a larger data packet. The spikes correspond to the duration of the animation, which was shorter in the first case.

The comparison of the two experiments of picture insertion reveals that, while the autocorrelation is fluctuating quite strongly in the case where the server MTU is set to 1394 byte. This is not the case, if the MTU is set to 1500 byte. Both autocorrelations are not significant after a given time which corresponds to the

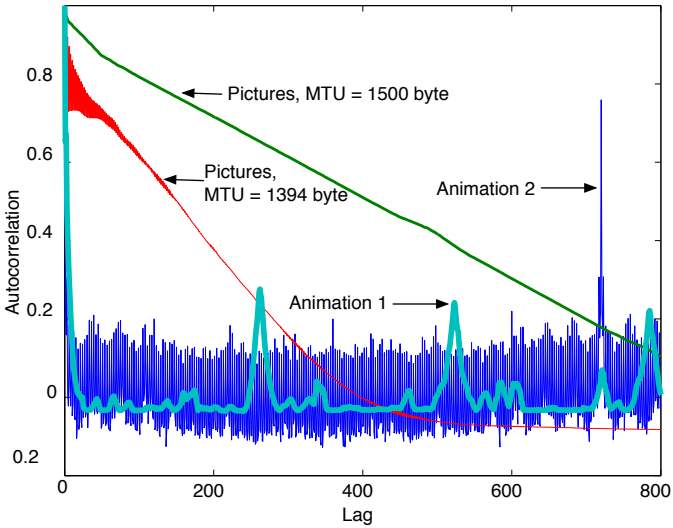


Figure 4.8: Packet size autocorrelation.

transmission of the picture. The reasons for the different shapes and slopes of the curves become apparent in Figure 4.9, where we show for each measurement a snapshot of 250 subsequent sent packets. In both cases, we chose the begin of a picture transmission. Observe that for the case where the smaller maximum packet size is chosen, very often we notice a large packet of 1394 byte if followed by a small packet of around 200 byte. This is not the case, if the MTU is set to 1500 byte: Under this configuration, the most common packet size corresponds to the maximal feasible value. Furthermore, smaller packet sizes are only used for acknowledgments which results in a higher autocorrelation value than in the first case. These results indicate that the server tries to send packets of 1500 byte in both cases, which are fragmented by the operation system under the first con-

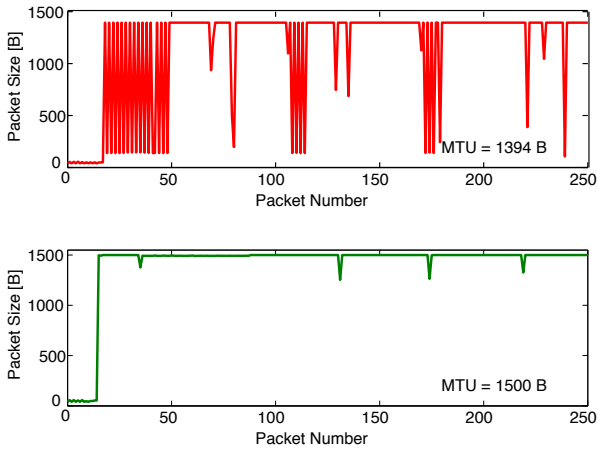


Figure 4.9: Snapshots of a picture transmission.

figuration. Thus, if large pictures or files have to be transmitted, the parameter settings of the operating system and the ICA protocol should be chosen with care to avoid unnecessary transmission delays and bandwidth consumption.

4.2.3 Network layer costs of QoE-improvements

In this section we revisit the application layer settings, which we already discussed from the QoE perspective in Section 3.4. Recall, we consider the Input Buffer, the Speedscreen Latency Reduction, and a combination of both. These application layer settings are triggered by the ASP and affect the perceived QoE of the user. However, modifying these settings also effects the traffic on the network layer. Hence, for the VNO it is crucial to know these effects and to agree with the ASP in which scenarios some features might be triggered. Our aim in this section is to provide additional inside in the effects the application layer settings have on the network layer. Combining these results with the ones of Sec-

tion 3.4.2, we are able to infer best practice guidelines. These can be used for operating virtual Office SaaS networks even if the virtual network is not able to provide 'perfect' QoS.

Input Buffer

The Input Buffer directly influences the frequency at which information is sent from the client to the terminal server. Therefore, using this option should provide a way to reduce the bandwidth consumed by the client in upload direction, as e.g. mouse movements can be subsumed. In order to provide a better understanding how far this feature actually influences the bandwidth sent by the client, we first set both mouse and keyboard buffer to the same value, cf. Section 3.4.1. Afterwards, we increase this value progressively and study the effects on the traffic sent by the client.

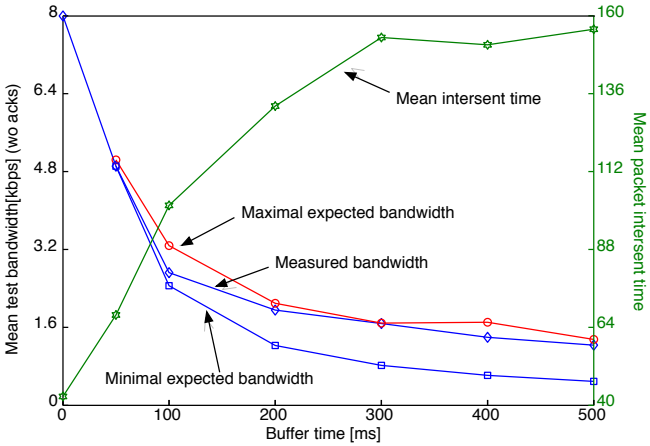


Figure 4.10: Influence of increasing input buffer settings

In order to illustrate the QoS benefits and the delay costs, we plot the bandwidth saved and the latency introduced by the Input Buffer in Figure 4.10. The curve depicts the mean bandwidth usage for increasing the Input Buffer settings, i.e. inter-sent times, during one hour tests using MS Word. For settings smaller than 100 ms the mean inter-sent time changes according to the preset value for the Input Buffer. Thus, the used bandwidth drops rapidly, for increased Input Buffer times in this range. Note, that for a disabled Input Buffer the mean inter-sent time is about 40 ms. For higher presets of the Input Buffer time the measured inter-sent time does not increase accordingly. Instead the mean inter-sent times of the packets converges against 160 ms. This seems to be an upper bound, which the client does not exceed. The mean bandwidth during each test decays for increasing Input Buffer times as expected. To better understand the traffic reduction, we additionally visualized the minimal expected bandwidth. We calculate these values from the bandwidth usage of the default value by dividing it through the relative increase of the buffer setting. If we e.g. double the Input Buffer setting, the minimal expected bandwidth is reduced by half. Furthermore, the curve for the maximal expected bandwidth represents the bandwidth usage derived by dividing the bandwidth without the Input Buffer by the increase of the mean inter-sent times as measured during the tests. Comparing these plots we see that the bandwidth usage is not as low as it would be expected from the preset value of the Input Buffer times. However, the bandwidth gain is better than we would expect from the measured inter-sent times in most of the cases, e.g. for an Input Buffer time setting of 100 ms.

It has to be noted that the traffic sent from the server to the client also decays for an increased Input Buffer setting. This is caused by the fact that in the default case, i.e. with disabled Input Buffer, about 80% of the packets sent by the server are TCP acknowledgements, cf. Section 4.2.2. Hence, as the input updates are sent less frequently, the number of acknowledgements per timeframe decreases accordingly.

Recall from Section 3.4 that the Input Buffer decreased the performance in the case all packets are delivered by the network. As soon as the network drops

packets, the Input Buffer smoothens the loss of QoE by triggering less waiting times for TCP-retransmissions. Recapitulatory, we conclude that the Input Buffer improves the QoE under loss conditions and reduces the used bandwidth. Thus, if the virtual network is dropping packets due to overload, it is advantageous to enable the Input Buffer to smoothen the quality loss perceived by the user and save network resources. In general, however, it is more reasonable to keep the Input Buffer disabled, as it decreases the responsiveness under normal operation, i.e. a lossless network with low latency. Hence, the VNO has to take care for the bandwidth requirements without using the Input Buffer option.

Speedscreen Latency Reduction

Recall, the Speedscreen Latency Reduction is a technique, which is supposed to improve the responsiveness of the system by sending client updates more often and anticipating the servers response.

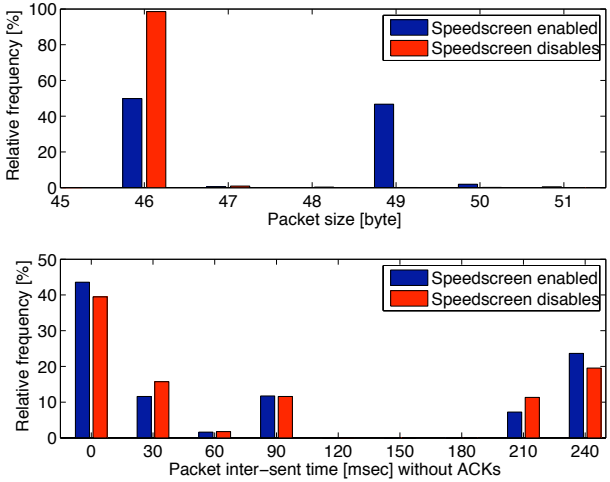


Figure 4.11: Packet pattern sent by the client enabling Speedscreen

The histograms in Figure 4.11 show how the Speedscreen option increases the used bandwidth. About 50% of the packet sizes sent by the client are three bytes larger, if Speedscreen is enabled. However the distribution of inter-sent times does not change much. Together this leads to a mean bandwidth increase of about 1 kbps between the client and the server, c.f. also Figure 4.12.

In the previous chapter, cf. Section 3.4, we discuss the effects of Speedscreen on the QoE and reveal that this option is able to speed up the task completion times. In summary, we see that the Speedscreen option improves the QoE while simultaneously increasing the bandwidth usage. Therefore, Speedscreen should only be activated for network connections which have a high delay caused by physical restrictions, i.e. long distance connections. If additionally packet loss occurs, i.e. due to an overloaded link, enabling Speedscreen might even worsen the perceived QoE, as it increases the bandwidth consumption and thus adds even more packets to the overloaded link. This implies higher packet loss rates, which might affect the QoE more severely than the anticipated server response can improve the QoE. Thus, the ASP might be interested in activating the Speedscreen option for long-distance connection with a high QoS. The VNO, however, needs to be aware of this decision, as it increases the bandwidth requirements for each connection.

Combining the Input Buffer with Speedscreen

The Input Buffer has lowered the bandwidth consumption whereas Speedscreen increased the used bandwidth. Thus, the question arises how a combination of both affects the network costs in terms of utilized bandwidth.

Figure 4.12 depicts moving average values of the client and server bandwidth measured. We smoothened the representation by using a window size of 100 seconds and chose a duration of 200 seconds, as the bandwidth heavily varies within each test run. The figure summarizes the bandwidth usage of the Textpad tests with default settings (no Speedscreen), with enabled Speedscreen (Speedscreen), and the combination of Speedscreen and Input Buffer (Buffer & Speedscreen). The lower half of Figure 4.12 reveals that the bandwidth consumption of a Citrix

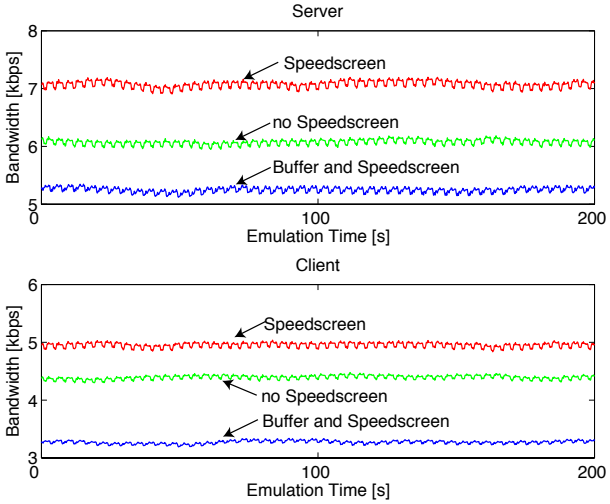


Figure 4.12: Bandwidth changes for Speedscreen and Input Buffer

client combining the Input Buffer and the Speedscreen option is smaller than the used bandwidth of both the default client and the client with only Speedscreen enabled. As it can be inferred from the Input Buffer measurements in Section 4.2.3, a change in the client upload bandwidth results in an accordingly modified bandwidth consumption of the server, i.e. a higher client upload bandwidth leads to an increased bandwidth received in download direction. Again, a change in the packet rate sent by the client causes the server to send TCP acknowledgments accordingly and therefore modifies the bandwidth usage in both directions.

Considering the QoE results described in Section 3.4, a combination of the Input Buffer and Speedscreen improves the performance compared to an unmodified client in a lossless scenario and outperforms both unmodified and Speedscreen only client in a scenario with packet loss. We come to the conclusion that combining Input Buffer and Speedscreen can further improve the responsiveness

in congested networks and might be used if short durations of packet loss are likely to happen. This combination is advantageous for the ASP as well as for the VNO. The ASP can improve the QoE of the user while lowering the network load. Hence, the ASP and VNO can agree on activating Speedscreen and the Input Buffer concurrently in cases of network QoS degradations.

4.3 Monitoring QoS

After considering the traffic characterization on the example of MS Office provided as SaaS, we now focus on measurement tools, which enable the VNO to measure the QoS in its network. We choose the Cisco IP SLA solution as an example, as it provides in-network measurements. This is a promising solution to be extended for virtual networks, as it can be easily integrated into virtual routers. Additionally, such a solution allows for precise measurements without deploying physical measurement components everywhere in the virtual network.

4.3.1 QoS Measurements within the Network

In practice, systems often do not exactly behave like expected from theory. This is especially the case for systems that use virtualization and where many virtual systems are run on the same hardware without perfect isolation. Hence, for a virtual network architecture supporting QoS over virtual networks it is necessary to implement features, which can measure the current QoS of a network service.

In the following we focus on an active measurement, which can be performed between two routers. The advantage is that the PIP does not have to deploy special measurement hardware. We examined the measurement quality of the *Cisco IP SLA UDP Jitter Test*. The IP SLA framework, which was formerly known as response time reporter (RTR) supports many tests from simple ping up to response time measurements for server requests, cf. [161]. However, we focus on the UDP Jitter Test, as it measures all interesting QoS parameters with a single configurable bulk data transmission. Other tests are able to monitor higher layer

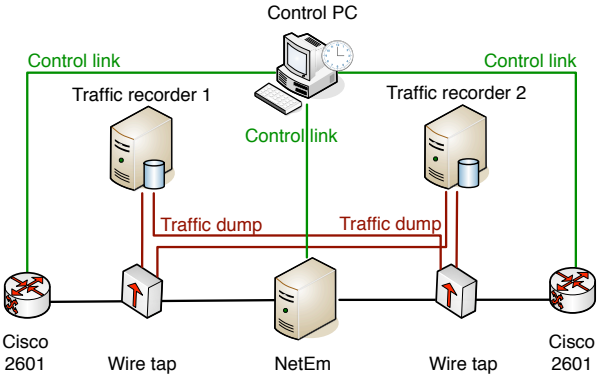


Figure 4.13: Testbed setup for IP SLA measurements

services like web response times and SIP connection setup times. The UDP Jitter measurement has the additional advantage that the two routers used can be configured to prioritize the measurement packets. Therefore it is possible to measure the network between the routers as a black box, and the measured QoS parameters are not affected by the load on the routers performing the measurement. Furthermore, if two edge-routers are able to perform these kind of tests, they can be used to characterize the global QoS of a network service or a specific part of the network. The prediction for a specific network service might additionally be improved by emulating the data transfer of the service. However, as we are only interested in the general precision and do not focus on a special service, we use the default setting. This sends out 1000 packets of size 214 bytes with a inter-sent time of 20 ms.

In order to verify the quality of the Cisco IP SLA UDP Jitter Test we installed a NetEM network emulator between two Cisco routers, cf. Figure 4.13. The router executing the IP SLA test was a Cisco 2801 router with Cisco IOS 12.4(9)T. As remote station we used a Cisco 2601 router with Cisco IOS 12.3(20). The IOS version of the Cisco 2601 is only supporting RTR, which is the former name of

the IP SLA Tests, but it can be used on a remote host for the IP SLA measurements. In order to compare the results of the router measurement with the exact values produced by the network emulation, we installed a wire tap on both sides of the network emulator and dumped all packets exchanged between the routers. The precision we can achieve with this setup is about 0.5 ms. The measurements are fully automated by a control PC machine, which (1) starts tests on the router with EXPECT scripts, (2) collects test results using SNMP, (3) modifies the network emulator over ssh, (4) controls the PCs dumping data and (5) provided a stratum 2 NTP clock for the complete testbed. This enabled us to measure each combination of a network delay of 0 ms to 250 ms in steps of 25 ms and ten equally spaced packet loss values between 0% and 2.1%. This range of parameters provides a good overview of the measurement precision for realistic WAN scenarios. Each measurement scenario was repeated thirty times. We furthermore measured the influence of jitter between 0 ms and 80 ms for a delay of 200 ms and repeated the test twenty times. Correlated jitter and packet loss are evaluated in cases of 75%, 90%, 99%, and 99,9% correlation and repeated seven and twenty times, respectively.

Quality of Delay Measurements

As a result of the IP SLA test, round trip times are reported. For each measurement the IP SLA test reports the minimum, the mean and the maximum delay during the measurement. In Figure 4.14 we show the maximum absolute error of our measurement results for the maximum delay value reported by the IP SLA tests. It can clearly be seen that the results of the router do not overestimate the maximum delay more than 6 ms in all tested cases. The accuracy for the mean delay and the minimal delay results are even better. The difference between the measured results of the emulated delay and the delay reported by IP SLA is 1.42 ms on average. It has to be noted that none of the test results underestimated the delay realized by the network emulator by more than 2 ms. Hence, the IP SLA results have a good precision estimating the network delay considering interesting parameter ranges for WAN scenarios. If both routers are synchronized with

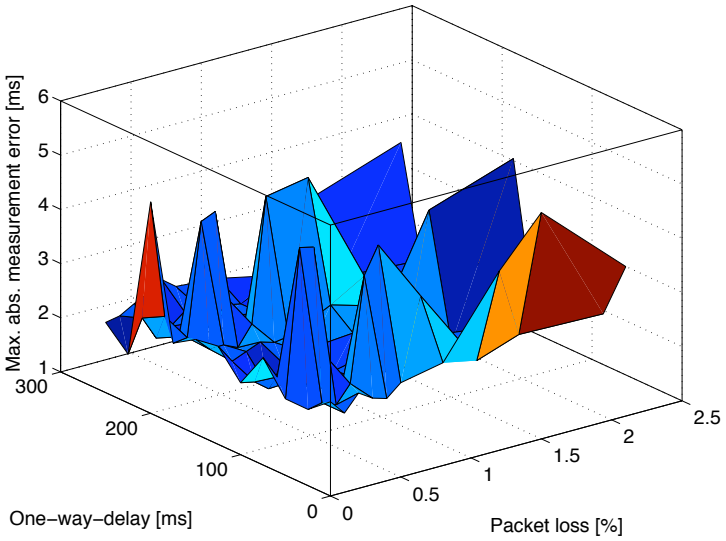


Figure 4.14: Maximum error of the delay measurement

a timeserver, it is also possible to measure one-way delays. Therefore, we set up some additional test series, in which we adjusted the delays for both directions in the network emulator differently. From these measurements we can confirm that both one-way delays work independently from each other and expose the same quality of results, which we measured in the other experiments described before.

Quality of Jitter Measurements

In order to study the quality of the jitter results, we first considered jitter values between 0 ms and 80 ms with a step width of 10 ms using a one way delay of 200 ms. Figure 4.15 depicts the absolute error between the reported jitter values and our reference jitter measurement. The line in the middle of each box

visualizes the median, whereas the upper and lower bound of the box show the inter-quartile range.

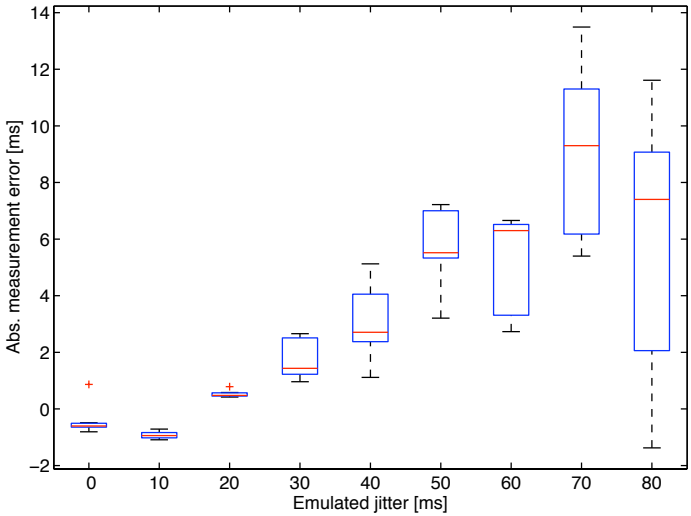


Figure 4.15: Error in reported jitter when increasing the emulated jitter

The minimum and maximum reported differences are marked by the ends of the vertical line. Differences more than three times the inter-quartile range away from the median are considered as outliers and marked by a '+'. The measurement quality for jitter beneath 40 ms is quite good, i.e. in the majority of tests resulted in a relative error below 10%. For higher jitter values, the absolute error range increases and the relative error also increases up to 20%. However, in practice the jitter values are mostly beneath 30 ms. If the reported jitter is higher, we can use the value as a worst-case approximation, as the reported values very rarely underestimate the jitter of the system in these cases.

In practice, another phenomenon is quite often observed. Jitter is rarely spread evenly. Sometimes there are periods of higher traffic and congestion in the net-

work, which additionally delays the transmission to some extend. However, there are also periods in which the network is less loaded and the transmission delay is only slightly increased. In order to simulate such a scenario, we adjusted the emulation to generate correlated jitter. We consider correlation values of 75%, 90%, 99%, and 99.9% for jitter values of 5 ms, 10 ms and 20 ms. We also consider average transmission delays of 50 ms, 75 ms, 100 ms, and 150 ms.

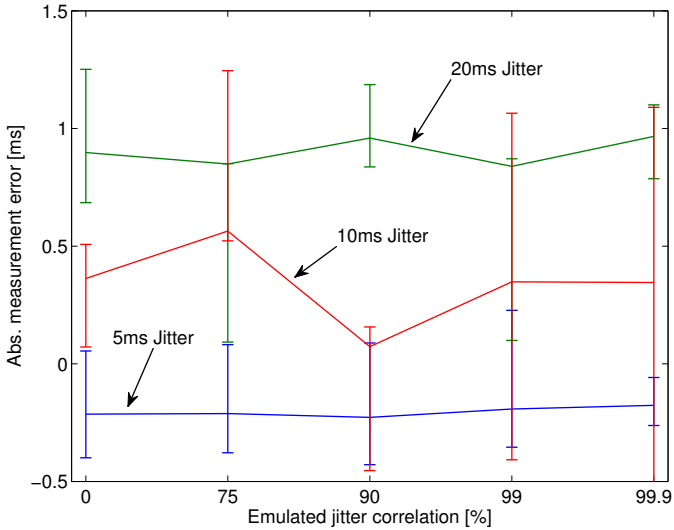


Figure 4.16: Absolute error in reported jitter when increasing the emulated jitter correlation

Figure 4.16 depicts the influence of jitter correlation on the error of jitter results of the IP SLA test for an average transmission delay of 75 ms. The curves depict the mean error between the jitter values, which we calculated from the time stamps recorded in the reference measurement and the results the router reported. For each measurement point, the error bars depicts the maximum and minimum difference between these values. Again, we see that the mean error increases for

higher jitter values. However, the range is rather small and correlation of jitter does not significantly effect the quality of the measurement. The results for the other emulated delay values are similar. Hence, a test like the UDP Jitter Test is a feasible tool for a VNO to measure the QoS of an end-to-end network connection or a partition of the network.

Quality of Loss Measurements

Packet loss in the network influences the sending behavior of a congestion aware protocol, e.g. TCP, which adapts its transmission rate. It is hard to estimate packet loss correctly with active measurements, as concurrent TCP based connections might lower their bandwidth and free capacity, which decreases the measured loss on the link as described in [162]. This general problem of active testing packet loss has to be considered, if measurements are done in a real network.

Figure 4.17 depicts the packet loss reported by the UDP Jitter Test for emulated packet loss values of 0.3%,1.5%, and 2.1%. The plot shows the mean values, the minimum, and the maximum of the loss reported by IP SLA. The number of lost packets reported for each direction is always correct. However, the deviation of the minimum and maximum reported packet loss value from the actual emulated packet loss value is quite large. This is a general problem, which occurs due to the fact that we probe a random process. One active measurement over a lossy link can be considered as a sample of this random process. It is therefore rather unlikely to estimate the real loss probability, but only within a certain range around the real value. The quality of the estimation for one test can only be improved by increasing the number of packets sent within the test.

This basic statistical problem gets more severe if we consider bulk loss patterns. As described in [162], packet loss in the Internet often occurs in short loss periods. Within such periods, a high number of packets are dropped by the routers because of congestion. Between two loss periods the router are able to deliver all packets. Some packets might be delayed but none is lost. In order to reproduce this behavior, we created correlated loss patterns with our network emulation. For a correlation of 75% and above, hardly any measurements did record lost

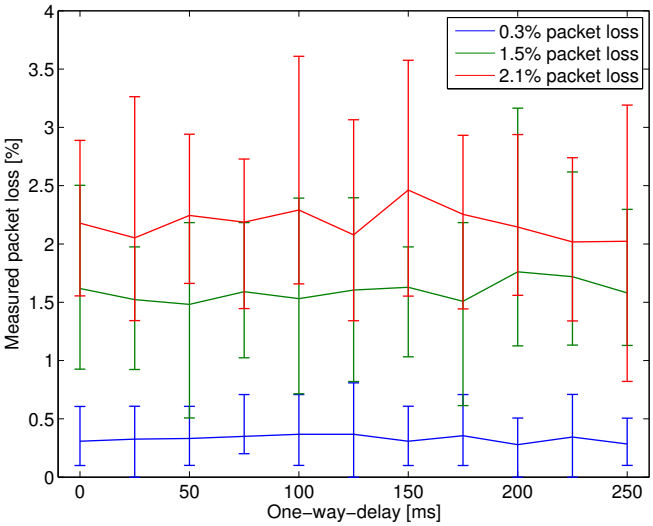


Figure 4.17: Measured loss rate for increasing delay

packets. Therefore, the mean values are also close to zero. However, in the rare case that packets of the test are lost, the probability is high, that many packets in sequence are lost, due to the high correlation. Hence, the maximum measured packet loss of all tests depicted in Figure 4.18 demonstrate these fluctuations. It is evident that even tests with a thousand packets are not sufficient to differentiate between 1.5% or 0.3% packet loss in this scenario. Under these circumstances the measured loss only depends on how much packets are lost during the test period. However the duration and the frequency of those loss periods cannot be estimated in this way. Hence, it is not feasible to estimate the packet loss by a single measurement sending one bulk of packets in practice.

For packet loss measurements relying on active bulk probing, there is a general trade-off between costs, i.e. the number of packets, and reliability. If a single

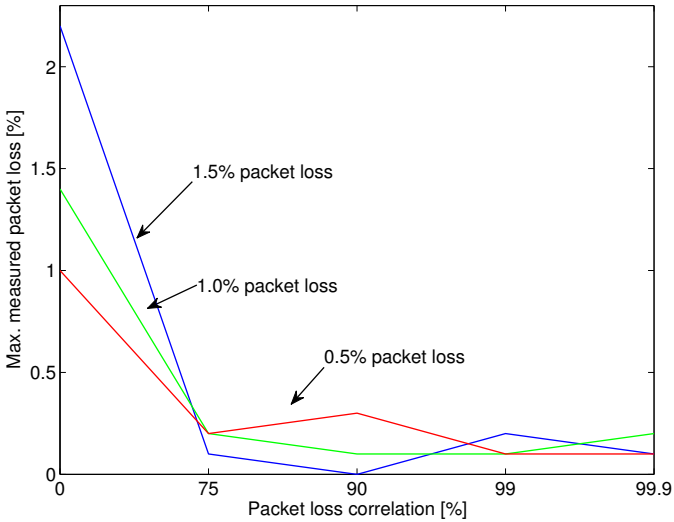


Figure 4.18: Maximum measured loss for correlated loss pattern

measurement has to provide a good estimation of packet loss, there has to be a high number of packets transmitted in the test. But there is also another solution. The mean loss values of many consecutive measurements can be used to estimate the mean loss value during the measurement time. This is correct as for each test the outcome is binomially distributed with the same parameters and therefore it is possible to sum them up. Furthermore, it is reasonable to design active tests in such a way, that the quality of the other estimated QoS parameters is sufficient, and to repeat these measurements in order to update these values over time and to better estimate the mean loss during a longer time span.

4.4 Calculating QoS for Patched Network Paths

In the previous section, we analyzed the precision of an in-network measurement tool. Such tools can be used by the VNO to verify the SLAs, it has for different parts of the network with the providing PIPs. However, the VNO is also interested in the end-to-end QoS of its network. The end-to-end QoS can be also measured with in-network measurements. However, it can also be calculated from measurements of a network partition. The estimation of the QoS parameters along an end-to-end path through n network segments, which could be routers or networks themselves, combines the measurements Q_i of the QoS parameter Q through all network segments $1 \leq i \leq n$. Let Q_i^+ denote the combined measurements through the first i network segments. After the next network segment $i+1$, the QoS parameter Q is observed as

$$Q_{i+1}^+ = Q_i^+ \circ Q_{i+1} = Q_1 \circ Q_2 \circ \ldots \circ Q_i \circ Q_{i+1}\,, \tag{4.1}$$

whereby the operator $\circ$ describes how the measurements are combined. The concrete definition of the operator depends on the actual QoS parameter. Without loss of generality it is sufficient to explain the combination of two QoS parameters due to the recursive description in Equation 4.1. In the following, we show how to calculate the QoS for patched network paths. To be more precise, the calculation of bandwidth, packet loss, as well as delay and jitter is presented for two measurements Q_i and Q_j.

4.4.1 Bandwidth

We consider two network segments i and j with throughput C_i and C_j, respectively. Then, the available throughput C along i and j is simply the minimum of both, i.e.

$$C = \min\left(C_i, C_j\right). \tag{4.2}$$

When the available bandwidth at a network segment is measured for a certain

time period Δt, the throughputs C_i and C_j are described as random variables. Assuming that the throughput on the network links is independent, i.e. C_i and C_j are statistically independent random variables, the cumulative distribution function (CDF) of the throughput C is simply

$$C(x) = P\left[C \leq x\right] = 1 - \left(1 - C_i(x)\right)\left(1 - C_j(x)\right) \ . \tag{4.3}$$

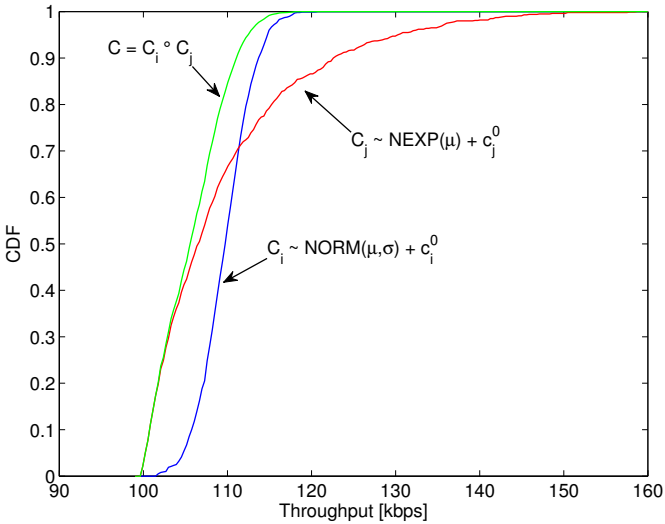


Figure 4.19: Example estimation of throughput for two network segments

Figure 4.19 shows exemplary the CDF of the throughput C for two measured CDFs of C_i and C_j. For C_i, we assume the sum of a minimum available bandwidth $c_i^0 = 100\,\text{kbps}$ and a normal distributed random variable with mean $\mu = 10\,\text{kbps}$ and standard deviation $\sigma = 3\,\text{kbps}$. For C_j, we assume the sum of a minimum bandwidth $c_j^0 = 100\,\text{kbps}$ and a negative exponentially distributed random variable with mean $\mu = 10\,\text{kbps}$. The obtained bandwidth along i and j is computed according to Equation 4.3 and plotted green in Figure 4.19.

4.4.2 Packet loss

The observed packet loss probabilities in the two network segments i and j are p_i and p_j, respectively. Since packets first traverse network i and then j, the resulting packet loss probability p follows as

$$p = p_i + (1 - p_i)p_j \, . \tag{4.4}$$

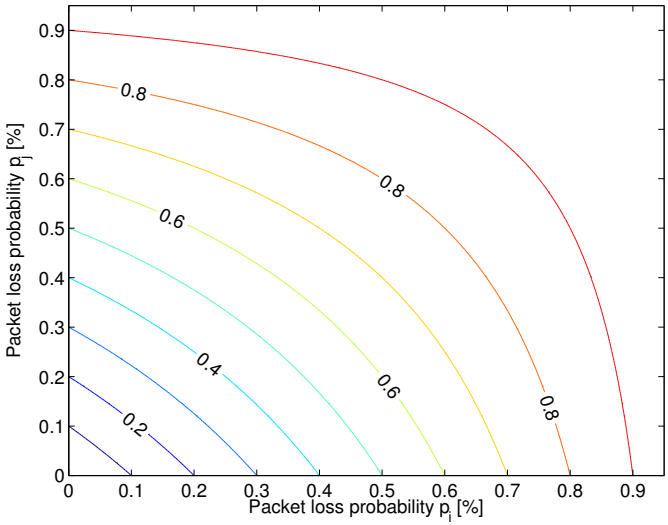


Figure 4.20: Contour plot of the packet loss for two consecutive networks i and j

Figure 4.20 depicts a contour plot of the resulting packet loss p and prints its value according to Equation 4.4 on the corresponding contour lines, while the packet loss probability of network segment i and j are given on the x-axis and y-axis, respectively.

4.4.3 Delay and Jitter

Let T_i and T_j denote the random variables for the delay through the i-th and j-th network segment. If the delays are independent, the joint distribution of the component delays can be formed using convolution. Thus, the total delay T observed along the path through i and j is

$$T = T_i + T_j \,, \tag{4.5}$$

while the probability density function (PDF) $t(x)$ as first derivation of the CDF $T(x)$, i.e. $\frac{d}{dx}T(x) = \frac{d}{dx}P\left[T \leq x\right]$, is calculated as

$$t(x) = \int_{\tau=0}^{x} t_i(\tau)t_j(x-\tau)d\tau \tag{4.6}$$

with the PDFs $t_i(x)$ and $t_j(x)$ of the corresponding delays.

Figure 4.21 shows the PDF of the delays T_i and T_j, as well as their sum T. We assume that T_i follows a lognormal distribution with mean $\mu = 12.18\,\mathrm{ms}$ and standard deviation $\sigma = 255.02\,\mathrm{ms}$ plus a minimum delay $t_i^0 = 10\,\mathrm{ms}$. For T_j, we assume the sum of a minimum delay $t_j^0 = 20\,\mathrm{ms}$ and a pareto random variable with parameter $K = 0.3$ and $\sigma = 4$. The PDF of T is calculated according to Equation 4.6) and depicted as the green curve in Figure 4.21.

Although the PDF of T allows to easily derive the average delay and the jitter, e.g. as standard deviation of the delay T, its calculation faces two problems. First, the computation is too time-consuming and therefore raises challenges in practical implementation of the Networking framework. Second, independence of the delays T_i and T_j is assumed, which might not be valid in practice. The same concerns may also arise for more complex bandwidth or packet loss processes. Nevertheless, in literature several approaches exist which approximate appropriate key performance measures. For example, [163] presents a simple approximation of delay-variation distributions, which is a convolution and transform-free method and allows for an accurate and less time-consuming estimation of jitter.

Furthermore, this calculation does not consider the delay which is caused by the handover of data between two consecutive networks. In some cases, this might

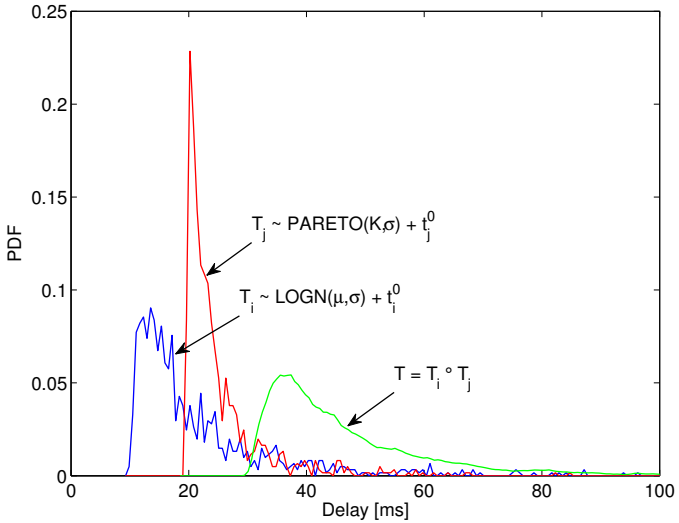


Figure 4.21: Estimation of delay of two individual network segments

be just some additional fixed time span, but might also depend on the data volume and how it is transferred in the different virtual networks. An implementation of the delay distribution therefore also has to consider the handover.

We conclude that besides extensive measurement, it is also possible to calculate the end-to-end QoS from measurements performed on partitions of the network. Depending on the trust between the VNO and the PIPs, this might be used to reduce the measurement effort of the VNO. For instance, the VNO can conduct random test measurements between some end points of its network. The results of these tests are compared to end-to-end QoS calculations based on the monitoring data provided by the PIPs using the interface described in Section 2.2.4. As long as the calculation fits to the end-to-end measurements and the SLA is kept no further action is needed. In contrast, if the calculation does not fit to the measurement data, further measurements are triggered to identify the PIP, which measurement data does not reflect the real properties of its network part.

4.5 Lessons Learned

In this chapter, we investigate topics that are relevant for the VNO. Due to the fact that virtualization hides the implementation of the network, QoS management is again a challenging research area. On the one hand, creating virtual networks for guaranteeing high quality for services like SaaS raises the need of traffic characterization. Many applications, which are neglected by now due to their small share in the overall resource usage in the Internet, move into the focus of attention, since they are mainly used on a professional level and offer new business opportunities. On the other hand, monitoring and controlling QoS is more complex for virtual networks, since many assumptions of current measurement tools do not necessarily hold.

Our traffic analysis of the business-critical example of MS Office provided as SaaS reveals some correlated major influence factors. First, the utilized Office application changes the way the user works with the system. A user working with MS Word is mainly typing on the keyboard, whereas using MS Powerpoint implies a high number of mouse movements and clicks. As mouse movements lead to a higher number of packets transmitted over the network, we infer from these differences that the traffic profile of both applications differs. Another important aspect is the number of external files, which the user wants to integrate in his document, and which therefore need to be uploaded to the server. The next major influence factor is the user itself. Users with a higher server interaction rate, i.e. users typing or moving the mouse faster, also increase the traffic demand between the client and the server. Furthermore, the WAN optimization options, which we already characterized in terms of the QoE improvement in Section 3.4, have a direct influence on the traffic pattern transmitted in the network. Our investigation revealed that at the cost of a lower client server interaction rate, it is possible to reduce the traffic significantly. In overload situations of the network, this also improves the QoE of the user, as we have shown in Section 3.4. An ASP that underestimated its resource requirement can therefore use this option to limit the QoE decrease of the user and lower the resource requirements, while waiting

for more resources provided by the VNO via the PIPs. Speedscreen can be used on long-distance connections to improve the responsiveness of the application. However, the VNO needs to be informed beforehand, as the bandwidth requirements are increased and packet loss might worsen the situation dramatically. In the case, the network might be congested for short periods of time, combining the options Input Buffer and Speedscreen is a reasonable solution for the ASP, as it improves the QoE of the user and reduces the traffic send in the network.

We discuss the aspects of network virtualization that cause problems with current QoS monitoring tools. The major challenges identified are that the implementing technology is unknown and that the precision of time stamps is uncertain in virtualized environments. One solution would be to trust the measurements the PIP provides with his access to the hardware layer. However, we cannot assume the VNO to trust the PIPs in real world scenarios. Therefore, we propose to integrate active measurement tools within the hypervisor, which provide packet templates, measurement functionality, and can be instantiated according to the VNOs protocol stack definition. Our investigation of a similar tool for nowadays technology show that such a router based implementation is able to provide results with reasonable precision to be used for QoS monitoring.

In a multi-domain scenario, i.e. a scenario in which a VNO is operating a network composed of network parts provided by different PIPs, it is necessary to combine the measurements provided by the PIPs to compare this to the VNOs own measurements. We applied decomposition techniques and provided formulas to calculate end-to-end results from partitioned network measurements using an recursive approach.

5 Conclusions

... and nothing else matters.
James Hettfield

In future, network virtualization will enable the concurrent use of arbitrary physical network hardware to build networks tailored to fit the service they provide. These virtual networks will be constructed based on virtual resources, which abstract functionality from implementation. Furthermore, virtual networks wil provide network isolation. This means, they will guarantee that the resources assigned to one network are not affected by whatever happens in concurrent networks.

In this monograph, we focus on network management for virtual networks in order to provide a satisfactory QoE to the user. We start with discussing use cases that demonstrate new business models and new functionality in the network in the future, e.g. green networking and the beta slice, that cannot be build with todays technology. Based on these use cases, we derive basic building blocks and five functional roles that build, compose, operate, and use virtual networks, namely Physical Infrastructure Provider (PIP), Virtual Network Provider (VNP), Virtual Network Operator (VNO), Application Service Provider (ASP), and End-Customer (EC). The interfaces between these role are not yet completely defined and need to be agreed by hardware vendors in order to speed up standardization and implementation. Network monitoring in this context is challenging, as besides the PIP no other role has direct access to the physical infrastructure. Hence, we deduce the need of additional monitoring interfaces, which allow the exchange of monitoring data between the roles, and virtual router interfaces that

allow to access a hardware clock or enable precise time stamping provided by the virtual router host.

To analyze the performance of virtual hosts on commodity hardware and the quality of isolation we can currently achieve in test-beds and data centers, we consider current hardware virtualization solutions including KVM, OpenVZ, VMware, Xen, and VirtualBox. We measure the overhead of virtualization each implementation implies and the quality of isolation in terms of achievable network throughput and compare it to the raw host system. We conclude that currently, the overhead of virtualization is noticeable for most platforms, even if we only consider a single network interface. The isolation quality of current solutions is relatively high, as long as other virtual guests only use CPU and RAM intensively. If instead another guest is handling high rates of network packets, the isolation is rather weak and overload in one virtual guest can have a severe impact on the performance of the other guests, depending on the packet size. Based on these results, we conclude that research and implementation of hardware virtualization has focused on computation performance recently and now needs to optimize the performance in the packet processing area. Implementing current solutions, operators deploying hardware virtualization on a professional level should consider these effects when scheduling virtual guests on physical hosts. Furthermore, research facilities offering virtualized resources need to provide detailed monitoring data to the test-bed users about all flows that run concurrently to an experiment. Otherwise, the researchers using the facility are not able to differentiate between an interesting behavior of the analyzed system and a measurement error due to the influence of concurrent experiments.

To operate a virtual network at a quality level that satisfies the user, we need to understand the requirements on the network layer. Therefore, we investigate the impact of network characteristics on the QoE of the user. We focus on the examples VoIP and MS Office provided as SaaS, since they are both business-critical services and well suited to demonstrate our approach. We provide a detailed analysis of the effects of packet loss to a VoIP conversation and compare the quality of three different codecs. The SILK-codec outperforms the other codecs analyzed

and can cope with minor network disturbances, i.e. application layer packet loss up to 2%, without decreasing the QoE to be unacceptable. Hence, we focus on this codec and show how to derive a mapping between QoS and QoE that can be used as a worst case assessment. Additionally, we consider sampling to reduce the monitoring effort. Our analysis of MS Office provided as SaaS reveals that the impact of a high delay connection effects all users equally, while packet loss leads to a wide area of possible quality perceptions. As long as the packet loss is in a region typical for WAN connections, i.e. between 0% and 2%, some users may only notice a small QoE degradation, while others can hardly use the application. These effects can be smoothened by adapting application layer settings of the SaaS implementation. Our results in this area demonstrate that the effects of these settings are in most cases dependent on the QoS the network is able to achieve. Hence, an ASP should not change the settings without considering the QoS assessment of the VNO. This result supports the necessity of an interface to exchange monitoring data between the functional roles, as it demonstrates that many application layer options directly affect and are affected by the network layer. Therefore, the ASP needs to access the QoS status from the VNO, before deciding to change the application layer. Adapting the settings on each layer with an autonomous network management control plane is only reasonable, if each layer is able to include the view of the adjacent layers. Otherwise, cross-layer problems will increase in severeness, since network virtualization hides all the relevant data of the adjacent layer, which we can currently use to asses how the lower layers might work.

Operating a network according to the requirements of an application or service cannot be achieved without the network being planned and dimensioned properly. Therefore, we consider traffic characterization as an important part of virtual network management. For many applications, we find precise traffic characterizations in related work. However, for the case of MS Office provided as SaaS, there is no such a model. Hence, we investigate the traffic generated by different MS Office applications and consider different kind of users. Depending on the application, the number of keystrokes and mouse movements of users differ

significantly and therefore, the bandwidth requirement changes. Furthermore, our analysis revealed that a 'power user', characterized by a higher input rate, is another reason for an increased traffic demand. Finally, the number of interactions, which need to transfer data between the cloud server hosting the application and the end-system of the user are another impact factor to the required bandwidth that cannot be neglected. For instance, if the user is including only locally available data or is printing the document at a local printer, the traffic increases noticeably. However, user models are provided in related work and together with our analysis, it is possible to tightly fit the required bandwidth. Combining this with our results on the QoE impact of the QoS, one can easily derive the QoS parameters required for a given MS Office SaaS solution.

Reconsidering the application layer settings of the exemplarily chosen SaaS implementation from a network perspective allows us to derive best practice guidelines. Whenever the network is overloaded due to an unexpected number of users, the ASP should request more resources from the VNO and smoothen the QoE degradation caused by its own miss-planning by switching on the Input Buffer. This results in an improved QoE, i.e. a shorter server response time, as less packets are sent and the user less often has to wait for TCP retransmission. Furthermore, the bandwidth consumption of the application is decreased, which will reduce the network overload until the VNO has increased the resources of the network and the Input Buffer can be disabled. In the case the network was designed to sometimes drop packets in order to save resources and be more cost-effective, the VNO and ASP can agree on enabling a combination of Speedscreen and Input Buffer. This combination reduces the bandwidth consumption and improves the QoE from a technical oriented perspective. However, surveys carried out with real test persons revealed that not all users like the visual effects of Speedscreen and, thus, this option has to be considered carefully.

In the course of this monograph, we show that efficient planning and operation of a virtualized network is reasonable. However, it needs to be supported by precise measurements of the QoS the network currently provides. Therefore, we discuss options of in-network measurements, which can be applied even if a vir-

tual router is allowed to roam within the network. We analyze the precision of the Cisco IP SLA tools, which are reasonable candidates to be further developed for in-network measurements of virtual routers. The accuracy of the active measurements is reasonably precise and could in future be improved by implementing a GPS based clock with better time resolution. However, some QoS parameter, e.g. packet loss rate, are hard to assess using active measurements and should better be monitored with a passive monitoring option.

The interface for exchanging monitoring data can be used to decrease the measurement effort of the VNO. This can be done by combining the measurements provided by the PIPs for their part of the network as we discuss in this monograph. If the VNO does not want to rely on monitoring data of PIPs, which might have different economical interests, the VNO can conduct random sampled end-to-end measurements. Comparing the results of the calculation based on the PIP monitoring data with his own spot tests allow him to verify the correctness of the PIPs reports.

The results provided in this monograph demonstrate that future virtual networks can be dimensioned and operated cost efficiently. The autonomous management of service specific networks monitors the user-perceived quality and adapts application layer settings as well as network resources as required. Such networks will use network resources cost-effectively while guaranteeing an optimal quality perceived by the user.

Bibliography and References

— Bibliography of the Author —

— Journals and Book Chapters —

[1] T. Hoßfeld, D. Schlosser, K. Tutschku, and P. Tran-Gia, "Cooperation Strategies for P2P Content Distribution in Cellular Mobile Networks: Considering Selfishness and Heterogeneity," in *Mobile Peer-to-Peer Computing for Next Generation Distributed Environments: Advancing Conceptual and Algorithmic Applications*. Information Science Reference, 2009.

[2] S. Meier, M. Barisch, A. Kirstädter, D. Schlosser, M. Duelli, M. Jarschel, T. Hoßfeld, K. Hoffmann, M. Hoffmann, W. Kellerer, A. Khan, D. Jurca, and K. Kozu, "Provisioning and Operation of Virtual Networks," *Electronic Communications of the EASST, Kommunikation in Verteilten Systemen 2011*, vol. 37, 2011.

[3] A. Fischer, J. F. Botero, M. Duelli, D. Schlosser, X. Hesselbach, and H. de Meer, "ALEVIN - A Framework to Develop, Compare, and Analyze Virtual Network Embedding Algorithms," *Electronic Communications of the EASST, Kommunikation in Verteilten Systemen 2011*, vol. 37, 2011.

[4] D. Schlosser and T. Hoßfeld, "Mastering Selfishness and Heterogeneity in Mobile P2P Content Distribution Networks with Multiple Source Download in Cellular Networks," *Peer-to-Peer Networking and Applications, Special Issue on Mobile P2P Networking and Computing*, vol. 2, 2009.

— Conference Papers —

[5] R. Pries, M. Jarschel, D. Schlosser, M. Klopf, and P. Tran-Gia, "Power Consumption Analysis of Data Center Architectures," in *Proceedings of GreenNets*, Colmar, France, Oct. 2011.

[6] M. Jarschel, S. Oechsner, D. Schlosser, R. Pries, S. Goll, and P. Tran-Gia, "Modeling and Performance Evaluation of an OpenFlow Architecture," in *Proceedings of the 23rd International Teletraffic Congress (ITC 2011)*, San Francisco, CA, USA, Sep. 2011.

[7] M. Jarschel, D. Schlosser, S. Scheuring, and T. Hoßfeld, "An Evaluation of QoE in Cloud Gaming Based on Subjective Tests," in *Proceedings of the Workshop on Future Internet and Next Generation Networks (FINGNet)*, Seoul, Korea, Jun. 2011.

[8] D. Schlosser, M. Duelli, and S. Goll, "Performance Comparison of Hardware Virtualization Platforms," in *Proceedings of the IFIP/TC6 NETWORKING 2011)*, Valencia, Spain, May 2011.

[9] D. Schlosser, M. Jarschel, M. Duelli, T. Hoßfeld, K. Hoffmann, M. Hoffmann, H. J. Morper, D. Jurca, and A. Khan, "A Use Case Driven Approach to Network Virtualization," *accepted at IEEE Kaleidoscope 2010, published via OPUS Würzburg under OpenAccess*, Dec. 2010.

[10] D. Schlosser, M. Jarschel, V. Burger, and R. Pries, "Monitoring the User Perceived Quality of SILK-Based Voice Calls," in *Proceedings of the Australasian Telecommunication Networks and Applications Conference (ATNAC) 2010*, Auckland, New Zealand, Nov. 2010.

[11] D. Schlosser, M. Hoffmann, T. Hoßfeld, M. Jarschel, A. Kirstaedter, W. Kellerer, and S. Köhler, "COMCON: Use Cases for Virtual Future Networks," in *Proceedings of the TridentCom 2010*, Berlin, Germany, May 2010.

[12] D. Schlosser, B. Staehle, A. Binzenhöfer, and B. Boder, "Improving the QoE of Citrix Thin Client Users," in *Proceedings of the IEEE International Conference on Communications (ICC 2010)*, Cape Town, South Africa, May 2010.

[13] D. Schlosser and T. Hoßfeld, "Service Oriented Network Framework Enabling Global QoS and Network Virtualization," in *Proceedings of the ITC Specialist Seminar 2009*, Hoi An, Vietnam, May 2009.

[14] B. Staehle, A. Binzenhöfer, D. Schlosser, and B. Boder, "Quantifying the Influence of Network Conditions on the Service Quality Experienced by a Thin Client User," in *Proceedings of the 14. GI/ITG Konferenz Messung, Modellierung und Bewertung von Rechen- und Kommunikationssystemen (MMB 2008)*, Dortmund, Germany, Apr. 2008.

[15] D. Schlosser, A. Binzenhöfer, and B. Staehle, "Performance Comparison of Windows-based Thin-Client Architectures," in *Proceedings of the Australasian Telecommunication Networks and Applications Conference 2007*, Christchurch, New Zealand, Dec. 2007.

[16] B. Emmert, A. Binzenhöfer, D. Schlosser, and M. Weiß, "Source Traffic Characterization for Thin Client Based Office Applications," in *Proceedings of the 13th EUNICE Open European Summer School and IFIP TC6.6 Workshop on Dependable and Adaptable Networks and Services*, Twente, the Netherlands, Jul. 2007.

[17] A. Binzenhöfer, D. Schlosser, K. Tutschku, and M. Fiedler, "An Autonomic Approach to Verify End-to-End Communication Quality," in *Proceedings of the Tenth IFIP-IEEE International Symposium on Integrated Network Management (IM 2007)*, Munich, Germany, May 2007.

[18] D. Schlosser, T. Hoßfeld, and K. Tutschku, "Comparison of Robust Cooperation Strategies for P2P Content Distribution Networks with Multiple Source Download," in *Proceedings of the Sixth IEEE International Conference on Peer-to-Peer Computing (P2P2006)*, Cambridge, UK, Sep. 2006.

[19] T. Hoßfeld, K. Tutschku, and D. Schlosser, "Influence of the Size of Swapping Entities in Mobile P2P File-Sharing Networks," in *Proceedings of the Peer-to-Peer-Systeme und -Anwendungen*, GI/ITG-Workshop in Kooperation mit KiVS 2005, Kaiserslautern, Germany, Mar. 2005.

— General References —

[20] B. Leiner, V. Cerf, D. Clark, R. Kahn, L. Kleinrock, D. Lynch, J. Postel, L. Roberts, and S. Wolff, "A Brief History of the Internet," *Internet Society*, vol. 10, 2003.

[21] M. Nierwettberg, "Das Telekom-Komplott gegen Youtube," *Blog.Telekom http://blogs.telekom.com/2011/05/18/das-telekom-komplott-gegen-youtube/*, 05/18/2011.

[22] M. Flannagan, *Administering Cisco QoS for IP Networks*. Syngress Media Inc., 2001.

[23] R. Sietmann, "Schmalspur," *c't magazin für computer technik*, vol. 08, 2011.

[24] B. Tudor, "Gartner Says SaaS Revenue Within the Enterprise Application Software Market to Total $9.2 Billion in 2010," Gartner, Inc., Tech. Rep. 1492814, 2010.

[25] C. Henke, A. Siddiqui, and M. R. Khondoker, "Network Functional Composition: State of the Art," in *Proceedings of the Australasian Telecommunication Networks and Applications Conference (ATNAC)*, Auckland, New Zealand, Oct. 2010.

[26] ITU-T Recommendation, "Oneway transmission time," *ITU G.114*, 1996.

[27] Skype Technologies S.A., "Skype," http://www.skype.com.

[28] A. Finamore, M. Mellia, M. Meo, M. Munafo, and D. Rossi, "Live Traffic Monitoring with Tstat: Capabilities and Experiences," *Wired/Wireless Internet Communications*, vol. 6074, 2010.

[29] Y. Wang, "Survey of Objective Video Quality Measurements," EMC Corporation, Tech. Rep., 2006.

[30] N. Chowdhury and R. Boutaba, "Network Virtualization: State of the Art and Research Challenges," *IEEE Communications Magazine*, vol. 47, 2009.

[31] ——, "A Survey of Network Virtualization," *Computer Networks*, vol. 54, 2010.

[32] A. Marikar, J. Mödeker, and K. Jonas, "Selbstverwaltung im Future Internet," *Electronic communications of the EASST*, vol. 17, 2009.

[33] N. Feamster, L. Gao, and J. Rexford, "How to Lease the Internet in Your Spare Time," *ACM SIGCOMM Computer Communication Review*, vol. 37, 2007.

[34] D. Hausheer and B. Stiller, "Peermart: The Technology for a Distributed Auction-Based Market for Peer-to-Peer Services," in *Proceedings of the IEEE International Conference on Communications*, Seoul, Korea, May 2005.

[35] K. L. Kroeker, "The Evolution of Virtualization," *Communications of the ACM*, vol. 52, 2009.

[36] T. Anthias and K. Sankar, "The Network's New Role," *Queue*, vol. 4, 2006.

[37] N. Niebert, I. Khayat, S. Baucke, R. Keller, R. Rembarz, and J. Sachs, "Network Virtualization: A Viable Path Towards the Future Internet," *Wireless Personal Communications*, vol. 45, 2008.

[38] K. Shiomoto, I. Inoue, and E. Oki, "Network Virtualization in High-Speed Huge-Bandwidth Optical Circuit Switching Network," in *Proceedings of the IEEE Infocom Workshops 2008*, Phoenix, AZ, USA, Mar. 2008.

[39] T. R. Banniza, "D-1.2 Project-Wide Evaluation of Business Use Cases," FP7 4WARD Project, Tech. Rep., 2009.

[40] S. Baucke and C. Görg, "D-3.1.1 Virtualisation Approach: Concept," FP7 4WARD Project, Tech. Rep., 2009.

[41] R. Bless and C. Werle, "Network Virtualization from a Signaling Perspective," in *Proceedings of the IEEE International Conference on Communications (ICC) Workshops 2009*, Dresden, Germany, Jun. 2009.

[42] M. Subramanian, *Network Management: Principles and Practice*. Addison-Wesley, 2000.

[43] A. Hanemann, J. Boote, E. Boyd, J. Durand, L. Kudarimoti, R. Łapacz, D. Swany, S. Trocha, and J. Zurawski, "Perfsonar: A Service Oriented Architecture for Multi-Domain Network Monitoring," *Service-Oriented Computing-ICSOC*, vol. 3826, 2005.

[44] M. Iliofotou, P. Pappu, M. Faloutsos, M. Mitzenmacher, S. Singh, and G. Varghese, "Network Monitoring Using Traffic Dispersion Graphs," in *7th ACM SIGCOMM Conference on Internet Measurement*, Kyoto, Japan, Aug. 2007.

[45] T. Repantis, J. Cohen, S. Smith, and J. Wein, "Scaling a Monitoring Infrastructure for the Akamai Network," *ACM SIGOPS Operating Systems Review*, vol. 44, 2010.

[46] N. McKeown, T. Anderson, H. Balakrishnan, G. Parulkar, L. Peterson, J. Rexford, S. Shenker, and J. Turner, "OpenFlow: Enabling Innovation in Campus Networks," *ACM SIGCOMM Computer Communication Review*, vol. 38, 2008.

[47] G. Consortium, "Global Environment for Network Innovations (GENI)," http://www.geni.net.

[48] Stanford OpenFlow Group (edt.), "OpenFlow Specification," Stanford University, Tech. Rep., 2011.

[49] N. Gude, T. Koponen, J. Pettit, B. Pfaff, M. Casado, N. McKeown, and S. Shenker, "NOX: Towards an Operating System for Networks," *ACM SIGCOMM Computer Communication Review*, vol. 38, 2008.

[50] J. Naous, D. Erickson, G. Covington, G. Appenzeller, and N. McKeown, "Implementing an OpenFlow Switch on the NetFPGA Platform," in *Proceedings of the 4th ACM/IEEE Symposium on Architectures for Networking and Communications Systems*, San Jose, CA, USA, Nov. 2008.

[51] R. Sherwood, M. Chan, A. Covington, G. Gibb, M. Flajslik, N. Handigol, T. Huang, P. Kazemian, M. Kobayashi, J. Naous *et al.*, "Carving Research Slices Out of Your Production Networks with OpenFlow," *ACM SIGCOMM Computer Communication Review*, vol. 40, 2010.

[52] L. van Doorn, "Hardware Virtualization Trends," in *Proceedings of the 2nd International Conference on Virtual Execution Environments (VEE)*, New York, NY, USA, 2006.

[53] F. Camargos, "Virtualization of Linux Servers: A Comparative Study," *Masters Abstracts International*, vol. 47, 2008.

[54] D. Ongaro, A. L. Cox, and S. Rixner, "Scheduling I/O in Virtual Machine Monitors," in *Proceedings of the Fourth ACM SIGPLAN/SIGOPS International Conference on Virtual Execution Environments (VEE)*, Seattle, WA, USA, 2008.

[55] F. Anhalt and P. V.-B. Primet, "Analysis and Experimental Evaluation of Data Plane Virtualization with Xen," in *Proceedings of the Fifth International Conference on Networking and Services (ICNS)*, Los Alamitos, CA, USA, Apr. 2009.

[56] J. Whiteaker, F. Schneider, and R. Teixeira, "Explaining Packet Delays Under Virtualization," *ACM SIGCOMM Computer Communication Review*, vol. 41, 2011.

[57] S. Tripathi, N. Droux, T. Srinivasan, and K. Belgaied, "Crossbow: From Hardware Virtualized NICs to Virtualized Networks," in *Proceedings of the 1st ACM Workshop on Virtualized Infrastructure Systems and Architectures*, Barcelona, Spain, Aug. 2009.

[58] S. Tripathi, N. Droux, K. Belgaied, and S. Khare, "Crossbow Virtual Wire: Network in a Box," in *Proceedings of the 23rd Conference on Large Installation System Administration*, Baltimore, MD, USA, Nov. 2009.

[59] M. B. Anwer, A. Nayak, N. Feamster, and L. Liu, "Network I/O Fairness in Virtual Machines," in *Proceedings of the 2nd ACM SIGCOMM Workshop on Virtualized Infrastructure Systems and Architectures (VISA)*, New Delhi, India, Sep. 2010.

[60] N. Egi, A. Greenhalgh, M. Handley, M. Hoerdt, F. Huici, and L. Mathy, "Fairness Issues in Software Virtual Routers," in *Proceedings of the ACM Workshop on Programmable Routers for Extensible Services of Tomorrow (PRESTO)*, Seattle, WA, USA, Aug. 2008.

[61] ——, "Towards High Performance Virtual Routers on Commodity Hardware," in *Proceedings of the 2008 ACM CoNEXT Conference*, Madrid, Spain, 2008.

[62] VMware, Inc. (edt.), "A Performance Comparison of Hypervisors," http://www.vmware.com/pdf/hypervisor_performance.pdf, 2007.

[63] XenSource Inc, "A Performance Comparison of Commercial Hypervisors," http://www.vmware.com/pdf/hypervisor_performance.pdf, 2007.

[64] M. Bredel, Z. Bozakov, and Y. Jiang, "Analyzing Router Performance Using Network Calculus with External Measurements," in *Proceedings of the 18th International Workshop on Quality of Service (IWQoS)*, Beijing, China, Jun. 2010.

[65] G. Bhanage, I. Seskar, Y. Zhang, D. Raychaudhuri, and S. Jain, "Analyzing Router Performance Using Network Calculus with External Measurements," in *Proceedings of the TridentCom 2010*, Berlin, Germany, May 2010.

[66] S. Bhatia, M. Motiwala, W. Muhlbauer, Y. Mundada, V. Valancius, A. Bavier, N. Feamster, L. Peterson, and J. Rexford, "Trellis: A Platform for Building Flexible, Fast Virtual Networks on Commodity Hardware," in *Proceedings of the 2008 ACM CoNEXT Conference*, Madrid, Spain, Madrid, Spain 2008.

[67] R. Olsson, "pktgen: The Linux Packet Generator," in *Proceedings of Linux Symposium*, Ottawa, Canada, Jul. 2005.

[68] G-Lab Consortium, "The G-Lab Project - National Platform for Future Internet Studies," http://www.german-lab.de, last accessed September 8th 2010.

[69] Wikipedia, "Comparison of Platform Virtual Machines — Wikipedia, The Free Encyclopedia," http://en.wikipedia.org/w/index.php?title=Comparison_of_platform_virtual_machines, 2010.

[70] VMware, Inc. (edt.), "VMware ESXi," http://www.vmware.com/products/vi/esx/.

[71] Citrix Systems, Inc. (edt.), "Xen," http://www.xen.org/.

[72] Oracle, Inc. (edt.), "Virtualbox," http://www.virtualbox.org/.

[73] KVM, "Kernel-based virtual machine," http://www.linux-kvm.org/.

[74] OpenVZ, "Open Virtualization," http://openvz.org/.

[75] Tcpdump, "tcpdump," http://www.tcpdump.org.

[76] IEEE Computer Society, "802.3 Part 3: Carrier Sense Multiple Access with Collision Detection (CSMA/CD) Access Method and Physical Layer Specifications," http://standards.ieee.org/getieee802/download/802.3-2002.pdf, 2002.

[77] A. Waterland, "Stress tool," http://weather.ou.edu/~apw/projects/stress/.

[78] S. Boslaugh and W. P.A., *Statistics in a nutshell.* O'Reilly, 2008.

[79] S. Han, K. Jang, K. Park, and S. Moon, "PacketShader: A GPU-Accelerated Software Router," *ACM SIGCOMM Computer Communication Review*, vol. 40, 2010.

[80] P. Reichl, J. Fabini, M. Happenhofer, and C. Egger, "From QoS to QoX: A Charging Perspective," in *18th ITC Specialist Seminar on "Quality of Experience"*, Karlskrona, Sweden, May 2008.

[81] M. Csikszentmihalyi and J. LeFevre, "Optimal Experience in Work and Leisure," *Journal of Personality and Social Psychology*, vol. 56, 1989.

[82] ITU-T Recommendation, "Mean Opinion Score (MOS) Terminology," *ITU P.800.1*, 2002.

[83] S. Winkler, "Video Quality Measurement Standards - Current Status and Trends," in *7th International Conference on Information, Communications and Signal Processing*, Macau, China, Dec 2009.

[84] ITU-T Recommendation, "Objective Quality Measurement of Telephone-Band (300-3400 Hz) Speech Codecs," *ITU-T P.861*, 1998.

[85] S. Voran, "Objective Estimation of Perceived Speech Quality - Part I: Development of the Measuring Normalizing Block Technique," *IEEE Transactions on Speech and Audio Processing*, vol. 7, 1999.

[86] W. Yang, "Enhanced Modified Bark Spectral Distortion (EMBSD): An Objective Speech Quality Measure Based on Audible Distortion and Cognition Model," Ph.D. dissertation, Temple University, Philadelphia, USA, 1999.

[87] ITU-T Recommendation, "Perceptual Evaluation of Speech Quality (PESQ), an Objective Method for End-to-End Speech Quality Assessment of Narrowband Telephone Networks and Speech Codecs," *ITU P.862*, 2001.

[88] Z. Wang, L. Lu, and A. C. Bovik, "Video Quality Assessment Based on Structural Distortion Measurement," *Signal Processing: Image Communication*, vol. 19, 2004.

[89] Z. Wang, "Rate Scalable Foveated Image and Video Communications," Ph.D. dissertation, University of Texas at Austin, 2001.

[90] Z. Wang, A. Bovik, and L. Lu, "Why is Image Quality Assessment so Difficult?" in *Proceedings of the IEEE International Conference on Acoustics Speech and Signal Processing*, Orlando, FL, USA, May 2002.

[91] S. Voran, "The Development Of Objective Video Quality Measures That Emulate Human Perception," in *IEEE GLOBECOM*, Phoenix, AZ , USA, Dec. 1991.

[92] S. Wolf and M. H. Pinson, "Spatial-Temporal Distortion Metrics for In-Service Quality Monitoring of any Digital Video System," in *Proceedings of SPIE International Symposium on Voice, Video, and Data Communications, Boston, MA, September*, 1999.

[93] Video Quality Experts Group, "Final Report from the Video Quality Experts Group on the Validation of Objective Models of Video Quality Assessment," http://www.vqeg.org/, 2000.

[94] L. Carvalho, E. Mota, R. Aguiar, A. F. Lima, J. N. de Souza, and A. Barreto, "An E-Model Implementation for Speech Quality Evaluation in VoIP Systems," in *Proceedings of the IEEE Symposium on Computers and Communications (ISCC'05)*, La Manga del Mar Menor, Spain, 2005.

[95] ITU-T Recommendation, "The E-model, A Computational Model for Use in Transmission Planning," *ITU G.107*, 1998.

[96] T. A. Hall, "Objective Speech Quality Measures for Internet Telephony, Voice over IP (VoIP) Technology," in *Proceedings of the Society of Photo-Optical Instrumentation Engineers (SPIE)*, 2001.

[97] ITU-T Recommendation, "Transmission Impairments due to Speech Processing," *ITU G.113*, 2001.

[98] T. Hoßfeld, P. Tran-Gia, and M. Fiedler, "Quantification of Quality of Experience for Edge-Based Applications," in *Proceedings of the 20th International Teletraffic Conference (ITC) on Managing Traffic Performance in Converged Networks*, Ottawa, Canada, 2007.

[99] T. Hoßfeld, D. Hock, P. Tran-Gia, K. Tutschku, and M. Fiedler, "Testing the IQX Hypothesis for Exponential Interdependency between QoS and QoE of Voice Codecs iLBC and G.711," in *Proceedings of the 18th ITC Specialist Seminar on Quality of Experience*, Karlskrona, Sweden, May 2008.

[100] M. Claypool and J. Tanner, "The Effects of Jitter on the Perceptual Quality of Video," in *Proceedings of ACM Multimedia*, Orlando, FL, USA, Nov. 1999.

[101] G. Rubino, "Quantifying the Quality of Audio and Video Transmissions over the Internet: the PSQA Approach. In Design and Operations," in *Design and Operations of Communication Networks: A Review of Wired and Wireless Modelling and Management Challenges*, J. Barria, Ed. Imperial College Press, 2005.

[102] M. Ries, O. Nemethova, and M. Rupp, "Motion Based Reference-Free Quality Estimation for H. 264/AVC Video Streaming," in *Proceedings*

of the 2nd International Symposium on Wireless Pervasive Computing (ISWPC), San Juan, Puerto Rico, Feb. 2007.

[103] R. Beuran, M. Ivanovici, B. Dobinson, and P. Thompson, "Network Quality of Service Measurement System for Application Requirements Evaluation," *Simulation Series*, vol. 35, no. 4, 2003.

[104] S. Khirman and P. Henriksen, "Relationship Between Quality-of-Service and Quality-of-Experience for Public Internet Service," in *In Proceedings of the 3rd Workshop on Passive and Active Measurement*, Fort Collins, CO, USA, March 2002.

[105] ITU-T Recommendation, "Estimating End-to-End Performance in IP Networks for Data Applications," *ITU G.1030*, 2005.

[106] J. Welch and J. Clark, "A Proposed Media Delivery Index (MDI)," IETF, Tech. Rep. RFC 4445, April 2006.

[107] S. Khan, S. Duhovnikov, E. Steinbach, and W. Kellerer, "MOS-Based Multiuser Multiapplication Cross-Layer Optimization for Mobile Multimedia Communication," *Advances in Multimedia*, vol. 2007, no. 1, 2007.

[108] ITU-T Recommendation, "Opinion Model for Video-Telephony Applications," *ITU-T G.1070*, 2007.

[109] S. G. Naegele-Jackson, "Network QoS and Quality Perception of Compressed and Uncompressed High-Resolution Video Transmission," Ph.D. dissertation, University Erlangen-Nuernberg, 2006.

[110] K. R. Walsh, "Analyzing the Application ASP Concept: Technologies, Economies, and Strategies," *Communications of the ACM*, vol. 46, no. 8, 2003.

[111] VMware Inc. (edt.), "PCoIP Display Protocol: Information and Scenario-Based Network Sizing Guide," http://vmware.com/files/pdf/VMware-View-PCoIP-Network-Sizing-Guide-IG-EN.pdf, May 2010.

[112] S. Zander and G. Armitage, "Empirically Measuring the QoS Sensitivity of Interactive Online Game Players," in *Proceedings of the Australian Telecommunications Networks and Applications Conference*, Sydney, Australia, Jul. 2004.

[113] L. Deri, "Open Source VoIP Traffic Monitoring," in *Proceedings of the 5th System Administration and Network Engineering Conference (SANE)*, Delft, The Netherlands, May 2006.

[114] W. Chiang, W. Xiao, and C. Chou, "A Performance Study of VoIP Applications: MSN vs. Skype," in *Proceedings of the Multimedia Communnicatinos Workshop (MULTICOMM)*, Istanbul, Turkey, Jun. 2006.

[115] R. Barbosa, C. Kamienski, D. Mariz, A. Callado, S. Fernandes, and D. Sadok, "Performance Evaluation of P2P VoIP application," in *Proceedings of the ACM Network and Operating System Support for Digital Audio and Video Conference (NOSSDAV)*, Urbana-Champaign, IL, USA, Jun. 2007.

[116] ITU-T Recommendation, "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-T P. 862*, 2001.

[117] A. Rix, J. Beerends, M. Hollier, and A. Hekstra, "Perceptual Evaluation of Speech Quality (PESQ) - a New Method for Speech Quality Assessment of Telephone Networks and Codecs," in *Proceedings of the IEEE International Conference on Acoustic Speech and Signal Processing*, Salt Lake City, UT , USA, May 2001.

[118] S. Pennock, "Accuracy of the Perceptual Evaluation of Speech Quality (PESQ) Algorithm," in *Proceedings of the Measurement of Speech, Audio and Video Transmission Quality In Telecommunication Networks Conference (MESAQIN)*, Prague, Czech Republic, Jan. 2002.

[119] K.-T. Chen, C. Huang, P. Huang, and C. Lei, "Quantifying Skype User Satisfaction," in *Proceedings of the 2006 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications (SIGCOMM'06)*, Pisa, Italy, Sep. 2006.

[120] B. Wah and B. Sat, "The Design of VoIP Systems With High Perceptual Conversational Quality," in *Proceedings of the Ubiquitous Multimedia Computing Conference*, Sliema, Malta, Oct. 2009.

[121] B. Sat and B. Wah, "Analyzing Voice Quality in Popular VoIP Applications," *IEEE MultiMedia*, 2009.

[122] L. Deboosere, J. D. Wachter, P. Simoens, F. D. Turck, B. Dhoedt, and P. Demeester, "Thin Client Computing Solutions in Low- and High-Motion Scenarios," in *Proceedings of the Integrated Communications Navigation and Surveillance (ICNS) Conference*, Athens, Greece, June 2007.

[123] Citrix Systems, Inc., "SpeedScreen Latency Reduction Explained," White Paper, December 2000.

[124] N. Tolia, D. G. Andersen, and M. Satyanarayanan, "Quantifying Interactive User Experience on Thin Clients," *IEEE Computer*, vol. 39, 2006.

[125] Juniper Networks, "Tuning Citrix for Optimal Performance with the WX/WXC Platforms," Application Note, August 2006.

[126] Citrix Systems, Inc., "Optimizing Citrix Technology for Operation Over Wireless Wide Area Networks," White Paper, April 2003.

[127] ——, "Networking Issues Affecting Citrix MetaFrame Environments," White Paper, April 2003.

[128] ——, "Citrix Independent Computing Architecture Client on Nokia 9210 Communicator," White Paper, October 2003.

[129] Skype Limited, "Silk: Super wideband audio codec," http://developer.skype.com/silk, last accessed May, 27th, 2010.

[130] C. Wu, K. Chen, C. Huang, and C. Lei, "An empirical evaluation of voip playout buffer dimensioning in skype, google talk, and msn messenger," in *Proceedings of the 18th international workshop on Network and operating systems support for digital audio and video*, Braunschweig, Germany, May 2009.

[131] ITU-T Recommendation, "Application Guide for Objective Quality Measurement Based on Recommendations P.862, P.862.1 and P.862.2," *ITU-T P. 862.3*, 2007.

[132] M. Fiedler, "On Resource Sharing and Careful Overbooking for Network Virtualization," *International Journal of Communication Networks and Distributed Systems*, vol. 6, 2011.

[133] M. Weiß, "Measurement-based Evaluation of the Quality-of-Experience of Thin-Client Architectures," Master's thesis, Chair for Communication Networks (Informatik III), University of Wuerzburg, 2007.

[134] M. Carson and D. Santay, "NIST Net: A Linux-based Network Emulation Tool," *SIGCOMM Computer Communications Review*, vol. 33, no. 3, 2003.

[135] WinPcap, http://www.winpcap.org/, last accessed: January 2008.

[136] AutoHotkey, http://www.autohotkey.com/, last accessed: January 2008.

[137] Arlington, VA, USA.

[138] V. Räisänen and J. Wiley, *Implementing Service Quality in IP networks*. Wiley Online Library, 2003.

[139] J. Cao, W. Cleveland, Y. Gao, K. Jeffay, F. Smith, and M. Weigle, "Stochastic Models for Generating Synthetic HTTP Source Traffic," in *Proceedings of the IEEE INFOCOM 2004*, Hong Kong, China, March 2004.

[140] D. Staehle, K. Leibnitz, and P. Tran-Gia, "Source Traffic Modeling of Wireless Applications," University of Würzburg, Institute of Computer Science, Tech. Rep. 261, June 2000.

[141] A. Dainotti, A. Pescapé, P. Rossi, F. Palmieri, and G. Ventre, "Internet Traffic Modeling by Means of Hidden Markov Models," *Computer Networks*, vol. 52, no. 14, 2008.

[142] M. Li and S. Lim, "Modeling Network Traffic Using Generalized Cauchy Process," *Physica A: Statistical Mechanics and its Applications*, vol. 387, 2008.

[143] N. Basher, A. Mahanti, A. Mahanti, C. Williamson, and M. Arlitt, "A comparative analysis of web and peer-to-peer traffic," in *Proceedings of the 17th International Conference on World Wide Web*, Beijing, China, Aug. 2008.

[144] D. Bonfiglio, M. Mellia, M. Meo, and D. Rossi, "Detailed Analysis of Skype Traffic," *IEEE Transactions on Multimedia*, vol. 11, 2009.

[145] M. Menth, A. Binzenhöfer, and S. Mühleck, "Source Models for Speech Traffic Revisited," *IEEE/ACM Transactions on Networking*, vol. 17, 2009.

[146] A. Lazaris, P. Koutsakis, and M. Paterakis, "A New Model for Video Traffic Originating from Multiplexed MPEG-4 Videoconference Streams," *Performance Evaluation*, vol. 65, 2008.

[147] M. Dai, Y. Zhang, and D. Loguinov, "A Unified Traffic Model for MPEG-4 and H. 264 Video Traces," *IEEE Transactions on Multimedia*, vol. 11, 2009.

[148] M. Zink, K. Suh, Y. Gu, and J. Kurose, "Characteristics of YouTube Network Traffic at a Campus Network-Measurements, Models, and Implications," *Computer Networks*, vol. 53, 2009.

[149] R. Antonello, S. Fernandes, J. Moreira, P. Cunha, C. Kamienski, and D. Sadok, "Traffic Analysis and Synthetic Models of Second Life," *Multimedia Systems*, vol. 15, 2009.

[150] The Tolly Group, "MetaFrame XP Presentation Server Windows-based Application Access Performance and Functionality," 2003.

[151] P. Simoens, B. Vankeirsbilck, L. Deboosere, F. Ali, F. De Turck, B. Dhoedt, and P. Demeester, "Upstream Bandwidth Optimization of Thin Client Protocols Through Latency-Aware Adaptive User Event Buffering," *International Journal of Communication Systems*, vol. 24, 2011.

[152] I. Humar, J. Bester, and S. Tomazic, "Characterizing Graphical Desktop Sharing System's Workload in Collaborative Virtual Environments," in *Proceedings of the Consumer Communications and Networking Conference (CCNC)*, Las Vegas, NV, USA, Jan. 2009.

[153] T. Yu, Y. Zhang, and K. Lin, "Efficient Algorithms for Web Services Selection with End-to-End QoS Constraints, Efficient Algorithms for Web Services Selection with End-to-End QoS Constraints," *ACM Transactions on the Web (TWEB)*, vol. 1, 2007.

[154] L. Huang, S. Kumar, and C. Kuo, "Adaptive Resource Allocation for Multimedia QoS Management in Wireless Networks," *IEEE Transactions on Vehicular Technology*, vol. 53, 2004.

[155] P. Arlos and M. Fiedler, "A Method to Estimate the Timestamp Accuracy of Measurement Hardware and Software Tools," *Passive and Active Network Measurement*, vol. 4427, 2007.

[156] ——, "Accuracy Evaluation of Ping and J-OWAMP," in *Proceedings of the Swedish National Computer Networking Workshop*, Luleå, Sweden, Oct. 2006.

[157] P. Arlos, "On the Quality of Computer Network Measurements," Ph.D. dissertation, Blekinge Tekniska Högskola, 2005.

[158] D. Croce, T. En-Najjary, G. Urvoy-Keller, and E. Biersack, "Capacity Estimation of ADSL links," in *Proceedings of the 2008 ACM CoNEXT Conference*, Madrid, Spain, Dec. 2008.

[159] P. Arlos and M. Fiedler, "Influence of the Packet Size on the One-Way Delay in 3G Networks," in *Proceedings of the 11th International Conference on Passive and Active Measurement (PAM)*, Zurich, Switzerland, April 2010.

[160] T. Santos, C. Henke, C. Schmoll, and T. Zseby, "Multi-Hop Packet Tracking for Experimental Facilities," in *Proceedings of the ACM SIGCOMM Conference*, New Delhi, India, Aug. 2010.

[161] Cisco.com (edt.), "Cisco IOS IP Service Level Agreements Command Line Interface," Cisco Systems, Inc., Tech. Rep., Mar. 2005.

[162] J. Sommers, P. Barford, N. Duffield, and A. Ron, "Improving Accuracy in End-to-End Packet Loss Measurement," *ACM SIGCOMM Computer Communication Review*, vol. 35, 2005.

[163] R. Armolavicius, "Simple Approximations of Delay-Variation Distributions," in *Proceedings of the 18th ITC Specialist Seminar on Quality of Experience*, Karlskrona, Sweden, May 2008.

ISSN 1432-8801