UNIVERSIDAD POLITÉCNICA DE MADRID

Escuela Técnica Superior de Ingenieros de Telecomunicación

**ARCHITECTURES FOR STREAMING IN P2P AND CLOUD
SCENARIOS WITH SDN-MANAGED QOS COST FUNCTIONS**

TESIS DOCTORAL

Irena Trajkovska
Ingeniero de Telecomunicación

Madrid, España
Julio, 2017

Departamento de Ingeniería de Sistemas Telemáticos

Escuela Técnica Superior de Ingenieros de Telecomunicación

**ARCHITECTURES FOR STREAMING IN P2P AND CLOUD SCENARIOS WITH
SDN-MANAGED QOS COST FUNCTIONS**

TESIS DOCTORAL

Autor: Irena Trajkovska
Ingeniero de Telecomunicación

Director: Joaquín Salvachúa Rodríguez
Doctor Ingeniero de Telecomunicación

Julio, 2017

Tribunal nombrado por el Magnífico. y Excelentísimo. Sr. Rector de la Universidad Politécnica de Madrid, el día 20 de Junio de 2017.

Presidente: _____

Vocal: _____

Vocal: _____

Vocal: _____

Secretario: _____

Suplente: _____

Suplente: _____

Realizado el acto de defensa y lectura de la Tesis el día 17 de Julio de 2017 en Madrid, habiendo obtenido la calificación de

El presidente,

El secretario,

Los vocales,

*To my parents Vesna and Zoran,
to my grandparents Vasko, Nada, Trajanka and Ladoimir.*

Acknowledgments

The research process that resulted in this thesis was accompanied by various challenges and acknowledgements, extrapolated over several years and involving important life aspects. I moved to Spain to dedicate to the phd work, which i completed one year after i moved to Switzerland. In this journey i met many exceptional people that made a significant impact not only to the work produced in this thesis, but to my personality as well. In this part i would like to give them my respect and gratitude. First and foremost my warmest gratitude goes to my parents and family, my cousins Ivana, Tanja, Anja and Jana, along with Rémy for their love, support and understanding that have been essential during the entire research process. Special thanks for my advisor, profesor Dr. Joaquín Salvachúa who was always full of ideas and positive thinking and who followed my advances after i moved to Switzerland. I also would like to express my thankfulness to profesor Dr. Juan Quemada who opened the door of the group *Internet de nueva generación* of *Departamento de Ingeniería Telemática, DIT* and let me take part in one of the most proactive and applied labs at the ETSIT, UPM. Thank you to the professors Dr. Gabriel Huecas and Dr. Santiago Pavón Gómez, and Dr. David Fernández Cambronero, also thanks Lola and Omar Walid Llorente from *Centro de Cálculo de DIT* for being so helpful. I would like to further give my regards to all my colleagues from the DIT group who i had the chance to work with or be around during the nice lunch discussions: Abel who always reflected back my positive mood, the ever smiling Sandra for her joint companionship in many nice discussions and shared experiences, Javi and Pedro - my close cooperators in this work and colleagues to learn many cool things from, and both of them with Alvaro for the nice times during the MWC hackathon we had as a team; Tapi the open source master, and our glorious "Los Macedonios" band performance between Tapi, Alvaro and Abel; my dear Carolina who valued me, and was always there to understand my point and help me out in big doubts; Dani and Ivan for the master days spent together and their proactivity as a big motivation; Kike, Barce, Emilio, then Aldo, and all the younger students along with the interns like Hao, Lui; our dear graphic designers Nestor, Juanca and Crispin who improved the user interface and the QoE of the group, and for many more who took part of DIT and with whom i shared valuable moments. I would like to furthermore mention my dear friends in Spain that mean a lot to me: Noemi, Laura, Eva, Amelie, Pina, Eva, Carlos, Sasho and Ivan (aka chicos), Ane, Kate, Perun, Isma, Leti, Augusto, Eva, Pina, Olivera, Dime, "los argentinos", Josemi and many more. The thesis took a different course after my research stay in Switzerland, and for all the confidence, activities related and the opportunities offered, i would like to thank profesor Thomas Michael Bohnert, who made me a

part of the great ICCLab team to date. In the same time big thanks to Andy Edmonds, Christof Marti, Dr. Piyush Harsh, Dr. Giovanni Toffetti, Dr. Josef Spilner and Dr. Sean Murphy for great cooperation in projects, proposals and interesting research discussions. Several people have i worked with at the lab, and some have taken an important role in my social activities. A vibrant, international ambient from experienced researchers and professors, to exceptional young students and engineers. It was a special experience working with Denis Baudinot and fruitful outcomes has been shared together from our SDN initiative. Thanks to Amrita Prasad for the help and cooperation in organizing the Swiss SDN workshop/meetup; Philipp Aschlimann who brought me into the SDN world together with Thomas, my ICCLab hackathon teammate Vincenzo who showed us Tuscany; Victor, Valon and Michael from the alumni guys that made a great job, Vojtech for the peculiar topics we could talk about, the fun Spanish crew - Lidia, Manu and Manolo, my dear Ukrainian colleagues - Oleksii who brought the ladies a rose for 8th of March and who shakes our hands for *good morning*, and the friendly hand and food specialties of Serhii, Martin for the fun pool game, Saken for the nice Kazakh dinner, Franco for the exceptional cordiality and wide smile, Tobi for sharing his ski expertise, Sandro - the movie expert for the nice choice of movies we watched, Sri for the amazing Indian wedding experience, Balazs for his honesty and big heart, Bruno for the OpenStack discussions, the laughs and all the fun times, foremost for being a good friend and very nice person to me. The young guys were always around, like Gabriel, my students Luca and Yannic, and thanks Janine for the great logos of Netfloc and Netflogi. I had spend nice moments and met new friends in Switzerland, so a special respect goes to Kriti and Andrea who always supported me in my work and have been of big understanding and always there as friends, Aleks my dearest flatmate and friend together with Ana who always believed in my potentials, boosting me up in accomplishing my research and professional goals. Thanks to Florian for the eternal discussions of the world and the universe, for showing me appreciation, encouragement, care and thanks for the faithful friendship. During the work in the projects, some remarkable contributions have taken place in cooperation with foreign partners and academic institutions. I would hereby express my acknowledgements to: Dr. Tasos Kourtis, George Xilouris, Eleni Trouva (NCSRD *Demokritos*), Dr. Panagiotis Papadimitriou (University of Macedonia), Dr. Letterio Zuccaro (CRAT), Michael Mcgrath (Intel), Dr. Ioannis Chochliouros (OTE), Dr. Begoña Blanco and Dr. Jose Oscar Fajardo (EHU), Dr. Pouria Khodashenas, Josep and Jordi (i2CAT), Javier, Aurora (Atos), Alexandros (PrimeTel), João (Alticelab), Kurt Bauman (SWITCH), and many other colleagues i've cooperated with over the course of this thesis. My appreciation for the researchers and developers Alexandros Kourtis and Christos Sakkas, for the many hours of integration, eternal OpenStack-SDN-NFV tweaks, tests and measurements, but also funny moments, and even funnier last minute dental arrangement by Akis. Last but not least, huge gratefulness to all Macedonian friends who has been a tremendous source of love, friendship, support and eternal relation - stronger than the long distance: Racka, Maca, Lide, Meri, Aleksandra, Nate, Kate, Snenska, Zlate, Goran, Aneta, Frose, Stole, Sanja,...

Abstract

In over three decades timeline the peer-to-peer technology was a foundation for numerous advancements in the domain of file sharing, video on-demand and real-time communication services. Among them, providing live streaming video over P2P networks was one of the most proactive and challenging services. Despite becoming a replacement of the costly client-server models from the past, it required reliable modular operation in order to make a holistic working service. Major issue in this approach is maintaining a robust overlay topology under dynamic network quality conditions and in the presence of peer churn. The user behavior furthermore has been a subject of many incentive mechanisms to encourage altruism vs. free-riding and mitigate malicious users. Finally, ensuring sensible QoS in live video delivery is a recurring topic to date.

The cloud computing milestone had a dominant influence in the design of new applications and services. It grew into de facto technology for value-added services, due to its enhanced business strategy bound to on-demand lease of elastic virtual resources with attractive pay-per-use policies. The cloud providers have moreover enabled flexible provisioning of robust and scalable services.

This thesis offers several contributions in terms of architecture, algorithms and technical solutions that have as objective enhancing the classical P2P streaming solutions with novel elements for achieving quality of service-based tiered business model, complemented with incentives based on discount to achieve improved system welfare. The set of contributions uses the converged P2P and cloud technologies to offer enhanced features and services. First of all we created a hybrid architecture for live streaming to address some of the common problems in P2P streaming - scalability, incentives and QoS. To assess the advantages and the feasibility of the proposal, we emulated realistic use-case scenarios in a mixed cloud-P2P environment. We elaborated later a discount incentive model bound to the streaming service that offers a monetary remuneration to motivate increased cooperation. The analytical study of the incentives pointed out the benefits in the streaming service to users' welfare and the provider's profit.

The second phase onboarded the SDN technology in the scope of contributions, driven by its attribute of full network manageability governed by a central control entity. The seamless control of network segments that were previously out of range for the service developers, catalyzed major opportunities in SDN. Founded on the QoS control primitives from the OpenFlow specification, a proof of concept service for SDN-based QoS adjustment is presented and APIs are offered to service developers in order to trigger external quality assessment for real-time video services. Finally a service function chain algorithm and proof of concept is delivered as value-added service contribution in the context of video quality adjustment using network function virtualization.

Resumen

En más de tres décadas, la tecnología peer-to-peer fue una base para numerosos avances en el dominio del uso compartido de archivos, video bajo demanda y servicios de comunicación en tiempo real. Entre ellos, la transmisión de video en vivo a través de redes P2P fue uno de los servicios más proactivos y desafiantes. A pesar de convertirse en un sustituto de los costosos modelos cliente-servidor del pasado, se requiere una operación modular fiable con el fin de hacer un servicio integral. El principal problema en este enfoque es el mantenimiento de una topología de superposición robusta bajo condiciones dinámicas de calidad de red y en presencia de "churn" de pares. El comportamiento del usuario, además, ha sido objeto de muchos mecanismos de incentivo para fomentar el altruismo frente a los "free-riders" y mitigar a los usuarios malintencionados. Por último, asegurar QoS sensible en la entrega de video en vivo es un tema recurrente hasta la fecha.

El hito de la computación en la nube tuvo una influencia dominante en el diseño de nuevas aplicaciones y servicios. Se convirtió en tecnología de facto para los servicios de valor añadido, debido a su estrategia de negocio mejorada vinculada al alquiler bajo demanda de recursos virtuales elásticos con políticas atractivas de pago por uso. Los proveedores de la nube han permitido, además, el aprovisionamiento flexible de servicios robustos y escalables.

Esta tesis ofrece varias contribuciones en términos de arquitectura, algoritmos y soluciones técnicas que tienen como objetivo mejorar las clásicas soluciones de streaming P2P con elementos novedosos para lograr un modelo de negocio basado en la calidad de servicio, complementado con incentivos basados en descuento para mejorar el bienestar del sistema. El conjunto de contribuciones utiliza las tecnologías convergentes P2P y cloud para ofrecer características y servicios mejorados. En primer lugar, hemos creado una arquitectura híbrida para la transmisión en vivo y así abordar algunos de los problemas comunes en la transmisión en tiempo real P2P: escalabilidad, incentivos y QoS. Para evaluar las ventajas y la viabilidad de la propuesta, emulamos escenarios realistas de casos de uso en un ambiente mixto de cloud-P2P. Más tarde elaboramos un modelo de incentivo de descuento vinculado al servicio de streaming que ofrece una remuneración monetaria para motivar una mayor cooperación. El estudio analítico de los incentivos destacó los beneficios en el servicio de streaming para el bienestar de los usuarios y los beneficios del proveedor.

La segunda fase de investigación incluyó la tecnología SDN en el ámbito de las contribuciones, impulsadas por su atributo de gestión de red completa gobernada por una entidad de control central. El control robusto y continuo de los segmentos de red que antes estaban fuera

de rango para los desarrolladores de servicios, dio lugar a grandes oportunidades de nuevos desarrollos basados en SDN. Fundada en las primitivas de control QoS de la especificación OpenFlow, se presenta un servicio de prueba de concepto para ajuste de QoS basado en SDN y se ofrecen APIs a los desarrolladores de servicios para activar la evaluación de calidad externa para servicios de video en tiempo real. Finalmente, un algoritmo de cadena de funciones de servicio y una prueba de concepto se entrega como contribución de servicio de valor añadido, en el contexto del ajuste de calidad de vídeo mediante la virtualización de funciones de red.

Contents

Acknowledgments	
Abstract	
Resumen	
List of Illustrations	
List of Tables	
List of Acronyms	

1	Introduction	1
1.1	Objectives	8
1.2	Research Methodology	10
1.3	Related Projects	11
1.3.1	FIWARE - A Core Platform of the Future Internet	11
1.3.2	T-NOVA - Network Functions-as-a-Service over virtualized Infrastructures	12
1.3.3	SESAME - Small cEllS coordinAtion for Multi-tenancy and Edge services	13
1.4	Structure of this Document	13
1.4.1	Chapter 2: State of the Art	14
1.4.2	Chapter 3: Architecture for Cloud-based P2P Streaming	14
1.4.3	Chapter 4: Business Model for Cloud-based P2P Streaming	14
1.4.4	Chapter 5: SDN-based QoS Assurance for Cloud-based Streaming	14
1.4.5	Chapter 6: Validation and Results	14
1.4.6	Chapter 7: Conclusions	15
2	State of the Art	17
2.1	P2P Overlay Design, Architectures and Algorithms	17
2.1.1	Video delivery architectures	17
2.1.2	P2P taxonomy and overlay design	19
2.2	P2P Multimedia Streaming	27
2.2.1	QoS in P2P steaming	28
2.2.2	Incentives in P2P steaming	30
2.2.3	Web-based P2P streaming	32
2.2.4	Cloud-based P2P streaming	35
2.3	Empowering SDN for Value-added Services	40
2.3.1	QoS- and streaming-related research trends in SDN	40
2.3.2	SDN technology specifiers	42
2.3.3	State of the Art in Service Function Chaining	44
3	Architecture for Cloud-based P2P Streaming	51
3.1	Problem Statement	51
3.1.1	P2P-based streaming	51

3.1.2	Cloud-based streaming	52
3.2	Architecture Description	53
3.2.1	APIs and service for streaming	56
3.2.2	Quality of service in streaming	59
3.3	Assessing Cloud Infrastructures for P2P Streaming	61
3.3.1	Measurements in distributed cloud scenarios	62
3.3.2	Characterizing cloud network for streaming	67
3.4	Contributions Summary and Conclusions	73
4	Business Model for Cloud-based P2P Streaming	75
4.1	Problem Statement	75
4.2	Behavior in P2P Streaming Systems	77
4.2.1	Cooperative users	77
4.2.2	Uncooperative users	77
4.2.3	Malicious users	78
4.3	Discount Incentives in P2P Streaming Service	78
4.3.1	Topology and business model	80
4.3.2	User's utility	83
4.3.3	Provider's utility	84
4.4	Extending the Incentive Model to a P2P-cloud Service	86
4.4.1	Welfare modeling	86
4.4.2	Case study	88
4.4.3	Cloud instances	91
4.4.4	Analysis and results	94
4.5	Contributions Summary and Conclusions	101
5	SDN-based QoS Assurance for Cloud-based Streaming	103
5.1	Problem Statement	103
5.2	SDN Application for QoS control in OpenFlow-enabled Networks	104
5.2.1	KIARA as SDN enabled middleware	106
5.2.2	Prototype application for QoS control	109
5.2.3	APIs for QoS control in residential networks	113
5.3	Service Function Chaining and Prototype Implementation in NFV Scenario	116
5.3.1	Netfloc: SDK for SDN-based solution in datacenters and NFV	120
5.3.2	Main architecture building blocks	121
5.3.3	Netfloc's SFC methodology and test scenarios	124
5.3.4	Validation of SFC proof of concept	132
5.4	Contributions Summary and Conclusions	142
6	Validation and Results	147
6.1	Validation in European Projects	147
6.1.1	FIWARE - A Core Platform of the Future Internet	147
6.1.2	T-NOVA - Network Functions-as-a-Service over virtualized Infrastructures	148
6.1.3	SESAME - Small cEllS coordinAtion for Multi-tenancy and Edge services	148
6.2	Dissemination of Results	151
6.2.1	Publications	151
6.2.2	Standards and open-source communities	153

CONTENTS

6.2.3	Research visit	155
6.2.4	Teaching and talks	156
7	Conclusions	157
7.1	Contributions	158
7.2	Future research	165
	Bibliography	167

CONTENTS

Illustrations

1.1	Timeline of the most famous network protocols and applications for sharing . . .	2
1.2	The contributions of the thesis	9
2.1	Taxonomy of P2P overlays [p2p b]	21
2.2	Multi-tree overlay with two delivery trees	23
2.3	Organized view of a random mesh	25
2.4	Early building blocks of WebRTC and additionally proposed ones to achieve P2P Web-based streaming	34
3.1	Hybrid client-server and P2P streaming cloud architecture	55
3.2	Example Topology with QoS Table	59
3.3	Node distribution	65
3.4	The 99.7th percentile of jitter in each scenario	65
3.5	Bytes received in each scenario	66
3.6	Packet losses in each scenario	66
3.7	Architecture of the measuring environment	68
3.8	Measurement results between datacenters EU-US	70
3.9	Measurement results between datacenters EU-AP	71
3.10	Measurement results between datacenters US-AP	72
4.1	SplitStream [Castro 2003b] overlay topology	78
4.2	P2P-cloud architecture with multi-tree overlay for video dissemination	80
4.3	Service topology showing the user placement according to the price plan	82
4.4	Provider's welfare change when 20% of the users are high capacity for 500 fractions of gold, silver and bronze users	89
4.5	Provider's welfare change for one user group depending on the users' cooperation and heterogeneity	90
4.6	Three types of system topology depending on the <i>Coefficient of effective cooperation</i>	93
4.7	Total users' discount. <i>x axis</i> : coop_highBW (100_80, 100_60, 100_20, 80_80, etc.); <i>y axis</i> : user fractions (gold, silver and bronze).	96
4.8	Provider's revenue (EUR). <i>x axis</i> : coop_highBW (100_80, 100_60, 100_20, 80_80, etc.); <i>y axis</i> : user fractions (gold, silver and bronze).	97
4.9	Instance(C_i) and transfer(C_i) cost change depending on the users' heterogeneity .	98
4.10	Profit change depending on the users' cooperation for different bandwidth capacities. <i>x axis</i> : coop_highBW(100_80, 100_60, 100_20, 80_80, ..etc.); <i>y axis</i> : user fractions.	98

4.11 Profit change depending on the users' cooperation for different bandwidth capacities. <i>x axis</i> : user fractions(from majority gold-left to majority bronze and mixed-right); <i>y axis</i> : coop_highBW	100
5.1 Diagram of the KIARA middleware	108
5.2 Negotiation with QoS retrieval and control in KIARA	108
5.3 SDN testbed for QoS retrieval and control	111
5.4 SDN based residential network for dedicated streaming services	114
5.5 Adjusting bandwidth between nodes in a multicast tree using OpenFlow	116
5.6 Components of T-Nova and their logical relations	119
5.7 Netfloc's internal components and South & Northbound APIs to T-Nova and third party applications (an early version of this figure appeared in Deliverable D4.31 of the T-Nova project [del a]).	121
5.8 Overview of the Virtual Traffic Classifier (Figure 3-1 of T-Nova Deliverable 4.32 [del b]).	122
5.9 Overview of the Virtual Media Transcoder (Figure 3-2 of T-Nova Deliverable 4.32 [del b]).	123
5.10 Simple traffic steering scenario in OpenStack all-in-one setup. Shows OVS and Linux interfaces on the Control node, OpenFlow based port matching, and header rewriting example.	125
5.11 NFVI-PoP Service chain details. The yellow line is common for both chains, the red line: Chain1, the blue line: Chain3. Chain2 is the combination of both, Chain1 and Chain3 (Figure 3-5 of T-Nova Deliverable 4.32 [del b]).	127
5.12 State Machine Diagram of the SFC mechanism in Netfloc	128
5.13 Flow entries programmed by Netfloc on the virtual switches of the OpenStack nodes (Control and Neutron) and the SDN switch. The uppermost terminal window shows the SFC process in Netfloc after the create chain API call. The highlights in yellow underline: virtual mac address matching on the switch and the control node (3rd, 4th upper window) and VLAN matching on the neutron node (2nd upper window).	130
5.14 NFVI-PoP physical overview (Figure 3-4 of T-Nova Deliverable 4.32 [del b] and Figure 9 of T-Nova Deliverable 7.1 [del c]).	131
5.15 High level view of the SFC test scenario (Figure 3-3 of T-Nova Deliverable 4.32 [del b]).	132
5.16 SFC sequence diagram	133
5.17 OpenStack representation of the replicated chaining scenario. It depicts the networks attached to each of the VNFs.	135
5.18 Throughput for: no-chain, Chain1 and Chain2 scenarios	144
5.19 Throughput for: 2 chains, 4 chains and 6 chains scenarios	144
5.20 TCP/UDP throughput for: no-chain, Chain1, Chain2, 4 and 6 chains scenarios	144
5.21 SSIM Video quality assessment results in relation to traffic interference (Figure 3-10 of T-Nova Deliverable 4.32 [del b])	144
5.22 Packet loss for: no-chain, Chain1 and Chain2 scenarios	144
5.23 Packet loss for: 2 chains, 4 chains and 6 chains scenarios	144
5.24 Latency for: no-chain, Chain1 and Chain2 scenarios	145
5.25 Latency for: 2 chains, 4 chains and 6 chains scenarios	145

ILLUSTRATIONS

5.26	Jitter for: no-chain, Chain1 and Chain2 scenarios	145
5.27	Jitter for: 2 chains, 4 chains and 6 chains scenarios	145
5.28	Video transcoding and watermark embedding by the vMT. Left: ffmpeg video processing output on User2. Right: final video result.	145
6.1	Joint SESAME-COHERENT demo architecture shown in EUCNC'16	149
6.2	LTE/LTE-A network deployed with Juju and OpenAir interface	150

ILLUSTRATIONS

Tables

2.1	Technical specification of the current SFC implementations	46
3.1	Streaming plans including current users and their QoS	56
3.2	User Status Table	57
3.3	Applied commands for client and server mode	70
4.1	Evaluation parameters	95
4.2	Profit improvement vs cost per hour (in euros)	101
5.1	Demo scenario hardware specifications	139

TABLES

Acronyms

P2P	Peer-to-Peer
QoS	Quality of Service
QoE	Quality of Experience
DARPA	Defense Advanced Research Projects Agency
ARPANET	Advanced Research Projects Agency Network
NNTP	Network News Transfer Protocol
UUCP	Unix-to-Unix-copy protocol
DHT	data hash tables
RTP	Real-time Protocol
RTCP	RTP Control Protocol
CDN	Content Delivery Network
DASH	Dynamic Adaptive Streaming
SDN	Software Defined Networking
CAPEX	Capital Expenditure
OPEX	Operational Expenditure
MDC	Multiple description coding
SVC	Scalable Video Coding
VoD	Video on Demand
IPTV	Internet Protocol Television
P2PTV	Peer-to-Peer-Television
RTC	Real-Time Communication
STUN	Session Traversal Utilities for NAT
TURN	Traversal Using Relays around NAT
ICE	Interactive Connectivity Establishment

TABLES

NAT	Network Address Translation
REST	Representational State Transfer
WebRTC	Web Real Time Communications
IntServ	Integrated Services
DiffServ	Differentiated Services
MPLS	Multiprotocol Label Switching
ToS	Type of Service
SNMP	Simple Network Management Protocol
NETCONF	Network Configuration Protocol
VLAN	Virtual Local Area Network
VxLAN	Virtual Extensible Local Area Network
GRE	Generic Routing Encapsulation
NSH	Network Service Header
ODL	OpenDaylight
SFC	Service Function Chain
MAC	Media Access Control
APR	Address Resolution Protocol
NIC	Network Interface Card
ML2	Modular Layer 2
OVS	Open vSwitch
OSGi	Open Service Gateway Initiative
ONOS	Open Network Operating System
ADSL	Asymmetric Digital Subscriber Line
xDSL	all types of Digital Subscriber Line (DSL)
UMTS	Universal Mobile Telecommunications Service
API	Application Programming Interface
IPGs	Inter-Packet Gaps
VoIP	Voice Over Internet Protocol

TABLES

ETSI	European Telecommunications Standards Institute
RTT	Round Trip Time
CDF	Cumulative Distribution Function
DC	Data Center
Light DC	Light Data Center
ISP	Internet service provider
SD	Standard-definition
HD	High-definition
PoC	Proof of Concept
RTI-DDS	Real-Time Innovations Data Distribution Service
FI-PPP	Future Internet Public-Private Partnership Programme
TSP	Telecommunication Service Provider
VNF	Virtual Network Function
VNFFG	Virtual Network Function Forwarding Graph
NS	Network Service
VIM	Virtual Infrastructure Manager
NFV	Network Function Virtualization
NFVI	Network Function Virtualization Infrastructure
SDK	Software Development Kit
vTC	virtual Traffic Classifier
vMT	virtual Media Transcoder
VNFaaS	Virtual Network Function-as-a-Service
WICM	Wide Area Network Infrastructure Connection Manager
CE	Customer Edge
PE	Provider Edge
VPN	Virtual Private Network
PoP	Point of Presence
WG	Working Group

ECN Explicit Congestion Notification

DPDK Dataplane Development Kit

SSIM Structural Similarity Index

GPU Graphics Processing Units

DSP Digital Signal Processors

FPGA Field-Programmable Gate Arrays

RAN Radio Access Network

LTE Long-Term Evolution

MANO Management and Orchestration

Chapter 1

Introduction

The chronicle of the peer-to-peer (P2P) concept has marked numerous milestones and technology advancements during a timeline for over three decades. As the name suggest P2P denotes a communication between two equal parties called peers, with the purpose of exchanging information or data files via server-less connection. The file sharing process dates since the Internet appeared in the seventies and it was further promoted by dedicated architectures created from several big universities and companies. For example, the Defense Advanced Research Projects Agency (DARPA) introduced the project ARPANET with the aim to interconnect computers of four universities: UCLA, Stanford Research Institute, UC Santa Barbara and the University of Utah, all as equal peers in the network. The post-coming Telnet and File Transfer Protocol followed this project, which although designed as a client-server, allowed to configure a direct connection between two equal hosts on the Internet. DARPA founded the research behind the TCP/IP protocol that appeared in the 1974 with the objective to provide end-to-end data communication by presenting a complete specification for packet manipulation (addressing, transmission, routing and reception) on the Internet. The Xmodem afterwards, was a point-to-point binary transfer protocol for exchanging files between computers. The high modem costs at the time, hindered the communication between the mainframe computers, which resulted in introduction of the microprocessors. This started the era of the Bulletin Board Systems (BBS) as computers that allowed users to share data files and information news (bulletins) via terminal. It started as a dial-up BBS to evolve in a Computerized BBS in the year 1978. The next year, 1979 was remarked by the publishing of the Unix-to-Unix-copy protocol (UUCP) that facilitated a direct connection between two Unix machines and with this open a communication pipeline for exchange of files, email and news. The UUCP was the underlying protocol used by the Usenet distributed discussion system, introduced in the 1979 with the joint effort between the Duke University and the University of North Carolina graduates. Having a similar purpose, the Network News Transfer Protocol (NNTP) introduced a bit later in 1986, was soon after adopted by the Usenet network. Early P2P applications have been adopted and supported by industry leaders such as Boeing and Intel, with clear benefits for e.g. to reduce the chip production cost in the case of Intel [p2p a]. Figure 1.1 reflects the historical evolution of the file sharing applications and protocols to date.

Napster revolutionized the P2P world as soon as it appeared in 1999 and it fostered a whole new era of communication on the Internet. The users downloaded a program that was able to search the local disk of the linked computers and send them directly the desired files. Mainly mp3

Figure 1.1 : Timeline of the most famous network protocols and applications for sharing

music files composed the traffic load within Napster that lead to legal lawsuits due to violated copyright laws and consequently led to the cease of Napster in 2001. Nonetheless, the huge success was followed by proliferation of new services including: Gnutella, eDonkey, LimeWire, Freenet, Kazaa, Audiogalaxy, iMesh, eMule etc., that marked the second generation P2P networks based on fully-distributed query search method using flooding and swarming. Immediately their popularity exponentially grew resulting in over 3.05 billion files downloaded in 2001 vs. the 2.79 billion downloads on Napster earlier that year. They substituted the central server database, with distributed file lists containing information of the nodes and the file availability in each of them. This topological shift from centralized to hybrid or fully distributed model not only mitigated the bottleneck issue, but increased the overall resource reserves in the system (memory, CPU, network), while providing highly available infrastructure (nodes and data resources for share). The network moreover became resilient to failure and rich in resources, especially bandwidth.

The high routing costs of flooding-based file querying lead to increased overhead across the entire overlay, due to the vast amount of control message exchanges. The need for "smarter" file sharing/lookup was satisfied by the third generation P2P systems based on data hash tables (DHT) as distributed structures that looked up a file within the organized overlay structure among the nodes. This new concept was brought forward by the following academic research projects: CAN,

Chord and Pasty in 2001, followed by Kademia and Tapestry. These P2P systems improved the overlay structuring, message routing, and object placement. The good scalability with $O(\log N)$ complexity and the resistance to dynamic changes, failures and attacks is what endorsed the DHT based P2P algorithms as a base for many applications that followed.

BitTorrent was published as in 2002 and has ever since remained the de-facto protocol and client for Internet content sharing to date. Based on a new strategy called "tit-for-tat", it promoted alternative way to confront the unfair distribution of files and resources. Unlike many of its predecessors it managed to circumvent all the legal concerns and has been an inspiration for many client applications to follow. The Pirate Bay was established in 2003 and currently still exists despite the charges it received. Spotify lunch was the legitimate come back of a digital music service, seven years after Napster shut down. Though with a different model from Napster, Spotify bases on a P2P underlying protocol. Finally Infinit appeared as a P2P file share application in 2013, Figure 1.1. This particular service however will be discontinued precisely on March 31, 2017, preserving on their alternative file storage platform.

Soon after the Internet took off as technology and inspired by the emerging file sharing momentum among the users, live streaming was set in the focus. The very first known live streaming was held on 5th September 1995 with the audio broadcast of a baseball game between the Seattle Mariners and the New York Yankees, based on a technology developed by Progressive Networks [str b]. This company that later renamed to RealNetworks, together with Microsoft will soon after engage in the most famous battle of the streaming market that will spread over the entire 90's [str a]. Many related technologies became popular during that time such as - Windows Media Player, Flash, Helix Universal Server; video codecs including MP3, MPEG2 and H.264 for which Microsoft spent a lot of efforts and investments; and protocols - HTTP and RTSP, dedicated to control streaming media servers and developed by RealNetworks, Netscape and Columbia University in a common 1996 IETF draft. Servers such as: FFmpeg, GStreamer, Helix Universal Server, QuickTime Streaming Server, Wowza Streaming Engine, YouTube and clients such as: QuickTime, VLC media player, Skype, Spotify, and many more. Other participants including Vivo, Xing, VDOnet, Vxtreme and Microsoft were also developing closed ownership platforms for streaming. Macromedia (acquired by Adobe) became a legitimate leader for streaming media on the Web after they launched the famous Flash Player. In spite of the rapid improvements inspired by a huge competition, the classical 56k home modem connections challenged the support of streaming technologies in user devices. Moreover the scarce bandwidth resources prevented continuous full quality video delivery in home receiver devices. Although the dot com bubble in 2000 affected critically many streaming media companies and products, the innovation in streaming technologies never stopped and instead took alternative directions towards successful expansion and growth.

The growth of the HTTP traffic reinforced the content delivery networks (CDNs) like Akamai in their on-demand delivery of popular content services. Its evolved version, the HTTP-based

adaptive streaming was designed to meet the heterogeneous bandwidth and CPU capacities in user's devices by encoding the video quality in multiple bit-rates. This way the receiver can adapt to the network condition by switching between different qualities in real time. These technologies were adopted by Netflix and Amazon and also used to stream live events like Olympics and tennis championships. The 3GPP body with the joint effort of companies like Apple, Microsoft and Netflix ended the proprietary protocol battle in 2012 by publishing the protocol Dynamic Adaptive Streaming over HTTP – MPEG-DASH, a first official international standard based on the individual solutions: Adobe Systems's HTTP Dynamic Streaming, Apple's HTTP Live Streaming (HLS) and Microsoft's Smooth Streaming. Since the bandwidth reserves of the streaming services were bounded by the overall bandwidth resources, the purpose of the codec was to provide better video efficiency with improved compression. For that purpose the announcement of the H.265 codec marked a new history in the digital video era - the beginning of the 4K video. H.265 increased compression efficiency and decreased bandwidth in order to provide a Ultra HD video quality (4 times 1080p full HD) at the same bitrate (3-4 Mbps) used for 720p video delivery, or increase the frame rate to 50/60Hz without requiring a proportional increase in bandwidth.

At the time when the development of file share solutions over the Internet culminated and while the industry was competing to establish own multimedia delivery products to overtake the streaming market, the peer-to-peer technology joined the game. Although P2P gained popularity with the file-swapping applications, it didn't stop there. When the well proven algorithms for file sharing reached their sweet spot, it was the time for broadening the scope of the P2P use-cases in order to keep up with the market. The main goal was to serve to the users a video content in real-time and not waiting till it is completely downloaded, as it was the case before. This resulted in a proliferation of new algorithms and applications created for real-time video delivery. The new concept imposed many challenges in the design of P2P algorithms for e.g. the inclusion of protocols for video slicing and encoding. Moreover the dynamic behavior of the users and their heterogeneous resource possessions started to play essential role in the design of the overlay topology for streaming purposes. Nonetheless, the hard community work produced successful results in this field and received a substantial amount of user attention. The number of voluntary supporters followed the same fashion as in the file share era, which soon led to unveil one important challenge – the scalability factor. Furthermore, since the systems designed for P2P streaming support were based on altruistic resource share by their users, ensuring quality of service (QoS) guarantees was ruled out as option. Instead, the well-known best-effort principle of the Internet applied also here. Therefore, in spite of the huge popularity of streaming systems like: PPLive, PPStream, SopCast, etc. it was time to acknowledge that the P2P network alone was far from robust and vulnerable to disruptions due to the increasing number of the users, their profiles and their behavior patterns.

The introduction of the cloud computing as a buzz technology in the year 2006 marked a

significant milestone in the technological era. The cloud computing was introduced as an evolution of the common virtualization mechanisms into a novel commercial strategy with several key concepts. The most important of those concepts was the provider's virtual infrastructure created on the top of the physical network of servers, offered to the users *on-demand* as a substitute or alternative for their legacy on-premise servers. The main goal was to make the cloud building blocks - Infrastructure-as-a-Service (IaaS), Platform-as-a-Service (PaaS) and System-as-a-Service (SaaS) available to all users as resources for rent, or to companies (SMEs) in order to increase flexibility and enable scalability of their own infrastructure. The price policy was bound to a *pay-per-use* base, i.e. the user is charged in a fine-granular manner for the amount of cloud infrastructure resources she effectively uses. The cloud computing served as immediate catalyst for many new technologies and applications to come, that were leveraged on the cloud provider's platforms. To foster a balanced adoption of the cloud, hybrid model was introduced as a baseline for distributed services across both, cloud and on-premise infrastructure. The price differentiation of the cloud providers, called for intelligent service hosting among different cloud providers. It was a time of unanimous cloud adoption and support across the industry and the academia altogether. By this time P2P was on the crossroad, either to continue the ever lasting discourse with the Internet Service Providers (ISPs), or grasp the momentum of the novel cloud technology in order to build up more sophisticated and scalable streaming services. All this was happening while the enhanced QoS in the last mile and the innovations in the Internet access technologies, were improving the end-users' experience of quality of video delivery and reception. The promotion of both peer-to-peer (P2P) and content delivery networks (CDN) architectures for video streaming raised interest in several fields, such as education, entertainment, TV industry, research communities, etc.

The important momentum of P2P was recognized in scenarios with heavy server load and bottlenecks, especially critical in streaming systems. However the abundant demands for video resources, raised the expectations for video provisioning at the source. It was impossible for a user alone to meet the numerous expectations for video dissemination as a single source, nor sustain the load of the subsequent connections in the P2P overlay. From this reason the cloud infrastructure came as an answer to this complex problem and offered fast and easy-configurable virtual machines to host multiple video instances as roots of the streaming tree/mesh. This approach fit perfectly in the concepts of both technologies P2P and cloud, albeit it was expected to be scalable across domains and cost-efficient for the provider of the streaming service.

This thesis proposed leveraging the P2P technology with cloud in order to create novel architecture for streaming that is based on QoS cost functions with the aim to offer value-added services to both, commercial clients and P2P users. Led by the advances of the cloud industry and the proven P2P successes, the challenge was to create a hybrid architecture that combined these two technologies in a mixed- centralized and -distributed fashion to solve some of the common problems of the P2P streaming services at the time - scalability, incentives and quality

of service. The initial approach to this end was to simulate a realistic scenario using cloud provider infrastructure and peer machines in order to assess the advantages and the feasibility of the proposal. The goal was to identify the constraints and the limitations of this approach and understand a video service performance deployed over a hybrid distribution network.

One of the major P2P principles is the awareness of cooperation. Since the entire model is tightly coupled to each and every single user connected in the swarm, it is very important to perceive this responsibility from individual point of view. Some of the problems that occur typically in a voluntary best-effort system, is the unfair behavior by the participants. For P2P this means that some users may report lower uploading bandwidth than the actual one (selfishness/free riders), some may deliberately disconnect in the middle a session disrupting the connections established with other users (churn), and some may even intentionally induce a system attack and service failure (malicious users). This demonstrates that even though the majority of the P2P users are altruistic and cooperative, there exist a notable amount among them who would abuse the system integrity for personal reasons. In this sense, the P2P model made a delicate trade-off to the traffic control and security in order to circumvent the server role, which turned more difficult to regulate with the P2P algorithms than it was considered before. To manage the dynamic behavior of different user profiles, designing efficient incentives became integral part of the P2P algorithms. This became a subsequent task in the process of this research, to contemplate an incentive mechanism for the introduced streaming service and in addition, formalize and evaluate the same.

We identified a lack of incentive models for mixed P2P-Cloud services at the time, especially for commercial streaming systems where the quality of service plays an important role in the service business model. As a result we introduced a new, discount based incentive model to the system we presented, as a monetary remuneration towards increasing users' cooperation and eliminating free riders and malicious users. Extending the notion of the taxation incentive schemes, we defined a utility model for the mixed P2P-cloud streaming system, based on different QoS profiles. By performing analytical study the objective was to showcase the effect of the incentive model and the quantitative advantage of the cloud infrastructure on the users' welfare and the provider's profit in a streaming service.

Assuring quality of service has been always a great challenge in real-time streaming services since the time they appeared. This is nonetheless a prevalent condition of the adoption and the popularity of those services. With the proliferation of commercial streaming serviced bound to the cloud, the assurance of the QoS was no longer a handicap for the streaming market, as those services were mainly served in a unicast fashion, providing enough resources to the users. However many of those services were either based on proprietary, non open-source protocols, or they shifted back to the old client-server concepts, bypassing the complexity of the P2P algorithms. The market was changing and so was the tendency in the P2P research when it came to "yet another streaming algorithm and/or application". When Software Defined Networking

appeared as a technological revolution in 2013, it changed the common perspective of the legacy network systems. The SDN paradigm decoupled the control from the data plane as a separate functional boxes in order to offer a full network programability and centralized control entity. With an entire abstraction of the physical layer, SDN enables the creation of controller implementations and application level services based on novel principles. The concept of operative system for networking, grants a lower level control of the fabric and thus permits a direct top-down interaction with the network devices. This was initially enabled thanks to the OpenFlow standard that warrants a full access to the setup parameters of the switch and enables among other, to tune down the QoS-related meters. After the research stay the candidate of this thesis had the chance to learn and work with this new technology, which lead to new ideas in the domain of QoS control based on SDN. When we completed our first set of testing and results, the forthcoming objective was to provide a proof-of-concept application for latency and bandwidth control that would potentially be employed as part of a larger streaming service. The results would essentially give a guidance towards the feasibility of such an approach and determine the next steps. Another objective was to define and test a set of QoS API, as it was initially described in the streaming architecture. The adoption of SDN in the further process of the research was a natural decision, leaded by its high potential and the possibilities it offered for a direct control of segments that was previously impossible to imagine. Moreover establishing the old known IP multicast functionality was now attainable using SDN, and the challenge on QoS provision and tuning was finally considered achievable. Related to QoS, the OpenFlow specification offered primitives for enforcing QoS constraints defined by user application directly into the networking stack.

When talking about SDN's efficacy in traditional environments of connectivity providers such as Telco and ISPs - CAPEX and OPEX cost reduction and revenue increase are mostly pointed out as main achievements. SDN, coupled with the new concept of Network Function Virtualization (NFV) raised the telco's attention to envisage the magnitude of the ad-hoc provisioning and deployment cycles. The NFV introduced a similar concept to the one the cloud computing did several years back - virtualizing the functions built in proprietary hardware boxes and offering them as Virtual Network Functions (VNFs) running as applications in a cloud datacenter. NFVs such as: virtual firewalls, proxies, NATs, security appliances, traffic classifiers, video transcoding units, etc. overtook the ground from the classical network functions facing the age of orchestrated, easy provisioning and deployment of integral telco solutions. This technological synergy have finally ended up the tedious telco's advancements, by splitting their monolithic infrastructure into several segregated, multi-purpose functional domains. At this stage, SDN was already paving its way towards the support of NFV architectures, recognized as a de-facto controller of the underlying networking stack in the NFV conceptual architecture [NFV 2014]. The northbound interfaces and API abstractions offered by the SDN controller, allowed a direct low level interaction to the network and service providers. This

process boosted up the raise of Service Function Chain as a prominent use-case application, to best demonstrate the direct intervention of the controller to support an end-to-end service to operators, providers or third-party end users.

The final objective in this thesis was to create a proof of concept SDN application in order to demonstrate the feasibility of some use-case services deployed on the top of a service chain abstraction. In particular and of most interest to this thesis, was the validation of the service chain related to video transcoding and quality of service assessment. With this work we wanted to correlate the VNFs such as video traffic classifier and the SDN-guided service chain, to the QoS control concept for video streaming application and the algorithm discussed in the thesis. This was the last validation in the process, to show that with the new technological advancements, the service that was initially presented became easier to realize. Of course the use of the SDN, cloud, P2P and NFV are just technological specifiers that should be used as tools in the process, but in essence they all have one goal in common - contemplating, implementing and verifying the idea of value added service to the users.

1.1 Objectives

The work carried away in this thesis, has a span across several years and therefore involves multiple objectives accompanying the main idea described in Figure 1.2. The principal objective was to exploit the possibilities of P2P and cloud computing as technologies to support architectures for real-time multimedia streaming. In a nutshell, the individual objectives include:

- *Pinpoint the opportunities and challenges for implementing real-time streaming services using P2P.* In this part the traditional P2P algorithms are studied followed by reasoning on their potential support for real-time streaming services. It also involves the understanding of the P2P development progress over the years, the issues it was facing, potential gaps to explore and finally envisage potential future contributions in the area.
- *Analyze the cloud ability to support real-time services.* In this study, a through analysis of the cloud network is pursued in order to measure essential QoS parameters in the context of streaming video across multiple distributed cloud instances. It also should determine whether the cloud is ready to support the deployment of real-time services and identify any scalability limits.
- *Propose architecture for streaming as a foundation of the research.* After assimilating the P2P and cloud solutions, this task should propose a potential architectural solution for the problems identified in the previous research steps. This would involve presenting an idea of joining the P2P and cloud in a common framework and presenting a streaming service as opportunistic use-case to explore in the further process of the research.

1.1. OBJECTIVES

Figure 1.2 : The contributions of the thesis

- *Emulate the proposed architecture in a real-life scenario.* This step relates to simulation/emulation of the proposed architecture across cloud virtual instances and nodes acting as peers in order to identify the feasibility of the architecture and the bounds that determine its advantages and trade-offs. Partly the work can include simulation tools in order to test some segments of the idea, understand the algorithms and potentially test the entire architecture.
- *Elaborate a methodology for business model of the streaming service.* In a proof-of-concept streaming service that has a business model bound to QoS cost functions and discounts as incentives, it is essential to understand the model and its contributions to all the parties involved in the service. This can be done by formalizing the business model and based on some assumptions, applying example price policies for PoC scenario in order to depict the provider's profit and the users' welfare.
- *Present interfaces for managing QoS.* With respect to the quality of service in the streaming and the business plan that is coupled with the QoS, there should be a set of interfaces in order to permit a seamless control of QoS in a streaming session. This study should identify the possibilities to offer APIs to service providers and create a validation scenario to showcase

the feasibility of the proposal.

- *Implement a mechanism that validates end-to-end service delivery and video quality adjustment within and among larger cloud DCs.* This is concluding step in the research dedicated to validate the feasibility of research components related to video service delivery and QoS adaptation mediate SDN technology in NFV scenario. It should proof the possibility to assess QoS on a network level and identify the opportunities for paradigm shift from application to network level in the process of implementing streaming architecture/services.

1.2 Research Methodology

Implementing protocols and applications for real-time streaming over the Internet is a complex process that involves several focus areas: overlay topology construction, data dissemination, incentive mechanisms, business model, video coding and compression, neighbor selection policy, application-layer QoS control, media synchronization mechanisms, transport protocol for video streaming, security, etc.

The initial part of this thesis was developed within the scope of topologies for overlay construction in the context of QoS optimization and control. In particular this stage revolved around an architecture for real-time streaming as original design proposal. The architecture was based on P2P and cloud as supporting technologies and it was used as a cornerstone to set up the thesis research statement. To determine the potentials behind the streaming service leveraged with the architecture, the next steps included: (1) detailed study of related solutions among the academy and the industry; (2) rigorous analysis and experimentation with some of the de-facto P2P algorithms and applications. Based on simulation packages dedicated to P2P streaming, the author performed in-depth analysis and testing of some P2P algorithms in order to identify a potentially applicable/adaptable protocol in order to meet the requirements of the system proposal. Measurement study based on set of tests was accomplished in order to verify the readiness of the cloud infrastructures to support traffic intensive real time delivery under stringent QoS constraints. The evaluation results furthermore established relevant thresholds as a guideline for implementing real-time services in the cloud. Extensive focus was directed in the formal representation and analysis of the business model of the streaming service. The adopted methodology for this was chosen as an economical study that was applied in the taxation incentive schemes used in a related research. This model was extended to reflect the system of interest. The evaluation study was based on assumptions and empirical data used in similar P2P services, mainly due to the limited access to statistical data from larger-scale streaming systems from the industry, but also as a result of the novelty of the proposed approach. This phase of the thesis was self-directing and did not form part of a research project.

In the later stage, the author was familiarized with the SDN technology, resulting from work during the research visit to the Zurich University of Applied Sciences. SDN at that time was in

1.3. RELATED PROJECTS

early stage of development with the OpenFlow protocol promising control and solution to many of the existing networking problems, which immediately catalyzed a massive interest among the community. During that work the author identified a potential liaison between the DHT algorithms used for overlay creation and SDN as technology to trigger network-level QoS adaptation requested by application-level services. This resulted in implementing proof of concept artifact to verify the feasibility of SDN-based QoS tuning and control. The work was incorporated within the KIARA component from the FIWARE project. The concluding effort involve design, sketch, implementation and evaluation of SDN-based SFC algorithm to support composite NFV services. Although not directly related to QoS support, the validation of an example service for video quality adjustment and control mediate a dedicated VNF demonstrated the technological evolution and paradigm shift during the entire research process, i.e. the feasibility to implement desired services while providing certain guarantee for quality using alternative technologies and approaches. This work was supported and exploited within the projects T-NOVA and SESAME, described in Section 1.3.

Over the course of the dissertation, each of the objectives described in Section 1.1 were carefully addressed and elaborated in order to certify the validity of the research contributions. Several novel contributions were presented and evaluated in order to show their efficiency and economical impact. A special focus was given on creating reference implementations that will be valuable and applicable in production-quality implementations. As a contribution to the community, all solutions were developed as open-source components, having in consideration the transfer of experience and alignment with the existing solutions (e.g. the SFC library).

1.3 Related Projects

While being involved in the thesis work, the candidate has been participating in several research projects, that were used as support frameworks to implement and validate some key concepts and solutions. Following is a brief summary of the European projects - FIWARE and T-NOVA from the FP7 program and SESAME coming form the new Horizon 2020 research framework.

1.3.1 FIWARE - A Core Platform of the Future Internet

The FIWARE project [fiw] started in 2011 and back then its was established as a Fi-PPP FP7 project for the Future Internet, with the main goal to provide community and framework for future applications development and support. FIWARE platform exists even at current date and is focused on serving APIs that ease the development of Smart Applications in multiple vertical sectors. It also provides multiple publicly available components offered to providers in order to facilitate the deployment of emerging market services.

There are diverse contributions within the project FIWARE carried along the research, although not all of them directly related to the topic of the thesis. A most applicable from FIWARE is the project KIARA - Advanced Middleware. The goal of this Java based

communication middleware was to provide support for modern, efficient and secure applications. KIARA was developed within the I2ND task for network interface management. The role of the QoS component in KIARA, was to define APIs that enable end-to-end QoS provision between two endpoints using KIARA client/server and/or KIARA endpoint and a web server. This was performed via SDN plugin on the network layer that offered the following set of parameters/actions:

- QoS parameters (bandwidth, latency, jitter and packet loss)
- Well arranged P2P multicast overlay
- Optimization of the datapath through the network
- Topology information i.e., the type of the underlaying network

As KIARA tackled SDN architecture and QoS provisioning, it was used as an opportunity to test and validate the part of the concepts related to QoS for latency and bandwidth control in SDN environment. The code of the application was further advanced by other project members involved in KIARA and included in the SDN transport module to be materialized in future FIWARE demo scenarios and IoT applications.

1.3.2 T-NOVA - Network Functions-as-a-Service over virtualized Infrastructures

The T-Nova project [t-n] focused on the design and implementation of an integrated end-to-end architecture for NFV services dedicated to automated provisioning, management, monitoring and optimization of Network Functions-as-a-Service (NFaaS) over virtualized Network/IT infrastructures. The aim was to address the key challenges of deploying VNFs in carrier grade environments by covering all layers of the technical framework: from the Marketplace down to the Infrastructure (NFVI). Having this target in mind, T-Nova presented a complete functional solution, which can be elevated to pre-operational status with minimal additional development.

During the period after the research work at UPM, the work focused on cooperative development of NETwork FLOws for Clouds (Netfloc) - an open source SDK for datacenter network programming. This component contains a set of tools and libraries packed as Java bundles that interoperate with the OpenDaylight controller. Netfloc exposes REST API abstractions and Java interfaces for network programmers to enable optimal integration in cloud datacenters and fully SDN-enabled end-to-end management of OpenFlow enabled switches. Netfloc was formerly included as a part of the I2ND domain in FIWARE. Providing Service Function Chaining in NFV received a substantial amount of attention among the community and was one of the target use cases inside the T-Nova domain. This capability came as a reference library implementation for traffic steering directly from Netfloc and was used to demonstrate and validate end-to-end connectivity service in action.

1.4. STRUCTURE OF THIS DOCUMENT

The efforts in the Netfloc SFC support and the demonstration of holistic service for video quality adjustment can be directly transferred to this thesis and the topic related to the QoS profiles adjustment of video streaming services in a cloud environment.

1.3.3 SESAME - Small cElls coordinAtion for Multi-tenancy and Edge services

The SESAME project [ses] has as objective creating multi-tenant cloud enabled (C-RAN) architecture included under the H2020 5GPPP initiative. This effort includes evolving commercial Small Cells (SC) towards a so-called Cloud Enabled Small Cell (CESC). A CESC integrates a virtualized execution platform (micro server) equipped with, e.g., IT resources (RAM, CPU, storage) and Hardware Accelerators (HWA) such as: Graphics Processing Units (GPU), Digital Signal Processors (DSP), and/or Field-Programmable Gate Arrays (FPGA). SESAME also proposes CESC clustering for creating distributed data center, denominated Light Data Centre (Light DC). This component in essence places the network intelligence and applications in the network edge with the help of NFV while preserving the QoS and 5G performance requirements such as latency and bandwidth.

Within the scope of this project, Netfloc was validated as a component to steer the traffic between different VNFs deployed in a LightDC. A relevant demonstration involving the SFC component was held between the SESAME and COHERENT projects, in the area of multi-domain orchestration and service function chaining. The objective of this demo was to showcase an inter-project solution for bridging the gap between Telco and cloud operators, by provisioning service composition using components and technologies from both sides and based on the following goals:

- Orchestration and automatic deployment of VNFs into CESC cloud infrastructure
- SDN-enabled SFC between network services involving both, Cloud-based and Telco-based VNFs (virtual traffic classifier and virtual media transcoder) .

Similarly as in T-NOVA, this work was identified as suitable for evaluating part of the concepts in the thesis. The second of the goals demonstrated an overall, top-down service provisioning in 5G scenario - from orchestration, through service composition, till delivery of optimized and tailored video to the end-users via their mobile device equipments. It moreover validated the ability of the current technologies to offer value-added services on the edge, like the QoS adjustment and control. From this point on, providing service streaming with SLAs based on tiered price models, is a matter of leasing dedicated functional blocks and adhering to a targeted deployment decision.

1.4 Structure of this Document

This part sketches the outline of the thesis and iterates briefly over the Chapters' topics.

1.4.1 Chapter 2: State of the Art

This Chapter offers a comprehensive literature overview in P2P streaming, cloud and SDN technologies. It is focused on the early days of the P2P algorithms and topologies, showing their classification and properties. It continues by presenting the vital concepts in P2P streaming followed by a description of the most prominent example services for streaming in P2P and mixed P2P-cloud environments. Finally it iterates through technological specifiers in SDN and its most relevant representatives in the area of QoS and streaming.

1.4.2 Chapter 3: Architecture for Cloud-based P2P Streaming

A mixed P2P-Cloud architecture for streaming is presented in this Chapter as a baseline model for the research. Along with the service APIs for QoS cost functions are presented. In the second part use-case scenarios are designed and an evaluation work is performed on validating the feasibility of a cloud provider's network to support the described streaming service.

1.4.3 Chapter 4: Business Model for Cloud-based P2P Streaming

This Chapter starts by analyzing the user behavior in P2P streaming. It proceeds with describing the discount incentive that was proposed in order to complement the business model of the streaming service. Finally it extends the model to a general cloud-P2P service in order to formulate and analyze the common benefits of the provider and the users of the service, in terms of profit growth and welfare improvement.

1.4.4 Chapter 5: SDN-based QoS Assurance for Cloud-based Streaming

SDN application for QoS control is implemented and described in this section as a proof of concept. Moreover APIs for QoS control are presented and offered as part of the service. Ultimately an SDN application for service chain is elaborated to demonstrate a holistic support of value-added service in NFV environment. This implementation and demonstration validate the feasibility to support customized video delivery with QoS triggered directly on a network level with the use of dedicated VNFs.

1.4.5 Chapter 6: Validation and Results

As a summary of the delivered evaluations, this Chapter consolidates the contributions delivered per each project. It furthermore lists the dissemination activities in terms of published papers, open source community contributions, and some details on the activities during the research visit.

1.4. STRUCTURE OF THIS DOCUMENT

1.4.6 Chapter 7: Conclusions

This Chapter sums up the activities, underlines the realization of the objectives and traces the future research steps.

INTRODUCTION

Chapter 2

State of the Art

2.1 P2P Overlay Design, Architectures and Algorithms

This Chapter describes the related work to the one presented in this thesis and the background of the used technologies. Initially we present the taxonomy of the P2P systems and algorithms and review the relevant topology models for building P2P services. We present further some works related to quality of service and incentive mechanisms as important aspects in P2P systems, describing relevant examples in these topics. Later, we discuss some inter-related technologies to the P2P such as Cloud computing as directly involved in the contribution of this thesis, covering P2P and cloud streaming services. Finally we discuss the SDN technology specifiers and contributions related to QoS and Service Function Chain to conclude the last part of the contributions in this doctoral work.

2.1.1 Video delivery architectures

Client-Server unicast. In early days of the Internet, Client-Server was widely adopted as a technology to stream the video to a broader audiences located in different parts of the world. In such architecture, a streaming server acts as a central entity that delivers the video to multiple clients. Each client can request the desired video and obtain it directly from the steaming server in unicast (one-to-one) fashion. The video delivery was enabled with the creation of new protocols such as Real-time Transport Protocol (RTP), designed specially for multimedia communication. A media player at the client side had to be enhanced with the support of such protocols in order to receive the streaming service.

IP-layer multicast. The fast growing interest in real-time communications by a large number of users revealed significant burden of the IP unicast communication. Scalability has emerged as a big issue due to the high number of one-to-one connections to be maintained for each new coming user. This lead to the proposal of IP multicast: a one-to-many communication on IP level permitting to simultaneously stream the same video to multiple users, as it was introduced by Deering [Deering 1990]. The concept of IP multicast consist of a video server that is connected to all participants of the streaming session through multi-tree connections among IP routers. Maintaining the common networking principles and the IP semantics, dedicated multicast address blocks were reserved in the IPv4 and IPv6 protocols to enable the creation of multicast groups. The routers created optimal distribution paths for packet delivery at a network level to a specific destination address, based on the multicast address domain. The scope of the IP multicast

has remained limited when it was introduced 20 years ago, from several reasons. From infrastructure point of view, the implementation is bound to dedicated routers (specialized in performing multicast), which imposes a burden on the network providers to install such equipment and thus incurs additional costs in implementation complexity, time and money. Moreover if not widely adopted, such model would show as an unscalable solution. Like the unicast model, IP multicast is maintaining separation of transport and routing layer, but it is difficult to provide higher level features such as error, flow, and congestion control with multicast. Increased router overhead to manage multicast groups and complex transport control for multicast session was another drawback. Due to the above mentioned reasons, including also security concerns and economic issues [Diot 2000], many Internet Service Providers (ISPs) either blocked or disabled IP multicast.

In conclusion, since the IP multicast at that time seemed too complicated and costly solution for the ISPs, it did not receive enough push in further developments related to streaming over the Internet. In a recent article from 2016 named "The Return of Multicast" (January/February 2016) of the Streaming Media magazine, the author Dom Robinson has an interesting vision for the IP multicast: "In the traditionally appliance-led and capital expenditure CAPEX-intensive network investment, most network services need a minimum level of traffic before they can garner political and financial support internally, and since IP multicast reduced costs for everyone, any successful long-term strategy was vulnerable to a counterargument of "that didn't increase our revenue. "Worse, the efficiencies could reduce revenues or reduce transit-buying power, which could affect the business more widely. Sometimes inefficiency can be valuable."

Nowadays, reducing CAPEX and OPEX is not the main concern any longer, rather the network operators and the telcos are directly involved in the innovation, forming also part of the end-user offers. With this in mind, one may say that the idea was too optimistic for that time, but it brought up some key principles, to be yet restored and applied in the modern systems. IP multicast was followed with more research in the area, which will remark the raise of a new way to distribute the video over the Internet, by bringing the multicast to application layer.

Application-layer multicast. Since the IP multicast introduced interesting design principals, some of its key features were shifted up to the application layer, in a so called Application Layer Multicast (ALM) [Chu 2000a]. IP multicast features such as: group membership, multicast routing, and packet duplication were implemented in the end-user systems. The ALM is considered an organized overlay structure among the end-users, communicated in a one-to-one unicast manner. This multi-point link communication serves as underlying topology for applications such as: content exchange, on-demand streaming, and audio/video conferencing. Raising as a product of several years' work aimed to overcome the client-server bandwidth limitations and the scalability issues of the IP multicast, application-based multicast captured developers' attention into creating variety of algorithms, protocols, topologies and novel applications based on this delivery technique.

2.1. P2P OVERLAY DESIGN, ARCHITECTURES AND ALGORITHMS

The ALM officially marked the start of the P2P era, with the basic design philosophy of encouraging the users called *peers*, to act both as clients and servers. There are two key drivers making this approach attractive. Firstly such technology does not require support from Internet routers and network infrastructure and consequently it is extremely cost-effective and easy to deploy. Second, in such a technology a participant that tunes into the broadcast is not only downloading a video stream, but also uploading it to other participants watching the program. This approach has consequently shown better scale performance with the group size, since greater demand also generates more resources. With this, P2P has become the new standard for building distributed network applications.

File sharing and BitTorrent. As it was captured with the timeline in the Introduction Chapter 1, the P2P applications that are well known to the majority of the users, had their predecessors even before the Napster (year 1999). P2P applications such as: Gnutella, eDonkey, LimeWire, Freenet, Kazaa, Audiogalaxy, iMesh, eMule etc., became well known by the fully distributed content search based on flooding and swarming. It wasn't a long before the third generation of DHT based routing algorithms appeared in the academia through proposals of CAN [Ratnasamy 2001], Pastry [Rowstron 2001], Tapestry [Zhao 2006], Scribe [Castro 2002], Kademlia [Maymounkov 2002], SplitStream [Castro 2003a] etc.

BitTorrent marked the file sharing era as the first protocol to successfully facilitate file transfers among multiple peers across unreliable networks based on an approach different than the existing ones. Designed by Bram Cohen [Piatek 2007, Varvello 2012], BitTorrent bases on *Tit-for-tat* (choking/un-choking) algorithm that consist in the following: each peer only uploads to 7 other peers at a time, 6 of these are chosen based on amount of data received from the neighbor in the last 20 seconds. The last one is chosen randomly, with a 75% bias toward newcomers. While the related studies mainly based on random neighbor selection, the strategy of BitTorrent to make a good neighbor selection, marked a high peer performance without increasing the ISP cost. This came due to the combination of both, random selection across ISPs and selection based on link preferences by the ISPs, which brought benefits to the users and the ISPs in the same time. The huge success and adoption of BitTorrent has remained to current date, offering free client application for content sharing [Bit a]. In the recent years, it has as well extend its business model to meet the requirements of the Live Streaming and On-demand market [Bit b].

The subject of this thesis is bound to video streaming systems, therefore the following sections will limit the focus to classification of overlay topologies and cover the systems relevant to P2P media delivery.

2.1.2 P2P taxonomy and overlay design

The P2P topology is composed of overlay network represented through a directed graph $G = (N, E)$, with N being the set of participating peers, and E the set of overlay links [Rodrigues 2010].

The nodes that are connected via the links, keep the notion of their public IP addresses for a direct one-to-one access. Overall, the P2P overlays can be divided in two main groups: centralized and decentralized. Figure 2.1, depicts a taxonomy of the P2P overlays as presented in the IRTF P2P RG Core Subgroup [p2p b]. It shows the main categorization of P2P overlays (based on the presence of central components in the system) on: decentralized overlays - that can be unstructured or structured; and centralized (client-server), or hybrid overlays. The structured overlays are further divided depending on: (1) routing and (2) topology, whereas the unstructured are classified in two categories: (1) flooding and (2) random walk. Finally few examples for systems belonging in each category are depicted as leafs in the tree.

Centralized overlay. In a centralized overlay, the peers contact the server in order to establish connection to the system and ask for a particular content. This organization is however the least stable of all, since the centralized server represents a single point of failure for the entire system, thus it is less scalable with the number of peers in the system [Lua 2005]. Napster was an example P2P file sharing system that appeared among the initial ones in this category. In this system, the peers registered file lists for share to the Napster server, which responded to them giving the list with hosting peers. The peers then could contact directly the peers holding the desired files by specifying a keyword in the file name and download the content form them. Although search performance showed efficient, the system was unscalable and essentially it was turned down due to music content copyright issues. After this, Napster was turned into online music store until it was acquired by Rhapsody [nap]. SETI@home was another P2P early adopter with a service for CPU sharing in idle computers across the world. More specifically, it was used for processing data chunks gathered form the Arecibo radio telescope from the SETI@home server. This approach of sharing free compute resources, together with its successor BOINC [Kurochkin 2016] entered in the category of P2P applications denominated as *Volunteer computing* [Rodrigues 2010]. The biggest advantage of SETI was the price-performance trade-off. However security appeared of significant concern due to the transparency of the SETI servers that were exposed to anonymous peers, which made them vulnerable to attacks. Both, BitTorrent [Bit a, Piatek 2007, Varvello 2012] and Skype [Guha , Baset 2006] also belong to the P2P applications based on decentralized overlays, the first having a node for tracking the set of peers involved in the upload/download process and the second, having a client application for user management.

Decentralized overlay. Decentralized topology is characterized by the logical relations among the peers. The peers register in the system by contacting an already existing peer, called "bootstrap node", whose address and port number they obtain from a dedicated P2P client application. Due to the absence of dedicated server node in this topology, all the peers share the responsibility of maintaining the overlay coherency. This involves their contributions of resources (software, data, media content etc.), as well as bandwidth for applications such as IP telephony, video streaming and conferencing. Nodes called "super-nodes" or "rendezvous nodes" may exist in some systems,

2.1. P2P OVERLAY DESIGN, ARCHITECTURES AND ALGORITHMS

Figure 2.1 : Taxonomy of P2P overlays [p2p b]

acting as a gateway for the new-coming peers, and responsible for indexing the shared content in the overlay. Due to their public IP addresses and large compute power, these nodes are highly available and accessible for all the peers. Skype has the concept of super-nodes in order to increase the efficiency of the content lookup and delivery. However super-nodes acting as isolated servers for a subset of peers, can also suffer attacks as a single point of failure in the swarm. Replicating their role in the system, is one approach to assure resilience. Nonetheless, the overall distributed topology is more robust and resilient to attacks, when compared to the centralized overlay.

Based on the organization, the decentralized overlays can be further classified as: unstructured and structured.

Unstructured overlay. This means a random construction of overlay graph without a particular structure. The nodes send a request message along a random walk in the overlay, for an a priori one-hop neighbor discovery and content advertising, beginning at the rendezvous node and ending at the last peer in the walk. The links are maintained statically or in a response to the changing conditions, determined by minimum and maximum degree (number of connections). To search for an object, a peer floods a query in the overlay (containing the object's key and metadata) that locates a rendezvous node (or other nodes) who have a pointer to the object's owner (copy of the object). One example of unstructured topology is Gnutella [gnu], the first fully decentralized peer-to-peer system for resource sharing. The Gnutella system was based on ping-pong protocol to establish connection among the peers who were fully autonomous. The resource lookup and navigation was based on flooding represented by a broadcast search query to the peer's immediate neighbors. Despite the flexible peer join/leave and fast search, this system showed unstable and

unscalable under load.

Structured overlay. This topology follows a specific graph structure and definite node relations. Each node has an identifier chosen from 160-bit integers numeric key space and used to distribute the node uniformly in that space. The placement of nodes and the set of links in the overlay is done based on the node identifier. Node id keys can also establish responsibilities to nodes and enable efficient key-based routing, which is one of the greatest benefits of these overlays. A typical implementation requires logarithmic complexity with the network size, for the amount of per-node state and forwarding hops. To keep the distributed state, *distributed hash table* (DHT) abstractions are used in the structured overlays. The idea in this approach is to enforce a replica placement policy, by replicating the key/value pair on the DHTs of those participating peers, whose identifiers succeed the key in the circular key space [Rodrigues 2010]. DHT approach has been used mainly in file sharing systems, such as BitTorrent. The structured topology can be flat or further organized hierarchically in tiers/levels. Chord is a flat, structured peer-to-peer overlay based on DHT. The protocol used in Chord is based on per-node routing tables maintained for peer discovery. This is a fast and scalable overlay solution, with expensive routing table maintenance.

Liu et al. [Liu 2007a] offers similar taxonomy representation of the architectures for Internet video broadcast, enclosing mainly two overlay groups: (1) Router-based or IP multicast and (2) No router support. The later further branches in "Infrastructure-centric", "End-systems with infrastructure support" and "End-systems only". The last two are also known as "End-System, Application-Level, Overlay or Peer-to-Peer Multicast". Infrastructure-centric is a topology where specific nodes or proxies are deployed strategically in certain locations on the Internet, in order to enable unicast data transmission to peers in their proximity. These architecture, analogous to the client-server hybrid approach discussed above, is referred to as Content Delivery Network, with Akamai [Aka, Pathan 2014] being the most known among them. The other two categories belong to the decentralized overlay approach, where the peers have equal rights in the multicast group, being managed and operated by a distributed end-user application.

Based on the abstraction model for data dissemination, the P2P overlays can be broadly classified in two categories: *Tree-based* and *Mesh-based*.

Tree-based overlay. The idea of the tree overlay dates back from the routers organization in IP multicast topology. In this overlay the peers are grouped in a structure containing multiple trees, with a source that represents the video server. Upon overlay creation, each peer join multiple trees at a specific level, which number is decided based on the peer's access link bandwidth. The peers have parent-children relationships which allows for push based data packet forwarding from the higher to lower levels in the trees, i.e. a parent node pushes a data to its successor children nodes. Each video packet is disseminated from the source to the destination using a precise route in the tree, determined by the overlay. The peer's upload bandwidth capacity, determines the number of children it will have, i.e. the fan-out degree. It is a good

2.1. P2P OVERLAY DESIGN, ARCHITECTURES AND ALGORITHMS

Figure 2.2 : Multi-tree overlay with two delivery trees

practice to have lower fan-out such that the delay is preserved minimal, especially for the peers that appear as leafs at the bottom level. Early examples of single-tree streaming systems include Overcast [Jannotti 2000], ESM [Chu 2000b], ZigZag [Tran 2003], Nice [Banerjee 2002] and Scribe [Castro 2002], MON [Liang 2005] and Plumtree [Leitao 2007].

The dynamic P2P behavior is a great challenge for maintaining the tree structure coherent. The tree should be self-healing and have a mechanism to adapt to the network changes affected by a peer's sudden departure, preserving continuous video service to the rest of the peers. Ensuring such performance in a single tree overlay with static paths, is severely affected by churn, where the risk of quality degradation is bound to a peer's departure. Another drawback is the leaf peers' resource inefficiency as they don't contribute their uploading bandwidth to other nodes. Since the majority of the nodes are leafs in the tree, their excess bandwidth remains underutilized, which highly impacts the efficiency of the tree structure.

As a solution, a multi-tree overlay is a resilient approach to alleviate the peer's strong dependency on its parents in the case of peer churn. To that end, a peer can appear as internal in limited number of sub-trees, being a leaf in the rest of the trees, reducing this way the chance of multiple video disruptions when the peer disconnects. Hence, the algorithm for tree construction is paramount to the tree-based P2P system [Magharei 2007]. Being a superset of the single-tree overlay, this structure owns some advanced benefits. Examples include the use of video coding algorithms that allows for videos with specific quality to be pushed to some trees, reduced latency and increased robustness [Goh 2013, Liu 2007a]. Figure 2.2, depicts an example of two streaming delivery trees and seven peers who are placed in both sub-trees, at a different level.

When streaming a video through a multi-tree overlay, the server slices the stream into

multiple sub-streams, pushing each stream to only one sub-tree. For optimal bandwidth utilization, the peer should appear as internal node in a number of sub-trees proportional to its uploading bandwidth. In a fully balanced multi-tree system having m sub-streams, a single peer is positioned as an internal node in only one sub-tree, remaining as a leaf in the rest $m-1$ sub-trees [Liu 2008]. The first and most known multi-tree P2P system is SplitStream [Castro 2003a]. Other examples include: ChunkySpread [Venkataraman 2006], CoopNet [Padmanabhan 2002] and Sepidar [Payberah 2010].

The tree construction and maintenance can be done either centrally or distributed. In the first when a peer joins the streaming, it contacts the central server or a bootstrapping node, who assign to the peer a position in the streaming trees, depending on the peer's location and network access. To keep well balanced trees, the new peer is usually added as internal node in the trees that have minimum internal nodes, whereas to keep short trees, it is placed as a child of the first available node at a lowest depth. The central server detects peer's departure either by a sign-off signal or by a time-out mechanism. In such case, the server recalculates the streaming tree and instructs the peers to modify the topology. Since the central server can be a bottleneck and a single point of failure for large streaming system, various distributed algorithms has been created to address this issue, such as for ex. Zigzag. MutualCast [Li 2005], as well as SplitStream, are examples of P2P systems that achieves reliable content delivery in a dense distribution forest. This is achieved by re-adjusting the tree delivery rate according to the change in case of frequent peer churn. As a downside, the large number of nodes prevent easy scale.

Mesh-based overlay. This is unstructured overlay in which the peers are organized dynamically and connected according to the availability of the data to share. Therefore they are also denominated as data-driven overlays. The mesh approach follows the topology of BitTorrent's swarming mechanism for file sharing, imposing additional constraint for video delivery in timely manner. Upon arrival, the participating peer is connected randomly in the mesh, mainly by following a gossip approach. This means that the peer sends a message to a subset of random peers, these follow the same procedure in the upcoming phase and so forth till the message spreads to all the nodes in the swarm. This way, each peer connects to a group of other peers maintaining a certain number of pairwise links (incoming and outgoing), which can vary during streaming session and can permit more dynamic peer selection according to both, peer and content availability. Though content delivery in mesh can flow in both directions (from peer to neighbor and reverse), unlike in tree overlay, it has been demonstrated that uni-directional outperforms bidirectional exchange in mesh overlays, ex. Prime [Magharei 2009]. BitTorrent [Bit a] and PPLive [ppl] are some of the most famous implementations of mesh-based overlay. Other examples include: Chainsaw [Pai 2005], CoolStreaming/DONet [Zhang 2005], Promise [Hefeeda 2003], PULSE [Pianese 2007].

The possibility to choose neighbors from a list and make a content mapping across the overlay, enables fast recovery in case of churn since the peer can replace the broken connections

2.1. P2P OVERLAY DESIGN, ARCHITECTURES AND ALGORITHMS

to download the remaining content. During streaming in mesh-based overlays, the video content is divided in portions called *chunks*. These are smaller parts of the video stored in the peer's memory buffer, each including a sequence number for ordering before video playback. The peers exchange *buffer maps* that contain the information of chunks and their owners. Both the buffer maps and the chunks are requested via "push" and "pull" mechanisms, and updated continuously during the streaming session to avoid redundant chunk transmissions. The pull mechanism is preferred in order to avoid bandwidth drain for pushing a duplicated content. A well-designed packet scheduling mechanism is essential for mesh-based streaming. It takes care of the following tasks: achieving optimal bandwidth distribution and usage by all peers in the mesh, adjusting the video quality by pulling precise number of chunks and finally keeping track of the playback deadlines. Chainsaw and CoolStreaming are systems that adopt the pull-based techniques.

Despite being distributed, a mesh streaming system has a tracker or bootstrap node for registering the active peers and their outgoing degree. A peer indicates its IP address and port to the tracker upon joining and obtains a subset of random peers that can accommodate its request. If affirmative, the connections are established, the new-coming peer's neighbor list gets updated and the video exchange initiates. Despite the content availability, the decision factors for an optimal peer's placement include as well the resource availability: bandwidth, latency, packet loss, CPU, memory, and connection quality. In case of graceful departure, the peer informs the tracker and its neighbors in order to get the information updated properly. Finally the peer is removed from the lists. Caching and "keep-alive" messages are one way to track an unexpected peer departure. Organized view of random mesh overlay in details can be observed from the Figure 2.3.

The mesh-based streaming systems create a decentralized overlay with stable peer relations. The random targets and the redundant connections makes them robust to churn and resilient to failures. The lack of specific structure however impacts the video distribution, by causing increased startup and transmission delays, degraded video quality and frequent playback freezes due to the dynamic and unpredicted behavior. Large buffers are moreover required to support chunks storage, which increases the signaling overheads for buffer maps exchange.

Multi-tree vs. mesh overlay. Both multi-tree and mesh approaches create similar overlay and follow the same push-pull method for disseminating different pieces of the video content. Peer's churn causes video disruption and broken links in either cases. The approaches to retain connectivity can be reactive (chose a new parent upon peer's departure) or proactive (replace broken path from the backup sets) [Goh 2013]. The main difference is the overlay construction, that has ordered structure in the case of tree, and random connections for the case of mesh. The packet delivery path in multi-tree has a static descriptions-to-trees mapping, whereas in mesh the the delivery path is formed dynamically during packet traversal through the overlay. Both approaches equally have been a valid choice for many streaming systems, although neither fully

Figure 2.3 : Organized view of a random mesh

solves the well known challenges of a dynamic P2P environment. In essence, tree-based model should be chosen in the case when a good video quality, low playback and a low end-to-end delay is vital. In a system prone to churn, a mesh-based overlay should be considered, thanks to its resilience to dynamic peer churn. The study by Nazanin et al. [Magharei 2007], offers a detailed comparison of the two overlays. Attempts also exist on defining a hybrid mesh-tree solution for combining the best of both in achieving optimal quality, like for instance in Chunkyspread [Venkataraman 2006], mTreebone [Wang 2010a], Bullet [Kostic 2003], CliqueStream [Asaduzzaman 2008], Prime [Magharei 2009], GridMedia [Zhao 2005], Bayeux [Zhuang 2001] and recently in a mesh-tree-mesh approach [Maheswari 2016].

Technical challenges. A fundamental characteristic in P2P streaming, is the support for different qualities, imposed by the heterogeneous bandwidth resources of the participating peers. Ten years ago, there was significant difference between the connection quality of home users vs. the internet capacity available in big companies and university campuses. To offer streaming available for all users, the research introduced different encoding techniques, such as scalable or layered video coding (SVC) [López-Fuentes 2015]. This method divides high quality video bit streams in layers with different lower quality. A peer can subscribe to basic layer for lowest quality, and gradually increase the quality by subscribing to additional number of layers. SVC can remove some parts of the video and generate smaller stream aimed for specific decoder, facilitating like this the video to users with limited bandwidth capacities. Multiple description coding (MDC) [Dong 2016] is another technique, where an MD coder generates sub-streams of the source video called *descriptions*, which be decoded individually and aggregated to achieve higher quality in function of the peer's incoming bandwidth. It is important in P2P streaming to

2.2. P2P MULTIMEDIA STREAMING

choose properly the transport protocol. Despite the doubts, there are P2P implementations based on TCP, which similarly as with network coding schemes, achieves a decreased delay. NATs and firewalls impact equally the system and can prevent direct connectivity between the peers in the overlay. Startup delay and channel switching enter in category of potential issues that impact the quality of the service in general. Finally, in order to achieve an optimal performance, the tree overlay (as less flexible) has to adapt the distribution path to the changing network condition. The mesh overlay instead, adjusts the path dynamically in accordance to the feedback, therefore it is easier to implement [Magharei 2006].

2.2 P2P Multimedia Streaming

In parallel to file sharing using P2P, a new way of distributing the content was introduced - *P2P multimedia streaming*. The P2P paradigm has been implemented in such a way to deliver the multimedia content to the audience as an alternative way to the traditional client-server model in order to ensure more scalability and robustness and in the same time to offload the bandwidth provisioning cost on the server side. Soon after this technique was announced, many new P2P algorithms for streaming content appeared, inviting different applications to be created on the top. This attracted many users to switch to the revolutionary way of watching media content online, without having to download the entire video content beforehand. The P2P streaming systems were divided in two categories according to the type of video distribution they provide:

- *Video on Demand (VoD)*. VoD delivery was the service offered by the Content Delivery Network (CDN) providers. In such case, the video content is replicated over various intermediate servers strategically placed at the network edges of the CDN. On video request, a client is directed to download it from the closest server. Video-on-demand users enjoy the flexibility of watching chosen video clips at any time they desire. There is no synchronization among the playbacks of the same video clip on different users. CDN is convenient for shorter start-up delays, reduces traffic and serves more clients at a time. Big disadvantage of the CDN streaming is the monetary limitation when scaling. Since higher number of clients means requirement for higher bandwidth provision, the CDNs should scale proportionally with the number of clients.
- *Live Video Streaming*. Live Video Streaming or P2P TV has evolved to be a cutting edge technology for delivering multimedia streaming to the Internet audience. In a live streaming session, the video content is sent to the users in real time. Synchronisation exist among all video playbacks. Build as an overlay network including several thousands peers, P2P live video streaming has received a lot of attention, as it appeared as an alternative to the conventional TV over IP (IPTV) and CDN services.

"On 5 September 1995, ESPN SportsZone streamed a live radio broadcast of a baseball game between the Seattle Mariners and the New York Yankees to thousands of its subscribers worldwide

using cutting-edge technology developed by a Seattle-based startup company named Progressive Networks. It was the world's first live streaming event" [str b].

Today, there is no doubt about the big achievements of the P2P technology, already proven through the popularity of the invented applications and the user rate success with an exponential raise over the years. Although voice over IP applications had as well some real-time requirements, live video broadcast showed to be much more challenging considering the requirement for simultaneous support of a large number of participants, dynamic peer behaviour and high bandwidth demands for the video streaming. Despite its advantages, the P2P streaming started to face different problems arising from the network-unaware peer selection which were moreover increasing with number of peers in the network. Reducing the P2P traffic could certainly improve the network performance, but this did not necessarily guarantee optimum performance experienced by the users of the real-time streaming service. To overcome this issue, various P2P models based on collaboration between ISP and P2P users have been consequently suggested.

The Internet users are aware of several well-known CDN and IPTV services for live streaming content, such as: Zattooⁱ, Octoshapeⁱⁱ, Livestationⁱⁱⁱ, Netflix^{iv}, PPLive^v, PPStream^{vi}, SopCast [Vieira 2013] and UUSEE^{vii}, CoolStreaming^{viii}, Vuze^{ix}. Zattoo [Chang 2009] is P2PTV service, offering more than 30 free channels plus a premium subscription for high quality (HiQ) channels. Initially being P2P-based, today Livestation operates as IPTV service and charges 4.99 EUR monthly subscription for their premium quality, on demand channels. Yet another propriety CDN/IPTV service, Ostoshape uses private multicast-enabled ISP backbones to provide high quality live and on-demand videos based on per-channel flat rate pricing. As most popular among the *free of charge* services, SopCast, UUSEE and PPLive, are channel-based services that provide different streams across different video channels. PPLive [Vu 2010] appeared as one of the earliest P2P-based IPTV application that supports both, live streaming and on-demand services. Youku [Li 2011] is a China-based *user-generated content service*, similar to YouTube. Its registered users can upload videos and enjoy the free archive of movies and TV shows. Peltotalo et al. [Peltotalo 2008] in their survey, summarize a large number of live and on-demand streaming services. Although widely available, a lack of QoS guarantee is a big downside on the performance of the free streaming services. In

ⁱ<http://zattoo.com>

ⁱⁱ<http://www.octoshape.com>

ⁱⁱⁱ<http://www.livestation.com>

^{iv}<https://www.netflix.com/>

^v<http://www.pptv.com>

^{vi}<http://www.pps.tv>

^{vii}<http://www.uusee.com>

^{viii}<http://www.coolstreaming.us/>

^{ix}<http://www.vuze.com/>

2.2. P2P MULTIMEDIA STREAMING

this context, Piatek et al. [Piatek 2010] showed that applying incentives to PPLive, significantly improves its performance up to 4 times that of native PPLive.

2.2.1 QoS in P2P steaming

Each of the emerging real-time communications over the Internet such as: gaming, videoconferencing, video and audio streaming, etc. demand individual traffic treatment due to: (1) the specific service type they offer and (2) the flows type - bursty vs. constant, or single vs. multiple flows. VoIP for example is more delay-sensitive, data is mainly loss sensitive, while video streaming, gaming and video conferencing are bandwidth intensive and delay sensitive. Therefore, having a mechanism for QoS assessment is essential for the underlying algorithms in order to deliver the promised service to the users and meet the SLAs that has been contracted, regardless of the varying network traffic load. A lack of QoS guarantee as in majority of the traditional P2P systems, impacts the stringent timing constraints of the live streaming applications. To provide efficient real-time streaming most of the services impose QoS requirements such as: minimum path bandwidth, jitter and bounded end-to-end latency for the delivery of their data streams. Traditional connection-oriented routing schemes for multimedia data streaming assume single-path routing, and often attempt to find the best path based on one or two QoS metrics, which can lead to packet losses and quality degradation along the route. From the viewpoint of a distributed multimedia application, a path is useful when it fully meets the QoS requirements. From a business perspective, assuring QoS is vital for users' perceived QoE.

To find a substitute of the "best effort" QoS, this topic has been treated by the Internet Engineering Task Force (IETF) in the architectures *IntServ* [Braden 1994] and *DiffServ* [Wang 1998]. In the first approach (also know as hard-QoS), the QoS is provisioned per flow-base by bandwidth reservation along the entire end-to-end path, thus making all routers in that path aware of the service state. The second protocol, DiffServ was proposed as alternative in order to alleviate the scalability and statefulness problem of IntServ. The way it works to make flow classification is on a hop-by-hop base, by treating each flow individually based on the ToS field in the header. Despite its simplicity, this solution has poor performance in terms of aggregated QoS guarantees on individual flows. Multiprotocol Label Switching (MPLS) [Rosen 2001] is a protocol that uses additional label for short path routing in order to avoid the complex routing table lookups. As a conclusion, despite providing some solution to the problem, these techniques have never been leveraged in real-time communication due to their complex configurations and lack of support by the service providers.

There are some examples in the literature on contributions related to QoS monitoring and adjustment. Mushtaq et al. [Mushtaq 2006] proposed an adaptive algorithm for video streaming over P2P based on active measurements on network links. This algorithm is based on a multi-clustered overlay where the super nodes of each cluster keeps a list of all candidate peers. A "Hello" message is sent as kind of a ping test to calculate the RRT time between the receiver

peer and the candidate peers, and also to get the information of the targeted peer's super-node. Peers with lowest RTT and belonging to different clusters are selected in order to avoid a common bottleneck link. The receiving peer monitors the RTT value regularly between all active peers and itself. Their adaptation mechanism uses a low-pass filter to calculate a smoothed value of RTT. A buffer is attached at the receiver peer whose threshold value depends on the playing rate and the packets arriving time. Efthymiopoulou et al. [Efthymiopoulou 2016] simulated a protocol that monitors peers' bandwidth and allocates dynamically additional resource from the cloud adapting the playback rate. Other researches are mainly bound to a single overlay topology or treat solely specific QoS parameter in individual setups [Brogle 2010b, Bernardini 2010, Brogle 2010a].

2.2.2 Incentives in P2P steaming

A P2P streaming system is powered by the shared bandwidth exchanged in the overlay. For optimal user experience, P2P live streaming requires sufficient bandwidth resources and topology integrity. In essence, the aim is to motivate users to participate actively in the distribution of data streams with the objective of increasing the global welfare in the streaming environment, by increasing the individually perceived video QoS. Therefore the presence of incentive mechanisms is crucial to deal with negative behavior and generate more productive and user-friendly environment. The incentives are important to detect and mitigate malicious users who threaten the system's integrity. The literature offers extensive research in this field and so far various mechanisms have been presented in order to motivate the users altruistic behavior in the streaming systems. The incentive mechanisms for sharing in P2P however, have a long academic rather than practical history. The authors in [Su 2010], divide incentive mechanisms applied to P2P media streaming into four groups: reciprocal, reputation-based, game-theoretic and taxation-based.

Reciprocal mechanisms base on the tit-for-tat strategy and they are mostly used in file sharing systems. Peers only exchange resources with neighbors with similar upload rates. Occasionally, they allocate their own upload capacity by assigning slots - first to those users with more transfer rate. In this scheme, free riders are alienated as they don't reach friendly relationships with their neighbors. The reciprocal mechanisms can be direct and indirect [H. Yang 2013]. The former represents a pair-wise incentive as in the PULSE system [Pianese 2007], based on the tit-for-tat strategy and it was initially proposed in file sharing systems due to its fair approach. Liu [Liu 2007b] proposes to combine tit-for-tat strategy with multiple layered video encoding. Layered videos provide better quality when more layers are received, thus by applying a tit-for-tat strategy, the users who transmit more layers receive more back and get a better video quality as a result. A similar mechanism to tit-for-tat aimed for P2P streaming services has been introduced by Ngan et al. [Ngan 2004]. They propose temporal reconstruction of the overlay to distinguish nodes with selfish behavior and avoid free riders, for the price of increased overhead. This mechanism is similar to tit-for-tat and it is aimed for P2P streaming services. The limitation

2.2. P2P MULTIMEDIA STREAMING

in this approach is that it limits the contribution to exchange between peers with similar contribution levels exclusively.

Reputation-based mechanisms deal with providing a global rating for every user in the system retrieved from the feedbacks of all of the peers who have interacted with this peer. This score improves with the user's contribution. Users consult the reputation of the possible receivers and choose content suppliers. A reputation is an indirect-reciprocal mechanism. This means that as opposed to the reciprocal systems based on mutual relation of two peers, here the reputation is global. The main drawback of reputation for media streaming is that it can take time to build up a relevant history. However, Pianese et al. [Pianese 2007] use a reputation history to improve on his reciprocal system, allowing users to perform informed choices when they select future candidates. This history bases on the previous interactions among the users. Other examples of systems based on reputation include: EigenTrust [Kamvar 2003]—uses the sum of positive and negative ratings and pre-trust peers to compute the global reputation scores; PowerTrust [Zhou 2007]—implements a trust overlay network to model the local trust and reveal feedback relationships among peers; PeerTrust [Xiong 2003]—introduces a transaction-based feedback system to integrate it into a model for evaluating the trustworthiness of the peers. Li et al. [Li 2007] claim that reputation and credit-based incentive mechanisms although successful in a file-sharing P2P applications, are unlikely to work in a streaming scenarios due to limited time framework. Therefore the study encourages further research for incentive models and price establishment in P2P streaming. The limitations of the reputation-based mechanisms are: peers with poor bandwidth are likely to have lower reputation scores and thus lower priority in selecting the hosting peer; there is a lack of reputation history for new coming peers; high-reputation peers can loose interest in further contribution; and free-riders can abuse the grace period given to newcomers to take advantage of the system.

Taxation are an indirect-reciprocal mechanisms that uses the optimal income taxation concept in the economy to establish different tax rates for peers with different capacities. All the other mechanisms tend to reward users based solely on their contribution. However, those peers with less capabilities in terms of bandwidth can be, in some cases, punished for the lack of resources rather than for the lack of willingness to participate. Chu [Chu 2004] motivates users to contribute with the entire available bandwidth, i.e., users with high upload bandwidth get encouraged to participate with more bandwidth than they need to receive the whole stream, in order to compensate to compensate for limited performance of the low capacity users. This altruistic approach aims to achieve a sort of social-welfare in the system where, ideally, more users are allowed to receive media in good conditions. Yang et. al [H. Yang 2013] incorporate taxation based incentive mechanism into SVC-based layered media delivery towards QoS aware neighbor selection. The drawback of this approach is the inability to cope with free rides due to the altruistic nature of taxation that has no means to deal with them.

Game-Theory mechanisms use the game theory approach to implement strategic play among

the participants in the streaming system. Every peer tries to optimize his play by maximizing the amount of data received, while minimizing the amount of data transmitted. Lin et al. [Lin 2009], provide a game-theoretic framework to model player's behavior and design incentive strategies for P2P live streaming. This approach faces a difficulty to deal with peer churn because of the precondition that every peer should stay until the end of the game, i.e. the multimedia streaming session.

Apart from the described incentives, the literature offers models based on combinations of some of the above techniques. One example is the approach to introduce the social network data into game-theory approach as incentive in file-sharing [Wang 2012]. Since mitigating churn, free riders or malicious users is limited with the aforementioned mechanisms, they are unsuitable for live streaming services. Commercializing a streaming service, moreover, calls for incentives that not only boost up participation and bandwidth share, but also promote the service among the online community.

2.2.3 Web-based P2P streaming

WebRTC [web] appeared in 2011 under the initiative of Google, Opera and Mozilla in order to facilitate JavaScript APIs for real-time communication in the browser. It is a free, open project, with an idea to enable web browsers with Real-Time Communications (RTC) capabilities and permit development of rich, high quality, RTC applications in the browser via simple Javascript APIs and HTML5. The main goal of this specification is to achieve an interoperability of real-time multimedia applications across multiple web browsers and liberate the application users of the plugin installation requirement. The common aspects of WebRTC and P2P was the possibility to implement P2P routing algorithms into JavaScript API through WebRTC and this way facilitate the creation of P2P web oriented real-time multimedia streaming applications.

Until then a user needed to download a dedicated application based on P2P algorithm in order to watch a streaming content on the web. In a research contribution [Trajkovska 2012a] aligned to the P2P protocols and in the early times of the WebRTC events, we analyzed the possibilities and challenges that are required in order to extend WebRTC with JavaScript APIs for P2P streaming as part of the WebRTC standardization process. This approach was necessary in order to leverage the browser with ability to host P2P multimedia streaming applications in a straightforward manner without having to install plugins or download a specialized application. This way the developer implements the multimedia application using APIs of algorithms such as PRIME [Magharei 2009] or LayeredCast [Moshref 2010] as part of a standard JavaScript library, rather than deploying a complex application based on these P2P algorithms. We believed that this approach should be address in order to satisfy entirely the necessity of the ever increasing multimedia streaming community.

Opportunities in P2P Streaming over WebRTC. Considering the different elements of WebRTC that could bring new opportunities and features in the context of the existent

2.2. P2P MULTIMEDIA STREAMING

peer-to-peer streaming solutions pointed out JavaScript as obvious key technology in common. WebRTC as a novel technology was designed for implementations in web browsers accessed via JavaScript APIs. However, this meant that the protocols used to establish connections and transmit data have to be standardized so that different browsers can communicate. We divided the potential pros of using WebRTC for peer-to-peer real-time streaming in two big groups: advantages derived from WebRTC being a web technology and pros of the standardized solutions defined for the transport and connection establishment protocols.

Web applications. As a web technology destined to work within web browsers, WebRTC inherits most of the known benefits of web applications. Some of them are:

- *Manageability:* In terms of actualization and patching, managing web applications is very easy as users always obtain the last version of the client application when they browse the web page. As a side effect, there is no possibility a peer can connect to a new P2P streaming application with an old, incompatible version.
- *Multi-platform support:* Application clients of WebRTC are coded in JavaScript and are run by a web browser. If a browser developer follows the standard, the application should run without modifications, regardless of the specific browser that the user has chosen. Furthermore, the operative system or even the hardware platform is also irrelevant as long as there is a compatible browser developed.
- *No installation:* At the time of publishing this contribution, the peer-to-peer based streaming services required an installation of dedicated applications or browser plugins. WebRTC could bypass this step with alternative embedded approach, thus reaching a larger user base.

Standardized protocols. The protocols used in WebRTC are defined by the IETF working group called rtcweb [rtc] that has published numerous set of proposals to date. Having a standard definition of the technologies and protocols being used provides opportunities in terms of interoperability with new and existing applications. Furthermore, the current state of the drafts points towards already established and proven protocols and mechanisms. While following this standards can seem like a loss of flexibility as some of the decision have already been taken, it can also be seen as an advantage as developers can focus on the overlay network and the way the media is distributed and take the lower protocol levels for granted. In this respect we discussed the following key aspects:

- *RTP/SRTP:* Real-Time Transport Protocol [Schulzrinne 2003] and its secure counterpart. RTP is extensively used in real-time applications, especially in real-time audio and video communication tools, as well as real-time streaming solutions.
- *ICE:* Interactive Connectivity Establishment [Rosenberg 2010] provides the mechanism needed to perform NAT (Network Address Translation) traversal using both STUN and

Figure 2.4 : Early building blocks of WebRTC and additionally proposed ones to achieve P2P Web-based streaming

TURN protocols to achieve this purpose. NATs can make peer-to-peer communication difficult in some cases; the advantage of using ICE in P2P streaming is that peers can always connect to each other.

Challenges in P2P Streaming over WebRTC. Next we discussed the key open questions that we identified in the WebRTC group pending at the time in order to support P2P streaming in the web browsers. For that purpose, as crucial components to be implemented as parts of the specification are: redirection and segmentation of media streams in clients, security aspects, data channel support and implementation of P2P algorithms in JavaScript. Figure 2.4 depicts a diagram of the scheduled for implementation components by the WebRTC project at the time of publishing this contribution paper, and the ones we propose based on the challenges we discussed. We believed that these questions were important to address prior to the implementation of P2P streaming standard mechanisms based on WebRTC.

- *Streams redirection and segmentation between multiple PeerConnection in the client:* WebRTC specification was not clearly stating that an incoming media stream in a PeerConnection can be redirected to another PeerConnection. This feature would enable the creation of P2P overlays using current WebRTC APIs for media streaming. If these overlays were tree-based, clients would send incoming data to peers, and these peers would redirect the same traffic to other peers in the overlay. In mesh-based overlays the solution would be more difficult for video and audio streaming because it would force WebRTC implementations to support splitting of the streaming content into chunks. In other P2P scenarios (file sharing, gaming, etc.), the implementation of the overlay would be simpler

2.2. P2P MULTIMEDIA STREAMING

since the stream only consists of data (different from video and audio) that makes it easier to get processed by custom JavaScript libraries.

- *Data channel must be finally supported:* At the time of writing this proposal, WebRTC specification did not completely support the transmission of data (this feature came few years later). The main progress was marked in providing audio and video transmission that is quite interesting for multimedia P2P scenarios, such as simple streaming or conferencing. But there are additional P2P cases in which the transmission of other type of data would be also mandatory: on-line gaming for distributing game events, file sharing, collaborative distributed systems, signaling for advanced multimedia streaming, etc.
- *P2P algorithms must be ported to JavaScript libraries:* The P2P algorithms and architectures that are currently implemented in standard P2P systems support the creation of P2P overlays: mesh-based, tree-based or hybrid. Assuming that previous challenges have been successfully solved, P2P developers would need to port existing algorithms to JavaScript language in order to make them work for web applications. In other words, algorithms such as Scribe or Pastry will have to be implemented in JavaScript. With this, the web applications could provide complete P2P services through the web browsers.
- *Security problems related to JavaScript and P2P Systems:* Apart from the functionality, if we manage to solve previous problems, P2P developers would be concerned about another important aspect: security. In this case there are several vulnerabilities, such as lack of both JavaScript code encryption and efficient trust management in new P2P scenarios. Nowadays JavaScript source code can be downloaded and the end users can read it. This can be considered as global flaw in JavaScript and can especially affect P2P scenarios because participating users could access foreign data of other users that is redirected through their clients. This could be solved if JavaScript code was ciphered and browsers prevent the user of accessing this data. Finally, P2P applications should implement some kind of mechanism for trust management in order to support faithful traffic redirection among clients.

Pointing out videoconferencing services over WebRTC as a parallel research line, the author contributed in a WebRTC publication dedicated to videoconferencing as alternative type of real-time communication service [Rodríguez 2012]. WebRTC has progressed with a fast rhythm in the yeas following these works and currently it whinesses a number of use cases and applications for real-time communication, which are aligned to the protocol updates of the related rtcweb protocol standard.

2.2.4 Cloud-based P2P streaming

Cloud computing brought back on the scene the concept of virtual centralized model. Initiated by the leading cloud providers such as Amazon, Google, IBM, Oracle, the new concept of infrastructure lease attracted service providers to move some or their applications to the cloud. In the business world, there was a short time span between the cloud computing initially being only a buzzword and its fast migration towards the public and private market. The research community was witnessing a number of attempts to qualify and quantify the network performance of a certain cloud provider. The cloud became a common infrastructure for hosting various applications and research use cases in matter of months.

Multimedia live streaming and VoD Software as a Service (SaaS) applications like Wowza [wow] were among the first to give a try on the new service hosting model in the cloud, in this case cloud provider being Amazon with its EC2 [ec2] and CloudFront [clo] solutions. M. Fouquet et al. [Fouquet 2009], presented a good overview of the cloud computing utilities. They describe a potential cloud adaptation plan of their distributed application-layer multicast for video streaming (engined by relay servers). In their paper, they elaborated the distribution trees topology and numbered various use cases of how to benefit from the cloud. Latency and bandwidth issue in video streaming was tackled but without further details. Research examples that deal with the long lasting battle between the P2P overlay networks and the ISPs is described in [Aggarwal 2007, Shen 2009, Wang 2008]. Based on locality-awareness method, overlay topology construction approach or on both, they suggest methods such as: oracle web service offered by the ISP, adaptive peer selection algorithm, algorithm based on a gossip protocol or hybrid approach adapting between biased and random peer selection, in order to reduce the cross-ISP traffic. These techniques aim to facilitate a P2P node to choose neighbors independently based on a certain criteria. The last example presents an algorithm for peer-assisted VoD in which peers bias transmission rates to their neighbors and adjust dynamically these rates.

The research work mentioned so far treats the trade off between the ISP and the P2P overlays build on the top of their network. However comparing among commercial cloud applications, no ISP's uniform payment policy for multimedia streaming and VoD service existed at the time of our research proposal in this domain. Offers varied depending on the required cloud disk space or CPU for PaaS and IaaS solutions, while SaaS providers defined their own cost models depending on the services they offered.

Initially we located several works related to analysis of the network I/O performance for applications in the cloud that focus on searching a way to optimize resource sharing, e.g. [Mei 2010] Mei et al. explain this considering the throughput and resource sharing effectiveness as parameters. Other research [Sivathanu 2010] conducted performance measurements of the storage management addresses, both for the application and the service providers to achieve increased efficiency in I/O performance, resource provisioning and workload

2.2. P2P MULTIMEDIA STREAMING

scheduling. Garfinkel [Garfinkel 2007] performed an analysis of EC2 management and security facilities in the same time considering evaluation measurements of Amazon S3 and SQS, finding EC2 good enough when considering cost-time trade-off.

Deelman et al. [Deelman 2008] studied the cost-performance analysis of various execution plans used by the same application example. They showed that good choice of both storage and compute resources reduces cost which doesn't affect the application performance. Moreover they pointed out "Nimbus Science Cloud"ⁱ as an example cloud provider to examine the use of cloud computing in science. In [Chan 2009] authors offer a semantic analysis for modeling and testing applications in the cloud through a cloud graph model, that can be used to dynamically compose cloud computations.

Research as [Pereira 2010] were of interest for our work because their Split&Merge architecture used for encoding time of high performance video served us as a guide for treating the video streams in the cloud i.e., coding/decoding and evaluating the cost of this operation. Finally, Ang Li et al. [Li 2010] have recently designed a tool that offers the possibility of comparing several aspects among various cloud providers by taking into consideration parameters such as, CPU velocity, I/O of the disk and latency, bandwidth and response time. The main objective is relating their performance and compare the cost each of them imports. Although their results in some part are similar with the measurements we have independently collected and performed during the period of three months, we focus on applying the knowledge obtained from our metrics to the area of multimedia streaming. Focusing moreover on exhaustive analysis of the Amazon EC2 cloud would lead to profound elaboration of the possibilities that each of its data centers offer, for deploying multimedia cloud applications and possibly of hybrid models with P2P solutions. Interpreting the results served as a base to establish more accurate solutions and investigate the network tolerance when such applications interrelate and communicate among each other.

We wanted in our research to characterize the network of the cloud providers and examine the reliability of QoS parameters such as latency, bandwidth, response time and jitter. Finally, we were interested in how these specific results can reflect in various applications for multimedia streaming and videoconference like [Rodriguez 2009]. Next we are going to present the literature review divided into two parts: first one reviewing the existing cloud infrastructures and managers, and second, we revise current methodologies for deployment in cloud and cross-cloud techniques and architectures.

Cloud infrastructures. Eucalyptusⁱ, Intel Hybrid Cloudⁱⁱ, Open Stackⁱⁱⁱ, OpenNebula

ⁱ<http://workspace.globus.org/clouds/nimbus.html>

ⁱ<http://open.eucalyptus.com>

ⁱⁱwww.intel.com/go/hybridcloud

ⁱⁱⁱ<http://www.openstack.org>

[Sotomayor 2009], Zimory^{iv}, VMWare vCloud^v, etc. are some of the early cloud technologies named infrastructure managers that allow cloud administrators to establish cloud hybrid architecture. OpenNebula is an IaaS (Infrastructure as a Service) application. Its main objective is to allow the creation and management of a private Cloud in a data center, but it offers APIs to directly manage virtual machines on Amazon EC2. It implements another standard API named OCCI which makes it very interesting for hybrid architectures which combines the usage of both own and third party resources. For our concern, the most important feature of OpenNebula is that it can schedule the creation/deletion of virtual machine instances. Unfortunately this feature does not work in the case of Amazon EC2 instances. Eucalyptus is also an IaaS application, containing similar features as OpenNebula. Due to the lack of a scheduler that can start and stop instances of virtual machines, we found this virtual infrastructure manager less interesting for our purposes. Apart from these, there are other important systems for managing hybrid infrastructures as the comparative table details in [Sotomayor 2009]. Although we found these systems very interesting for future purposes as they achieve the objective of making hybrid clouds, they didn't meet all the needs of scheduling. Therefore we decided to create a service that would interact directly with Amazon EC2 as well as with our current virtual machine solution, VMWare.

Methodologies for deployment in the cloud. With the objective of quick responsiveness to the business challenges, hybrid solutions in the cloud have shown to be an inevitable approach especially common among the industry-specific applications. To avoid risky undertakings by migrating entire systems on the cloud, some companies committed themselves to the hybrid approach which, as the literature states, has remarkably increased profitability. There are interesting research examples revealing the challenges of hybrid cloud systems. In his paper, Goyal [Goyal 2010] describes the steps to take in order to implement the cloud computing environments in the companies and the way to adapt the entire business oriented architecture to hybrid clouds. A combination of Microsoft Azure cloud-based application and enterprise application developed over a large-scale campus network, is used as a test-bed in Hajjat et al. [Hajjat 2010] to establish the benefits of hybrid clouds for enterprise application migration. This is achieved through a model that describes the required approach conditioned by specific constraints, cost savings, transaction delays, as well as privacy and security policies associated to the enterprise applications.

Combining cloud with other technologies in order to increase the advantage of such composite architectures is another interesting approach. A through overview of the challenges for building multimedia applications in the cloud were presented by Zhu et al [Zhu 2011]. The authors moreover presented an idea of media-edge cloud (MEC) architecture, placing CPUs and GPUs at the edge in order to provide distributed parallel processing and QoS adaptation for diverse device types. Using public clouds for mobile technologies was described by Manjunatha

^{iv}<http://www.zimory.com/>

^v<http://www.vmware.com/products/vcloud/>

2.2. P2P MULTIMEDIA STREAMING

et al. [Manjunatha 2010] and using cloud with P2P topologies in [Trajkovska 2010a, Dagher 2008, Carlini 2010]. In the first one we presented a P2P-Cloud architecture for multimedia streaming including APIs for QoS cost functions, which is one of the contributions that will be described in the thesis. The later work handles the P2P concepts combined with cloud servers in order to build a social multimedia application on the cloud. It has proven to adjust traffic generated by the users and reduce provisioning cost for large servers. Carlini et al. [Carlini 2010] showed a combination of P2P and cloud computing as successful in providing good performances such as scalability, elasticity, load balancing and resource provision to massive multi-user virtual environments. This include cost optimization methods for either QoS cost or provisioning cost that is similar to our model. The authors claimed that the cloud when mixed with other techniques can bring benefits related to cost. Similar research study was presented with the CLive system [Payberah 2012]. The authors used cloud resources for P2P live streaming to complement the lack of peer resources in order to guarantee a predefined QoS. The cloud was furthermore recognized as a suitable platform to leverage improved video transcoding, for example in solutions based on Scalable Video Coding (SVC). In this respect, Chang et al. proposed CloudPP [Chang 2012] - a Cloud-based P2P streaming platform based on public cloud servers to construct an efficient and scalable video delivery platform with SVC, putting accent on saving cloud resources. CloudStream [Huang 2011] offered multi-level transcoding parallelization framework with two mapping options to optimize transcoding speed and reduce the transcoding jitters while preserving the encoded video quality. This with the objective to deliver high-quality streaming videos based on streaming adaptation to network dynamics. Studies related to combined approaches with cloud, involved also the Video on Demand (VoD) services, such as [Wu 2011]. In the storage systems domain, Stormy [Chang 2012] used the cloud storage for efficient data processing, providing optimized resource utilization and increased cost efficiency.

It is important to highlight the emerging middleware technologies for workload management in hybrid cloud environments. For example Zhang et al. [Zhang 2009] enable seamless and flexible allocation and communication between heterogeneous resources in hybrid clouds, job scheduling, virtual ipmage deployment and adaptability application requirements. Comet-Cloud ⁱ is a framework for creating applications on federated and hybrid infrastructures that implements an autonomic computing engine for Cloud and Grid environments. GridGain ⁱⁱ is Java and Scala based middle-ware software to build HPC cloud applications over Cloud and Grid resources. Claudia ⁱⁱⁱ is a middleware toolkit for governing federated services on hybrid clouds in a manner that best matches the service requirements. ProActive Suite ^{iv} is another middleware solution for

ⁱ<http://nfsccac.rutgers.edu/CometCloud>

ⁱⁱ<http://www.gridgain.com/>

ⁱⁱⁱ<http://claudia.morfeo-project.org>

^{iv}<http://proactive.inria.fr>

parallel, distributed, multi-core and cloud computing. It implements a scheduler for inter-user resource sharing to run workloads.

A hybrid grid-cloud platform is used as a testbed in [Kim 2010] to investigate the way hybrid clouds improve the changing computational requirements and the QoS constraints for a dynamic application workflow. Application infrastructure adaptability support together with the subsequent performance advantages is stated to achieve improved overall application performance and more effective use of the hybrid platform. Koala [Baun 2011] is a cloud management Web-based toolkit that enables working with various services across multiple cloud platforms aimed at helping customers detect the best matching cloud infrastructure according to functionality and price. It works with both public and private clouds and it has Amazon Web Services compatible API for storage and infrastructure services. Although not dealing with exactly the same issue, they contribute to a large extent to optimization techniques in the cloud, while some of them implement job schedules. We analyzed some of these solutions in the context of our related study on videoconference service over hybrid-clouds, in particular, for a videoconference automatic price control management. We found similar research in the area of cost-optimization techniques for hybrid clouds, for ex. [den Bossche 2010]. They focus on scheduling deadline constrained workloads with a minimum execution cost on a cloud by following the application's QoS requirements such as CPU and network. However this model does not consider application migration on various instance types nor on clouds.

This gap is exactly what we find interesting to fill-in, by offering a model to guide the service developers on how to divide their application into CPU, storage and bandwidth-sensitive components and place them on the best matching clouds in terms of cost [Cerviño 2013]. This is a continuation of our previous research [Cervino 2011a, J. Cerviño 2011] in which we proposed a new methodology focused on implementing cost effective services composed of units distributed over hybrid clouds, with the focus on videoconferencing service as a use case. Li et al. [Li 2009] presented an optimization approach for profit maximization on cloud. This method is aimed at applications running on one cloud and the cost optimization is conditioned by QoS and SLAs. In our technique on the other hand, we are guided by price constraints across multiple clouds. A research study related to cloud provider comparisons and measurements [Cerviño 2011] was required in order to find the best-matching cloud for certain systems. Li et al. [Li 2010] proposed a measurement tool for comparing different Cloud providers in order to select the best-performing provider for a given application of a Cloud customer. This tool can be used to take a comprehensive measurement study over four major cloud providers, namely, Amazon AWS, Microsoft Azure, Google AppEngine, and Rackspace CloudServers. However the results obtained are ephemeral due to the periodic changes that every Cloud provider introduces into their software and hardware, and because of varying user demands over time. While these methods try to define only one best matching cloud, in our study [Cerviño 2013] we established a general methodology to enable cost planning for system deployment on multiple provider hybrid

2.3. EMPOWERING SDN FOR VALUE-ADDED SERVICES

clouds. We have created a videoconference system by implementing a scheduler, deployed over hybrid private-public or multi-provider clouds based on a price trade off methodology that provides a cost effective and high performance system. Additional contribution related to cloud-based service is included in our related work [Harsh 2014]. It describes an approach to implement a generic rating, charging, and billing engine that addresses the requirements of the telco and the cloud providers together. The RCB service launches the telcos service providers into the emerging cloud service business market.

2.3 Empowering SDN for Value-added Services

To accommodate the ever increasing demand for computing resources required by the real-time multimedia applications, SDN-based QoS work has raised in past years the attention of the academia and the developers' communities worldwide. Thanks to the efforts in this field, each subsequent version of the OpenFlow protocol included more QoS capabilities in the specification. The version 1.3.2 enabled to measure and control the quality of service parameters in the switches and tune the end-to-end quality between various endpoints. For the first time the researchers faced the opportunity to reconsider earlier developed protocols and rely on the "open" network to improve them. The OpenFlow support of traffic differentiation and QoS, makes SDN-based testbeds convenient for performance evaluation of real-time multimedia applications. We use this as incentive to approach the multicast algorithms from the network layer and create a protocol for multimedia streaming based on aggregated QoS parameters.

2.3.1 QoS- and streaming-related research trends in SDN

With the modification that introduced the *meter bands* and *band limiters* in the OpenFlow protocol, it was allowed an immediate intervention by the network application developer for resource bandwidth allocation in the switch. This has motivated the surge of various proposals based on the new features related to QoS provision on a network level. Several of them made an intention to enhance the already existing protocols such as *IntServ* and *DiffServ* with SDN-based QoS architectures. Egilmez et al. [Egilmez 2012] for example, implement a novel OpenFlow controller design called OpenQoS, aimed for multimedia delivery with end-to-end QoS support. Their approach is based on distinguishing and optimizing the multimedia traffic from the rest of the Internet traffic. Our approach in contrast, uses the meter bands to dynamically adjust the desired QoS with the objective to make a scalable QoS levels inside the same multimedia traffic. We match on the IPv4 field in the routing tables to distinguish between the different users in the network and adjust accordingly their required QoS. Jeong et al. [Jeong 2012] proposed a QoS-aware Network Operating System (QNOX) for SDN with Generalized OpenFlows that includes IP, MPLS-TP and WDM/ASON. Sezer et al. [Sezer 2013] offered a high level analysis of the current SDN state and described in detail the issues and the upcoming challenges regarding

network performance, scalability, security, and interoperability, proposing potential future directives.

The recent years account for a massive interest in QoS research trends based on SDN. Some of the topics include: multimedia flows routing mechanisms, inter-domain routing, resource reservation, queue management and scheduling mechanisms, QoE-aware mechanisms, etc. All of these are summarized and classified in several categories in the survey study by Karakus et al. [Karakus 2017]. The authors provide a thorough analysis of the OpenFlow protocol timeline per version and QoS features per SDN controller. For example, the ODL controller offers PacketCable MultiMedia (PCMM), a southbound plugin and supporting modules to allow management of service flows with dynamic QoS. Alternatively, the reservation module in ODL tends to allow dynamic low-level resource reservations to "rent" to the users network services, connectivity or a pool of resources (ports, bandwidth). The ONOS controller, on the other hand, (1) supports the OpenFlow metering mechanism, but with limited implementation in the switches, and (2) implements a new high-level instruction SetQueueInstruction. All these features are however in early stage of development and testing until they are exploited in fully-functional use-case services.

The authors of the survey moreover provide a summary of the most recent works classified by type of QoS parameters in Table 1 of their study. What is most interesting to our research is the Table 5 that outlines the Multimedia flows routing mechanisms as common to the work performed in this thesis related to our QoS-related research and solution [Civanlar 2010, Yu 2015, Yan 2015, Jin 2016, Egilmez 2011, Tomovic 2014, II 2015, Yilmaz 2015]. Although most of these solutions came along a posteriori to our solution, their contribution is worth mentioning especially due to the impact made in the academia with respect to further promotion on the QoS assurance research line, present to date. Most of these proposals e.g. (HiQoS, OpenQoS, ARVS and [Egilmez 2011]) are based on the OpenFlow protocol native QoS service support. The Video over Software-Defined Networking (VSDN) [II 2015] selects optimal path for QoS delivery in their network architecture, while [Yilmaz 2015] provides a load balancer solution based on the ODL controller to ensure dynamic rerouting in case of overload. Finally [Jin 2016] is a mediator solution between traditional DASH service-based and SDN-per-flows based QoS adaptation.

Collaboration between the ISPs and the SDN networks has been discussed in a recent work [Herbaut 2016a, Herbaut 2016b] pinpointing NFV and SDN as best-matching technology enablers for such a collaboration. The authors propose a virtual CDN network (vCDN) towards the concepts of shared CDN-ISP infrastructure in order to reach a fair balance of revenues in the streaming market. In the described collaboration types they advise SDN's OpenFlow protocol as a flexible solution for QoS provisioning and control. The NFV business-oriented blocks would additionally facilitate the SLA negotiation between the ISPs and the CDNs for a video delivery service. MS-STREAM [mss, Borcoci 2017] for example is a contribution towards the evolution

2.3. EMPOWERING SDN FOR VALUE-ADDED SERVICES

of the MPEG-DASH that proposes streaming by downloading the video from multiple sources as opposed to the DASH that use one server alone. The authors claim their solution to be adjustable to different streaming architecture models (P2P, CDNs, Clouds) with proven QoE advantages in front of DASH.

These examples are among the several pioneering contributions in the domain of streaming solutions over multiple novel topologies. Streaming and CDN solutions in SDN and NFV eco-systems are starting to prevail the research scene at present time, which proofs again that the right direction to gear the native P2P streaming solutions is in the direction of these novel and prospective technologies. Most importantly because they are coherent with the Telcos' and the ISPs' visions for extensive investments in the 5G-NFV-SDN domains as a promising market for increased revenues. Furthermore they provide a good opportunities for the ISPs to use their infrastructure for new innovations, while in the same time gaining more visibility and clients. Finally by joining the infrastructures and leveraging them with novel technologies, opens an extensive research filed for not only adapting the well-known P2P and CDN algorithms to the new setup, but also enhancing them with new flavors in terms of services and components coming from the diverse technologies and resolving specific problems. Despite live streaming, other real-time services such as the one from our related work [Cerviño 2013] could be adapted as additional over-the-top service deployed using L7 technologies over virtualized Telco and NFV infrastructures.

2.3.2 SDN technology specifiers

In the research related to SDN and QoS we used Mininet 2.1 and Ryu as experimental SDN controller to simulate an experimental testbed composed of several network nodes connected to a single switch. To implement the service chain and deploy the proof of concept services, we used OpenStack cloud and OpenDaylight. Next, we described briefly each of these technologies.

Mininet [min] is a network emulator tool that can run a complete network of hosts, links, and switches on a single Linux machine. We chose this tool due to well documented library and the implementation of Python API that is convenient for interactive development and testing of OpenFlow-based SDN networks. Moreover it is highly compatible with Ryu - the choose of our controller.

Ryu [ryu], on the other hand is a software defined networking framework. It provides software components with well defined APIs that make it easy for developers to create new network management and control applications. Ryu supports various protocols for managing network devices, such as OpenFlow [OF] Netconf [Enns 2011], OF-config[OF-] etc. Regarding OpenFlow, Ryu supports fully 1.0, 1.2, 1.3 and Nicira[Nic] extensions. Since implemented in Python, Ryu interacts very well Mininet and permits easy porting of simulation testbeds to real hardware. Moreover it is an ongoing development tool and open for contributions. Finally Ryu's support of OpenStack, enables the creation of real testbed scenarios over multiple isolate virtual

machines. We used the Ryu controller implementation running along with OpenFlow protocol version 1.3. The topology is based on the OpenFlow 1.3 Software Switch [ofs a], a simple switch compatible with the latest version of - OpenFlow 1.3 and with Mininet 2.1.

Floodlight [Flo 2013] is an alternative OpenFlow controller based the Java programming language and supported by Big Switch Networks. With respect to QoS, it has a system that relies on the OpenFlow 1.0 enqueue action and the Open vSwitch project. It allows management of OVS by exposing Rest APIs via the QueuePusher module. The queues have to be configured in advance in the virtual network device and the services are differentiated by the ToS byte in the IP-header.

OpenDaylight [odl b] is an SDN project that was launched by the Linux Foundation and several networking companies as an open source framework focused on facilitating programmable platform for network developers, end-users and customers. ODL supports a variety of networking projects, standards and protocols making it currently the number one SDN platform in terms of number of users, project and code base repository. It was one of the first SDN controllers that covered multiple protocols on the northbound and the southbound. It is based on Apache Karaf - a small OSGi based runtime that provides a lightweight container onto which various components and applications can be deployed. This implementation approach makes ODL fully modular and extensible for new projects, APIs, and protocols on the both, north- and the southbound. The ML2 plugin [neu] makes ODL interoperable with OpenStack, currently the most prominent cloud infrastructure manager. This allows a complete synergy between the both worlds and permits to create ecosystem for various Cloud-SDN applications and services.

OpenStack is an open source cloud management and operative system that provides virtual resources such as: compute, storage, and networking, that are packed in the common IaaS, PaaS, and SaaS cloud stack and deployed in a cloud data center. The project has a modular approach just as ODL, which encourages increased participation among developers into delivering collaborative project solutions. It also enables the integration with legacy and third-party technologies. The OpenStack source code is available under an Apache 2.0 license and it is organized in multiple correlated projects and components for integral cloud services. OpenStack aims on providing cloud solutions that are scalable, reach in features and bound to related cutting edge technologies. All the services are managed from a dashboard and command line interface library, in order to facilitate seamless resource management to the cloud administrators and the cloud users.

2.3.3 State of the Art in Service Function Chaining

Service Function Chaining creates an ordered list of network functions such as: virtual firewalls, classifiers, transcoding units, proxy services etc., by glueing them together in service aimed for: network administrators, programmers, cloud and telco operators. This section summarizes the recent research efforts in the field of SFC. An insight is given furthermore on the industry

2.3. EMPOWERING SDN FOR VALUE-ADDED SERVICES

implementations, open source solutions and standardization activities.

Related academic research. The following part gives a literature overview for existing traffic steering solutions. Ghaznavi et al. [Ghaznavi 2016], developed an optimization model for service function chain deployment based on distributed and resource efficient placement of VNFs. This decomposes the service chain based on performance requirements and eliminates the throughput bound of a chain to a single VNF or a single physical machine. The authors target mainly a mathematical model for optimal VNF placement in a simulated network setup.

The VNF optimal placement has been also studied by Bellavista et al. [Bellavista 2015]. They analyze the problem of network-aware virtual data center and VM placement in OpenStack platform, taking into account the computing overhead due to intense communication load of VMs co-located on the same physical host (as in the case of Virtual Network Functions (VNF) embedding). This research is build over a similar environment setup as the one described in this paper (OpenStack used as virtual infrastructure manager (VIM) to deploy VNFs in multi-tenant scenario). Even though their results could leverage our SFC approach, the research focus is different - our model offers L2-based OpenFlow-OpenDaylight chaining solution, built on a simplified network overlay when compared to one of OpenStack.

SDN-based architecture for NFV chaining support is presented by Ying-Dar Lin et al., [Lin 2015]. This approach proposes extending the OpenFlow protocol *Match field* messages to classify traffic based on policies. The goal is to reduce the control traffic overhead by promoting data-plane traffic classification when providing NFV support. The proposed model has limitation in real-life deployments, due to several modifications required on the switch, the NFV modules, and the OpenFlow protocol together.

Coming from the Unify project under the FP7 EU research program, it is ESCAPE prototyping system based on the Click architecture for implementing Virtual Network Functions (VNF). It uses NETCONF for managing Click-based VNFs and POX for taking care of the traffic steering. The presented SFC solution makes a different controller choice from our implementation. Moreover we implement SFC on L2 layer using OpenStack as virtual infrastructure manager and OVS as virtual switch. Finally the prototype is implemented in simulated environment, whereas we deploy and test our solution over physical fabric in real-life scenario. At current stage no performance evaluation of the ESCAPE is presented [Csoma 2014].

A service function chain construction scheme is proposed by Kim et al. [Kim 2016] that uses the VLAN tag as chain ID. The evaluations are limited to a simulated Mininet scenario using Floodlight controller, which make it incompatible to compare to our implementation that uses ODL. Similarly, Callegati et al. [Callegati 2015] suggest traffic steering based on VLAN tags in OpenStack. This may cause traffic inefficiency to the chained traffic, due to the additional packet overhead. Avoiding VLANs and GRE header tags, has been considered a precondition for our development strategy.

The work in this thesis related to SFC, differs from the ones previously described in the holistic

approach it follows for SFC delivery. We offer ODL-based traffic steering method and variety of components and services, each of them contributing individually to the PoC and the community. Furthermore, we deploy these components on real SDN-NFV setup, to test the feasibility of the integration and to validate the PoC. Finally, we perform end-to-end tests for different traffic types, to showcase a chained traffic characterization and performance implications of the same. Up to our knowledge, an integrated SFC solution resulting from implementation, to deployment and testing, have not been fully explored and demonstrated in a scenario similar to the one shown hereby.

Industry, standards and open source communities. The SDN and NFV community have been actively engaged during the last couple of years, in the design and development of new ideas for enabling service chaining in NFV environment. This resulted in the design of several SFC prototype solutions coming from academic partners involved in the open source community, as well as, company products releases. The ones that relate to the use of SDN principles are yet in a premature state and lack a thorough evaluation study. In this section we present a number of representative works of the current chaining mechanisms, mainly from the open source community, but also including a production-quality deployments. In the same time, a comparison is made between the state of the art and and this contribution.

To enforce traffic steering on a network level, namely two key approaches can be followed in SDN: *packet-based* and *flow-based*. The first one assumes that packet header encloses an ID field associated to the specific network chaining service. This can be achieved by tagging the packets or rewriting the header fields [Akyildiz 2015]. In this process, the end-to-end traffic is treated as an overlay connection either (1) between the service nodes or (2) independently of the network topology. Alternative option is to comply to a dedicated standard, designed specifically for this service type. The spectrum of related RFCs by the IETF community, has grown fast with more than 30 proposed SFC drafts [sfc 2016] only in the last year 2016.

Packet-based traffic steering: Network Service Header (NSH) [nsh] was introduced by Cisco and it has been adopted by the SFC project developed by the Open Daylight (ODL) community. NSH is an IETF data-plane protocol that represents a service path in the network. NSH is injected as expandable header to the packet, by a classifier function at a service plane entry, in order to enforce chain identification and encode the virtual network graph mapping, onto the physical service data path. Limited to a lifespan bound to the SFC domain solely, the packets carry the NSH header along the chain traversal process. The drawback of using header encapsulation is the increased network complexity and undesired throughput overheads.

Flow-based traffic steering: This approach does not rely on packet headers, proxies, or additional third party components. Instead, it provides a routing logic independent of the overlay topology, and it has the benefit of keeping untouched the packets along the chain. Preserving the original packets, brings transparency to the network functions, because they see the traffic as if it was routed through a chain-less connection. The NEC team from the VTN project in ODL, contributed an alternative to the NSH-based chaining, that relies to OpenFlow rules for traffic

2.3. EMPOWERING SDN FOR VALUE-ADDED SERVICES

Table 2.1 : Technical specification of the current SFC implementations

SFC Solutions	OpenDaylight	OpenStack	ONOS	OpenContrail	Netfloc
NSH spec.	✓	✓	✓	✗	✗
OVS	✓	✓	✓	vRouter	✓
Plugin	✓	✓	✓	✓	✓
Encapsulation	VxLAN/GRE	MPLS	✓	MPLS over GRE/UDP & VxLAN	✗
Match	NSH-based	Neutron ports	OpenStack-based	N-tuple match packets	Neutron ports

steering [NEC]. The benefit is that this approach is OpenStack compliant, since it has been designed to coexist with the Neutron ML2 Plugin. The disadvantage however, is that the agent that manages the virtual graph and the chaining logic is dependent on the VTN abstraction model, and coupled to the NEC experimental environment.

Ericsson delivered a flow-based proposal [eri] that implements traffic steering by forcing the routing over a specific network path. This solution uses OpenFlow 1.3. where the L2 reachability of Service Functions (SFs) and Service Function Forwarders (SFFs) is provided as a description in the Yang files of ODL. At every SFF, the packets are reclassified using VLAN ID in the header. The service hop is based on the MAC address of the previous hop.

SFC in OPNFV. The OPNFV community [opn b] promotes the SFC initiative through the *OpenStack Based VNF Forwarding Graph* and the *Service Function Chaining* projects. The main objective is to deploy the native OpenDaylight SFC approach and test it using VFNs developed in the scope of the OPNFV working group. Leveraging both, the OpenStack work on Virtual Network Function Forwarding Graph (VNFFG) and the ONF OpenFlow work on service chaining [sfc 2016], this project tends to show automatic set up of end-to-end VNF services through VNFFG so that different tenants' flows can be steered through various VNF sequences. The aim is to bridge projects from the Linux Foundation and the OpenDaylight (ODL) community to the OPNFV. The first SFC solution under the OPNFV umbrella was released in the 2016th February's release called Brahmaputra [opn c] and it was implemented on the top of a custom NSH-patch for OVS. The latest release Colorado, includes a second revision of the SFC project based on the following project specifiers: Yi Yang NSH patch, OpenStack Mitaka, OpenDaylight Boron, and Open vSwitch 2.5.90 releases [opn d].

SFC in OpenStack. Neutron is the networking service project associated to the OpenStack VIM. Neutron provides an abstraction of the underlying network through virtual network components and a plugin collection for third party project support. The ML2 plugin in particular, provides the essential libraries that allow for third party substitute of the OpenStack networking service [neu]. It acts as a main gateway for integrating OpenDaylight as a unique control entity in OpenStack.

In its Liberty release, OpenStack included an open-source SFC solution based on a Neutron plugin extension for OVS [ope f], using a standard networking deployment paradigm (Flat Networking with OVS). This implementation comes from the notion of Neutron ports as a

sequence of connection points between the VNFs involved in the chain. In OpenStack SFC, the service function path is represented via: (1) ordered set of neutron ports that define the sequence of the functions, ex: ['p1': 'p2', 'p3': 'p4', 'p5': 'p6']; and (2) set of classifiers for the traffic flows of the chain. The chain lays on the existing overlay formed among the tunneling bridges of the Compute nodes. The tunneling bridge (br-tun) is used to transport traffic coming from/to different ports (egress, ingress and internal) on the Compute Nodes. The chaining technique manipulates MPLS headers used by the OVS agent, in order to carry along the route, the chain path identifier and chain flow classifiers.

Netfloc's vs. OpenStack's SFC: The OpenStack SFC solution has been introduced with the Liberty release that appeared few months after the implementation and testing of the our solution (which is based on the previous - Juno and Kilo releases). Principal differentiator of the two solutions is the overlay approach (VxLAN/GRE vs. L2-forwarding). It is currently impossible to compare these two chaining approaches on a same infrastructure. In our deployment for example, the tunneling bridge is omitted as opposed to the OpenStack-ODL deployment, because our implementation relies on Netfloc as a lightweight ODL version. Last and foremost, the Neutron based SFC in OpenStack is a work in progress, with a limited sandbox testing capability, and provided exclusively as a DevStack all-in-one solution working with OpenStack Mitaka and a dedicated 2.5.0. version of OVS. Aside from the possibility to do basic testing, it is still a challenge to perform a complete benchmarking evaluation, because of the project's incubation phase status and its versions/plugins incompatibilities to out proposal.

SFC in OpenDaylight. OpenDaylight [odl b] is an open source framework focused on facilitating SDN programmability for network developers, end-users and customers. ODL supports a variety of networking projects, standards and protocols, and ranks as a catalyst controller reference for SDN research and innovation. Large number of members from the academia and the industry are actively supporting ODL through a number of project contributions in various development cycles. Organized in a modular architecture, ODL ensures extensibility through various third-party projects, including north/south bound protocols. It is based on Apache Karaf, a small OSGi based runtime that provides a lightweight container for component and application deployment. By using Karaf, one can import different bundles in the controller's runtime environment to achieve the desired functionality. According to the Github code statistics, ODL have reached an unprecedented dominant role among the other controller solutions to date. Due to the former reasons, and with the objective to provide a generally adopted and stable solution, we opted for ODL as a reference SDN controller for our solution. Netfloc also provides an ODL compatible library support that allows an easy integration with OVS, Neutron and OpenFlow. The integration plugins of Neutron has matured over the last three ODL versions (Hydrogen, Helium and Lithium), which lead to more stable and reliable network representation and persistency in Netfloc.

The Open Daylight framework includes SFC southbound protocol *sfc-ovs* (introduced in the

2.3. EMPOWERING SDN FOR VALUE-ADDED SERVICES

Lithium version) that is compliant to the NSH specification. This SFC project depends on platform with augmented OVS version (NSH-aware), since the official OVS release does not yet include NSH support. When we designed Netfloc, the ODL SFC proposal was still in a premature state with a rudimentary technical documentation, similarly as the ones by NEC and Ericsson, thus early adoption was challenging at that stage. To circumvent the lack of tangible solutions, we designed an own chaining implementation the builds on the network and service model established in Netfloc. This is a solution leveraged by OpenFlow, designed to work over standard SDN hardware and software.

Netfloc's vs. ODL's SFC: The early phase of the ODL SFC project, lacked the support and integration with OpenStack, being the the main VIM technology used in the T-Nova project and maintained at our premises. To apply the NSH solution in OpenStack environment, the SFC mechanism would need to extract the following data from each VNF: IP Address, encapsulation details (VxLAN, NSH enabled), OVS switch, and port number. Therefore, the NSH solution is perpendicular to the one described here and a possible adaptation would require a complete redesign of the entire chain mechanism. Finally we aimed to create a native, protocol/product agnostic SFC algorithm, with the perspective to deliver improved traffic performance and lightweight solution, alternative to the OpenStack networking [ope e].

SFC in Tacker. Tacker [tac a] is an OpenStack project dedicated to the creation of unique technological ground for Management and Orchestration of VNFs and network services. Tacker follows the ETSI MANO NFV Architectural Framework [ets], specifically to the components Virtual Network Function Manager (VNFM) and Network Function Virtualization Orchestrator (NFVO). It embraces TOSCA [tos] specification for the description of NSs and VNFs. Taker's SFC initiative [tac b] is an attempt of the community to standardize the approach of implementing joint SFC solution based on OpenDaylight and OpenStack. Currently Tacker does not yet support SFC per se, and it is highly dependent on the development outputs of the ODL and OpenStack community.

Netfloc's vs. Tacker's SFC: As previously described for ODL and OpenStack, the authors have opted for alternative approach in order to provide simplistic working solution, independent of third party implementations. Taker's SFC project is coupled to the OpenStack and ODL, therefore the previous arguments apply.

SFC in ONOS. ONOS [ono c] is another SDN controller that captured the public focus, especially in the recent months. Under the premise of open source, collaborative and bind to the community, ONOS has aligned to the OpenStack and neutron-sfc developments in August 2016 [ono a]. ONOS includes a component that implements the SFC requests from Neutron, allowing resource creation via REST APIs [ono d]. The flow rules in ONOS are created upon request and supported via the NSH patch included in OVS v2.5.9. SFC is implemented as a bundle in the VTN application of the ONOSFW project, which forms part of the OPNFV [ono b] community.

SFC in OpenContrail. Coming from the industry, Juniper based their SFC solution on the OpenContrail controller [ope c], that was released as an open source version of the commercial SDN controller, Contrail. Lacking support of OpenFlow, It is build on the top of OpenStack and customized by Juniper. The routing is based on vRouter, a counterpart module of OpenvSwitch, that was build as an alternative to the Linux bridge in the kernel. vRouter is in charge for the data plane forwarding and uses OpenStack standard based solution that allows to talk to MPLS enabled routers (MPLS over GRE/UDP and VxLAN). On the control plane, it uses BGP and Netconf.

The chaining is implemented in the vRouter via forwarding policies and using tunnel encapsulation/decapsulation [ope d]. Similar as in the OpenFlow protocol, it applies on the received packets, N-tuple match criteria and respective actions such as: packet dropping, packet allowing, or packet redirection to another routing instance. The traffic steering in OpenContrail relies on data model objects, such as "routing instances" and "route targets". A virtual network is divided into one or more routing instances, and implemented as such in the vRouters. The route targets are used to control route leaking between routing instances. In the latest version of OpenContrail (v3.0), the concept of port tuples has been introduced, which similarly as in OpenStack, denotes a group of virtual interfaces belonging to the same machine: VM, container, or bare metal.

Netfloc's vs OpenContrail's SFC: The OpenContrail chaining is different from ours in the following key technologies: SDN controller - Juniper uses its own version of controller and we use Netfloc based on ODL; virtual switch - unlike our solution that uses OVS, Juniper uses vRouter in charge of the routing and forwarding logic; overlay - Contrail uses MPLS/VxLAN, and our solution is based on non-tunneling approach; plugins - Contrail requires dedicated agent and drivers to be installed and configured on the Compute and Control node in OpenStack, and our solution uses the native Neutron ML2 plugin. Therefore a possible comparison between the two chaining solutions, is unfeasible and would result in a different research from the one presented here. Before performing SFC benchmarking in heterogeneous environments, a study of this kind should consider the architectural differences and their effect on the service chaining, beforehand.

Resume: It is evident from the state of art, that the research for enabling traffic steering, have been increasingly present in the past year [Bhamare 2016]. The experience however, shows that at a current stage it is difficult to make accurate one-to-one comparison between the chaining mechanism studied in this manuscript, with other solutions, mainly due to: (1) heterogeneous deployments, (2) incompatible architectures, (3) work in progress specific limitations etc. Table 2.1 provides SFC-features summary of the formerly described solutions.

We would like to point out that the scope of the study was to present a novel SFC approach implemented and supported by Netfloc [net c] (a component based on SDN to manage networks and provide services), in order to verify its feasibility in a proof of concept deployment scenarios, and to finally perform evaluation study on a common set of QoS. With this complete solution

2.3. EMPOWERING SDN FOR VALUE-ADDED SERVICES

(sketch, implementation, development and testing) the objective is to offer useful insights of the service chaining effects to the traffic in a ready-made cloud-NFV solution. We believe that further research in SFC should focus on bridging the gap between increased development vs. lack of evaluation. An upcoming activity is offering the SFC project in Netfloc, as a contribution to some of the open source communities (ODL, OpenStack or OPNFV), the same way as the vTC component was formerly submitted to the OPNFV Yardstick project [OPNFV]. The aim is to encourage further adoption of this solution, receive community wide feedback, and facilitate extended tests in variety of setups.

The rationale on engaging the SDN technology in the research of this thesis is tightly coupled to the synergy of the SDN and NFV paradigm and their dependance on the cloud datacenter infrastructure. Similarly as goal of the cloud technology, NFV targets on providing on-demand, elastic and scalable network infrastructure. The role of a centralized SDN component in such eco-system is to provide full network visibility and management, by optimizing the forwarding decisions. The downside of a central entity is the lack of resilience in the presence of networking anomalies, overloads, loops and failure issues originating from the limited resources of the environment. Such setup can affect the responsiveness of the control plane and hence, degrade the overall quality of service. In this respect, the solution of one problem can lead to another, if not properly addressed. With the objective to handle those shortcomings and provide vigorous, production-quality solutions, combining SDN-NFV and datacenter technologies is a way to obtain efficient outputs by each of them. Analogously to the cloud that was promoted as a technological pillar for existing applications and technologies, SDN and NFV coupled with the cloud, are evidently the new paradigm shift towards contemporary value-added services.

STATE OF THE ART

Chapter 3

Architecture for Cloud-based P2P Streaming

As soon as it appeared in the year 2001, the peer-to-peer technology has given rise to applications for streaming among the others. When the cloud computing concept emerged ten years ago, one of its attempts was to offer various service models as a base for successful end user applications. The initial idea of this thesis dates back from the early research related to examining these two technologies, in order to identify their joint potential in the streaming domain. Guided by the advances of the cloud features and the proven successes of the P2P, we proposed a hybrid architecture that combined these two technologies in a mixed- centralized and -distributed fashion to offer a value-added streaming service.

3.1 Problem Statement

The improvements in Internet access technologies e.g., the availability of high bandwidth and the spread of portable media player devices have opened new opportunities to receive high quality, on-demand, and interactive multimedia applications. In the recent years, the use of both peer-to-peer (P2P) and content delivery networks (CDN)ⁱ architectures for real time video streaming have aroused much interest in several fields, such as education, TV industries, research communities, etc.

When the P2P success culminated with the applications for file sharing, a new way to distribute the content appeared – real-time and on-demand streaming. The subsequent result was the rise of new type of application built on the top of the P2P overlay, dedicated to support of sharing a sequences of a video. This concept extended the initial scope of the existing P2P applications, and it also attracted many users to a new way of watching the video content. The objective of the P2P streaming was to offer to the users a possibility to watch the video content without it having to download.

3.1.1 P2P-based streaming

P2P networking has favorable characteristics, such as high scalability, self-configuration and organization. Many people consider them as suitable infrastructures for supporting real time streaming. However, P2P networks possess dynamic characteristics that can decrease drastically the performance of these real-time applications. Among these characteristics, it is worth taking

ⁱ<http://www.akamai.com/>

into consideration the ability of peers to enter and leave the system without any prior notification. Such situations however can lead to service interruptions, if the adaptive mechanism does not properly address variations in the overlay. There exist also problems of making the right decision in routing schemes and bandwidth utilization, or dealing with heterogeneity of terminals, such as TV, Laptops, mobile phones or tablets. This also includes the variety of network connections including ADSL, Cable Modem, UMTS, WiFi, etc.

From other side, private content delivery networks have reached a great success in real time streaming, mainly for IPTV, life casting and live video streaming of events. They even allow users to broadcast from mobile phone applications or websites. The implementation of these networks has one great flaw, that is the need of dedicated infrastructure which makes CDN services much more expensive than P2P based systems, mainly for broadcasters.

As it was discussed before, it is essential in real-time streaming to guarantee a set of quality of service parameters that are critical to the service. Since the algorithms are not able to ensure this to the users QoS in streaming are dynamically changing and dependent on many factors in the system. Since P2P is mainly a best effort service, assuring QoS was not one of the primarily objectives in streaming, which negatively impacted the users' experience. Therefore, ensuring timely video delivery with proper quality, in a dynamic churn-prone environment, has been one of the greatest challenges for the streaming-based P2P protocols and applications.

3.1.2 Cloud-based streaming

The cloud computing evolved the well known virtualization mechanisms in a new business strategy based on revolutionary features. Some of the most important to mention are: *on-demand* lease of virtual resources instead of the previously used legacy on-premise servers, and elastic and scalable approach in the provision of different type of resources and services, with attractive *pay-per-use* policies.

Before the cloud drew into various new applications, we were witnessing a debate between P2P applications – using unevenly the network resources, and the Internet service providers – trying to discriminate the P2P traffic. The cloud platforms gained points in this debate, creating new opportunities for both parties. Several research works dealt with the objective to offer a solution for mutual P2P-ISP cooperation [Aggarwal 2007, Shen 2009, Wang 2008]. Some alternatives to the problem could come from the cloud world with a proper application of the cloud features and services, in order to offer new business models. The cloud computing immediate effect was a notable strain on the business market. In the same time, it inspired researches to think of adapting to the cloud the already existing technologies in order to raise benefits in different ways.

To take advantage of the cloud paradigm and make multimedia streaming more efficient, the work described in this part established a set of cloud APIs for automatic QoS calculations and adaptations. The aim was to enable negotiation of QoS, such as bandwidth, jitter and latency, between a streaming service provider hosted in the cloud, and the clients willing to contract the

3.2. ARCHITECTURE DESCRIPTION

service. This resulted in a cloud-P2P architecture aimed to offer services for multimedia streaming based on different QoS setups.

This chapter is organized as follows: Initially in Section 3.2, we present the hybrid architecture for streaming, giving details of the streaming service to be deployed on that topology. This includes description of both, the overlay topology and a streaming service example, as well as QoS-based APIs that give a context to a potential business plan for the service. The proposed architecture has commercial nature, merging private and public clouds, with scalable design that can extend to non-profit P2P scope.

In the second part, Section 3.3, we sketch the experimental plan for evaluating a cloud infrastructure's ability to distribute streaming sessions with respect to some key streaming QoS parameters: jitter, throughput and packet losses. The experiments run in a hybrid architecture that consist of mixed connections among P2P and cloud nodes that can interoperate together. To simulate a scenario close to the described architecture, we used PlanetLab machines to represent peers in the system, and cloud nodes to represent clients with better QoS characteristics. With this we created various hybrid test-case scenarios, that consisted of gradually moving streaming nodes located in different continents, into a cloud provider's infrastructure. The aim was to measure the improvement of the multimedia QoS parameters in relation to the location of the participating nodes. We also wanted to confirm the feasibility of the proposed hybrid concept in a large-scale system. The overall conclusion is that, unlike in traditional P2P systems and CDNs, it is possible to efficiently improve the QoS of a streaming service, by deploying a hybrid streaming architecture. This enhancement can be obtained by strategic placing of certain distribution network nodes into the cloud provider's infrastructure, taking advantage of lower cross-datacenter latency and packet loss.

3.2 Architecture Description

A commercial multimedia streaming application is expected to meet certain QoS values. Due to the lack of mechanism that will provide this QoS assurance in a pure P2P service, a hybrid architecture for multimedia streaming emerged as an idea, that defines APIs for QoS assessment as a key feature. This puts into perspective a QoS-bound pricing model for the commercial cloud part, and a completely non-profitable and voluntarily P2P part of the service. The architecture takes advantage of the scalable cloud infrastructure that facilitates deployment of stable and robust services. This opens an additional perspective for SaaS applications in the cloud, that will be in charge for management and orchestration of such a service. They can be deployed as value-added applications by third party service providers or ISPs.

The P2P-Cloud architecture is composed of a distributed overlay that offers a transparent approach towards QoS guided business model. One of the novelties in this architecture is the automatic API functions based on Quality of Service (QoS). Moreover the author of this thesis believed the cloud paradigm to be a good technology driver for novel research in P2P multimedia

communication. Therefore engaging the cloud technology in its roots for hosting prospective ideas could bring benefit to both, service providers and end users. This idea was the base of the research in this thesis and a framework that inspired the forthcoming contributions. It is described in details in the paper by Trajkovska et al. [Trajkovska 2010a].

The Isabel [Isa] platform for videoconferencing service that preceded this research, included a feature in the infrastructure, that registers and stores streaming packet time stamps of the participants in a video conference. One way to implement automatic functions for calculation of QoS parameters (bandwidth, jitter and latency) could potentially come as an extension of such a feature. The APIs could be offered as a toolkit or a library to the clients of the service via graphical web interface. This would enable them to access the streaming service's details, by previously registering to the provider's portal. The queries could include: a list of the currently offered streaming sessions, a detailed status of a particular streaming content, currently connected users, and their QoS status (for example their bandwidth to share).

The API would furthermore offer a price model, that permits straightforward decision on whether to watch streaming in a client-server or a P2P manner. Unlike the models for ISP-guided choice of peer selection, this model has a new and completely independent way for selection guided by the clients' preferences of QoS parameters and price packets. This encourages the clients to take maximum advantage of the model, while enabling to the provider a complete overview of the dynamic QoS changes. With a monitoring tool for QoS representation, the provider could make a quick decision to reinforce its resources and scale the service out in case of flash crowds. At the time of proposing this architecture, it did not exist a similar streaming service for commercial use that is fully based on QoS assurance. The next research objective was to explore the possibility of a price model for the multimedia streaming service that will base on different QoS levels.

Figure 3.1 presents an overview of the proposed architecture. It depicts the cloud that hosts video servers, the clients and the peers. The number of video instances depends on the dimension of the streaming service and it can scale with the number of customers' requests. The service has two types of users: clients and peers. The idea behind this model is to offer the video service to both, customers who are willing to pay for higher quality, and peers as a free of charge service for lower quality. The clients can be *first level* of directly connected clients (Client C1, C2 , C3,..C6), or *lower level clients* (Client C4_1, C4_2, C5_1 and C5_2). The first level clients after registration, access the provider's portal in order to choose among the offered price plans that are bounded to video quality. Afterwards, they choose a price plan that offers high streaming quality and make a contract directly with the service provider.

Similarly, lower level clients are those who consult the APIs in order to obtain a list of the existing clients and QoS status. Depending on the QoS intervals, they pick up a suitable a client that can stream the video with a lower quality for a cheaper price model. The plan lets them connect to the selected host client, instead of connecting directly to the video source hosted in

3.2. ARCHITECTURE DESCRIPTION

Figure 3.1 : Hybrid client-server and P2P streaming cloud architecture

the cloud. Exists also a possibility for a peer to connect to lower level clients who want to offer their service for free (Peer P1, P2, P2_1, P3). Although represented as a multi-tree, the streaming topology does not restrict the overlay type.

The service provider is responsible for centralized management of the customers' contract policies and the streaming service's control. If the load in the system increases in a case of a sudden arrival of new customers (flash crowds), the provider can extend and scale the service. This can be done within the domain of the same cloud provider, across the infrastructures of different providers, or on premise. The video provider can also offer to its clients a streaming service form third party providers.

Such a cooperation where the service provider acts as a mediator between third party servers

Table 3.1 : Streaming plans including current users and their QoS

QoS	UserID: Bandwidth (Mbps), Latency (ms), Jitter (ms)		
Gold	C1: 25, 20, 0.0435	C8: 19, 10, 0.5366	C4:..
Silver	C5: 13, 40, 1.4785	C9: 11, 60, 1.3299	C6:..
Bronze	C2: 4, 80, 2.3401	C5: 5, 90, 2.7832	C7:..

and its own clients, makes supplementary streaming contents available. The provider of the video service profits from the cooperation with the cloud service provider, in a way that it provides a reliable service that can scale. By utilizing the excess bandwidth reserves in the P2P model, it additionally increases the popularity of the service. The service offers higher bandwidth, lower latency, better load balancing, scalability and robustness, all thanks to the topology organization and the cloud infrastructure resources.

The author was researching similar cost model examples for video streaming services offered by cloud and content providers. Some examples include the idea of Amazon Reserved Instances [res] and a payment method implemented by some telecommunication companies. The idea was to promote the model proposed with this contribution, as a base to build uniform model for coordination between streaming service provider and cloud provider.

3.2.1 APIs and service for streaming

In continuation the API functions will be described following an event flow scenario. A user logs in to the service portal willing to watch a live streaming event (or video on demand).

After selecting the video content, the user invokes the service API function (1) that returns a table with the current subscribers that are watching the specified streaming content. Call of the function (2) would return a table, with type of price packets and users of each packet watching the streaming content.

- (1) `get_user_table(streaming content)`
- (2) `get_price_packets(streaming content)`

Shorten version for this output is shown on Table 3.1. It shows the user ID and the three QoS values for the streaming packets. Price packets are defined in function of the QoS bounds guaranteed to the user of the streaming service. We suggest a generic model containing three types of price packets: Gold, Silver and Bronze.

Gold plan as a most expensive, offers the best streaming quality with highest bandwidth and lowest jitter and latency. *Silver* and *Bronze* offer qualities targeted to users who prefer less quality at a lower cost. The service could be set up on demand in minutes, hours, per streaming content etc., which together with the price of the packets is determined by the provider.

3.2. ARCHITECTURE DESCRIPTION

Table 3.2 : User Status Table

QoS / Client (C) Peer (P)	C	C-C	C-P	C-C, P	P	P-P
User ID	C1	C9	C5	C2	C10	C12
Band. (Mbps)	25	11	5	4	2	1
Latency (ms)	20	60	90	80	90	110
Jitter (ms)	0.04	1.32	2.78	2.34	3.03	3.44

Besides direct client-server connection, clients can connect to other clients and extend the streaming tree. New users can obtain information for connected clients invoking the associated API functions. For example invoking (3) would return all the clients watching a streaming video along with the current QoS values. One can require a view of all peers registered in the session by calling (4).

```
(3) get_clients(streaming content)
(4) get_peers(streaming content)
(5) get_bandwidth(streaming content)
```

API (5) returns a list of all clients and their current values for bandwidth. Similar API would be available for latency and jitter respectively. It should also be possible to combine several APIs to infer a broader picture of the service or specify desired values for some parameters via Post APIs.

```
(6) get_bandwidth_and_latency(streaming content)
(7) get_peers([bandwidth], [jitter], [latency], [packet loss])
(8) get_direct_clients(price-plan, bandwidth, jitter, latency)
(9) get_clients_status(streaming content)
```

Function (6) returns a list of clients together with their bandwidth and latency values, while (7) lists all peers whose values for bandwidth, jitter and latency are in the specified interval (some of those could be left empty if not required). Number (8) is complex function returning list of direct clients with specified QoS who have contracted a specific price plan (gold, silver or bronze). Having visible this information, a new user can choose to either become a first level client, or connect to other client(s) in the streaming tree depending on their QoS and price preferences. The choice can be also to connect as a peer, if the user is not willing to invest in better quality for the particular content. Once the plan selected, the client takes care of the payment method.

This way, potential clients and peers have transparent view on the streaming status related to the QoS parameters, intervals, prices per interval, available clients and their status. This allows them to easily choose most suitable contract and connection type depending on the QoS. Another API function (9) would return the Table 3.2 containing the status of the clients sorted according to the service they offer.

The first row denotes the status categories. The tuple denotes if a user identified with its ID number is either client or peer, and the second one denotes whether they permit their service to other clients, peers or both. In a case they do not offer further streaming, the second letter is dropped. For ex. $C9(Client - Client)$ denotes that user $C9$ is client and permits only clients as lower level connections. $C2(Client - Client, Peer)$ states that user $C2$ is client and permits connection from both clients and peers. Note the absence of category $ID(Peer - Client)$ from the obvious reasons that once a user becomes a peer, it is not permitted to offer service to clients, due to low QoS values and therefor this category has no price established. As a consequence a client(peer) with QoS values lower than a certain threshold, could offer a further streaming in a P2P fashion exclusively.

Lower level clients (silver or bronze) can also offer their service as a P2P. For example, lets say that a bronze client has the following QoS tuple (Bandwidth: 2Mbps, Latency: 90ms, Jitter: 2.55ms); this user has no benefit to charge another client(s) due to the low bandwidth values it has, thus he sets up his status to $C(Client - Peer)$ contributing to P2P expansion of the streaming tree. A possible incentive scenario for a $(Client - Peer)$ case could be a special offer by the provider in terms of improved price packet, better position in the topology, or discount to the cooperative clients. This will makes the entire service more resilient since it will: (1) strengthen the client-server model opening new connections at higher level (2) improve the P2P delivery tree. A reward for lower level client can be placing it directly in the cloud as a first level client, while preserving the current price plan.

Each client/peer decides their own status. As noted earlier the goal is to join client-server and P2P streaming under the same model for achieving a hybrid P2P-cloud system with improved welfare. Once the choice is made, the client gets an ID number, pays for the service, set up its status and start enjoying the streaming.

Figure 3.2 depicts reduced streaming scenario including a table that contains the user's in the system classified according to the bandwidth intervals of the price plans. The first column contains the QoS upper and lower bound values of the bandwidth, and the second – the price associated to that interval. The third column contains the current user in each category together with their status. This table serves for a new coming user to chose a convenient price plan and a host according to his preferences for QoS.

As previously described, the cloud architecture enables an enhanced centralized control of the involved clients and peers. At the same time the QoS interface reflects current state of the resources, which facilitates to the service provider better monitoring and management. Scaling the infrastructure by provisioning new video servers, improves the QoS of the service in general. In order to avoid service abuse and frauds, all contracts are established through the cloud control system and the provider's management portal. This also prevents clients to offer service setting personal random prices. Payments between a new and a current client can be arranged through a service payment method after login.

3.2. ARCHITECTURE DESCRIPTION

Figure 3.2 : Example Topology with QoS Table

Considering the problem of churn, the fact that the clients pay for the service, increases the likelihood that they will stay connected until the end of the streaming. However, legal commitments should be specified in the SLA for higher level clients who want to leave the session temporally or permanently. The system should then reconnect the affected clients to some others from the same price range with a status type set to $ID(Client - Client)$. Churn among peers does not directly affect the global integrity of the architecture.

At the time of proposing this architecture, it was still difficult to combine client-server and peer-to-peer streaming within the same service. One challenge for this was the novel topology and overlays that have to be created to host this service model. Another aspect was the requirements for new and accurately established cost policies. Therefore this conceptual architecture was one of the pioneers to present a combined solution between these two technologies. The cloud computing was identified as a catalyst, which offered attractive price plans and structural model to host novel services and applications. It was also believed that it will reinforce the interaction between the service providers and the end users, through the newly build applications in the cloud.

3.2.2 Quality of service in streaming

The QoS values mentioned in the previous section could be retrieved via automatic QoS calculation algorithm. As it was mentioned in the introduction, a feature that could offer the

acquisition of packets' timestamps can be used to make the QoS calculations for the service. We derived the packet time stamps from a component of the Isabel platform for videoconferencing service (developed in a related work at our research laboratory). The idea is described on Figure 3.2.

As a streaming session initiates, time stamps of outgoing and incoming audio and video packets are registered. The received data is stored on the provider database server and on the clients' machines. We will consider only time stamps of outgoing packets assuming the processing packet time is very small, thus irrelevant for the calculation. Following are the well known formulas for QoS calculation as they would be used in our scenario, provided we have included the time stamp feature in the web service logic:

Latency: We observe Client C2_2 on Figure 3.2. If t_{C22} is packet time stamp at Client C2_2, then its latency will be calculated by summing all the previous latencies from the server down to client C2 in the tree, adding the new time stamp for Client C2_2: $latency(C2_2) = t_{C22} + latency(C2)$, where $latency(C2)$ is the latency of client C2 for the same packet. Latency for both audio and video packets is calculated according to the same formula and so for every new user in the streaming tree.

Jitter: Jitter represents the statistical variance of the real time protocol (RTP) data packet inter-arrival time [Schulzrinne 2003]. Therefore if S_i is time stamp for packet i , and R_i is the time of arrival in RTP time stamp units for packet i , then for two packets i and j we have the difference $D(i, j) = (R_j - R_i) - (S_j - S_i) = (R_j - S_j) - (R_i - S_i)$. The inter-arrival jitter of any client is calculated instantaneously as the data packets are received by the client using the difference D for that packet i and the previous packet $i - 1$ according to the formula $J(i) = J(i - 1) + \frac{(|D(i-1, i)| - J(i-1))}{16}$. The division by 16 is in order to reduce noise.

Bandwidth: Finally to calculate the bandwidth it was assumed that the packet size is constant during the streaming session. Supposing that the streaming channel is asymmetric, then only the upload bandwidth will be considered of interest for new users. If the streaming bit rate br is known, depending on the audio/video codec used, the bandwidth of Client C2 is simply calculated as $bandwidth(C2) = n * br$, where n is the number of clients to which peer C2 is forwarding the streaming packets (two in this case). The bandwidth can be also calculated as $bandwidth(C2) = (ps_a * n * p_a / t_a) + (ps_v * n * p_v / t_v)$, where ps_a and ps_v denote the packet sizes of the audio and video the stream respectively (ex. bits/packet). Analogically p_a / t_a and p_v / t_v are the audio and video packet rates (ex. packet/time). To show updated and more accurate QoS representation, new time stamp values can be used in the equations to recalculate the QoS within smaller time intervals.

Contribution. This chapter presented a novel use case for cloud infrastructure, introducing an architecture for P2P multimedia streaming in both client-server and P2P style. At the same time it was shown QoS API functions to be considered by the cloud streaming providers. This model facilitates to users a clear representation of the status with the objective to choose the best

3.3. ASSESSING CLOUD INFRASTRUCTURES FOR P2P STREAMING

matching model: establish contract with the service provider, or join as peer in the free-of-charge streaming session. This idea was a natural approach towards solving the famous P2P-ISP issue, by offering a transparent solution for both the service providers and the end users. It moreover set the stone for scalable cloud SaaS based on QoS guarantees to be deployed by streaming service providers.

3.3 Assessing Cloud Infrastructures for P2P Streaming

In the study covered with the previous section, we proposed a hybrid and distributed architecture for multimedia streaming. Combining P2P and cloud computing, the study was focused on the streaming model required to make the architecture commercially usable. It offered in relation, an idea to derive APIs for the streaming service in the cloud, based on QoS ranges. This solution is an alternative to the P2P-CDN hybrid architectures because it is not based on the CDN principles. On the contrary, it implements P2P systems over public and private cloud infrastructures.

This section discusses the research following the architecture previously presented, with a focus on characterization of the cloud provider network in its early days of expansion. Introducing novel features, the cloud paradigm has immediately attracted a large academic audience to follow up on bashing the cloud's capabilities and capacity in supporting various applications. Similar to those studies, we wanted to focus on testing the quality of service for a real-time streaming hosted across geographically distance cloud instances. The main objective was to perform tests that will demonstrate the variation of the quality of service, as perceived by the participants in a large-scale cloud streaming scenario. In cloud terms, a resource can be any compute network and storage component that is leased by a cloud provider to the users, under certain conditions and price. Since the video streaming is supported on application level and it is dependent mainly on compute resources, a streaming architecture was created and deployed using IaaS virtual machines called *cloud instances*. Parallel to the cloud adoption process, the P2P technology was witnessing a growing number of online services and portals for real-time and on-demand (VoD) streaming. Yet many of those have demonstrated the weakness of the current networks to support services with stringent constraints, like for example P2P video delivery. We argued that a possible solution of this issue can be seen in a well-balanced hybrid distribution network. The description the follows states the detailed objectives and elaborates the steps performed in order to test the feasibility of the proposal.

The test scenarios enclosed in this section report an experimental validation of the previous architecture in terms of QoS and can be divided in two parts:

Benefit of cloud in multimedia streaming. This involved measuring some common network QoS parameters in order to identify the achievements of employing cloud infrastructure in real-time streaming. During that period, researches like the one from Ferretti et al. [Ferretti 2010] have studied QoS such as: timeliness, scalability, availability, trust and security required in cloud SLA agreements. The authors presented an architecture for mobility support of wireless devices

for executing multimedia applications on the cloud, unlike our proposal that assumes fix nodes as participants. Related studies [Wang 2010b, Li 2010] have also failed to address QoS related to real-time streaming. Due to this we extended this research to know exactly the response of those networks. Despite the similar concepts, there was a lack of analysis for QoS in real multimedia scenarios, which combined cloud and external nodes together. We addressed this issue and extended similar studies [Wang 2010b, Li 2010] that failed to assess the cloud network behavior during real-time streaming, in the first part of the measurements.

Quality of cloud for multimedia streaming. The next round of measurements involved mixed P2P-cloud scenarios that we designed in order to approximate real use cases. These scenarios represent a streaming distribution network, where the network nodes can be located on a cloud infrastructure, or elsewhere on the Internet. The objective was to: (1) verify the hybrid network's ability to transmit multimedia streams and (2) relate the QoS trade-off to the number of nodes placed in the cloud. Instead of making comparison among public, private or community cloud, we relied on PlanetLab [Chun 2003] nodes to act as peers that will interact with nodes placed in the cloud. The reason for using PlanetLab was the great expansion of this voluntary network and the possibility to obtain variety of machines over the globe that fit perfectly for representing the P2P part of the experiments. There was a similar work [Weissman] that relied on PlanetLab nodes utilized as edge routers in a cloud proxy network. The authors examined optimal data distributed operations for improving cloud applications and optimizing cloud-proxy network. For the scope of the measurements, we chose Amazon ⁱ as one of the leading cloud providers at that time.

As a summary, this section demonstrates a complete study [Cervino 2011b] for two worlds joined in non-trivial experiments. Instead of using software simulators, we implemented all the scenarios across real infrastructure. The obtained results conclude that when using cloud network to forward streaming sessions among distant peers, some cases allow to enhance the overall QoS of the service. Thus, the results can be taken as a real proof that validates the feasibility of the architecture proposal along with the assumptions for QoS improvement.

3.3.1 Measurements in distributed cloud scenarios

This part presents several scenarios deployed in a tree based distribution network for streaming. The nodes of the distribution network are located in different geographical points around the world. Initially, all the nodes are located in the Internet but outside the cloud infrastructure. In each successive scenario, some of them are gradually moved into the cloud. The rationale of this experiment is to measure the improvements in terms of quality of service when part of the network nodes are moved from Internet to the cloud. A real video traffic was used for the tests in order to better approximate a realistic scenario. This enabled to authentically compare the received QoS results, to the ones required in multimedia transmission. As previously stated, we used PlanetLab

ⁱ<http://aws.amazon.com/ec2/>

3.3. ASSESSING CLOUD INFRASTRUCTURES FOR P2P STREAMING

to geographically spread the distribution network nodes, and Amazon as cloud provider. In the following part, the tools used in the measurements and the measured parameters are described. After that, details from the experimental setup are shown, finalizing with an explanation of the results.

Tools and statistics

In order to implement the measurements, a set of components to generate and redistribute media are needed. It is also necessary that they are flexible and easy to automate in order to make the measuring process as smooth as possible. For that purpose, we based the tools on a previously developed project toolkit, changing them as required in accordance to the scenario, making also the necessary environment adjustments in order to be able to gather statistics.

The tools came from a system called Isabel [Quemada 2005] that was previously developed in the research group. Isabel is a multipoint group collaboration tool which includes video and audio conferencing. The low level components of Isabel are daemons coded in C/C++ that can be operated via a control port using a set of primitives. The video component is able to encode, decode, transmit, receive and present video using a variety of codecs. Furthermore, it can be configured to broadcast video with different sizes, bit-rates and codecs, either from a file, or from a synthetic video source. Real-time Transport Protocol (RTP) is used to transmit media.

The second component and the most important in terms of characterizing the network is called *irouter*. Its main objective is to receive and forward packets from and to other components including other "irouters". In our scenarios, the irouter receives data from the video daemon, presents it and then forward it to other irouters. This component was modified to gather data from all incoming packets and to process the statistics that are presented later in the text. For practical reasons, the irouter component dumps the raw data in text files that will later be collected and processed statistically in order to separate the actual measurement process from the calculations. The data accumulated is roughly equivalent to that explained in the Intra-Cloud measurements described in this document. Following are the parameters collected by the irouter:

- *Packets received*: The total packets received in an interval.
- *Bytes received*: The number of bytes received in an interval.
- *Packets lost*: The amount of packets lost in an interval. A discontinuity in the sequence number of RTP packets is perceived as a loss.
- *Packets disordered*: A packet is considered disordered if it arrives in time to fill a gap in sequence numbers. It is considered to be "in time" if it arrives in the same interval.
- *Packets duplicated*: A packet is considered duplicated if two packets with the same sequence number arrive anytime.

- *Packets recovered*: When forward error correction (FEC) is active, packets recovered via this mechanism are recorded here.
- *Inter-Packet Gaps (IPGs)*: The time difference between the arrival of two consecutive packets. IPGs are used in order to calculate the average jitter introduced by the network. As explained in [Imran 2007b] you can theoretically obtain an accurate approximation to an upper bound of a jitter using:

$$\sigma^2 = \frac{1}{N_i - 1} \sum_{j=2}^{N_i} (IPG_j(i) - E[IPG(i)])^2$$

$$J(i) = 3\sigma$$

For a normal distribution of random jitter this value means that the 99.7% packets will arrive with a jitter lower than $J(i)$. But in practice it does not follow a Normal distribution because there are several external causes other than only random noise. So we preferred to calculate directly the 99.7 percentile of values in:

$$|IPG_j(i) - E[IPG(i)]|$$

Description of the streaming scenarios

To test the already introduced hybrid model, we designed 4 scenarios of nodes connected in 4 levels depicted in Figure 3.3. In each scenario the nodes of each level were gradually moved from Internet to the cloud infrastructure. As hosts for the cloud nodes, we used Amazon instances equivalent to those explained in section 3.3.2 and PlanetLab for the non-cloud nodes, all distributed on machines running in different countries. Different geo-locations for the peers were targeted in order to identify the new challenges in cross-continent multimedia content distribution. Other goal was to determine the geographical importance when designing the streaming topology. As shown in the figure, the scenarios are deployed in three different zones: Europe, America and Asia, containing 4 levels of nodes connected in a tree topology.

The node distribution was organized according to the location of the available Amazon EC2 regions at the time of planning the tests. This included: West USA, East USA, Europe and Asia. Similarly, PlanetLab nodes were located in the same regions, in order to effectively compare the streaming performance between them and the nodes of the Amazon network infrastructure. The most common PlanetLab hardware has a specification of 4x Intel 2.4GHz cores with 4GB RAM, well beyond expected for the measurements. The traffic type used in the tests mimicked a real video streaming as used in the Internet services, i.e. a test video pattern of 640x480 pixels encoded in MPEG-4 with a bitrate of 500 Kbps, transmitted using RTP. In order to discard outliers, three executions of each experiment were run per scenario and average values were calculated. Executions were run from Monday to Friday during business hours when PlanetLab

3.3. ASSESSING CLOUD INFRASTRUCTURES FOR P2P STREAMING

Figure 3.3 : Node distribution

Figure 3.4 : The 99.7th percentile of jitter in each scenario

nodes were available.

With those assets, we defined four scenarios of nodes placed in a tree hierarchy, chosen within the closest location given by the provider. Thus, in Scenario 1, all the nodes are PlanetLab machines located in the country specified in the figure. In Scenario 2, the level 1 node (the source node in Norway) is replaced by one node located in the European datacenter of Amazon. In Scenario 3, the second level nodes USA-1, France and Taiwan are replaced by corresponding Amazon nodes located in the respective datacenters in: USA, Europe and Asia. Finally, Scenario 4 substitutes the level 3 nodes (Puerto-Rico, USA-2, Italy, Germany, Japan and China) by the the corresponding Amazon nodes located in datacenters within USA, Europe and Asia.

Figure 3.5 : Bytes received in each scenario

Figure 3.6 : Packet losses in each scenario

Quality of service tests and results

The figures 5.26, 3.5 and 3.6 show the values for jitter, bytes received and packet loss, as observed by each node in the distribution tree. The X axis groups in consecutive blocks the formerly described four scenarios, with nodes in each scenario ordered according to the geographical criterion.

Figure 5.26 shows the 99.7th percentile of jitter values obtained for each of the four scenarios. The figure indicates clearly that the jitter values decrease when levels of the distribution tree are moved into the cloud network. We also noticed a high variability in the obtained results for the first scenario. As expected, significant jitter degradation appeared in Scenario 1 where all the communication links are outside the cloud network. This happened mainly for packets crossing intercontinental links – from Europe (Norway) to America (USA-1), affecting the whole subtree (USA-1, Brazil, PuertoRico and USA-2) with big jitter values. Even when the source node was moved inside the cloud provider (Scenario 2), the jitter values were still big when the packets left the cloud network to cross from Europe to America. However, when the root nodes of each continental sub-tree (USA-1, France, and Taiwan) were moved into the cloud network (Scenarios 3 and 4), and intercontinental traverse was fully enclosed inside the cloud network, the jitter values decreased. Observing the jitter values obtained in Scenario 3 and

3.3. ASSESSING CLOUD INFRASTRUCTURES FOR P2P STREAMING

4, it is not expected to have severe jitter problems [Karlsson 1996] even if a multi-conference service was deployed. Also, it must be observed that jitter values did not improve significantly when the third level nodes of the distribution network were moved into the cloud. Therefore we can conclude that the poor jitter values are bound to crossing intercontinental links (from first to second level) and so, regardless of the cloud-enhanced connectivity between second and third level nodes, a significant jitter improvement was not achieved.

Figure 3.5 shows in *box plot* format the distribution of the streaming session throughput for each scenario and network node. The number of bytes received in each node is accumulated during 400 seconds period, and a summary of five values is shown in each *box plot*: minimum value, lower quartile, median, upper quartile and largest value. Again, Scenario 1 shows a high variability in the number of bytes received (mainly in American nodes). The rest of scenarios exhibit a nearly uniform distribution, and so, we can conclude that a throughput degradation appears during intercontinental traversal when Internet nodes are used exclusively. The problem can be overcome by moving the source node in the cloud network (scenarios 2, 3 and 4). In these scenarios, some links of the intercontinental path traversed by the session packets, are internal transmission links of the cloud network. These links exhibit a better throughput than the corresponding Internet external links, used by the PlanetLab nodes to pass over continents. This indicates to improved overall throughput in the scenario where the source node is hosted in the cloud.

Finally, Figure 3.6 shows the packet loss during the streaming session, represented per scenario and per network node. Analogously to the previous results for jitter and throughput, the cloud transmission links experienced lower packet loss than the corresponding Internet links of the PlanetLab nodes. Since the packet loss values were accumulated for each link of the streaming session path, more nodes placed in the cloud implies less packet loss occurrence. This case can be observed clearly in Scenario 4, where the unique nodes outside of the cloud (Brazil, Spain and Korea) are the only ones to accumulate bigger packet loss than the rest.

The conclusions in this section are: (a) jitter degradation is nearly eliminated if both, the source and destination node of each intercontinental traverse are placed in the cloud, (b) throughput degradation is alleviated by placing the source node of each intercontinental traversal in the cloud, and (c) incrementally moving nodes to the cloud, decreases packet drop.

3.3.2 Characterizing cloud network for streaming

In this part we initially elaborate on the the common network parameters for streaming. In the next steps the measuring environment is presented, describing the datacenters and every component used in the scenario. Finally the results are presented by interpreting the collected data, and offering a final thoughts about the proficiency of the Amazon cloud to host real-time streaming service.

At the time of completing these research, related studies were focused on similar analysis of the Amazon network for different application use cases. One example is Wang et

Figure 3.7 : Architecture of the measuring environment

al. [Wang 2010b], who measured the network performance over different virtual machine types. They demonstrated that using Amazon Small Instances is harmful to the network responsiveness. The approach they take includes performing measurements inside a local datacenter exclusively. We use this work as a foundation to measure the network performance across multiple Amazon datacenters.

Li et al. [Li 2010] show how the variation of clients' request and the infrastructure itself can alter the application's execution when run over different clouds, with this putting into perspective the relevance of network performance and traffic responsiveness between different datacenters of the same provider. Same as in the previous research is the lack to consider various datacenters, which we believed was an essential test for the hypothesis of a given architecture created for the cloud.

Relating to the initial set of measurements we considered bandwidth, delay, jitter and packet loss, as essential network parameters to be reported. These QoS parameters has been traditionally considered in real time communication since the appearance of VoIP applications [Klepec 2001, Imran 2007b].

Bandwidth: The architecture of this study is supported by hybrid system (P2P and centralized) used for video/audio streaming. As alternative use case is a scenario in which end users could watch Internet TV channels. These users are accessing the Internet typically through a wide range of access networks such as xDSL, Wifi, etc. As a result of different setup and heterogeneous connections, a streaming service must allow a configuration of different bandwidth levels. We assumed no bottlenecks exist in the core network for any of the cases examined. For this reason

3.3. ASSESSING CLOUD INFRASTRUCTURES FOR P2P STREAMING

we also measured the limits of available Amazon network's bandwidth.

Delay: The users can access a streaming session from any point in the planet. This means that they can join the service from access points with additional network delays in their connections. Even the core network that will be used as a transport network may introduce inherent propagation delays.

Jitter: It is the variation in the time between packet arrivals caused by network congestion, timing drift, or route changes. Jitter could manifest in different ways depending on the application. Web browsing is fairly resistant to jitter, but any kind of streaming media (voice, video, music) is very sensitive to it. As for these reasons jitter is an undesired parameter in any network connection, and according to ETSI [McLellan 2010] it is very important that the amount of jitter is lower than 20 ms in VoIP applications therefore we applied it as a default limit.

Packet loss: Packet loss can occur for variety of reasons including: link failure, high levels of congestion that lead to buffer overflow in routers, Random Early Detection (RED, a technique used in routers to apply congestion control), Ethernet problems, or an occasional misrouted packet. Packet loss degrades voice and video quality. If Packet Loss Concealment is used (PLC, a technique used to mask the effects of lost or discarded packets), then isolated losses may be less noticeable. Nonetheless, packet loss occurring in bursts of 20-30% during 1-3 seconds may seem as a low rate in average, although the user reports call quality problems. For achieving more reliable results in our scenarios, it is favorable to have a core transport network with low packet loss rate.

Environment setup

The resources used for testing involved three amazon datacenters: US-East-1 in Northern Virginia, EU-West-1 in Ireland and AP-Southeast-1 in Singapur. Another region located in California was used, which we considered irrelevant because of its proximity to the US-East-1. Figure 3.7 represents the architecture of this measuring phase. It depicts the three Amazon datacenters used to run multiple virtual machines, used as clients and servers. In the clients we run the tools *iperf* and *ping* to measure the connections established with the servers. Each virtual machine was a Large Amazon instance that ran Ubuntu Server operating system (Ubuntu Karmic 9.10 AMD64 Server). We performed measurements of network delay using the *ping* test tool. For the bandwidth, *iperf* sends and receives packets using only TCP connection. It uses read and write buffer size of 8 KB, maximum segment size equals to 1460 bytes, and window size is set to 85.3 KByte. This command is the same for both client and the server. For the jitter and packet loss the read and write buffer size is set to 8 KB. The set up commands used in *iperf* for server and client modes are listed in Table 3.3.

Table 3.3 : Applied commands for client and server mode

Server (S) & Client (C) QoS	iperf commands
Bandwidth (S)	<code>iperf -i 10 -s -p 6031 -f k -y C</code>
Jitter and packet loss (S)	<code>iperf -i 240 -s -u -p 6032 -f k -y C</code>
Bandwidth (C)	<code>iperf -t 3000 -i 10 -c server -p 6031 -r -L 6030 -f k -y C</code>
Jitter and packet loss (C)	<code>iperf -t 3000 -i 240 -u -c server -p 6032 -L 6030 -f k -y C</code>

Figure 3.8 : Measurement results between datacenters EU-US

3.3. ASSESSING CLOUD INFRASTRUCTURES FOR P2P STREAMING

Figure 3.9 : Measurement results between datacenters EU-AP

Figure 3.10 : Measurement results between datacenters US-AP

3.4. CONTRIBUTIONS SUMMARY AND CONCLUSIONS

Measurement results between different datacenters

The figures 3.8, 3.9, 3.10 show the related QoS graphs respectively: the bandwidth graph represents the cumulative distribution function (CDF) of the tests that measured the amount of available data bitrate in each connection between two regions, shown in bits per second. Analogically, the jitter graph represents the CDF of the jitter in these connections. The packet loss graph represents the probability density function (PDF) of the packet. Finally the Round Trip Time (RTT) graph indicates CDF of the RTT delays between two nodes in different regions.

Europe-USA: Figure 3.8 shows the results between the Europe and USA regions. The bandwidth graph marks an average over 45 MBits per second for a single sender thread. Meanwhile jitter results show a jitter bounds between 0.1ms and 1ms, which acceptable even for VoIP communications. Regarding the loss rate, the low results proved it suitable for streaming purposes. Finally RTT is on the margins between 90ms and 92ms, that are typical values for connections between distant peers.

Europe-Asia: Figure 3.9 represents the measurements between the Europe and Asia regions. There is an increment in this case, of the outgoing bandwidth that reaches over 70Mbps value for a single sender thread. There is however a divergence between the incoming and outgoing values, such that the incoming traffic values decrease and oscillate in the interval between 8 and 20Mbps. Just as in the previous scenario, jitter and packet loss are kept low, thus satisfactory for streaming purposes. In the case of RTT, there is an increment in the mean value reaching 270ms. This is mainly because the peers are geographically far away from each other.

USA-Asia: Figure 3.10 represents the measurements between USA and Asia regions. Most of the bandwidth measures show a value of 90Mbps for outgoing data while for the incoming data the bandwidth drops to values over 20Mbps. This case notes a small increase of the jitter values exceeding above 0.3 ms. Despite this, it is kept too low in order to become a concerning value for streaming services. Low packet loss rate and RTT over 250ms are a typical values for distant connections.

As a recap from measurements above, using Amazon cloud resources to stream multimedia content, is highly efficient and meets the QoS requirements of a distribution service for streaming.

3.4 Contributions Summary and Conclusions

The measurements scenarios presented in this section, validated the idea of in the proposal described in Section 3.2. We wanted to identify the benefits of the hybrid architecture for streaming by choosing Amazon cloud as the most dominant provider at the time (that uses cloud infrastructure to leverage the service model in terms of QoS). We demonstrated that when used as a core in the P2P architecture, the cloud can bring several benefits to the service:

- (1) Provides video access from a public domain, rather than through other peers that are usually behind limited Internet connections.

- (2) Regulated intra- and inter-DC QoS policy, improves the overall quality in streaming vs. the one in traditional P2P and CDN.
- (3) Strategical placement of distribution nodes in the cloud can lead to more robust and coherent streaming service.
- (4) Opens new market possibilities for harvesting business models based on resource share.
- (5) Facilitates a trade-off between streaming capacity cost and ownership cost for multimedia provide.

The proof of concept topology that we simulated with the scenarios, shows that a video service provider can take advantage of reduced packet loss and low latency that exists among the datacenters of a cloud provider, and shape up accordingly the service topology. As a result of controlled QoS within the cloud infrastructure, we managed to reduce the jitter in the end points, and we improved the available bandwidth. We also minimized the packet loss in each of the peers involved in the streaming scenarios. This effect is especially visible in the results of the Scenario 2 and 3, which demonstrate that using connections between distant cloud datacenters can help to improve the QoS response of the P2P streaming service. This notion comes handy in the design of a robust P2P overlay, knowing the physical location of the peers.

In the forthcoming chapter, we describe the incremental results from the later research in context of a business model for the streaming service, by tackling the economic achievements and social welfare that it offers.

Chapter 4

Business Model for Cloud-based P2P Streaming

Peer-to-peer (P2P) streaming is still a challenging technique for commercial multimedia delivery. To cope with dynamic behavior of different user profiles, designing efficient incentives takes essential part of the P2P algorithms. In parallel, a dedicated large-scale and flexible cloud infrastructure on pay-as-for-go basis has become a legitimate asset to deploy streaming services. This chapter analyzes the effect of user behavior in a multi-tree P2P overlay and describes a business model based on monetary discount as incentive in a P2P-Cloud multimedia streaming system. We use the previously presented model of the P2P streaming service, to apply the idea and validate the quantitative effect on both the users' and the provider's welfare in the system. A billing policy based on QoS levels for the perceived video, that includes alternatively a remuneration incentive for the cooperative users, contributes to the P2P streaming due to the two main reasons: increases provider's welfare in terms of monetary benefit and saves infrastructure cost; increases users' welfare in terms of increased cooperation and motivation to participate in the streaming and enhances the overall system's integrity and performance.

Furthermore a methodology is described on how to apply discount-based incentive mechanism in mixed P2P-cloud streaming services. In particular, we modeled an economic projection of the cost-profit tendency of a P2P-cloud service that tackles a tiered, quality of service (QoS) based billing model. With this we aimed to assess the content providers with a fundamental understanding of the billing model tight relation to the user profiles and to the topology of the service. Based on heuristic parameters, the evaluations diagnose a significant effect of the discount incentive, on the provider's cost and profit. The presented model moreover sets the bounds for a provider's revenue and cost if the P2P system is leveraged on a cloud infrastructure. Our case study shows that a streaming system provider can establish or adapt his business model by applying the described bounds to achieve a good discount-revenue trade-off and promote the system to the users.

4.1 Problem Statement

The growing trend of watching TV on the Internet, made the P2P a prominent cost-effective technology for real-time content delivery to end-users. As evidence, various P2PTV and content delivery network (CDN) services today dominate the Internet with more than half of the current online traffic. Nowadays even several telecom providers have adapted their internal infrastructures to offer real-time and video on demand (VoD) residential services. Unlike VoD where the contents are pre-stored and served upon request, live streaming is a delay critical

technology with stringent bandwidth demands during the entire streaming lifecycle. In a centralized model where each user acquires the video directly from the server, a flash crowd scenario may cause bottlenecks at the source node or at the access links, impacting this way the users' overall experience. As alternative, the P2P delivery model offloads the server, in a way that it disseminates the video only to the direct users. Afterwards, the rest of the users (peers) take over and push it further in the swarm. Such a fundamental advantage of the P2P facilitates scalability and contributes to the system's resilience. The downside of P2P, affecting both, the system's stability and the quality of service, is its highly dynamic and voluntary nature occurring in a presence of free riding users and churn. Therefore any P2P-based streaming service must incorporate an incentive mechanism designed to motivate the users to cooperate in the P2P network by offering their free upload bandwidth. In order to achieve a cost-effective streaming, it has become a natural approach recently, to leverage the streaming services with dedicated cloud infrastructure. Such pay-as-you-go resource leasing, provides flexible framework in the presence of few thousands simultaneous users.

P2P with its proven success together with the cost advantage and the flexibility of the cloud, can lead multimedia services towards novel business directions. Relying on these two technologies, the forthcoming research work proposed a methodology towards applying discount incentive mechanism to P2P streaming services. We extended our work described in the paper [Trajkovska 2014c] with an evaluation study to represent both: the effect of the incentive model and the quantitative advantage of the cloud infrastructure on the users' welfare and the provider's profit in a streaming service.

The existing online streaming proposals vary from free, best-effort applications, to commercial P2PTV services with static, flat rate billing models, generally based on proprietary protocols for topology management. Unfolding these services is challenging but crucial burden towards innovative streaming solutions. Recent studies about pricing on the Internet [Ma 2013, Sen 2013] reveal the upcoming trends of sophisticated pricing, by analyzing the historical charging mechanisms, the current business decisions, as well as the pricing competition between application and network providers. Though recently considered by some Internet service providers (ISPs), the authors in the latter find QoS based priority pricing yet to be absent today.

This is one of the pioneer economic study for a commercial streaming service designed to run in a mixed P2P-cloud topology that tackles tiered, QoS based billing model applying discount as incentive. Based on heuristic parameters, it analyzes the significant discount incentive effect on the provider's operating costs and profits. This study presents a general methodology addressed to instruct the designers of streaming services and the content providers, on the constraints to consider and the advantages to assume for their prospective P2P streaming services deployed in the cloud. In parallel, it offers insights into the provider's changes in costs and profits based on the interrelationship between the billing model and the user profiles. Proof of concept as SDN-based

4.2. BEHAVIOR IN P2P STREAMING SYSTEMS

QoS control algorithm to achieve QoS governed user allocation in P2P service, was described in a related work [Trajkovska 2014a], and will be presented in the following Chapter 5.

This chapter is organized in two parts: First the tight relation of the multi-tree P2P overlay topology to the user behavior in the system is discussed. It follows up with presenting the types of behavior in P2P streaming systems. Finally the proposed business model based on monetary discount is presented and applied to the model of the streaming service we proposed in Chapter 3.

4.2 Behavior in P2P Streaming Systems

This section focuses on the influence of both cooperative and uncooperative users in the system based on their position in the streaming trees. To remind, P2P relies on collaborative network of users that are expected to contribute, which in practice, is not always the case. Cooperative users share determined upload bandwidth with other users in the system, unlike the uncooperative also called *free riders*, who enjoy the streaming but don't forward the video to other users in the tree. Malicious peers join the system exclusively to harm its integrity and degrade the overall QoS.

4.2.1 Cooperative users

In a multi-tree overlay the cooperative users improve the overlay construction and contribute to more balanced sub-trees. Moreover they foster the service performance by increasing its shared bandwidth reserves. According to the multi-tree constraints in SplitStream [Castro 2003b], to achieve a feasible overlay a client should contribute upload bandwidth at least as the amount he downloads. For a well-balanced overlay, the authors suggest to connect each user as inner node in only one sub-tree, and as leaf in the rest of the sub-trees, Figure 4.1. Sweha et al. on the other hand [Sweha 2012], suggest connecting the users in minimum number of sub-trees. From those, they appear as inner nodes only in a number of sub-trees corresponding to their upload bandwidth.

Imagine cooperative users with high upload bandwidth. Following the above recommendation, a provider can put them on a higher level in a number of sub-trees equivalent to the amount of their upload bandwidth. This opens proportional number of connections for additional high level users. Therefore, placing the cooperative users in higher sub-trees of a multi-tree P2P overlay, make significant contribution to the overall structure in terms of well-balanced bandwidth reserves. In a mesh overlay, cooperative users increase the availability of the streaming content to the peers across the entire overlay. They also improve the overlay integrity by forming more stable connections among the peers.

4.2.2 Uncooperative users

In a P2P system that is based on voluntary cooperation, it is inevitable the existence of users who profit from the optional bandwidth share. These uncooperative users do not contribute any uploading bandwidth to the service and increase for free their personal welfare. In a multi-tree

Figure 4.1 : SplitStream [Castro 2003b] overlay topology

overlay, this behavior generates elapsed branches in the sub-trees where the uncooperative user appears as inner node, leaving his successors to be orphans nodes. Higher level free riders make big impact to the overlay integrity as they leave empty connections and hence provoke sub-trees mis-balance. From the above reasons, uncooperative users should be always connected at a lower tree level to reduce the impact on the tree expansion and to avoid system's resource consumption.

Hosting selfish users in a mesh overlay impacts to the same extent, since the peers would lack the resources of the selfish peer and thus experience a lower video quality. The selfish node on the other hand take the advantage of the available bandwidth of the peers he is connected to.

4.2.3 Malicious users

Unlike self-interested users, malicious ones are even worse candidates who threaten the system integrity. They exploit the system by harming other peers and prevent them to enjoy the streaming. They do this by performing various actions such as: presenting false identities, data drop and data pollution, collusion attacks with other peers, assigning false reputations, etc. It is yet a challenging task to detect and deal these users even for a widely used P2P streaming systems.

4.3 Discount Incentives in P2P Streaming Service

The exponential growth of online video traffic has raised the demand for better video quality. In a search for increasing revenues, several content providers have refined their flat-rate billing models with new premium offers for high definition (HD) video delivery. Deploying a P2P streaming service in a centrally controlled system opens as many possibilities for the streaming content providers as for the users. The *elastic* and *on-demand* cloud infrastructure avoids

4.3. DISCOUNT INCENTIVES IN P2P STREAMING SERVICE

underutilized on-premise resources for the service providers. As a result, the trend of using cloud infrastructure to leverage multimedia services has increased in the last years. In our related study [Cervino 2011b] we showed that using cloud-based P2P streaming, increases overall system performance and improves QoS for the participants in the multimedia streaming service. This traces the way towards innovative user-oriented price policy as a prerequisite to define a good business model for streaming service in the cloud. During the time of this research, we did not find any commercial streaming service that based the video QoS as a fundamental of their business model.

The baseline example for a commercial service on the Internet was to classify the clients in premium and basic membership based on subscriptions. Premium users usually pay monthly/yearly price and enjoy certain privileges such as, commercial-free, complete service access. Basic users on the other hand, are charged per content or have restricted access of contents and may also receive commercials during the streaming. In a P2P streaming service with a subscription or on-demand billing model, the idea of a remuneration-based policy is to achieve an increased participation and collaboration among the users, by voluntarily sharing their free upload bandwidth. The cooperative users are rewarded with a discount that is applied to the basic price, and it is proportional to the amount of bandwidth they decide to share.

In this study a tiered pricing model is proposed with three video qualities, similar as the one which was studied by Netflix [net b] back in 2014, after this work was published. Nowadays Netflix offers a subscription plan [net a] related to quality of video (SD, HD, HD + Ultra HD) and number of devices from where a user accesses the video. To achieve different qualities, a single video stream can be handled using techniques e.g. Multiple Description Coding or Scalable Video Coding. These transcoding mechanisms divide a single video stream into several sub-streams (chunks). Combining multiple sub-streams, proportionally improves the perceived video quality. This facilitates to the provider, a streaming service offer with diverse QoS-based pricing schemes.

Since this study tackles the economic aspect of the cloud and its price relevance to support a P2P streaming service with monetary incentive, we referred to Xu et al. [Xu 2013] study related to pricing in cloud computing. The authors explore the cloud's economical impact based on several aspects of the cloud systems, such as: best-effort nature, performance guarantees, and capacity adjusting. Niu et al. [Niu 2012] present a pricing theory for BW guarantees in multiple VoD providers, based on various cloud providers. Focused on optimal pricing models, He et al. [He 2014b] provide a guideline for the cloud-based video providers, on how to optimally procure virtual machine instances to satisfy the user's demands.

This work differs from previous works, in that it focuses on a particular streaming service setup and instead of achieving an optimal model, it establishes the billing model constraints with the objective to offer a comprehensive representation of the cost-profit tendency under different user profiles. In particular, our evaluation considers different video qualities with tiered prices and discounts, based on empirical values from the literature and the existing commercial services.

Figure 4.2 : P2P-cloud architecture with multi-tree overlay for video dissemination

As a case study, relying on novel topology and billing model setup, this contribution aimed to establish a fundamental analysis of the current Internet streaming tendency and guide for the content providers and the ISPs towards novel business models. In the next section we present a case study for a business model based on QoS which applies a discount-based incentive.

4.3.1 Topology and business model

To meet the stringent QoS constraints of a live streaming service, a cost-effective and highly flexible infrastructure is required. The emergent cloud infrastructure facilitates a grand choice of *on-demand* virtual machines called *instances*. The possibility of dynamic resource scale reduces significantly the cost for underutilized on-premise resources. Therefore such elastic provision can be considered as legitimate content provider's choice to deliver real-time streaming content to the users. In this section we base on our previously proposed multimedia streaming system [Trajkovska 2010b] to illustrate what are the provider's cost constraints to establish a commercial P2P service. We also describe discount model based on monetary incentive and discuss its application on the system. This system relays on cloud infrastructure for QoS improvement and service-scale assurance in case of overload.

The trend of using cloud infrastructure to leverage multimedia services has increased in the recent years. In a related study, Cerviño et al. [Cervino 2011c], we showed that using cloud-based P2P streaming increases the overall system's performance and improves the QoS for the participants in the multimedia streaming service. Offered as *pay-per-use*, the cloud liberates the

4.3. DISCOUNT INCENTIVES IN P2P STREAMING SERVICE

service providers from expensive long-term investments. To improve the QoS, the cloud providers continuously expand their infrastructure by opening various datacenters across continents. The use of cloud marks a continuous CAPEX and OPEX savings to its customers and opens the possibility for profit adjustments and QoS tuning of their cloud-based services.

The decision to include cloud resources in the analysis of the streaming service of this study is based on several key factors such as: (1) favorable price offers, (2) well-proven performance, (3) flexible and elastic infrastructure. These features permit the content providers to offer reliable streaming services deployed over multiple cloud instances with the possibility to scale. Moreover the content providers can pickup different cloud instances deployed in distant geographic zones in order to fine-tune an adequate video delay.

The service is deployed in the cloud aided by P2P multicast to disseminate the video to the users, Figure 4.2. Pointing out here that a multi-tree multicast is analyzed ahead of mesh multicast, intuitively guided by the particular service description, but also to avoid delay overhead. This means that for accurate service modeling, the users are assigned to specific number of subtrees depending on the different QoS profiles. The video is divided into several substreams, each routed over different subtree. The billing model defines the user's perceived QoS and consists of three different price profiles, *gold*, *silver*, and *bronze* [Trajkovska 2014c]. The gold and most expensive packet provides the users with full video quality and low latency, jitter, and packet loss. Because proximity to the source improves the users' QoS, gold users are mainly placed on higher subtree levels. Silver and bronze packets offer proportionally lower video qualities. Therefore the silver users are assigned to fewer subtrees (18 out of 20), while the bronze users appear in only 15 subtrees at a lower tree level. Note that for simplicity, the figure shows an example of a video divided in three substreams, where gold user appears in all subtrees, silver in two and bronze in only one. Cooperative users with high forwarding capacities and stable users with fiber optic connections should appear higher in the subtrees, in order to optimize the overlay, exploit the bandwidth surplus and avoid coupled failures.

Figure 4.3 depicts the service topology in more details. The content provider transcodes the video in multiple sub-streams, m , and pushes them in the m number of trees formed among the clients. To assure low latency and jitter, the gold users are mostly placed on the higher levels of the sub-trees, as the picture shows. The silver and the bronze packets consequently, offer lower video qualities with higher latency, jitter and packet loss when compared to the gold quality. Therefore they are placed in less sub-trees, which are mainly located on a lower level. The provider has the choice to geographically adjust the video delivery depending on the location of the users. For that purpose he can rent a cloud instance in a zone closer to the position of the nodes in order to host the video delivery from that location. This decreases video delay and facilitates scalability, since once a sub-tree is saturated, the system can be scaled on-time with a minimal delay.

Such organization permits to organize the heterogeneous peers at a best matching level in the tree hierarchy according to the amount of bandwidth they are willing to share. The commercial

Figure 4.3 : Service topology showing the user placement according to the price plan

model will stimulate the peers to stay until the end of the streaming session once they have paid for the service. Moreover it intuitively decreases malicious behavior since it demotivates the users to join and pay with the only purpose to harm the service. Should a disconnection appears due to a deliberate or accidental churn, the children nodes from the elapsed brunches are reconnected to other trees in a timely manner and before they experience quality degradation. The same way free-riders who forward less bandwidth then the declared amount, can be penalized by reconnecting them at a lower tree level and cutting off the discount they receive.

The key driver of this service is the monetary incentive model that stimulates cooperative behavior and motivates the users to offer *bandwidth as a service* from their excess upload capacity. This concept potentially decreases the dynamics in the swarm, by discouraging intermittent behavior during the streaming session. It should as well eliminate the churn caused by ungraceful departures, since the assigned remuneration commits the users to stay connected till the end of the session. As a rule of thumb without it guaranteeing to fully eliminate malicious users, such a policy may discourage them from joining due to the cost of causing harm. Although out of scope for this study, a mechanism for malicious behavior mitigation should be integrated in commercial streaming services, to assure reliability and trustfulness. The fact that the video is paid at the beginning and the discount applied at the end of the streaming facilitates the provider to detect malicious users beforehand and expel them by using some of the common algorithms

4.3. DISCOUNT INCENTIVES IN P2P STREAMING SERVICE

that deal with malicious behavior. A possible technique to apply in such situation can be the PeerSorter system [He 2014a] that makes a real-time classification of the P2P traffic and detects malicious traffic from the applications. In our service, after a node leaves the service, the affected connections from the elapsed brunches can be immediately reconnected to spare connections before the users notice a degraded video quality. The multi-tree topology is very convenient to prevent a service interruption in case of churn, since one user is connected simultaneously to multiple subtrees. The free riders that forward less bandwidth then announced can be penalized with lower-level tree repositioning and discount cancelation. The same way, unstable users can be periodically moved towards the outskirts of the tree, as Ullah et al. [Ullah 2012] suggest. Next we represent the utility functions of the user and the provider in order to formally represent the described service.

4.3.2 User's utility

We want to model the benefit of the discount from both provider's and user's point of view. To that end, we refer to the work in Chu [Chu 2004] and [Shrivastava 2005]. In these P2P incentive models, the authors work with a utility function inferred from the linear taxation studies in the economy, to ensure a budget balance. After applying approximations, the model is assimilated in the distributed communication networks to represent the marginal effect of the bandwidth on the user's viewing experience. Here we extend the function and adapt it for a monetary based steaming system.

User's utility function $u_u(r_i, d_i, f_i, p_i, F_i, t_f)$ captures a singular experience of a participant in the streaming. It encapsulates the user's benefit for watching the video and cooperating, decreased by the video cost. The user's utility function serves as an indicator for the individual welfare in the streaming and therefore it should always have a positive value:

$$u_u = b_i(r_i, d_i, f_i, t_f) - c_i(p_i, r_i, t_r, F_i, t_f) > 0 \quad (4.1)$$

The *benefit* function $b_i(r_i, d_i, f_i, t_f) = \lambda (\sqrt{r_i} + d_i f_i t_f)$ models the user's overall satisfaction of the streaming service. The first member captures the benefit of the perceived video quality with a bit rate r_i . It is a monotonically increasing and concave function that expresses the diminishing benefit for the perceived video quality as the bit-rate increases. In the second member, d_i denotes the discount per user's forwarding bandwidth unit f_i during a time t_f or forwarding the video to other users in the service. Similar as in the Shrivastava's method [Shrivastava 2005], the coefficient $\lambda = 1$ is applied to both, the video quality and the discount to represent the same importance of the two in the overall user's benefit function.

The *cost* function $c_i(p_i, r_i, t_r, F_i, t_f) = \alpha (p_i r_i t_r + \sqrt{F_i} p_i (f_i, F_i) t_f)$ indicates the aggregated cost the user pays for watching a streaming content, increased by the cost of forwarding the content to

the other users. In the equation, p_i denotes the price per bandwidth unit r_i during video reception time t_r ; F_i is the user's maximum upload bandwidth and $\alpha = 0.25$ indicates the desire to forward the video. The expression $p_i(f_i, F_i) = \beta \frac{f_i}{F_i} + (1 - \beta) (\frac{f_i}{F_i})^4$ is the fractional cost to use f_i given F_i , whereas $\beta = 0.5$ shows the equal weight of the two components: the direct forwarding cost, $(\frac{f_i}{F_i})$ and the congestion cost, $(\frac{f_i}{F_i})^4$. To represent the cost comprised by the underlying network variation and the cross-ISP effect on the users' forward capacity, a study can refer to some of the well-studied congestion cost models from the literature [Chu 2004, Shrivastava 2005].

4.3.3 Provider's utility

Provider's utility function u_p models the provider's *benefit* from a user involved in the streaming service reduced by the remuneration *cost* he pays back to the potentially cooperative users:

$$u_p = b(p_i, r_i, t_r) - c(d_i, f_i, t_f) > 0 \quad (4.2)$$

Provider's *benefit* function $b(p_i, r_i, t_r) = \lambda p_i r_i t_r$ denotes the provider's per-user revenue for the offered streaming content that can be for example: a football match, a live concert, a keynote presentation, a press-conference etc. in a duration t_r . Here, the same coefficient λ is assigned as in the user's benefit in order to stress the equal revenue relevance to the provider as the discount to the users.

Provider's *cost* $c(d_i, f_i, t_f) = \alpha d_i f_i t_f$ is the monetary remuneration the provider offers to a user who shares his free upload bandwidth. The user's average forward bandwidth is denoted as f_i and t_f indicates the time a user needs to disseminate the video to his successors, where t_f is less then or equal to t_r to denote the shorter time associated to user's sudden departure. Finally d_i denotes the discount per unit of shared bandwidth. The same coefficient $\alpha = 0.25$ is used as in the user's cost function, to represent the provider's willingness to give discounts to the cooperative users.

Form these equations, we can establish general price-discount restrictions as a function of the received bandwidth r_i , the forwarded bandwidth f_i , and the price for the streaming p_i . Equation 4.1 is a necessary condition for the price-discount constraints to satisfy the user's expectations. A sufficient condition, equation 4.4, is that the discount per user doesn't exceed the initial price the user pays. As additional constraint, the provider may limit the discount per user to a maximum of 50%. In such case, the price p_i and the discount d_i should be the solution of the following in-equations:

$$\begin{cases} \frac{\alpha (p_i r_i + \sqrt{(F_i p_i)}) - \lambda \sqrt{r_i}}{\lambda f_i} < d_i < \frac{\lambda p_i r_i}{\alpha f_i} \\ d_i < \frac{0.5 \lambda p_i r_i}{\alpha f_i} \end{cases} \quad (4.3)$$

4.3. DISCOUNT INCENTIVES IN P2P STREAMING SERVICE

It may be of a crucial interest for the provider to tune the price-discount ratio under the above condition. Such incentive can bring mutual benefit to both, the provider in form of profit by the increased number of motivated users that join, and to the users by having wider quality and price choice with the option to cooperate and get a discount. Provided the users announce their excess bandwidth to contribute, the provider can part from the above utility constraints to create a targeted billing model for a specific service.

Having defined the provider's utility per user, we can establish the constraint for the aggregated provider's utility that depends on the total number of users n_i . The users can be heterogeneous according to the video quality they receive and the forwarding capacity they share. The users' aggregated utility has to be always positive:

$$U_p(n_i, p_i, r_i, t_r, d_i, f_i, t_f) = \sum_i n_i (\lambda p_i r_i t_r - \alpha d_i f_i t_f) > 0 \quad (4.4)$$

To represent the provider's utility in a mixed P2P-cloud scenario, the equation should additionally include the operating costs for the cloud infrastructure. In a cloud scenario, the video can be encoded in a streaming media server on premise or inside a cloud instance (virtual machine). If the provider decides to use the cloud, he may consider contracting extended hosting services and tools such as: load-balancers, optimized utilization of instance use across data centers (in case of Amazonⁱ), monitoring, etc. Since these services are optional they will be omitted from the provider's profit-cost analysis. To establish a general price definition for a streaming service in the cloud, we define constraints for the cost per streaming event. From here, the total provider's cost depends on the cost per cloud instance and the data transfer cost to forward the video from the publisher to the users that are connected directly to the cloud. This cost can be essentially split into: cloud *instance cost* C_i - the cost for leasing a virtual machine instance, and bandwidth *transfer cost* C_t - cost for the outgoing traffic coming from the virtual machine instance to the Internet.

Instance cost $C_i = c_i t_r m$ is the total cost the content provider pays to the cloud service provider, where c_i is the cloud instance cost per hour, t_r is the time the instance is used (for video streaming) and m is the number of cloud instances. To take advantage of the flexible cloud infrastructure, the content providers can choose among multiple types of instances to be used during the service peak hours. The cloud providers classify the compute instances mainly depending on their hardware performance into: standard - small, medium, large, extra large; micro; high-memory; high-CPU; cluster-compute; cluster-GPU, and high-I/O.

Transfer cost $C_t = c_d u_i t_r m$ is a price the content provider pays to the cloud provider, for the outgoing traffic from the cloud instance to users that are connected directly to the m instances. In the equation, t_r is the duration of the video being streamed on a cloud instance with u_i upload

ⁱ<http://aws.amazon.com>

bandwidth capacity and c_i cost, whereas c_d is the data transfer cost. Finally, the total cost function resumes to: $c(f_i, c_i, c_d) = \alpha [d_i f_i t_f + (c_i + c_d u_i) t_r] m$. Equation 4.1 is a necessary condition for the discount model to satisfy the users' expectations. As a sufficient condition, Equation 4.3 indicates that the total provider cost (operating and discount cost), should not exceed the total benefit received by the users. The cost depends on the number of cloud's direct connections. In a scenario where all users are connected to the provider in a star topology, the provider's transfer cost increases linearly with the number of users. P2P topology instead, decreases the number of direct connections by organizing users in an overlay network. Consequently, less direct connections to the cloud reduces the provider's transfer cost.

To calculate the final utility, the provider's OPEX costs for cloud resources (C_i, C_t) should be added in the cost function and subtracted from the provider's utility, which boils down the Equation 4.4 to the following: $U_p = n_i \sum_i [b - c - (C_i + C_t)] > 0$.

As shown, a discount incentive compensates the users' expenses as a reward for their cooperation. When the users announce beforehand the bandwidth they are willing to share, the provider can rely on the utility constraints to establish the most suitable price-discount trade-off. Naturally, a billing model based on remuneration policy, can be a good promotion strategy for the service among the competitors, which can additionally boost the service performance by motivating the users to cooperate. Finally using the Equations 4.3, 4.4, a content provider can create a suitable rule engine for each streaming service depending on various decision criteria. The work described above has been published in several research and journal papers [Trajkovska 2014e, Trajkovska 2013, Trajkovska 2012b].

4.4 Extending the Incentive Model to a P2P-cloud Service

Next, the model from our previously studied streaming service [Trajkovska 2014e] is further elaborated in order to shape up a general billing paradigm for a streaming service that is leveraged by each of the three topological setups: *P2P*, hybrid *P2P-cloud* and *cloud*. We furthermore increment the work published in [Trajkovska 2014c] to establish a generic billing model for the streaming service involving new directives brought by the cloud technology. To that end, a brief reflection is made of the utility functions defined in that study and they are adapted to achieve a general utility definition for similar streaming services.

4.4.1 Welfare modeling

To determine the qualitative effect of the described billing model to the overall welfare, the utility functions 4.1 and 4.4 are used to model the described service. Let's say that n_g , n_s and n_b are the number of gold, silver and bronze users in the service and $n_{h,g}$, $n_{h,s}$, $n_{h,b}$ and $n_{l,g}$, $n_{l,s}$ and $n_{l,b}$ denote the number of gold, silver and bronze - high and low capacity users respectively. The incoming bandwidth varies depending on the quality levels, so r_g , r_s and r_b represent the video bit

4.4. EXTENDING THE INCENTIVE MODEL TO A P2P-CLOUD SERVICE

rate that each price profile is entitled to. The cooperation is represented as a k percentage of the total number of users.

Users' welfare: Based on the previously defined variables, we can now represent the aggregated value for the users' utility, commonly known as *welfare*. The discounts for gold, silver, and bronze - high and low capacity users respectively, are denoted as: $d_{h,g}$, $d_{h,s}$, $d_{h,b}$ and $d_{l,g}$, $d_{l,s}$, $d_{l,b}$. F_h and F_l are the users' average low and high forwarding capacities and f_h and f_l are the actual low and high forwarding bandwidth shared with other users. Having defined this, the users' welfare can be represented as $W_u = B_u - C_u$, where B_u equals the users' total benefit:

$$B_u = \lambda k * \left[\sum_{j=g,s,b} n_j \sqrt{r_j} + f_h \sum_{j=g,s,b} n_{j,h} d_{j,h} + f_l \sum_{j=g,s,b} n_{j,l} d_{j,l} \right] \quad (4.5)$$

In the users' cost equation, p_g , p_s , p_b denote the prices for gold, silver and bronze users respectively; $p_i(h)$ and $p_i(l)$ stand for the high and low fractional costs. From here, the users' total cost is:

$$C_u = \alpha k * \left[\sum_{j=g,s,b} n_j p_j r_j + \sum_{j=g,s,b} n_{j,h} \sqrt{F_h} p_i(h) + \sum_{j=g,s,b} n_{j,l} \sqrt{F_l} p_i(l) \right] \quad (4.6)$$

Provider's profit: To calculate the content's provider profit in terms of real money, Equation 4.4 is applied. The provider's profit then equals to $P_p = R_p - C_p$, where R_p represents the *revenue* earned from the users, decreased by the total cost C_p . The revenue is k independent, because all users have to pay the video price regardless of the willingness to cooperate:

$$R_p = \sum_{j=g,s,b} n_j p_j r_j \quad (4.7)$$

The total *cost* is tightly coupled to the cooperation level and the fraction of high-low capacity users. It also depends on the prices defined by the chosen cloud operator. From here:

$$C_p = k * \left[f_h \sum_{j=g,s,b} n_{j,h} d_{j,h} + f_l \sum_{j=g,s,b} n_{j,l} d_{j,l} + (c_i + c_d u_i) m \right] \quad (4.8)$$

Note that in a P2P-cloud scenario, the provider's total cost depends mostly on the number of direct connections to the cloud. In a client-server model, where all users are connected to the source in a unicast topology, the provider's transfer cost increases linearly with the number of

users. P2P topology, instead, decreases the number of direct connections by organizing the peers in an overlay network. Consequently, for increased profit, the provider should tend to reduce the transfer cost by decreasing the direct connections between the users and the source (cloud instance).

4.4.2 Case study

In this section we perform an experimental study to depict effect of the incentive model on the system's welfare. We want to graphically represent the influence of the three factors: users' forwarding capacity (high, low), cooperation level (100%, 80%, 50%, 20%, 0%) and user fractions on the system's welfare. For this, we calculate the users' and provider's welfare, as defined in the previous section. To formally represent the streaming service and perform a fair modeling of the prices, we make some assumptions and assign statistically inferred values to some of the variables in the utility functions. To that end, we base on our previous experience with video systems on the one hand, and apply common parameters that have been used in similar research works [Chu 2004] and [Shrivastava 2005] on the other. In the latter, the authors use live event traces to set up the parameters in their experiments. We model 500 fractions of 1500 gold, silver and bronze users in total, assigning different forward and receive bandwidth capacities to the users in each group. Moreover we classify them in five levels of cooperation (from altruistic to selfish), to evaluate the effect of the different user profiles to the utility formulas of the users and the provider. Following are the rest of the parameters used in our experiment:

- **Tree depth:** $h = 3$. We choose this value in order to assure a minimum latency for the users connected at the leaf levels in the sub-trees (three hops from the video source).
- **Video bitrate:** 1.5Mbps encoded in 20 equal stripes of 76.8Kbps bitrate. In a previous research [Cervino 2011b], we used video encoded with 500Kbps. Nowadays the streaming video can reach a quality of more then 1.5 Mbps for high definition [hua Chu 2003].
- **Quality level.** Gold users receive all 20 stripes for best quality; silver - 18 stripes or 1.35Mbps and bronze - 15 stripes - 1.125Mbps.
- **Forward capacity.** Low capacity users use at most $f_l(s)$ of 500Kbps forwarding bandwidth capacity and high capacity users, $f_h(s)$, reserve a video forward rate of 1Mbps.
- **Maximum forward capacity.** Low capacity users have a maximum forwarding capacity $F_l(s)$ of 1Mbps and high capacity users, $F_h(s)$, can forward a video with a maximum rate of 2Mbps.
- **Price** is charged per amount or bandwidth received. For gold, silver and bronze packet it is: 3euros/GB, 2.5euros/GB, and 2euros/GB.

4.4. EXTENDING THE INCENTIVE MODEL TO A P2P-CLOUD SERVICE

Figure 4.4 : Provider's welfare change when 20% of the users are high capacity for 500 fractions of gold, silver and bronze users

- **Discount** is fixed regardless of the price groups, forwarding capacity or cooperation level and can reach 50% off the unit price. This means that the fully cooperative, high capacity users in all the three groups can save up to 1.5euros/GB, 1.25euros/GB and 1euros/GB.

Provided the users have sufficient available download bandwidth to receive the desired video quality, gold users will have to pay a price of 3.96 euros to watch two hours of life video. In this case, the discount they receive can reach 33% for high capacity users and 16.5% for low capacity users.

Users' welfare: We focus here on the users' welfare and see its relation with the user profiles. We generated 500% different user fractions ordered in groups of dominated gold, silver and bronze users and assigned them high and low forwarding bandwidth capacities. Overall, the users' welfare when 20% of the them have high forwarding capacity, is lower compared to the case when 80% are high capacity users. This happens because high capacity users have more bandwidth to share and therefore they increase the total bandwidth reserves in the system, which increases the overall benefit in terms of video quality. Increased cooperation moreover, increases the total discount. We also note that the welfare increases proportionally to the cooperation, where the altruistic users increase the welfare index for approximately 1600 units above its value in the case of selfish users.

Provider's welfare: Figure 4.5 captures the provider's welfare grouped by dominant subscriptions, as a function of the cooperation for high and low capacity users (shown on x axis) and depicted for a random user group. We note that when all the users are selfish, the provider's welfare is the same regardless of the fraction of high capacity users. This result is expected, since the uncooperative users don't generate any discount and therefore the welfare stays independent on the peer's heterogeneity. The provider's welfare decreases however, as the cooperation

Figure 4.5 : Provider's welfare change for one user group depending on the users' cooperation and heterogeneity

increases in all the three groups, because increased cooperation increases the discount, which on the other hand, reduces the provider's welfare. We note that the high capacity users reduce the provider's welfare more than the cooperation does (the case of 100_20 and 80_80). This difference is small and appears because high capacity users increase the bandwidth reserves which is a critical resource in a P2P system. Having enough bandwidth resources, improves the system's integrity and with this - the provider's welfare.

We can conclude from the analysis that the incentive model may be of a general benefit for both the users and the provider. A billing model based on remuneration, encourages the users for increased cooperation and attracts more users to join the system knowing they could save money if they cooperate. From a provider's point of view, visualizing the welfare change with respect to different factors is of high interest for creating a sound billing model. For the analyzed system, although the welfare is similar across all cooperation levels, the provider will be rewarded in long-term perspective. A billing model based on discount incentive, will increase the popularity of the streaming service across the users on the Internet. This will bring to the provider both, monetary and benefit in terms of infrastructure. The provider's welfare can be adjusted depending on the complexity of the system. One example is the case of large-scale service deployed in the cloud, where the provider's infrastructure cost can be reduced by increasing cooperation in the P2P. Similarly, the provider may adjust the discount, to always obtain an increasing welfare as the cooperation increases.

Discussion and conclusions: The previous analysis used a system that bases on the QoS of the perceived video, as a show case of a system that could potentially benefit from the described incentive model. The system uses a multi-tree topology to achieve scalable quality (using

4.4. *EXTENDING THE INCENTIVE MODEL TO A P2P-CLOUD SERVICE*

multiples of the video stream) and in the same time to deal better with peer churn. Since the users in the system can be located in distant geographic locations inside different ISP networks, the quality of the network may vary. In order to ensure all users receive the desired quality under the current network conditions, the P2P system would periodically evaluate the underlying network state and reconfigure the distribution tree if needed. It depends on the requirements and the constraints of specific business model of how it will adopt the described incentive scheme. In this study we included the congestion cost in the user's cost function, however the underlying network conditions are not transparent to the provider of the streaming content and therefore we did not address cost related to the network in this study.

The P2P dynamic nature increases a system's vulnerability in the presence of uncooperative and malicious users, as they impact the system integrity and the users' perceived QoS. In this section we initially observed the characteristics of P2P and the challenges it faces for increased adoption in commercial live streaming systems. We sketched the influence of the uncooperative and malicious users to the overall system topology. It was elaborated how the P2P dynamic nature increases the system's vulnerability in the presence of uncooperative users, since they impact the system integrity and the users' perceived QoS. In continuation we presented a business model with incentive based on monetary discount. We based on the utility model defined in the taxation incentive schemes and adapted it for a commercial streaming service based on multi-tree overlay. To evaluate the qualitative effect of the model, we made experimental evaluations that depicted the users' and provider's welfare in the system. The model establishes the constraints for users' satisfaction and provider's profit in a cloud-hosted P2P streaming service. We find that the discount as a potential incentive in P2P-based streaming systems can contribute to: increased shared bandwidth and better system's performance and user's QoE. Moreover when implemented as a multi-tree overlay with central authority, it increases the trust among the peers and deals better with malicious behavior and peer's churn.

Finally we made an analysis for different groups of peer's compositions for four cooperation levels—from all cooperative users to only 20% who share bandwidth and concluded that an appropriately chosen discount and price values can boost up users participation with low effect on the global provider's revenue. To summarize, we concluded that a billing model based on QoS levels for the perceived video that uses remuneration as incentive model, is feasible and can increase the system's welfare. An economic-oriented live streaming today faces a slow tendency in adopting P2P for large-scale systems. At the time of publishing this work we were among the first to present an example for a business model based on QoS which applies a monetary discount incentive. Moreover the guidelines of this model can be adopted in a commercial streaming system to boost increased participation and improve the overall's systems integrity.

4.4.3 Cloud instances

Performing scalability prediction analysis is a vital study for the robustness and the market evolution of the real-time streaming services. In this part we define a method that estimates the number of cloud instances required to host the currently connected users in the multicast. This establishes the threshold for the number of connections, which if exceeded requires from the provider to start a new video instance to make up for the elapsed tree connections. Moreover the bounds of the system topology (P2P, P2P-cloud or cloud) are set as a function of the different user profiles. Following is the equation to infer the required number of cloud instances, m :

$$\begin{aligned}
 m &= \text{total_number_direct_stripes} / \text{stripes_instance} \\
 &= \frac{\text{trees_number} * \text{direct_stripes_tree}}{\text{instance_capacity} / \text{stripe_capacity}} \\
 &= \frac{\frac{\text{total_stripes_required}}{\text{max_stripes_tree}} * \text{direct_stripes_tree}}{\text{instance_capacity} / \text{stripe_capacity}} \\
 &= \frac{\frac{n(g)s_g + n(s)s_s + n(b)s_b}{s + s^2 + s^3} * (s + (1 - \min(C, 1))(s^2 + s^3))}{\text{instance_capacity} / \text{stripe_capacity}}
 \end{aligned} \tag{4.9}$$

In these equations, s denotes the number of stripes (sub-streams) that compose the video, n_g, n_s, n_b are the number of gold, silver, and bronze users; s_g, s_s, s_b are the number of video strips assigned to each profile. The max_stripes_tree is the most optimal case when the tree is full and all the users cooperate in the streaming. The $\text{direct_stripes_tree}$ models the sum of direct and elapsed connections to be reconnected directly to a new video instance, in a case when the tree is not entirely full. The reason why all of the elapsed connections are assigned directly to a video source, rather than to the remaining empty branches, derives from the nature of the tree structure.

As previously explained, the majority of the connections (stripes) appear at a leaf level, Figure 4.6(a), and therefore there it low probability that the remaining bandwidth will suffice. In the current case with a maximum tree depth equal to three, involving 20 subtrees, nearly 95% of the connections are leafs. Thus, the majority of them lack a sufficient forwarding capacity to accept the forthcoming connections as children. Derived from the previous said, if we assume a "worst case scenario" for the provider then all the elapsed connections will be reconnected directly to a new video instance. To infer the second member in $\text{direct_stripes_tree}$, the users' cooperation status and their forward bandwidth capacities have to be known at any time. Afterwards, the number of second and third level elapsed connections is summed to the number of direct ones. To justify this tendency, C - a *coefficient of effective cooperation* is introduced:

4.4. EXTENDING THE INCENTIVE MODEL TO A P2P-CLOUD SERVICE

Figure 4.6 : Three types of system topology depending on the *Coefficient of effective cooperation*

$$C = \text{shared_capacity_tree} / \text{required_capacity_tree}$$

$$= \frac{\sum_k (n_h f_h + n_l f_l) k}{\text{max_stripes_tree} * \text{stripe_capacity}} = \begin{cases} \geq 1, & \text{P2P} \\ (0, 1), & \text{P2P\&cloud} \\ 0, & \text{cloud} \end{cases} \quad (4.10)$$

Here, n_h and n_l are the number of high and low capacity users in the service, f_h and f_l - their average forwarding bandwidth and k expresses the cooperation percentage, as previously defined. The rationale for introducing this coefficient is the need for a dynamic calculation of the number of cloud instances as the user load changes in the system. Therefore it is used in equation 4.9 to calculate the *direct_stripes_tree* depending on the current user profiles. Moreover, equation 4.10 shows that based on the aggregated bandwidth reserves, C can define the current service topology as: P2P (multicast), P2P-cloud (mixed), and cloud (unicast). The first one (P2P) is the baseline model when the service is self-sufficient to perform in P2P manner with only one cloud instance. In this case, *direct_stripes_tree* = s , Figure 4.6(a). When the coefficient of effective cooperation is in the interval between 0 and 1, then the topology is hosted in a mixed P2P and cloud overlay, Figure 4.6(b). The third case (cloud) implies to an overlay as if the entire tree was directly connected to the cloud. This means that the required number of direct stripes equals to the total stripe number of a full tree, *direct_stripes_tree* = *max_stripes_tree*. This is a case of unicast overlay (0% cooperation) where all the users connect directly to the cloud, Figure 4.6(c).

In the experimentation section that follows, we represent the relation between the different cooperation levels and the provider's cost for starting a new video instance for the elapsed connections (the ones that would be hosted on the same tree, if all users were altruistic). The instance cost will moreover shape the mapping of the different user profiles to the provider's profit.

4.4.4 Analysis and results

The objective of this section is to conduct a comprehensive analysis of the proposed methodology in a real streaming setup. It bases on varying degrees of three key factors: (1) price group (*gold*, *silver*, *bronze*), (2) users' forwarding bandwidth capacity (*high*, *low*), and (3) cooperation level (100%, 80%, 50%, 20%, 0%), with the objective to capture their effect on the provider's cost and profit. These cases are examined when the service is deployed in three different environmental setups: cloud, P2P-cloud, and P2P topology. We emphasize the operating cost savings owed to the P2P topology, in the case when the service is leveraged in the cloud. Zhao et al. [Jia Zhao 2013] applied a similar evaluation approach in their research. The authors model four types of peering relationships among the ISPs - from egoism to altruism that characterize the willingness to cooperate. This method is used to evaluate the efficiency of peering for content

4.4. EXTENDING THE INCENTIVE MODEL TO A P2P-CLOUD SERVICE

Table 4.1 : Evaluation parameters

Gold price 2.5 euro/GB	Low f.BW 500 Kbps	Total discount (Gold) 1.25 euros/GB
Silver price 2 euro/GB	High f.BW 1 Mbps	Total discount (Silver) 1 euros/GB
Bronze price 1.5 euro/GB	Average f.BW 750 Kbps	Total discount (Bronze) 0.75 euros/GB
Gold quality 20 stripes	Discount(high) 50,40,30% off/GB	Cloud instance capacity 160 Mbps
Silver quality 18 stripes	Discount(low) 30,20,10% off/GB	Cloud instance cost 0.1 euros/hour
Bronze quality 15 stripes	Video quality 1.5 Mbps	Cloud outgoing transfer cost 0.15 euros/hour

caching and sharing between interdependent ISPs within content centric networks.

Evaluation parameters: Even though the number of online streaming services shows a continuous raise, during the time of this research there was a slow tendency among the commercial providers to introduce prices differentiation based on QoS. To promote innovative multimedia services, it is fundamental for a content provider to observe and examine the users' behavior based on the streaming service history. Gathering data for scientific purposes is however difficult, due to the closed nature of the commercial services that use private streaming protocols. Furthermore, the lack of empirical data from providers like Netflix, Zattoo, and Octoshape prevent the economic-related academic studies from performing accurate analysis using realistic traffic patterns.

The experiments use the previously presented welfare equations. In order to make a sensible projection of the described service, this study makes some assumptions and assigns statistically inferred values to the variables that are shown in Table 4.1. We based on our previous experience with video services [Cervino 2011c] and chose documented parameters from real event traces [Shrivastava 2005, Chu 2004]. A combination of gold, silver and bronze users in a total number of 1500 were setup to watch a 1.5Mb/s HD video quality. The incoming and outgoing bandwidth capacities were fixed based on the Akamai's report [Aka] for the current Internet speeds. For setting reasonable video prices, Zattoo's model was used, despite that its subscription-based pricing is perpendicular to our tiered per-transfer pricing. In order to make an approximation, we inferred the initial price per GB transfer by taking the Zattoo's announced 4.5 EUR monthly subscription and the reported average use of 6 hours per month for 1.1Mbps HiQ video. This calculates an average price of 1.56 EUR/GB, which mapped to our tiered quality model, resulted in the example price setup included in Table 4.1 (Note: this study was performed and it is related to the Zattoo's numbers relevant to year 2014). The discount values are chosen according to price group and the forwarding capacity in the following way - users who pay for higher quality, can accumulate higher discounts. The same applies for cooperative users with high bandwidth capacities. Since the prices of the cloud offers at the time of performing this

Figure 4.7 : Total users' discount. *x axis*: coop_highBW (100_80, 100_60, 100_20, 80_80, etc.); *y axis*: user fractions (gold, silver and bronze).

analytics were alternating within a small range, we used the Amazon EC2 average instance and transfer cost as a reference parameter in the provider's cost equation. In addition to this, we required the information regarding the outgoing bandwidth capacity if the cloud instance in order to estimate the number of simultaneous videos that can be hosted on that instance. The instance capacities in the cloud market are based on their scope and their price range is similar among the common providers like: Amazon, RackSpace, CloudSigma, etc. For the calculus we used a reference value of a RackSpace instance capacityⁱ. Since most of the cloud providers charge on hourly base, one hour was used as time interval for the video duration. Finally, we omit the churn analysis in these evaluations with the assumption that $t_r = t_f$ i.e., all users stay in the streaming until the end of the video.

Results: This part summarizes the results from the experimental evaluation based on the discussed parameters from Table 4.1.

Discount: Figure 4.7, shows the discount change trend across all user profiles represented as percentage of the total provider's revenue. The X-axis represents the cooperation level (starting from altruistic to selfish) grouped by high capacity users. The Y-axis contains a total of 450 different user fractions starting with majority gold users. The discount is set down per GB to 50, 40, 30% and 30, 20, 10% for gold, silver and bronze - high and low capacity users respectively. As anticipated, the discount increases proportionally with the cooperation, reaching highest

ⁱ<http://www.rackspace.com>

4.4. EXTENDING THE INCENTIVE MODEL TO A P2P-CLOUD SERVICE

Figure 4.8 : Provider's revenue (EUR). x axis: coop_highBW (100_80, 100_60, 100_20, 80_80, etc.); y axis: user fractions (gold, silver and bronze).

values when all users are altruistic. The discount also depends on the forwarding bandwidth, as high capacity users increase the overall discount. For example, when all users cooperate, having only 20% high capacity users decreases the total discount to half compared to case of 80% high capacity users. Across different cooperation levels, the same effect happens when 50% of the users cooperate. From a provider's viewpoint, the total discount remains below 30% off the total revenue, which shows an example of a well-chosen discount setup. Such discount choice rewards similarly the users who pay more, with the ones that are able to share more. Pointing out that this example setup tends to demonstrate the convenience of the discount model for both, the provider and the users. Note that even in a case with maximum of 50% discount, the provider's revenue is impacted by only 20%. This example shows clearly that applying the utility constraints can bring a feasible tradeoff between the discount setup and the provider's profit. This can be chosen in a way that adjusts best to the business model of the provider and more importantly, promotes fairly the user interaction and cooperation.

Revenue: The heat map, Figure 4.8, shows a pattern of the provider's revenue. The revenue reaches higher values for the user fractions with majority gold and silver users and decreases as the fraction of bronze users becomes more dominant. The revenue is tightly coupled to the provider's billing model and does not depend on the users' forwarding capacity, nor their cooperation. Based on this representation, the content provider can: (1) dynamically estimate the revenue from the current user setup, (2) fine-tune the discount to comply with the constraints. For a business model

Figure 4.9 : Instance(C_i) and transfer(C_t) cost change depending on the users' heterogeneity

in which the current service load defines the discount bounds, making a real-time status acquisition is essential in order to enable fair discount application.

Operating cost: Here, the instance and transfer cost is analyzed when the provider leases cloud instances to leverage the streaming service, Figure 4.9 . Note that if all users are altruistic, the transfer cost is geared towards very small values and stays similar up to the case of 50% cooperative users. Above that level, the cost increment is significant, rising lineally as the cooperation decreases. The highest cost is registered in the case of a single cloud topology, where it keeps constant as a result of cooperation absence. Increased number of cooperative users reduces significantly the operating cost. It is interesting to note that above certain threshold (50% coop., 60% high bandwidth), the cost remains the same regardless of the cooperation - Figure 4.9.

Discussion: The graphs showed that increased cooperation significantly reduces the operating cost. This cost is moreover tightly coupled to the user profiles – more expensive video price results in increased cost, whereas more high capacity users increase the spare bandwidth in the service, which reduces the cost. Note that the point of sudden cost change in the cooperation-cost relation may be interesting for the provider to node in order to make a dynamic cost prediction based on the current cooperation. This threshold delimits two situations: (1) the case of a self-sufficient P2P architecture, as shown with the straight line on Figure 4.9; (2) the case of instance's cost increment as the altruistic behavior decreases. The case of (1) reflects the system on Figure 4.6(a) where there is enough bandwidth reserves in the system supported by the users, which makes the cloud a non-essential bandwidth source. The case (2) traces the maximum costs for instances in the case of selfish users, Figure 4.6(c). Even in such case, the results show that the cost spent to rent an instance, is significantly lower than the one spent for data transfer. Nonetheless, predicting the point of the cost increment is important for the content provider and may affect his profit if not properly addressed, as will be demonstrated in the later analysis. The chosen evaluation setup

4.4. EXTENDING THE INCENTIVE MODEL TO A P2P-CLOUD SERVICE

Figure 4.10 : Profit change depending on the users' cooperation for different bandwidth capacities. *x axis*: coop_highBW(100_80, 100_60, 100_20, 80_80, ..etc.); *y axis*: user fractions.

involving one hour of video and 1500 users, raises the transfer cost higher beyond the instance cost. Especially in the case of unicast, the total transfer cost notes a fast increment and in some cases it exceeds the revenue. This clearly demonstrates that centralized model increases the cloud provision demands that culminate in high transfer cost. Likewise, increased bandwidth claims saturate the instance capacity, resulting in increased instance cost due to the demands for new cloud instances, Figure 4.6(b).

This analysis showed a descriptive prognosis of the cloud infrastructure cost and demonstrated the quantitative effect of the incentive model. The provider's goal should be keeping to a minimum the bandwidth reserves required in order to maintain an optimal P2P-cloud topology, while always preserving a positive revenue. In terms of money, the instance cost doubled when the cooperation decreased from 50% to 20%. The nearly 3000 EUR operating cost for cloud based streaming service, compared to the less than 500 EUR (when only half of the users cooperate) in case of P2P-cloud, clearly states the provider's operating cost savings thanks to the P2P topology. For simplicity, the graph depicts the cost for one user fraction alone in order to capture its variation with respect to the cooperation level, rather than showing the entire cost change across all user groups. When all profiles considered, the highest cost is registered when gold and silver users compose the majority, due to increased bandwidth demands for better video quality. In that case the content provider has to extend the cloud bandwidth reserves, which impacts considerably the cost for transfer. The transfer cost decreases when the majority of the users have high forwarding capacities.

Profit: The last analysis, represents the content provider's profit using Equation 4.7 and

Figure 4.11 : Profit change depending on the users' cooperation for different bandwidth capacities. *x axis*: user fractions(from majority gold-left to majority bronze and mixed-right); *y axis*: coop_highBW

Equation 4.8. Figure 4.10, and Figure 4.11 depict from two different perspectives, the profit change as a function of the five cooperation levels across all user groups, in the three cases of the high and low capacity users. The total profit rises with the cooperation till it reaches highest values for the cases of 50% cooperation, where the profit difference becomes marginal and independent of the heterogeneity. When the cooperation exceeds beyond 50%, the profit starts to gradually decrease. The jagged lines appear due to the mixed user fractions. For this setup, selfish users bring a negative profit due to the high infrastructure cost to host a unicast streaming scenario. With regard to money, the first graph captures a profit change from 2000 to 2700 EUR for cooperation level from 50% to 100%. The profit is moreover significantly higher than the one when only 20% of the users cooperate, with a difference of nearly 1000 EUR gain, per one-hour video duration. Furthermore, there is 1115 EUR profit improvement with majority of gold fractions as opposed to majority of bronze fractions, Figure 4.2.

Observations. The registered profit drop when cooperation is above 50%, is due to the previously analyzed cost stagnation as the global cooperation increases. There is a moment when the missing bandwidth capacity in the service is fulfilled and the provider does not benefit any longer from the newly shared bandwidth, as the P2P service becomes self-sufficient. Henceforth, any further cooperation results in excessive bandwidth and new discounts starts to affect the provider's profit. In order to avoid negative profit impact, the provider may adjust the discount

4.5. CONTRIBUTIONS SUMMARY AND CONCLUSIONS

Table 4.2 : Profit improvement vs cost per hour (in euros)

Coop.	0%	20%	50%	80%	100%
Profit	[-630, 276]	[690, 1820]	[1300, 2580]	[1200, 2600]	[1100, 2100]
Cost	[1500, 2700]	[500, 1500]	[0.4, 250]	[0.4, 0.5]	[0.4, 0.5]

according to the cooperation levels, such that when bandwidth saturation is achieved ($C \geq 1$), the discount drops conforming to the current service load. Should cheaper users join the service, the provider may assign a discount limit proportional to the subscription price or the shared bandwidth, in order to attract more expensive subscriptions, while maintaining accurate and fair share in the service.

When nobody cooperates ($C = 0$), the service topology bases to a unicast model, where all the users are connected directly to the streaming instance. In this case the billing simplifies to a basic *price-per-video* model without discount. Table 4.2 represents the profit-cost intervals for different cooperation levels. Note that for one-hour video, the provider takes the most advantage with only 50% of user cooperation. In that case, the profit improvement is nine times higher than the profit during zero cooperation. Likewise, the cost savings are significant. This table highlights the achievements of the cloud and its potential to bring remarkable profit returns with minimal investment (Note the profit for 0% vs. 50-100% coop.). In a large-scale cloud a minimal contribution can result in significant profit increase, if applied to multiple services over longer period of time. The described evaluations offer a quantitative analysis of the tiered billing model and illustrate the profit-cost advantage to establish a commercial P2P service in the cloud. The values used in the evaluations are only rough estimates to provide guidance on how a content provider can adjust the service based on the current load and the user profiles. Following the profit-cost distribution, they may not only fine-tune prices and discounts, but also adjust the topology model.

4.5 Contributions Summary and Conclusions

The absence of commercial P2P services with direct user participation based on discount incentive and QoS based tiered billing model, makes the P2P-cloud service analyzed in this study outstanding among the existing offers. As a guide towards transparent pricing strategy, this proposal confronts the currently available commercial services with proprietary algorithms and flat rate billing models. The contribution of this research is twofold. First, it presented a methodology for discount incentive application in P2P-cloud based streaming services, bringing out the idea of open, tiered, QoS-based billing. Second, it offered analytical study and representation of the billing model effect applied over different user-profile setups. The experimentations evaluate the economic impact of the billing model, represented through the users' welfare and the provider's profit. The evaluation confirms that the discount incentive

significantly reduces the provider's operating cost and increases his profit. Moreover it suggests that increased users' participation and cooperation, improve their own welfare. The described methodology can be applied on a commercial streaming service and easily adapted in accordance with the service specification. The constraints can be instructive for content providers to achieve a balance among their own profit, the required cooperation and the allowed discount. In conclusion, this contribution not only offered the idea of innovative streaming service having tiered, QoS-based billing model, but it also facilitated a comprehensive approach and guidance towards dynamic price adjustment in similar billing models.

Chapter 5

SDN-based QoS Assurance for Cloud-based Streaming

Quality of Service is important aspect for the participants in a real-time multimedia streaming. Therefore the provision of end-to-end QoS takes an important role in the deployment of commercial P2P streaming applications on the Internet. This topic is tackled in first part of this chapter. We describe the proposed QoS control algorithm that is leveraged by applying the SDN technology into the mode. We explain how SDN tends to return the multicast functionality on the network layer by programmatically defining the rules and the constraints for enforcing QoS defined by user application directly into the networking stack. This algorithm is addressed for creating P2P multicast in order to leverage the previously described P2P-cloud streaming service, where the QoS levels define both, the overlay topology and the billing model. We followed the algorithm to implement a PoC for QoS control based on SDN as a part of the KIARA project - an advanced middleware for wide scope of use-cases inside the European FIWARE project [fiw]. We demonstrated the novelty of controlling a set of QoS parameters for real-time communication on the network layer, using an SDN controller with OpenFlow-enabled capabilities and APIs.

5.1 Problem Statement

At the beginning of the Internet era the initial process of implementing multicast at the IP layer, resulted in an alternative approach for supporting audio/video streaming by shifting the multicast functionality to the application layer. Since then, the users called peers, took a part of a new cooperative model for direct user-to-user communication called P2P. The P2P paradigm bases on a virtual overlay network created on the top of the physical layer and uses various algorithms to provide a seamless topology creation and different P2P services. As completely distributed and server-independent network, it attracted the researchers' interest for exploiting P2P in applications such as: streaming, audio/video conferencing, file sharing, gaming etc.

Real-time multimedia streaming as a counterpart of Video on Demand (VoD) replaced the traditional way of fetching the content beforehand, with watching it without having to download. The users are offered the possibility to watch the live events online, in just few steps, within a click of a button. Although very convenient and easy to use, P2P is based on voluntary behavior which makes it highly exposed to peer churn. Moreover P2P faces a challenge of providing continuous end-to-end quality of service due to its highly dynamic nature. To achieve this, many efforts have been done during the last decade to improve the algorithms for P2P overlay creation and streaming towards ensuring more reliable services in higher definition quality. Yet dedicated to multimedia streaming, no standardized or common QoS protocols

existed at the time this work was published. The requirements for timely delivery and delay restrictions in real-time multimedia, makes the challenge even bigger when the streaming applications are implemented over a dedicated P2P network.

The SDN paradigm enables a direct control and interaction with the network resources and decouples the controller role from the physical network. Such abstraction of the physical layer, enables the creation of OpenFlow-based controller implementations and application level services based on the novel SDN principles. The novel concept of operative system for network devices, authorizes lower level interaction with the devices and thus permits a direct top-down control of the service. This was initially enabled thanks to the OpenFlow standard that warrants a full access to the setup parameters of the switch and enables among other, to tune down the QoS-related meters.

In the first section of this chapter we will focus on the QoS-based algorithm. Leveraged by the Software Defined Networking (SDN) paradigm, our approach uses the idea of our previously studied P2P streaming system [Trajkovska 2010b] to implement a prototype application for QoS assurance. The application described in this chapter is of a curtail importance for the deployment of commercial P2P streaming applications on the Internet. As stated in the early part of this thesis, QoS provision and control is extremely important from the customers point of view. Furthermore it facilitates to the content providers, to establish billing models with granulated pricing schemes based on scalable QoS levels as one example. We focus in section 5.2 on presenting the SDN principles and the OpenFlow-based QoS approach. Section 5.2.1 describes the SDN component developed as a prototype within the KIRARA project. This component includes the the prototype application with the algorithm for QoS retrieval and control described in Section 5.2.2.

5.2 SDN Application for QoS control in OpenFlow-enabled Networks

Software Defined Networking: The concept of Software-Defined Networking [SDN a], emerged as a novel paradigm for computer network architectures as soon as the OpenFlow standard was published. The key concept of SDN is the separation of the control plane and the data plane. The control plane is represented as an appliance that takes the accountability for the communication between the business applications and the data plane. In this respect the control functionality is entitled to a central controller unit, whereas the forwarding duty is preserved in the routers and the switches. The data plane is represented by the network infrastructure that can be either physical or virtualized. For network administrators and application developers, the control plane provides abstraction of the network in both directions: high level access through a so called, northbound APIs, and lower level – infrastructure that is accessed by the control plane over a southbound API. The SDN paradigm makes no declaration for the protocol to use on the north- or southbound API but clearly defines the placement of the functionality like forwarding, control and monitoring. HTTP protocol and REST specification were used primarily on the northbound, whereas the southbound relied on applying traditional protocols like Netconf or SNMP as well as

5.2. SDN APPLICATION FOR QOS CONTROL IN OPENFLOW-ENABLED NETWORKS

the emerging OpenFlow protocol.

QoS in OpenFlow: OpenFlow [OF] is the first open standard currently managed by the Open Networking Foundation, which successfully implemented the concept of SDN. The idea in OpenFlow is to decouple the network of the control part and provide a communication interface between them allowing effective use of the network resources. In a conventional network, each switch is both configured and controlled by a proprietary software i.e., packet forwarding (the data path) and high-level routing (the control path) occur on the same device, while the logic is distributed through the entire network. In OpenFlow however, this approach is substituted with a centralized unit, called controller that takes over the global control. The controller abstracts the network control of the underlying network devices. This way any software that supports the OpenFlow protocol can be used to control the forwarding decisions, provided that the network devices are also OpenFlow configured. The routing and packet forwarding functionality is maintained in the routers that are called forwarders. OpenFlow introduces the concept of flows and rules that can be defined per set of flows. The flows can be installed by network operators or application developers in a proactive or reactive manner. Depending on the rules, the controller takes per-flow actions and updates the forwarding (flow) tables in order to steer the traffic as expected by the application.

Several universities including Stanford, Georgia Tech and Princeton and established companies including IBM, Google, Amazon, etc. were among the early adopters of the OpenFlow principles that have to date developed and maintained large SDN solutions on premise [Jain 2013]. By early 2012, Google's internal network ran entirely on OpenFlow as a first large scale SDN deployment in production [blo]. In parallel, several switch and router vendors committed to provide full OpenFlow support in their devices. The possibility of partitioning the traffic into production and research flows, engaged the research community in exploiting increasingly the OpenFlow principles and SDN for research purposes. This came as a convenience for the researchers and helped them to create isolated OpenFlow testbeds to experiment with new routing protocols, schemes and security models [McKeown 2008]. OpenFlow provided among others, to achieve a complete network visibility while facilitating full network abstraction and transparent monitoring of devices. Currently there are multiple streamlines for engaging OpenFlow in large network operators and ISPs. SDN is directly plugged as the networking component in charge of NFV architectures, and recently projects like SESAME [ses] are also using SDN controllers to manage the networking layer in the mixed telco-cloud eco-system.

Efficient QoS in computer networks has to cover both, performance assurance and service differentiation. Due to the limited network resources, an application should be able to utilize the minimum resources available in order to work correctly. Although any network device comes with a performance assurance, the advantages of an environment that follows the SDN paradigm is that the performance might be fine-tuned and adjusted by a controller in a more granular way. Those

adjustments are enabled by the OpenFlow adjustments implemented in the traditional devices that support the specification. OpenFlow 1.0 introduced the queues as a way to create network slices and provide QoS operations for bandwidth control. An SDN controller instead of adding a queue to the device, can ask the device for existing configured queues, alter them and also assign flows to enqueue packets.

The meter-bands in the OpenFlow specification 1.3 are a new way to implement bandwidth control with rate-limiters. A *rate-limiter* in OpenFlow, belongs to a meter-band and is available in two different forms: *drop rate-limiter* that drops packets that exceed a certain packets per seconds threshold and *simple DSCP policer* that remarks the drop precedence in the IP header of the packet. The information needed for service differentiation - which application needs what type of QoS - is defined by the application itself. The matching functionality in OpenFlow, implemented as *FlowModification messages*, is a way to differentiate the services either in a fine granular way - IntServ, or a class based way - DiffServ. The facilitator for effective service differentiation is the northbound API, which in the SDN paradigm is a gateway to provide access to applications and third party developers. This way they can connect to the controller and ask for a specific service differentiation. The OpenFlow controller can then setup and adjust the QoS required for the application by using rate-limiters, queues or creating an optimized network path.

5.2.1 KIARA as SDN enabled middleware

The KIARA [kia a] middleware was a project within the frames of the technology platform FI-WARE [fiw], developed under the program of Future Internet Public Private Partnership [Consortium 2012]. The goal of the KIARA project was to provide a "Middleware for efficient and QoS/Security-aware invocation of services and exchange of messages" for the FI-PPP program and beyond. KIARA based on the OMG standard [dds], for RTI-DDS Middleware (OMGcompliant DDS implementation) and extended by several new technologies: just in time complement of function stubs/skeletons, new security concepts and enhancements on the network layer. The aim was to provide an advanced middleware layer that targets the specific requirements of the Future Internet, see Figure 5.1. KIARA aspired to improve the state-of-the-art in multiple ways:

- Improve developer productivity and greatly simplifies application integration.
- Dynamically and transparently select the optimal communication mechanisms, protocols, and data representations to be used between two peers.
- Embedded compiler dynamically at run-time generates highly optimized code that transfers messages directly from application data structures to the network.
- Simple, high-level specifications of QoS/QoE and security requirements.
- KIARA uses a "Secure By Design" approach for the communication architecture.

5.2. SDN APPLICATION FOR QOS CONTROL IN OPENFLOW-ENABLED NETWORKS

To address the above challenges, the KIARA Project was internally divided in four dedicated sub-projects/activities:

1. *Development of Networking library.* Covered a concrete implementation of the network library called *ktransport* [kia c]. The KIARA networking library was written in the C language and it was based on ZeroMQ [zer], a high-performance asynchronous messaging library for distributed and concurrent applications.
2. *A modern QoS system for computer networks.* This research activity focused on assuring QoS control and the results were implemented and used in *ktransport*. The QoS library took part of the SDN Plugin shown in Figure 5.1.
3. *Negotiation between endpoints.* This task included the negotiation from a network perspective between KIARA and the desired endpoint. The other endpoint could be also using KIARA or a native endpoint like a web server or a web browser. The result of the negotiation was used to configure the communication on a network level. The general negotiation mechanism was based on trivial network protocol and was related to the QoS system, as well as to the general transport mechanisms that are offered by the operating system.
4. *Communication over Infiniband with RDMA.* This was the implementation of RDMA over Infiniband in KIARA and it was also included as part of the *ktransport* library [kia b].

QoS component in KIARA: The role of the QoS component in KIARA was to define APIs that enable end-to-end QoS provision between two endpoints using KIARA client/server and/or KIARA endpoint and a web server. This was to be performed via SDN plugin on the network layer that will be in charge to offer the following set of parameters/actions:

- QoS parameters (bandwidth, latency, jitter and packet loss)
- Well arranged P2P multicast overlay
- Optimization of the datapath through the network
- Topology Information i.e., the type of the underlaying network

The QoS parameters depend on the type of the offered communication. The goal was to offer: (1) a support for QoS parameters based on RTI-DDS, (2) third party application/user defined QoS, and (3) SDN-based QoS under direct supervision of the SDN plugin. It was analyzed that to achieve a high level of QoS provision, KIARA needs an SDN controller that apart from the OpenFlow protocol, supports other standards and technologies - SNMP (similar to the ones in the project OpenDaylight [odl b]) and makes them transparent to the consuming application. The third group of QoS parameters are the same as the ones described in Section 3.3.2 of Chapter 3:

Figure 5.1 : Diagram of the KIARA middleware

Figure 5.2 : Negotiation with QoS retrieval and control in KIARA

5.2. SDN APPLICATION FOR QoS CONTROL IN OPENFLOW-ENABLED NETWORKS

bandwidth, delay, jitter and packet loss. Those are the traditional QoS network parameters considered as relevant in real time applications ([Cervino 2011d] and [Imran 2007a] are two example studies).

Figure 5.2 depicts the negotiation process for aggregate set of QoS parameters together with the SDN component that acts as QoS controller and adjusts the parameters negotiated between the endpoints. In the part that follows, we'll describe in further details the depicted scenario. The remaining part of this Section will focus exclusively on the library and QoS prototype that was developed in KIARA as the most essential part for this thesis. The work related to this project took part during the three month research visit at the ZHAW University with the aim to enhance the current research with mutual cooperation in the domain, complement the achievements at the time, and advance the process.

5.2.2 Prototipe application for QoS control

In this work we builded an algorithm that guarantees the QoS on a network level, such that the users in a multicast application running on the top of the network, can have chosen QoS parameters injected by an SDN application. So far in the research work we relied on algorithms such as Pastry to build an overlay between the nodes. Pastry [Rowstron 2001] is a scalable, distributed, decentralized object location and routing substrate used in P2P applications such as Scribe [Castro 2002], Past [Druschel 2001], etc. to build multicast trees. Pastry is a DHT based and uses proximity metric routing to find the closest node in the Pastry set of nodes using best effort approach. We wanted to extend the idea of this algorithm and improve the routing between the nodes on a network level, for the KIARA middleware and beyond. The advantage of this approach is the ability to measure and for the first time - control the QoS, independently of the overlay algorithm and the application used on the top of the network. In common P2P practice, in order to build a multicast overlay for P2P streaming, the application had to use a dedicated algorithm for routing overlay and creation leveraged by the application layer. With the emergence of SDN this step can be omitted and the overlay can be created directly on a network layer, by directly tuning the physical connections and the dynamically adjusting the routing between the nodes.

For advanced middleware like KIARA, this brings an easier and scalable approach to maintain the negotiated QoS between the endpoints that want to communicate. The SDN-based QoS component, facilitates to KIARA the information for the available resources of the two endpoints and without having to use a dedicated algorithm, adjusts best matching communication with the desired QoS provision. For the example application used in this work, this means that the nodes who wish to participate in a P2P multicast communication, will already have the QoS adjusted, before the start of the streaming. In this case, the providers can simply create their own streaming applications on the top of KIARA, extend the KIARA's negotiation with own parameters, or simple use the API library to establish the desired parameters. This kind of

Algorithm 1 QoS adjustment and control

Require: $QoS(gold, silver, bronze)$
Ensure: $QoS(minBW(XMbps), maxLatency(Yms))$

```

1:  $nodesSet = [n1, n2, \dots, n6]$ 
2: arrival of  $newNode$ 
3:  $candidateNodesSet = []$ 
4: while  $nodes$  in  $nodesSet$  do
5:    $avgRTT = ping(newNode, nodes)$ 
6:   if  $avgRTT < maxLatency$  then
7:      $candidateNode = node(avgRTT)$ 
8:     add  $candidateNode$  to  $candidateNodesSet$ 
9:   end if
10: end while
11: while  $nodes$  in  $candidateNodesSet$  do
12:    $candidateBW = iperf(newNode, nodes)$ 
13:   if  $candidateBW < minBW$  then
14:      $candidateNode = node(candidateBW)$ 
15:      $SDNplugin\_adjustBW(newNode, candidateNode)$ 
16:      $connect(newNode, candidateNode)$ 
17:   else
18:      $connect(newNode, candidateNode)$ 
19:   end if
20: end while

```

abstraction, brings a high benefit to the providers also from an economic perspective, because it enables the introduction of novel billing modes based on more granulated QoS-based pricing schemes.

Application description: Next we describe the QoS application as a proof of concept for the SDN based QoS retrieval and control. Figure 5.3 depicts the testbed we used to evaluate the SDN based QoS application. We create a novel, network based QoS routing algorithm 1 for multicast streaming applications. We base on the SDN capabilities and take over the control of the network, to offer an improvement to the best effort routing among the network nodes. Using Mininet, we created a network of six nodes connected to a switch and from the other side, we connected the switch to a Ryu controller. Then we wanted to represent the arrival of a new node in the network, that has specific QoS requirements. These QoS refer to the end-to-end QoS parameters for the perceived video in a multimedia streaming session. For a gold user for instance a HD video encoded with $1.5Mbps$, means he wants to watch the video with maximum streaming bit-rate and full quality. The new node negotiates the following set of QoS: $(min_BW(1.5Mbps), max_latency(200ms))$. Afterwards, the application performs *ping* and *RTT* measurements between the new node and the rest of the nodes to find out the best matching candidate (or a subset of candidate nodes) that match the required QoS. Initially the program

5.2. SDN APPLICATION FOR QOS CONTROL IN OPENFLOW-ENABLED NETWORKS

Figure 5.3 : SDN testbed for QoS retrieval and control

chooses the closest node with the lowest latency (that is lower than the specified maximum) and pass the information via REST API to the Ryu controller application, see Algorithm 1. The controller application then performs adjustment of the bandwidth between the new node and the candidate node using the bandwidth rate limiters of the OpenFlow protocol specification [OFs b].

The total amount of bandwidth a network device can process is a variable size that differs even for identical hardware depending on the place in the network topology and the complexity of the traffic forwarding. The general idea for bandwidth control with the OpenFlow rate-limiters, is to slice the maximal bandwidth a device can offer into one or more rate-limiters. For the management of the associated rate-limiters as well as for the traffic that is related to it, a bandwidth control Ryu application (BWC) was developed. Every network device that is newly attached to the network rises an *OFPSwitchFeatures* event in the BWC application. The controller then adds in it's initial state a rate-limiter associated to that device. Its size equals to the total amount of bandwidth the device can offer. This size will be discovered by *OFPCongRequest* message or, if not possible, set to a default size out of the configuration file. All traffic that passes through this network device, now identified as 'common-queue', is processed by the rate-limiter. After the common-queue is set, an application can request the BWC, further queues via REST based interface. If a new queue is added, only the size of the common-queue will be shrink by the size of the new queue. If the new queue is bigger then the remaining size of the common-queue, an error message is returned and the requested new queue will be rejected.

The assignment of installed queues to traffic is made via the *table-pipelining* - introduced in the OpenFlow protocol version 1.3. Table zero in the devices is used to identify nodes by their MAC-address. All packets that match their MAC-source address, are forwarded to table one where the queue is applied to the packet. If no matching queue is found in that table, the common-queue is attached and the packet is forwarded to table two. This is the last table in the pipeline in charge to add the correct output port and forward the packet to its destination. To process further traffic matches among the existing queues, it is possible to attach more then one node to an existing queue. At the end we use a REST API call to retrieve the current QoS values and validate the correctness of the algorithm.

By creating this experimental testbed and perform simulating QoS retrieval and adjustment, we validated the idea of the QoS control protocol and the billing model idea of the P2P streaming architecture. A direct control over the network QoS parameters will enable providers by using their own streaming applications (unicast (one-to-one) or multicast (one-to-many)), make a direct call on the KIARA's QoS component and negotiate and adjust the QoS in the streaming service. The advantage of this approach is that the QoS module can be decoupled from the KAIRA middleware and plugged in to other applications with a similar scope. This can be done via simple web-based communication by using REST calls in order to pass the required parameters and receive the adjusted values. Of course, the idea of using KIARA in the end-to-end communication, is to meet the need for wider range of use-case scenarios and host heterogeneous communications over different transport layers in a most optimal manner. Regarding the QoS component, as described in Section 5.2.1, the final objective inside the FIWARE project, is to complete the entire set of QoS supported by KAIRA and extend it further with an amplified list of non-SDN related QoS.

Conclusion: In this part, we presented a prototype application for adjustment and control of aggregated QoS. We performed the implementation using an SDN-enabled testbed with an OpenFlow compatible switch and several users in the network to support the idea of P2P multicast communication on a network layer with SDN-aided control of latency and bandwidth. We tested the possibilities of the current OpenFlow protocol version and the meter bands API implementation, to verify the possibility of QoS control for the architecture of our related research. This architecture defines a billing model and user profiles based on scalable QoS for commercial P2P systems. The work described here formed part of the KIARA middleware inside the FIWARE project. With the scope to offer an advanced middleware between endpoints and improved end-to-end communication, it delivered a QoS module in order to provide a wide set of QoS parameters between endpoints in the network. The described work this section included is focused on QoS provision and assurance, as part of the multicast communication offered by the middleware.

We engaged in the progressive advance of the SDN field and applied its current tools and libraries in order to represent the P2P streaming service in alternative manner. This work was based on the current OpenFlow version at the time and it was limited to a small subset of initially

5.2. SDN APPLICATION FOR QOS CONTROL IN OPENFLOW-ENABLED NETWORKS

supported QoS parameters. Nonetheless it included the most prevalent parameters that are critical for real-time communication applications, which helped us verify the idea on a network level. With this we showed that there is a straightforward way to support real-time multimedia delivery with QoS provisioning on a network level, provided that the infrastructure is SDN-enabled. From that perspective we believed that the emergent adoption of SDN for research purposes and the increasing market for OpenFlow-enabled devices will empower the enhancement of SDN features. The advanced functionality will lead to the possibility of implementing more complex protocols in a near time frame, which will extend the SDN scope up to the application level.

5.2.3 APIs for QoS control in residential networks

Various Internet Service Providers (ISPs) today rely on their own network infrastructure to offer Internet TV services. Those services require costly dedicated servers and constant load balancing to deal with flash crowds during peak hours. Thanks to the Software Defined Networking (SDN) [OFs b] paradigm, the ISPs can implement more flexible infrastructure and take over the network control in order to offer better and optimal services to the users. SDN delegates the control of the network to a software component inside a switch, enabling transparent interaction with the network resources according to the application's preferences. This is enabled thanks to the OpenFlow protocol that defines a communication interface between the control and the network plane. Any software that supports OpenFlow, can be used to control the forwarding decisions in OpenFlow configured network devices. Following these directives, the ISPs can deploy online streaming services with critical quality of service demands over dedicated SDN infrastructure, while maintaining the rest of the services run over conventional network.

In this part we discuss SDN based QoS management for online streaming services deployed over proprietary residential networks. To demonstrate its technical feasibility, we propose APIs to adjust the users' QoS by using the OpenFlow protocol that are developed following the Algorithm 1 from the previous Section. As a validation of the idea, we tested our QoS control APIs on P2P multicast scenario created in SDN based simulation environment.

Most of the proprietary TV services charge their users a flat-rate price for unlimited real-time/on-demand videos with best-effort QoS. In the work previously described, we studied a P2P streaming service with variable prices depending on the QoS levels. Following the same idea we study here the SDN ability to support such a service in practice. The opportunity for QoS management in the OpenFlow protocol have triggered some research efforts to approach the QoS problem from a network level. As example, a QoS driven management was studied by Yiakoumis et al. [Yiakoumis 2012]. The authors argue that the user's preferences on the application should guide the network traffic from home to the ISP network. Although similar to our idea for SDN based QoS adjustment, the proposal differs as it the focuses of direct user-ISP interaction and on-demand QoS change, while we leave the QoS maintenance responsibility to the content providers/ISPs, once the service preference has been done. Similarly, the authors

Figure 5.4 : SDN based residential network for dedicated streaming services

in [Bueno 2013] propose SDN prototype architecture for future dynamic QoS provision in end-to-end applications. Next we describe the residential streaming service architecture.

Residential SDN Streaming Service: Figure 5.4, shows a use case scenario of SDN-based residential topology. Instead of using an already existing network infrastructure, the ISP deploys a separate SDN solution dedicated to online streaming service. Each neighborhood has SDN-enabled network, supervised by an OpenFlow-configured controller. If based on a client-server solution, a real-time service delivery requires very stable and scalable underlying topology that is expensive. As alternative, the P2P topology for media delivery is becoming increasingly present in real-time services, despite the high demands for constant QoS assurance.

Let's apply the same price schema described in this thesis to this service, i.e. the provider offers the streaming service with three types of prices (highest to lowest): gold, silver and bronze based on video quality. In order to provide different qualities on application level, the provider should slice the video into smaller portions (sub-streams). Then the provider forwards the sub-streams to the users in unicast or multicast fashion, adjusting the quantity of sub-streams to the desired video quality. By using the SDN paradigm, the provider can achieve the same quality effect on a network level and provide the desired QoS by adjusting the underlying network flows between the source and the users, or between two users in the multicast. Although the P2P algorithms show significant improvement lately, when it comes to multimedia streaming, there is no standardized QoS protocol for P2P multicast. We part form this gap and the SDN advantage in order to leverage

5.2. SDN APPLICATION FOR QoS CONTROL IN OPENFLOW-ENABLED NETWORKS

the P2P service from our previous work [Trajkovska 2014d] and enhance the scope of the library for QoS management in multicast streaming services.

The advantage of the SDN based QoS control, is the possibility to offer end-to-end quality for real-time video delivery that is tuned down on a network level. This approach abstracts from the complex P2P overlay algorithms on the top of the network infrastructure. Instead, it shifts the interconnection decisions to the SDN control panel, thus enabling a direct mapping between the users' virtual and physical connections. Moreover it provides a better management of the physical topology with direct QoS supervision. The drawback of this solution is its current limitation to work only in a proprietary SDN infrastructure – enclosed in a residential ISP network or a private cloud deployment. For the moment, there is undergoing initiative of the switch producers to facilitate their solutions with OpenFlow support. Yet a global coordination and control interface across multiple SDN domains is about to come.

APIs for QoS Management: We proposed QoS control APIs for residential SDN networks [Trajkovska 2014b]. As previously discussed, these APIs can be adapted by the streaming providers for QoS management in online streaming services or used by the ISPs for QoS adjustment on a network level. We describe here the APIs in the context of streaming service scenario. When a user connects to online service to watch a streaming video, he sets up the desired QoS and his willingness to participate in a P2P multicast.

```
(1) get_connected_users()
(2) get_closest_nodes_list()
(3) get_current_bw(newNode, [closestNodes])
(4) connect(newNode, hostNode)
(5) adjust_bw(newNode, closestNode)
```

Using the *Mininet 2.1* [min] emulator, we simulated a network of nodes connected to an OpenFlow 1.3 Software Switch. The switch implements a *Ryu* [ryu] controller application. Ryu is SDN framework that supports various protocols for managing network devices.

Upon this negotiation, the provider invokes API (1) to retrieve a list of the connected users. Afterwards, the candidate nodes are calculated, that are close to the new node by using the *ping* tool in Mininet for the *RTT* distance between the new node and the rest of the nodes, API (2). All the nodes that respond within a delay specified in the negotiated interval, are chosen as potential host nodes for the new node. The API (3) calculates the bandwidth between the new node and the potential nodes. The node that has a bandwidth value that fits within the requested interval, becomes a host for the new node, API (4). Otherwise with API (5), the algorithm adjusts the bandwidth between the new node and the closest node to match with the requested value, as shown on Figure 5.5.

The bandwidth control algorithm bases on the OpenFlow *Meter Bands* [OFs b]. The basic idea is to slice the maximal bandwidth a device can offer into one or more rate-limiters. The controller application detects any new device connected to the network and adds in its initial state, a rate-limiter associated to that device. This rate limiter is equal to the total amount of bandwidth that the

Figure 5.5 : Adjusting bandwidth between nodes in a multicast tree using OpenFlow

device can offer. We validated the correctness of the API calls by invoking REST calls to retrieve the current QoS state. Testing the APIs with the use case scenario, showed the SDN approach to be convenient for QoS control in P2P multicast environment. The described proposal moreover had in purpose to present a use-case solution as a stimulation of the ISPs towards increased adoption of SDN in residential networks. Such infrastructure should serve as a baseline for new streaming services. The SDN approach optimizes the use of the infrastructure resources, enabling the ISPs to host QoS-critical and elastic services.

Conclusions: In this part we presented our contribution to the idea of SDN based residential network. We showed the APIs for QoS management in a streaming service that can be implemented in a similar SDN-enabled networks. As a use case, we applied the APIs to a multicast service and proved the feasibility to offer a network level QoS adjustment for real-time streaming services.

5.3 Service Function Chaining and Prototype Implementation in NFV Scenario

The fast growing development of Network Function Virtualization (NFV) trends and the remarkable progress of Software Defined Networking (SDN) have yielded a synergy between both, towards the provision of convergent networking solutions. Since the traditional networking principles were not designed according to the NFV principles, a work has emerged dedicated to protocol redesign among the academic fellows and the industry representatives. A typical

5.3. SERVICE FUNCTION CHAINING AND PROTOTYPE IMPLEMENTATION IN NFV SCENARIO

example that has been introduced in such eco-system is the concept of network service chaining, as a composition of virtual network functions stitched together in a logically ordered fashion, to create an integral networking service that can be owned by network administrators, programmers, cloud- and telco-operators. This appoints SDN as an essential technology enabler in administering advanced traffic steering techniques, thanks to the SDN controller's capability to dynamically manage VNFs virtual connections and underlying dataplane flows. Despite the rapid progression, the strategies for traffic steering in NFV environment, are still facing imminent challenges to be addressed. The work described in this section joins NFV and SDN technology into a novel traffic steering solution based on open source reference implementations (such as SDK for SDN, virtual Traffic Classifier, virtual Media Transcoder, and WAN Infrastructure Connection Manager) designed with performance and network optimizations in mind. We have successfully deployed and tested the system prototype in a real datacenter. The evaluation results of the prototype system: (1) validated the presented chain use cases, (2) affirmed an efficient performance and scalability of the chaining method, and (3) certified good quality of video traffic transmission and transcoding. The results from this work have been published in the following contribution [Trajkovska 2017].

Introduction: As the Network Function Virtualization (NFV) trend is moving towards deployment in production environment, the latest developments have in focus the synergy between NFV and Software Defined Networking (SDN), towards more efficient network management and service delivery. NFV introduces a novel approach to the deployment and management of network functions, exploiting COTS equipment and virtualization. In particular, the main focus of NFV is to decouple the physical appliances from the network functions that operate inside them, i.e., transition from hardware based network functions (ex. middleboxes) to software modules [Han 2015, organizations 2012]. This paradigm has gained a big momentum among the Telecommunication Service Providers (TSPs) as it aims, among others, to achieve a significant reduction in Operating Expenses (OPEX) and Capital Expenses (CAPEX) [Wu 2015, Institute 2011]. Other important goal is widening their telcos' service portfolio with novel, value-added software network offerings.

In NFV context, software appliances in the form of Virtualized Network Functions (VNFs) run on standard high-volume servers, which consolidate the operation and management of various network devices onto a common infrastructure. VNFs own the ability to be dynamically initiated, reconfigured and even reallocated to different network locations, without the requirement of installing new hardware. NFV aims to deliver more open and extensible network environment, in which the deployment of new network services becomes easier to execute.

In order to deliver the transition from physical NFs to VNFs, various fundamental developments have to be performed, many of which are still under implementation and continuous evolution stage. The deployment of NFV and its adoption into a Telco grade system, subsumes various open issues as laid out in the Network Functions Virtualisation (NFV)

Architectural Framework by ETSI [NFV 2014]. One of the key elements in this transition is the management layer, which must be capable of providing support for novel features in the NFV-enabled systems. NFV will significantly challenge the current network management systems, and will introduce additional levels of complexity over the currently deployed systems.

Within the ambit of NFV appears the notion of Network Service (NS) that hooks the VNFs into a connectivity service. The path sequence that the service traffic is expected to traverse in the network and across the VNFs is described by the VNF Forwarding Graph (VNFFG) [ets]. In this context the challenge that this work addresses, is the implementation of a mechanism in order to efficiently steer the traffic through a set of VNFs, enforcing the requested VNFFG, hence creating the Service Forwarding Chain (SFC). The traffic steering mechanism is based on SDN, more specifically it relates to the employment of OpenFlow (OF) based switches that are programmatically configured by an SDN application using OF based rules.

The converging trends between NFV and SDN are launching the design of tools (for network developers and application end-users) and advanced use-cases, required to meet the expectations of the new software paradigm while keeping the old Telco's legacy. In order to address this challenge and offer a ready-made prototype tested in a real scenario, we created a toolkit library for service chain support and a PoC deployment to backup the solution. This work relies on open source components and tools developed collaboratively across several premises, in order to deliver an integral problem solution and enhanced implementation prototype.

The main contribution is the design of novel traffic steering mechanism and its implementation and testing on a real-life NFV-cloud eco-system over physical, SDN-enabled hardware. The solution uses L2-based method, and it was designed omitting the use of overlay traffic encapsulation. Targeting simplicity and robustness, the following objectives were initially outlined: ensure tenant isolation, minimize tunneling overheads, maximize throughputs, and enable datapath redundancies. OpenStack was chosen for the cloud infrastructure, due to its maturity and extensive support by the open source cloud community [ope g]. OpenStack is also promoted by Open Platform for NFV Project (OPNFV) [opn a] as a reference NFVI tool, used specifically for the deployment of the Virtual Infrastructure Manager (VIM).

The PoC described in this section, covers a total of four components listed as open source reference implementations, which represent the fundamental building blocks of the entire SFC solution: (1) *Netfloc* as SDN-driven SDK for datacenter network programming, developed by the ZHAW ICCLab and based upon the ODL controller. Netfloc's main objective are: (a) optimizing cloud datacenter networks and providing end-to-end management of OpenFlow enabled switches, and (b) supporting the network application development in the cloud. In the scenario, Netfloc's traffic steering library was used to form an SDN-modeled virtual network graph for the SFC service; (2) *Virtual Traffic Classifier* and (3) *Virtual Media Transcoder* were introduced with the main objective to provide enhanced VNF references in the system prototype implementation; and (4) *WICM* was used to provide traffic redirection as a component for advanced traffic

5.3. SERVICE FUNCTION CHAINING AND PROTOTYPE IMPLEMENTATION IN NFV SCENARIO

Figure 5.6 : Components of T-Nova and their logical relations

management in a multi-datacenter (DC) scenario. In the continuation, each of these components will be elaborated individually.

We further designed several coexisting service chain use cases, which served to successfully test the aforementioned traffic steering implementation. The measurement results of this prototype, validate the presented SFC mechanism and confirm effective and correct traffic steering. In detailed scalability evaluations, a QoS performance comparison is presented between a single chain service and several instances of the same, tackling as well, quality analysis of video traffic transmission and transcoding.

T-Nova: The service chaining prototype of this contribution, has been implemented in the framework of the European T-Nova project ⁱ. Figure 5.6 shows a high-level view of the overall project's architecture and building blocks. Fundamentally, T-Nova introduces a novel platform that implements a Network Functions as-a-Service (NFaaS) concept, which allows the operators to not only deploy VNFs for their own needs, but to also offer them to customers as value-added services. Virtual network appliances (gateways, proxies, firewalls, transcoders, analyzers etc.) are provided on-demand as-a-Service, eliminating the need to acquire, install and maintain specialized hardware at customers' premises. A key aspect in T-Nova, and broader in NFV context, is the support of service chaining as an ability of the SDN Control plane to configure network flows, in order to steer packets through a sequence of network services (physical or virtual).

The part that follows - Section 5.3.1, describes in details the proposed architecture and the VNFs under analysis are included in Section 5.3.2. Section 5.3.3 describes the SFC scenarios along with the specific implementation details of the SFC OpenFlow-based methodology. The

ⁱ<http://www.t-nova.eu>

SFC scenarios are executed and evaluated in Section 5.3.4. At the end we discuss the concluding remarks and trace the future steps of this work.

5.3.1 Netfloc: SDK for SDN-based solution in datacenters and NFV

Netfloc is an open source SDK for SDN [net c] developed as a Karaf feature, composed of multiple OpenDaylight bundles. It provides a set of libraries aimed to facilitate datacenter network management, and to support the development of network applications in the cloud. The SFC scenario described in this part, is a prototype implementation based on the Netfloc's SFC enabling library.

The Netfloc's libraries are derived from real application use cases, geared towards ensuring tenant isolation, optimizing throughputs, support of data path redundancy, monitoring and QoS. We studied the current gaps in datacenter network management, in order to engage SDN for creating new solutions and resolving common networking issues. Based on the analysis, several use case applications were designed and developed:

Isolation Application: Ensures tenant segregation using lightweight non-GRE/VxLAN tunnelling mechanism for optimized packet header.

Resilience Application: Provides SDN control on layer 2, providing a redundant data path abstractions to enable ad-hoc data link recovery and virtual switch reconfiguration.

Service Function Chaining: Traffic steering support for NFV.

On startup, Netfloc registers itself to ODL via OSGI, while rendering the following ODL feature dependencies: neutron-networking on the northbound, ovsdb and openflow plugin on the southbound. Design primitives in Netfloc, such as Chain Path Patterns and Flow Path Patterns [net d] are wrapped as northbound APIs generated using YANG models. The data received by these APIs that relates to the network and the chain, is stored in the ODL SAL datastore inventory.

What makes Netfloc essential to this scenario, is the optimized network abstraction model that supports the delivery of OpenStack based SDN applications. The motivation behind this approach was inspired by the OpenStack native support for multi-tenancy, that we found very complex and cumbersome to implement, especially due to the use of inherent protocol encapsulation. The header encapsulation approach in a traditional networking, hinders the modern needs of a current datacenter networks where throughput efficiency and service scalability comes first. Netfloc overcomes these problems by offering alternative end-to-end isolation strategy based on pure OpenFlow L2 approach. Netfloc's [net c] implementation details are out of scope in this work, nonetheless some primitives specific to the described chain scenario, will be elaborated in the Implementation Section.

Netfloc North and Southbound APIs: Figure 5.7, depicts Netfloc's internal components and the external interfaces with OpenDaylight, OpenStack and OpenvSwitch. It also shows on the northbound, the Karaf and REST APIs it exposes to third party developers. At the management

5.3. SERVICE FUNCTION CHAINING AND PROTOTYPE IMPLEMENTATION IN NFV SCENARIO

Figure 5.7 : Netfloc's internal components and South & Northbound APIs to T-Nova and third party applications (an early version of this figure appeared in Deliverable D4.31 of the T-Nova project [del a]).

layer, Netfloc offers a plugin for OpenStack Heat [net e] in order to permit resource usage of its service function chaining (SFC) library. Similar functionality is also provided through orchestrator components, such as the T-Nova TENOR Orchestrator [ten] based on the ETSI MANO NSD specification [ets] and the ICCLab Hurtle [hur], in order to enable chain service specification and its automated provisioning via Netfloc. Creating a service chain in Netfloc, consists of invoking the *Create chain* REST API, with a dataset containing a service description as an ordered sequence of VNFs' connection points. This information is passed in a JSON data format as a list of the corresponding ingress and egress Port ID tuples.

5.3.2 Main architecture building blocks

In this section the main components of the service chain eco-system are described. At the beginning, the virtual Traffic Classifier and virtual Media Transcoder VNFs are presented, describing their architecture, functionality and role in the proposed system, followed by description of the WAN Infrastructure Connection Manager.

Virtual traffic classifier – vTC: The tasks of the vTC include: analyzing a real-time network traffic, recognizing specific applications, and prioritizing traffic flows according to the service description policy. Since such operation generates a significant workload, it is highly recommended to employ specific hardware acceleration features for efficient traffic processing. The vTC was developed within T-Nova frames and is currently being used as part of the OPNFV Yardstick [OPNFV] project, in the domain of NFV workload characterization. Next, we describe briefly the techniques employed to achieve the vTC's functionality.

Figure 5.8 : Overview of the Virtual Traffic Classifier (Figure 3-1 of T-Nova Deliverable 4.32 [del b]).

The Linux network stack, on which the vTC has been developed and that is commonly used as a basis for cloud networking solutions, sets as a primer goal, the provision of a general purpose network stack for a fully functional operating system. Therefore a standard Linux network stack cannot scale to the performance level requested from a software network appliance. The proposed Traffic Classification solution is based upon a Deep Packet Inspection (DPI) approach, which is used to analyze a small number of initial packets from a traffic flow, in order to identify the traffic type. After the flow identification step, no further packets are inspected. The Traffic Classifier follows the Packet Based per Flow State (PBFS) in order to track the respective flows. This method uses a table to track each session based on the 5-tuples (source address, destination address, source port, destination port, and transport protocol) that is maintained for each flow. The vTC's architectural overview is shown in Figure 5.8.

The vTC employs various technologies in order to offer a stable and highly performing VNF, compliant to the strict standards of legacy physical network functions. The implementation of traffic inspection functionality used for these experiments, is based upon the open source nDPI library [ndp]. The packet capturing mechanism is implemented using various technologies in order to investigate the trade-off between performance and modularity. The packet forwarding technology is based upon PF_RING [ntop]. PF_RING is a set of library drivers and kernel modules that enable high-throughput, packet capture, and sampling. The vTC uses the PF_RING kernel module for polling the packets through the LINUX NAPI. The packets are copied from the kernel to the PFRING buffer; afterwards they are analyzed using the nDPI library.

Virtual media transcoder – vMT: In the SFC architecture, the video service function is used in order to enhance the value-added capabilities of the PoC. A video server is included that hosts

5.3. SERVICE FUNCTION CHAINING AND PROTOTYPE IMPLEMENTATION IN NFV SCENARIO

Figure 5.9 : Overview of the Virtual Media Transcoder (Figure 3-2 of T-Nova Deliverable 4.32 [del b]).

the videos in its original encoded form. To maintain an acceptable video service delivery, our testbed was equipped with a video adaptation system, which is capable of performing a real-time transcoding. This resulted in a video stream adapting to the current networking conditions, which caused a degraded video quality as a tradeoff (due to infrastructure limitations). The proposed virtual Media Transcoder (vMT) is shown in Figure 5.9. This VNF is built on top of the widely used *FFmpeg* library for multimedia data manipulation [ffm], and it was implemented to leverage the prototype scenario. The vMT is instantiated as a VNF at the egress point of the SFC-PoP (being the testing environment for the chain scenario), shown in Figure 5.15 and described further in the text. The triggering event for adaptation varies across use cases, depending either on the user terminal specifications, or on the available bandwidth of the delivery channel. For these evaluations, the triggering event was controlled manually, in order to perform the experimental validation of the proposed NFV-based transcoder. For this case, the control was defined as a function of the congestion bottleneck that is artificially created at the access link of the end-user.

WAN Infrastructure Connection Manager – WICM: The emergence of multi-tenancy and VNF offerings collocated on different servers of a modern DC premises, raises the need for a component responsible to control cross-PoP traffic emerging from different clients. The idea behind is to provide multiple isolated connectivity services to clients, depending on the assigned client ID, and the purchased NS (and VNFs). Achieving this, includes the process of switching traffic direction depending on, for ex. VLAN tags, to different coexisting physical PoPs inside the DC. Therefore, a central control unit would enable full visibility and traffic management across several PoP deployments together. A component developed upon this

premises, is the WAN Infrastructure Connection Manager (WICM), Figure 5.15. The role of WICMs among the others, is to offer a seamless and transparent *VNFaaS* to both, the clients' and the provider's edge routers, while having minimum impact on the network topology.

For this scenario, the WICM was used to tag User1's packets with VLAN numbers, steer the traffic into the SFC-PoP, and forward the resulting packets to their destination, User2. Bi-directional traffic communication was enabled (one tag for each direction). WICM [wic] controls the OpenFlow Pica8 switch via integrated ODL application, that connects one or multiple clients' edge routers to one or multiple provider's edge routers. By default, WICM configures the switch to connect the CE router to the PE router by closing the MPLS IP VPN attachment circuit. This process ensures a connectivity to the clients, but not yet a VNF access. Once a client purchases a network service, the WICM reconfigures the switch in order to redirect the client's traffic into the assigned PoP. WICM also configures the switch to properly forward back the resulting packets coming from the PoP. If at any time the client decides to stop using the VNFs services, the WICM configures the switch back to the default state, ensuring always a connectivity service. The switch is strategically placed between the edges, so it can intercept and redirect all the traffic coming in, or going out of the endpoints (the clients).

5.3.3 Netfloc's SFC methodology and test scenarios

With the aim to provide a simple working SFC solution in a joint NFI-cloud environment, we have completed several phases of algorithm development and testing. This was a fundamental process in order to come up with the most suitable methodology, now implemented in Netfloc and leveraged in the proof of concept prototype. This section iterates over the early SDN adoption and SFC tests in Netfloc, and elaborates on how we derived the currently implemented solution.

Netfloc validation test cases and scenarios

To explore possible flow patterns and technical challenges w.r.t. the implementation of the service chain algorithm, we created initial test cases using VNFs released by T-Nova. Their description is enclosed in the following sections:

Test Case 1 – OpenStack native networking workarounds: For this test case, VNFs and VMs emulating end-points (i.e. Alice and Bob) were deployed in a single OpenStack Compute Node. Figure 5.10 depicts simplistic all-in-one scenario, including VMs and virtual bridges and interfaces. The Alice VM connects its eth0 interface to a tap device (tapxxxx) on the Linux bridge. On the other endpoint, the interface (qvbxxxx) connects the (qvovxxxx) interface to the (bt-int) integration bridge. This complex establishment in OpenStack is required by design, with the purpose to isolate the VM-to-VM traffic, by applying security group rules to the packets.

In this test case, a chain was emulated from Bob to Alice via the VNF VM. To generate traffic from Bob's source to Alice's destination address, tools such as *iperf* and *ping* were used. We set up flows on the OVS *br-int* in order to steer the traffic through the VNF, based on MAC address

5.3. SERVICE FUNCTION CHAINING AND PROTOTYPE IMPLEMENTATION IN NFV SCENARIO

Figure 5.10 : Simple traffic steering scenario in OpenStack all-in-one setup. Shows OVS and Linux interfaces on the Control node, OpenFlow based port matching, and header rewriting example.

matching in each of the nodes. We used *tcpdump* in order to identify the mapping between the OVS ports and the physical/virtual ports. As the Figure 5.10 shows, the traffic has to cross a minimum of 16 interfaces in the case of a simple chain containing one VNF only. Two OF rules are required on the integration bridge in order to steer the traffic through the VNF. The figure also includes on the right, a MAC rewriting example that Netfloc installs on the OVS integration bridge.

In this initial deployment, we run into a common problem in OpenStack Nova, discussed also by Callegati et al. [Callegati 2015] – the default application of anti-spoofing rules to the Linux bridge. On VM instantiation, nova service applies its own, or the neutron’s default security group rules that prevents a VM to forward packets originated from a different MAC/IP address. This burden falsified the previous flow setup, and the ping was interrupted due to packet drops, because the packets did not carry the original source MAC/IP address. To bypass the security groups towards achieving service chaining, the *–no-security-groups* flag was installed (OpenStack Kilo release). The drawback was however, that it worked only once after the initial setup. If the ping is restarted, the rules are automatically re-applied, therefore the iptables have to be manipulated manually to allow the traffic flow within the 2 chains (in/out), for each of the tap interfaces. As a workaround, we applied to the OpenStack nova-compute and neutron, the *Noop* driver that prevents the creation of iptables rules [ope a]. This technique stopped nova from using the hybrid VIF plugging strategy, where it places a Linux Bridge in-line between the instance’s tap interfaces and the integration bridge, br-int. Instead, it plugged the VIF (tap interfaces) directly into br-int, whereas the rest of the interfaces disappeared (the orange boxes on

Figure 5.10). This patch simplified the architecture, decreasing by half the number of existing interfaces.

Discussion: This preliminary test case, proved the feasibility of applying flow programming approach in OpenStack and solved some issues and challenges related to the environment. The flow programming approach applies actions on the OF ports of the switches, in order to enforce the desired route of the packets in the chain. This proves that implementing SFC is unbound to header encapsulation, as a mean to achieve traffic classification and steering. As mentioned before, this forwarding mechanism is simplified compared to the one that uses a dedicated NSH header, as it avoids cumulative header encapsulation (ex. the case of inter-tenant communication in OpenStack [ope h]), leaving the VNFs completely agnostic of the chain overlay. A downside of the previous approach is that it may not fully work for VNFs above L2, for ex. VNFs that alter the mac addresses. In this case, the flow rules need to be maintained to reflect these alterations and preserve the sanity of the service chain in place.

Test Case 2 – Escalating to multiple Compute Nodes: In this test case, we repeated the procedure from above by manually installing flows on the intermediate OVS bridges. The difference in the scenario was that we applied more than one VNF and more than one physical host, both involved in two separate chains. For this matter, we used the nDPI traffic classification functionality offered by the vTC. The flow programming for this SFC example is tightly coupled to the output traffic of the vTC, because it manipulates packets based on the traffic type, and determines the chain matching pattern using a distinctive property. The distinctive property for this case was the ToS field that determined the traffic type (UDP/ICMP or TCP) and enforced steering through one VNF or another, branching from the vTC egress ports onwards.

Discussion: This approach faces an obstacle when in a scenario where two VNFs involved in the chain reside on the same physical host, and there are more than two hops involved in the chain. The reason for this is a nondeterministic path in the chain, because the OVS of the physical host, does not know the hop order i.e., to which of the two VNFs should the flow be redirected next. This problem can be formally modeled as: $\text{hops to VNF A} + 1 < \text{hops to VNF B}$ (where VNF A and VNF B are the VNFs residing on the same VM). To further explain this, the core of this problem lays in the fact that a bridge/switch does not have the context of a whole connection path along a chain. Therefore a minimal information has to be encoded in the packet header. With other words, a tuple of two values (a, b) where "a" maps to topological switching instructions (like in source routing), or to a unique chain identification, and "b" describes uniquely the next hop.

A similar issue appears when employing the described VNF: If the ToS field is the only distinctive property for the forwarding decisions, then some paths cannot be deterministically established. Same as the above explanation, this relates to a case when the chain path has to cross an OVS bridge (from the switch) multiple times. In such situation, it is not certain which is the next VNF in the order of the chain when the packet comes to the bridge. Taking for example Figure 5.3.3 (that will be discussed in depth in the following section): if the chain was defined in

5.3. SERVICE FUNCTION CHAINING AND PROTOTYPE IMPLEMENTATION IN NFV SCENARIO

Figure 5.11 : NFVI-PoP Service chain details. The yellow line is common for both chains, the red line: Chain1, the blue line: Chain3. Chain2 is the combination of both, Chain1 and Chain3 (Figure 3-5 of T-Nova Deliverable 4.32 [del b]).

the following order $SF \rightarrow vTC \rightarrow vMT \rightarrow vTC-f$, then the path is indeterministic on the eth2 and the eth0 interfaces, because it is uncertain if the traffic that reaches br-int, comes from the SF or the vMT (arriving on em2 via eth2). Same applies for the the traffic reaching the bridge via the eth0. Therefore an additional field is mandatory to denote the current hop in the chain. As will be explained further, Netfloc resolves this ambiguity by assigning to the packets coming from the vSF, a virtual MAC address $\langle SFC\ ID \rangle:03:00:00:00:00$ and a MAC address $\langle SFC\ ID \rangle:01:00:00:00:00:00$ to the packets coming from the vMT. In such case, the OVS can easily rewrite the two mac addresses by successfully determining the next hop in the chain.

As a conclusion of this test case, higher level chaining abstractions and programming languages are needed, in order to enable the service developers to programmatically declare the sequence of the VNF traffic, leaving up to the underlying runtime system the actual implementation of the rules [Anwer 2015, Riggio 2016]. For the chain routing to be deterministic, there has to be a field in the header that tracks the chain hops. Using the *VLAN ID* for that matter could be a potential solution, since the chain routing does not follow the standard Ethernet routing. The optimal SFC solution should be an environment agnostic and must eventually work for any SFC scenario, regardless of the deployment scope.

Netfloc's SFC mechanism

By addressing the obstacles from the discussed test-cases, and following the guidelines for best-matching approach to enforce traffic steering, the SFC library based on OF was created in Netfloc. We present further on the details of the approach.

Figure 5.12 : State Machine Diagram of the SFC mechanism in Netfloc

Network and Datagram State: To begin with, the State Machine Diagram on Figure 5.12 represents the states that the datagrams follow through the service chain networks. It defines the minimal amount of information that needs to be read for each state transition. The "/" is a delimiter for the input condition and output transition from one state of the chain to the other, i.e. <read>/<write> transition as denominated in the legend. For example "port:1/sfc:1 vnf:1" denotes port:1 as a condition - packets that are coming on port 1 will be sent to vnf:1 form Chain1, i.e. sfc:1 vnf:1 being the transition. Similarly "wicm:1 sfc:2 vnf:2/wicm: 2" denotes the transition from state vnf:2 from Chain 2, to WICM.

Each data point is mapped in the ethernet packet header and modified along the way, such that the bridges can determine the forwarding. The MAC addresses of the datagrams are rewritten in the SFC PoP so they can be matched for service chaining purposes in a reactive manner. Packets entering a service chain invoke newly created flows in the first and the last bridges of that service chain, which rewrite the original MAC address to a virtual address. The SFC ID and the VNF ID are encoded in the destination MAC Address. The VNF ID is decreased after each VNF's visit.

- *Traffic Classification:* The SFC implementation in Netfloc is based on the following mac

5.3. SERVICE FUNCTION CHAINING AND PROTOTYPE IMPLEMENTATION IN NFV SCENARIO

rewriting approach: two fields are reserved in the MAC address: one for the chain ID, and the second, as a hop counter for the port traversal in the chain. Upon create chain API call in Netfloc, the destination MAC address in the header of the packet is rewritten to the following format: <SFC ID>:<VNF ID>:00:00:00:00. The original mac address is saved in Netfloc, and rewritten back to the packets in the final hop iteration (once <VNF ID> reaches zero).

Here, SFC ID denotes the Service Chain ID and VNF ID - the order of the VNF in the chain. The resulting virtual MAC address is matched along the chain path to forward the classified traffic to the *Nth* VNF in the chain:

Dest. MAC = ID:N:00:00:00:00/00:00:FF:FF:FF:FF

- *Hop Rewrite*: For each VNF, there is a rewrite flow that matches the egress port of the VNF to its virtual destination MAC address and decreases by 1 the VNF ID number. This way the forwarding is always clearly determined. In a similar fashion, forwarding to the next VNF is done using the following matching, down to N reaching 0:

Dest. MAC = ID:N-1:00:00:00:00/00:00:FF:FF:FF:FF

- *Endpoint Rewrite*: The virtual MAC addresses are rewritten to their original values, at the last bridge of the chain. This is done via the following match:

Dest. MAC = ID:00:00:00:00:00:00/00:00:FF:FF:FF:FF

Dest. IP = Destination host IP address

The Netfloc's SFC application keeps the MAC-to-IP mapping. This notion for the src/dst addresses comes from neutron. Netfloc can recover the original destination MAC address through the neutron API call. There is also a mechanism included for reactive source and destination IP learning. Afterwards the packet gets resubmitted and matches on a standard L2 forwarding. An example of SFC flow enforced by Netfloc is shown in Figure 5.3.3. The figure shows the process of Netfloc's service chain instantiation via the logs gathered during a create chain API call. Moreover it highlights the described MAC address rewriting, on the example packets shown in the terminals of each OpenStack node.

Discussion concerning traffic encryption: The above described solution concerning hop and endpoint rewrite is applied to the header of the packets without affecting the payload, whereas traffic encryption is applied on the payload of the packets and not before they reach L5 (for ex. IPSEC). Therefore the packet forwarding and rewriting is transparent and does not affect the payload. Netfloc's chaining method extracts the destination mac of the packet (which cannot be encrypted), and applies rewiring rules before putting back the original mac address, at the ultimate chain hop. Therefore, even if the traffic is encrypted, this does not affect the chaining logic, since the destination mac addresses are preserved by the SDN controller. The problem of

```

SHOT | new service chain reactive flow programmed
2016-10-07 14:45:10,341 | INFO | 124617488-668673 | NetFlocServiceImpl | 269 - ch.icclab.netfloc.impl - 1.0.0.SNAP
SHOT | NetFlocServiceImpl chainNetworkPorts: [86b86912-4813-4651-94c1-d111d986ed30, f2423112-6ff1-4490-9eef-b24a7444a2c9, 90c80fa
2-fbfb-46ba-a6cb-f49429f57de6, f948b219-4568-49ad-8cfc-4af5a5062b80]
2016-10-07 14:45:10,341 | INFO | 124617488-668673 | ServiceChain | 269 - ch.icclab.netfloc.impl - 1.0.0.SNAP
SHOT | ServiceChain chainPath: [-1601]->(00:00:d4:ae:52:c9:c2:77)->[1597]=, =-[1598]->(00:00:d4:ae:52:c9:c2:77)->[1]->(00:00
:00:0d:88:53:37:c4)->[2]=[1]->(00:00:d4:ae:52:c9:f3:55)->[214]=]
2016-10-07 14:45:10,345 | INFO | 124617488-668673 | NetFlocServiceImpl | 269 - ch.icclab.netfloc.impl - 1.0.0.SNAP
SHOT | NetFlocServiceImpl Created chain: Chain_6: [86b86912-4813-4651-94c1-d111d986ed30, f2423112-6ff1-4490-9eef-b24a7444a2c9, 90
c80fa2-fbfb-46ba-a6cb-f49429f57de6, f948b219-4568-49ad-8cfc-4af5a5062b80]
2016-10-07 14:45:10,345 | INFO | CommitFutures-3 | NetFlocServiceImpl | 269 - ch.icclab.netfloc.impl - 1.0.0.SNAP
SHOT | New service chain stored in OPERATIONAL data store

root@network:/home/localadmin

root@network:/home/localadmin# ovs-ofctl dump-flows br-int | grep dl_vlan
cookie=0x0, duration=237867.558s, table=0, n_packets=1194257, n_bytes=782780064, idle_age=65534, hard_age=65534, priority=14
,in_port=43,dl_vlan=400 actions=output:13
cookie=0x0, duration=237884.470s, table=0, n_packets=132906, n_bytes=86739107, idle_age=65534, hard_age=65534, priority=14,i
n_port=13,dl_vlan=400 actions=output:43
cookie=0x0, duration=237880.606s, table=0, n_packets=1254110, n_bytes=1025359281, idle_age=65534, hard_age=65534, priority=1
4,in_port=13,dl_vlan=401 actions=output:43
cookie=0x0, duration=237872.342s, table=0, n_packets=103471, n_bytes=72407001, idle_age=65534, hard_age=65534, priority=14,i
n_port=43,dl_vlan=401 actions=output:13

root@nfvipop2-switch:/home/localadmin

root@nfvipop2-switch:/home/localadmin# sudo ovs-ofctl dump-flows br0 | grep priority=20
cookie=0x0, duration=227878.549s, table=0, n_packets=709938, n_bytes=399566316, idle_age=135, hard_age=65534,
priority=20,in_port=1,dl_dst=01:00:00:00:00:00/ff:ff:00:00:00:00 actions=output:2
cookie=0x0, duration=227836.999s, table=0, n_packets=451504, n_bytes=565686632, idle_age=65534, hard_age=655
34, priority=20,in_port=1,dl_dst=02:00:00:00:00:00/ff:ff:00:00:00:00 actions=output:2

root@control:~

root@control:~# ovs-ofctl dump-flows br-int | grep priority=20
cookie=0x0, duration=228023.159s, table=0, n_packets=709932, n_bytes=396739296, idle_age=285, hard_age=65534, p
riority=20,in_port=1,dl_dst=01:00:00:00:00:00/ff:ff:00:00:00:00 actions=output:431
cookie=0x0, duration=227981.609s, table=0, n_packets=3472, n_bytes=4275264, idle_age=65534, hard_age=65534, pri
ority=20,in_port=1,dl_dst=02:00:00:00:00:00/ff:ff:00:00:00:00 actions=output:434
cookie=0x0, duration=224164.133s, table=0, n_packets=0, n_bytes=0, idle_age=65534, hard_age=65534, priority=20,
in_port=435,dl_src=02:82:00:00:00:00/ff:ff:00:00:00:00,dl_dst=02:00:00:00:00:00/ff:ff:00:00:00:00 actions=mod_dl
_src:b8:ae:ed:77:73:bc,mod_dl_dst:00:90:27:22:d2:68,output:428

```

Figure 5.13 : Flow entries programmed by Netfloc on the virtual switches of the OpenStack nodes (Control and Neutron) and the SDN switch. The uppermost terminal window shows the SFC process in Netfloc after the create chain API call. The highlights in yellow underline: virtual mac address matching on the switch and the control node (3rd, 4th upper window) and VLAN matching on the neutron node (2nd upper window).

5.3. SERVICE FUNCTION CHAINING AND PROTOTYPE IMPLEMENTATION IN NFV SCENARIO

Figure 5.14 : NFVI-PoP physical overview (Figure 3-4 of T-Nova Deliverable 4.32 [del b] and Figure 9 of T-Nova Deliverable 7.1 [del c]).

SFC chaining has been discussed in the IETF SFC WG Draft [Nadeau 2015]. Regarding the existence of encrypted traffic, Chapter 5 of this document assumes that the embedded VNFs in the traffic path are able to decrypt the traffic and process the flows, if this is expected by the service owner. This can be applied only for end-to-end VPNs and authorized by a 3rd party trusted entity. Moreover this problem is bound to the nature of the VNFs, i.e. having a middlebox video transcoder, assumes that the encrypted video traffic will be decrypted before processing, in order to apply the desired functionality.

L2 encryption, as for instance MACSec, has a limited scope to switches that are able to perform such encryptions. Even in this case, the destination and source address, and the SecTAG fields are not encrypted [Altunbasak 2005]. This encryption is done per hop (switch) and therefore does not affect the destination address. Moreover the current SDN switches do not support L2 encryption, ex. MACSec, whereas the current OpenFlow specification does not state anything regarding L2 encryption. Finally, Netfloc uses native overlay solution and does not support tagged traffic (VLAN/GRE). Consequently, a support of alternative tags, like SecTAG, is perpendicular to this solution. From the aforementioned, encrypted traffic should not hinder the described implementation, as it assumes that the IP/MAC tuple (src:dst) is well-known and provided by the service owner. This makes transparent the application of SDN rules to the packets that match the anticipated end-to-end path.

Discussion concerning packet forwarding vs. header rewriting: The final SFC mechanism in Netfloc was implemented as an alternative to our originally discussed and tested traffic steering

Figure 5.15 : High level view of the SFC test scenario (Figure 3-3 of T-Nova Deliverable 4.32 [del b]).

approach. In that approach based on simple flow rewriting, the goal was to maintain intact, the packet structure along the service chain (as if the packets are forwarded normally between endpoints). Nevertheless, this approach was not feasible in the current setup from the following two reasons:

- *VNF placement* (physical location of each VNF) may lead to nondeterministic paths.
- *VNF service type* may alter the chain dynamically (such as the vTC) by encoding the service chain information directly into the datagrams.

This MAC rewriting solution is implemented into the basic chain library of Netfloc and it offers reduced complexity as opposed to the alternative chain solutions, that require specific environment setup and non-standard OVS version. It is also a pure OpenFlow based library for implementing SFC with minimal overhead. Alternatives for service chain implementation, to the one described in this work, could be based on header encapsulation (MPLS/GRE/NSH) or rewrite patterns. A support for designing new solutions or extending the basic library, is provided in Netfloc via the Flow and Chain Pattern abstractions [net d].

5.3.4 Validation of SFC proof of concept

This section evaluates the functional operation of the proposed SFC solution, and presents in detail the experimental process that validated its applicability in a real NFV environment. The experimental results are related to performance and scalability of different network service chains.

5.3. SERVICE FUNCTION CHAINING AND PROTOTYPE IMPLEMENTATION IN NFV SCENARIO

SFC experimental setup

Figure 5.15 depicts the overall test case scenario. The orchestrator is a main instigator of the underlying infrastructure that manages both, the WAN and the NFVI-POPs. At a higher level, once a new client registers in the NFVI marketplace to request a specific network service, the orchestrator triggers the WICM with a new request for an isolated Client ID space. The WICM then creates a client's instance ID and reserves VLAN IDs for the client's network service management. Once the orchestrator has a green light from the WICM, it requests that the traffic coming from User1, traverses the SFC PoP before going to User2. The orchestrator is in charge of managing the network services' lifecycle by mapping the requested network service (NS) and instantiating the involved VNFs in the OpenStack Cloud infrastructure (VIM). Figure 5.14 depicts the physical details of the NFVI-PoP (denominated as PoP2 inside the NFVI ecosystem). The figure shows the VIM on the left and the SDN services, the WICM, and the access points to the VIM on the right.

As the sequence diagram depicts, Figure 5.16, after the initialization process (steps 1–4) and once the VNFs has been created and their services successfully started (steps 5–7), the orchestrator: (1) calls the WICM API to start traffic redirection (step 8), and (2) calls the Netfloc's API (step 9) to start the traffic steering through the requested service path. The vTC and vMT services are responsible for: packet classification, analysis and packet modification depending on the particular service specification. Details about the orchestrator's network service specification and provisioning is out of scope for this work, as the main focus is to show the function chaining prototype implementation based on the Netfloc's overlay model.

Figure 5.3.3 depicts in details the prototype scenario. User1 and User2 are used as client-server in the simulations scenario and they are connected to the SDN Pica8 switch. This switch is controller by the WICM and instructed to redirect the designated traffic to the SFC PoP. The traffic is bidirectional, tagged with VLAN IDs 400 and 401 (for each direction). The VNF components are deployed as OpenStack VMs. The virtual Service Forwarder (vSF) serves as an aggregator VNF that sends all the chained traffic back to the WICM outside the cloud. As a result of using Netfloc, the entire setup gets simplified when compared to the baseline OpenStack networking service (ex. the existence of tunnelling bridge is fully eliminated) [ope e].

Service chains

In order to create a realistic network service, we combined the VNFs in different network services. The first network service represents a baseline model of direct User1-User2 communication, without using chain. The second service is a chain of two L2 VNFs, VTC traffic classifier and vTC traffic forwarder. The third one consists of two chains - the previous one and a chain that uses the same vTC and L3 VNF – the vMT. The vTC is a common starting VNF for both chains, used to classify UDP video from other traffic types. Then it sends the video to the next VNF in the chain (vMT) to get transcoded, while sending the rest of the traffic to the vTC

SDN-BASED QOS ASSURANCE FOR CLOUD-BASED STREAMING

Figure 5.16 : SFC sequence diagram

5.3. SERVICE FUNCTION CHAINING AND PROTOTYPE IMPLEMENTATION IN NFV SCENARIO

forwarder. Finally the fourth scenario focuses exclusively on video traffic. To perform initial scalability tests of the chaining method, we replicated twice the same scenarios as in Chain2, described in Figure 5.3.3, deploying a total of 4 and 6 chains among the replicated VNFs. Figure 5.17 shows an OpenStack network representation of the final deployment. Following is a summary of the network chain services represented as an ordered sequence of VNFs:

- **Direct:** $User1 \rightarrow User2$
- **Chain1:** $User1 \rightarrow vTC(class.) \rightarrow vTC-f(frwd.) \rightarrow User2$
- **Chain2:** $User1 \rightarrow vTC(class.) \rightarrow vTC-f(frwd.); vTC(class.) \rightarrow vMT(transcoding) \rightarrow User2$
- **Chain3:** $User1 \rightarrow vTC(class.) \rightarrow vMT(transc.) \rightarrow User2$

Measurements and discussion

The experimental results demonstrate the characterization of the SFC deployment and include a performance comparison of the following QoS: throughput, packet loss, latency and jitter for all chain services.

Initial datacenter network measurements: Preliminary evaluations were performed in earlier development stage in order to evaluate the network representation model and the SDN performance of Netfloc and OpenStack. These measurements focused on the SDN-based tenant isolation using native OpenStack GRE/VxLAN encapsulation vs. the Netfloc's overlay model based on L2 OpenFlow forwarding rules. Unlike OpenStack, where after the OVS integration bridge *br-int*, the traffic gets forwarded to the tunneling bridge *br-tun* where it gets encapsulated/decapsulated, in this solution we enabled an explicit traffic forwarding between the physical NIC interfaces on the nodes (that are attached to the external OVS *br-ex* bridge) [ope e].

In these series of tests, latency and bandwidth were measured in order to evaluate the flow-based SDN approach. A significant latency gain was measured with nearly same bandwidth values for thousand standard sized packets sent over 0.5 seconds intervals. These initially collected results, confirmed that omitting encapsulation for tenant isolation can contribute to overhead reduction. Moreover they fostered the use of SDN principles in resolving data-center networking issues, and for creating advanced services and applications, as the aforementioned traffic steering mechanism – being currently a prominent NFV use case.

NFVI-PoP measurements: The experimental setup consisted of three nodes running OpenStack Juno release, one controller node running Netfloc, and a physical SDN switch, Figure 5.3.3. Table 5.1 includes the hardware specification of the entire demo setup. There are two SDN controllers in the scenario: an OpenDaylight instance installed on the WICM node being in charge of the Pica8 switch (type 48 x 1GbE RJ45 port base unit, 4 x 10GbE SFP+

Figure 5.17 : OpenStack representation of the replicated chaining scenario. It depicts the networks attached to each of the VNFs.

5.3. SERVICE FUNCTION CHAINING AND PROTOTYPE IMPLEMENTATION IN NFV SCENARIO

uplinks), and Netfloc that controls the entire physical and virtual network of the SFC PoP nodes, all plugged in the physical SDN switch. The network interfaces of the SDN switch have a limited bandwidth of 100 Mbps and MTU set to 1500. To generate traffic and process the results, D-ITG [Botta 2012] tool was used. Bursty frames were sent repetitively in 5 seconds interval and packet sizes varying between 64 and 1450 bytes. Each of the tests were repeated 100 times. For the video evaluations we used a Matlab implementation of the Structural Similarity Index (SSIM) as described in [Wang 2004].

The evaluation objectives to consider are: (1) effectiveness and correctness of the traffic steering mechanism in a real NFVI setup (including several components, networks and controllers); (2) robustness of the chain setup; (3) efficiency of the traffic redirection and packets manipulation; and (4) scalability of the chain deployment. Other goal of the extended chain pilot was to proof a seamless integration among the described components within a large-scale NFVI environment. The experimental setups covered in this work considers SFC as a whole, thus the results are based on end-to-end measurements that appear as part of the following "SFC performance measurements recommendations" listed in the IETF draft [Agrawal 2015]:

- Packet Loss, Delay and Jitter.
- Hop by Hop, E2E, and SFC segment(s)
- Granular Flows in SFP as well as SFP as a whole
- Continuous/proactive and selective/on-demand
- Packet Out of Ordering should not impact the PM calculation
- Compliance with NSH Standards
- PM applicability between different component in SPF path to exactly locate the problem area (this includes Measurement between SF-SF, SF-SFF, SFF-SFF etc.)

Hop by hop experiments on isolated SFC segments are out of scope for this part of the work and would be further elaborated in a future work.

Results and analysis: This part presents the evaluation analysis of the demo measurements organized as a comparison among TCP, UDP, ICMP and video traffic patterns for the established chain scenarios.

Throughput: Figure 5.18 depicts the throughput values as a function of different packet sizes, for each of the three chain scenarios. It is represented in a normalized scale on the y axis, with maximum values reaching 12 Mbps. As the figure shows, the throughput raises linearly as the packet size increases for all traffic types (ICMP, UDP, TCP). The ICMP traffic throughput is the highest among the other traffic types and has a minimal impact from the chains, whereas the highest TCP throughput appears in the baseline model. Significantly low TCP throughput was

registered in both chains, lower than the UDP throughput, both oscillating in the 0.2 – 0.6 MB interval. This shows that the chain impacts the UDP and TCP traffic throughput, which can be better observed in the Figure 5.20.

Figure 5.19 shows the parallel throughput values for the scaled chain scenarios (2 vs. 4 and 6 chains). We note identical traffic tendency and matching values for all the traffic types. This confirms that the chained traffic has lower throughput compared to the baseline case. The results also show that for the scenarios with 4 and 6 chains running in parallel, the throughput quality is maintained similar as in the case of 2 chains. The D-ITG tool used for the measurements, does not include extended statistical results for the throughput, to allow deeper understanding of this tendency. Therefore, dedicated scalability analysis could further support these outcomes and establish the number of chains threshold, in relation to the throughput. Such study is out of scope for this set of measurements, because the current objective is to test the PoC robustness and the tendency of some quality parameters within the constraints of the PoC environment setup, rather than testing the chain scalability bounds. To evaluate the described chaining method in production environment, it is important to choose advanced testbed setup and sketch scenarios with multiple chain services. This would unbound the results from the topology limitation and only relate the effect of the number of chains and chain services, to the scalability of the chaining method.

Several factors can impact the throughput values in the case of chained traffic, such as: system limitation, physical and virtual switch interfaces, behavior of OpenStack and OpenDaylight, as well as the effect of each individual network function along the virtual traffic path (with the specific actions they perform to the packets). By nature, TCP has increased overhead for acknowledgments, which may be one of the reasons for decreased TCP throughput values. In the concrete scenario, the generally low throughput values mainly appear as a direct consequence of the on-premise hardware limitations. To achieve higher TCP throughput in service chaining, one option is to apply TCP congestion mechanisms. Other alternative is to use dedicated hardware acceleration mechanisms, like for instance DPDK [dpd].

We would like to remark the importance of applying TCP congestion mechanisms, for example ECN [IETF], for traffic quality improvement. However considering similar tools is out of scope for these measurements. The evaluations in this manuscript characterize the chaining effect to the traffic, accounting an established infrastructure and a standard measurement setup, without the intention to demonstrate an optimal chain performance. For this, we relied on the available TCP commands set for traffic generation available in the D-ITG toolkit. The benefits of DPDK hardware acceleration mechanisms, on the other hand, have been demonstrated in a parallel research work [Kourtis 2015].

Packet loss: The results for the basic chain scenario are shown in Figure 5.22, whereas the packet loss patterns for the replicated chain setups are reflected in Figure 5.23. While ICMP and TCP have nearly zero packet loss regardless of packet sizes, protocol or scenario, UDP shows decreasing tendency when the packet size drops. Similarly for both cases, the UDP packet loss

5.3. SERVICE FUNCTION CHAINING AND PROTOTYPE IMPLEMENTATION IN NFV SCENARIO

Table 5.1 : Demo scenario hardware specifications

Server	CPU	Memory
Network Node	Intel Pentium 4 CPU 3.00GHz 2 MB Cache	2.4GB
Control Node	4Core Intel Xeon CPU E3-1240 V2 3.40GHz 8 MB Cache	15GB
Compute Node	4Core Intel Xeon CPU E3-1240 V2 3.40GHz 8 MB Cache	15GB
Switch	AMD Sempron Processor 2800+ Cache 256KB	993MB
SDN Node	4Core Intel Core2 Quad CPU Q8400 2.66GHz 2 MB Cache	3GB
WICM Node	4Core Intel Core2 Quad CPU Q6600 2.40GHz 4MB Cache	7.8GB
User 1	AMD Sempron Processor 2800+ 256 KB Cache	457MB
User 2	2Core Intel Core i7-5557U CPU 3.10GHz 4MB Cache	15GB
Pica 8	P2020 / Triumph 2	2GB
Servers' Operative System	Ubuntu 14.04.4 LTS	
Virtual Switch Open vSwitch v2.3.2	OpenFlow versions 0x1:0x1	

reaches maximum values of nearly 25% for 64 bytes packets, and values lower than 5% for 512 bytes packets. From that point on, the values remain constantly low and close to zero. In conclusion, the chains affect a traffic with low packet sizes. Similar behaviour has been registered in a related work [Kawashima 2013], showing increased UDP packet loss for very small and big packet sizes. In summary, traffic with packets sizes between 512 – 1280 bytes are the best candidate for achieving optimal packet loss performance in the chain. A study of the Internet packet size distributions [Sinha 2007] confirms that 60 – 80% of the Internet packet sizes are in the interval between 500 – 1400 bytes, which conforms to our measured chaining constraints. Alike the latency, we can observe that it's the traffic type that defines the quality within the chain, not the replicated chain scenario.

Latency: The latency results including the standard deviation are shown in Figure 5.24 with normalized values on the y axis. Most of the cases, except TCP chained traffic, register low latency and standard deviation. UDP and ICMP show a slight increment in the latency within the chains. TCP on the other hand, due to its nature, reaches higher values in the two chains, mainly for packets over 1000 bytes, followed by fragmentation for packet sizes between 1280 – 1450 bytes. Chain1 has lower latency than Chain2 both alone and scaled, as shown in Figure 5.25. At a current stage, due to the complexity of the pilot scenario and in general, the service chain as a work in progress NFV use case, It is difficult to identify a unique source that hinders the optimal performance of a service chain. Advanced measurements on particular endpoints and links could potentially provide an insight on which of the formerly numbered reasons for the throughput (if not altogether) affect the performance of an end-to-end traffic steered through the SFC-PoP. Even though this architecture follows a non-tunnelling solution that bypasses the OpenStack inherent cascading and encapsulation protocols for multi tenancy support [Open e], a processing overhead in the VNF services could be also a valid reason for bottlenecks that can affect the latency for the TCP traffic. Reaching an optimal traffic performance in the chain, can be as well constrained by the physical switch (Ubuntu machine with several integrated NICs of a limited capacity).

Jitter: The jitter shown in Figure 5.26 akin to the latency, registers highest values in the case

of chained TCP traffic. Following a raising curve, this appears mostly for packets above 1000 bytes with a maximum of 0.09 seconds in Chain2. Since jitter and latency are related parameters, the aforementioned latency discussion applies also here. UDP jitter values are oscillating in the range 0.03 – 0.05 seconds and they show negligible decrement for bigger packets. Ultimately, the chains cause minor impact to the ICMP traffic's and keep the jitter values below 0.01 seconds. Equivalent jitter behaviour to Chain1 and Chain2, is measured as well for the multiple chain scenarios, Figure 5.27. In summary, TCP traffic's jitter is bound to the packet size, whereas UDP and ICMP have marginally decreasing values with the increment of the packet size.

Chain3 results: The next experiment includes an evaluation of a video quality signal within Chain3. In order to measure the SFC effect on end user application, a video signal was transmitted from User1 with destination of User2. The quality was measured as the video is being saved in User2. The stored video sequence was compared to the original sequence using the Structural Similarity Index (SSIM) [Wang 2004], which is a widely used and recognized video quality assessment (VQA) method. SSIM is a full reference video quality metric, which means that in order to measure the video's quality both, the reference and the transcoded video are required. This explains the saving of the received video in User2.

To simulate a real life scenario, BigBuckBunny [vid] was used as reference video, shown in Figure 5.28, as a good candidate for evaluating video quality over error-prone service environment. The video comprises of 14.315 frames in a total duration of 10 min. The video was sent from User1, through the chain, to User2 in 5 different scenarios. In each scenario the network was flooded with traffic, each time at a different rate. The throughput rates varied in the range 0 – 95 Mbps. Figure 5.21 depicts the SSIM video quality index of the received video, as a function of the corresponding traffic throughput injected in the network in parallel with the video transmission.

The figure shows that the traffic injection impacts considerably the received video signal quality as the traffic throughput increases. In terms of SFC impact, we can see that for values of 0 Mbps and 10 Mbps, the SSIM index for the received video signal is satisfactory, confirming acceptable impact of the SFC and the vMT on the video degradation. One solution for improved video quality perception in a service chain environment, is to deploy the chain in a more powerful testbed setup, including switches and machines with lower impact to the performance. Other option is to integrate a packet acceleration framework such as Intel's Dataplane Development Kit (DPDK) [dpd] in the current architecture. In order for DPDK to work properly, the installation of a custom build OVS is a prerequisite.

The described video framework VNF is based upon the PF_RING kernel module and the implementation is using the standard OVS version, which is integrated in the current OpenStack distribution. The DPDK version of the vTC, would not be able to show the benefits of DPDK utilization over the described environment setup, since the standard OVS merely exceeds a 1 Gbps throughput threshold, as shown in the presentation from Madhu Challa [Challa]. DPDK hardware acceleration increases the network's capacity and throughput. There is a related study

5.3. SERVICE FUNCTION CHAINING AND PROTOTYPE IMPLEMENTATION IN NFV SCENARIO

that demonstrates in details the benefits and limits of DPDK in a NFV environment [Kourtis 2015].

As a summary of the evaluations, the results validate the SFC methodology and confirm a consistency across the tested chain scenarios. The cause for a degraded quality within the chains can come from different sources: a network outage (physical and virtual), SDN controller limitations, and VNFs' service type constraints, occasional ARP message exchange within the PoP, etc. As the results show, scaling the chain scenario twice or three times, does not affect significantly the overall traffic quality when compared to single chain topology. These preliminary scalability results appoint on the one hand, the environment setup as a principal responsible for the holistic traffic quality bounds. The NFV's scenario specifics, on the other hand, shape the behaviour of each protocol's traffic quality individually. We proved furthermore the correctness of the chaining method and demonstrated a generally solid results for latency, jitter and packet loss of the chained traffic. The performance evaluation instructs that leveraging the NFV testbed setup with mechanisms to increase throughput, can improve the performance of the UDP and in particular the TCP traffic to a great extent.

Future research tendency should focus on identifying the key building blocks that cause a degraded QoS behaviour in SFC architecture. This includes performing a through analysis involving hop-by-hop measurements on different segments along the chain, in order to contemplate these results with a more deterministic vision of the intermediate traffic patterns. The choice of our SFC setup consisting of two endpoints and SFC-PoP in between, is the most intuitive approach for testing the validity of the chaining solution. The complexity of the SFC mechanism is fixed by the network service use cases and the functionality of the VNFs as main factors, rather than the client/server endpoints. The overlay mechanism for traffic steering and the underlying topology setup, summed to the former, contribute equally to the complexity of the SFC demo PoC.

The fundamental objective for this work was testing a proof of concept SFC mechanism using several use case chains. Performing extensive scalability and stress tests is out of scope, being planned as future work when the tool improves and the infrastructure upgrades. Though User1 and User2 suggest of a simple scenario, deployment of the same is rarely a trivial case, at the time when most of the components (Netfloc, VNFs, ODL, Wicm) are under continuous development and sometimes prone to errors. Even today the native OpenDaylight and OpenStack frameworks endure occasional anomalies (though being backed up by large communities). Integrating new components in such eco-system is always challenging and requires substantial tuning. Therefore, a significant time of the evaluation is devoted to ensure proper and ready-to-test environment. The initial version of the results presented in this section have been included in the Deliverable D4.32 of the T-Nova project archives of the European Commission repository [del b].

In conclusion, performing advanced and extensive tests is a must forthcoming activity in the SFC era, in order to achieve a production-level performance and scalability. It is still in the focus

of OpenDaylight and OpenStack, the process of versioning, testing new features, methodology correctness and validity of new solutions. Despite working on a daily basis with those frameworks, the authors have a current potential to contribute the working solution and its complete evaluation study. Moreover there is a research initiative related to multi-pop deployment in the current focus of the project T-Nova. This may serve as a solid ground for triggering future research related to complex large-scale SFC deployments. To date, the authors have not identified a complete study identical to this one, where a chaining algorithm is followed up from sketch, to implementation and testing - neither among the commercial products, nor among the research and open source community. Therefore the presented outcomes could be a relevant contribution for further research initiatives of a similar kind.

Conclusions and Future Challenges: This contribution describes the design of OpenFlow-based SFC mechanism and its implementation in PoC, targeting effective and robust end-to-end traffic delivery for cloud-NFV services. It offers an integral service chain solution, from design to implementation, deployment and testing. Providing through evaluation measurements in a real-life scenario, it complements the current SDN-NFV traffic steering solutions with SFC characterization study. We developed and evaluated four open source component, forming part of the integral SFC service: Netfloc as SDN-driven SDK for datacenter network programming; vTC and vMT as enhanced VNFs references; and WICM as a component for advanced traffic management in modern DC scenario. Finally we performed extensive testing on ICMP/TCP/UDP and video traffic patterns repeated over multiple chain scenarios and validated the correctness, effectiveness and scalability of the proposed solution. The evaluation results confirm an efficient performance of the chaining method and a decent video quality for the transcoding virtual service. As a future research challenge, we will focus on extending the measurements by embracing a diverse set of VNF instances and use case scenarios, in order to achieve production-quality implementation. Dominant focus will be given in pursuing a decoupled hop-by-hop benchmarking measurements on the intermediate chain segments, to accurately test the fault-tolerance and to identify implementation improvements in the process towards ensuring a seamless behavior of complex NS.

5.4 Contributions Summary and Conclusions

The work in this Chapter leveraged the SDN technology as a most prominent candidate to ensure QoS adjustment and control. The contributions were structured in two parts. In the first part, we presented a prototype application for adjustment and control of aggregated QoS. This implementation was tested in OpenFlow-enabled SDN environment using the meter bands specification for adjusting different quality queues on the traffic in the network. The implementation based on the Ryu controller to offer a proof of concept implementation of the multicast architecture with the ability to ensure QoS for different user groups. This works also took part of the research in the European KIARA project, more specifically in the SDN module

5.4. CONTRIBUTIONS SUMMARY AND CONCLUSIONS

developed for the purpose of providing library for network-based QoS control for end-user applications, including video streaming services. We further showed the extended scope of this component in SDN-controlled residential network. For that purpose we presented APIs for QoS management and applied them to a multicast use-case service. By doing this we proved the feasibility of the service and the SDN library to offer a network level QoS adjustment for real-time streaming services that can be implemented in a similar SDN-enabled setups.

The work described in the second part complements work on SDN-based QoS model, by offering an alternative approach in SDN, to control and adjust the quality of a video transmission. The described vTC VNF, includes the mentioned functionality as its main objective and it is able to provide different video qualities based on traffic classification parameter. Combining the use of VNFs like the vTC with the mechanism for traffic steering proves that the desired tuning of the traffic characteristics can be achieved on a network level by applying dedicated VNFs and traffic steering methods. Focusing exclusively to the video traffic and considering the model of tiered video qualities, the functionality of multi-quality video dissemination can be achieved by building different chains, each one destined to provide the desired video quality. This entire process can be abstracted from an endpoint perspective, just the same way as the P2P overlay schemes do with the underlying infrastructure. In this case the video server traffic can be steered via dedicated chain and VNFs, experiencing adjustments as per QoS specification, before sending it to the receiver.

Having one goal in common, the previous works made a direct contribution and wrap up of the QoS study tackled throughout the entire thesis work. In this sense, by controlling the final stack of the network today a provider can ensure firmer guarantees on the quality of traffic and it can offer granular price models in respect. Although the described work focuses on single-PoP/datacenter implementation and analysis, we have further validated the feasibility of the method in a multi-datacenter eco-system within the previously described T-Nova project. This shows that while the initial SDN contributions were enclosed in a limited infrastructures managed by a single provider, with the advances of the 5G/LTE networks and the use cases in SDN-NFV scenarios, this work can be extrapolated on large-scale services that can be co-located in the premisses of different network operators and service providers. This is well demonstrated with the concept of multi-tenancy in 5G, where the network and its functions can be sliced among several service providers located on different geographic locations. With extended share of the network among different providers, SDN and NVF are closing the gap of having isolated networks and functions, by providing isolated network slides across a common large-scale network. Such approach enables the provision of value-added services across several provider networks, with the possibility to ensure certain SLAs, towards improved QoS and QoE. A multimedia service can be easily deployed and supported in such multi-PoP setup, relaying on different VNF functionalities to guarantee its best performance. For example, in a real-time P2P streaming characteristics such as peer churn can be handled in a transparent way across the network, while malicious traffic can be easily detected eliminated with using VNFs like firewalls and security appliances placed in critical points of the network.

SDN-BASED QOS ASSURANCE FOR CLOUD-BASED STREAMING

Figure 5.18 : Throughput for: no-chain, Chain1 and Chain2 scenarios

Figure 5.19 : Throughput for: 2 chains, 4 chains and 6 chains scenarios

Figure 5.20 : TCP/UDP throughput for: no-chain, Chain1, Chain2, 4 and 6 chains scenarios

Figure 5.21 : SSIM Video quality assessment results in relation to traffic interference (Figure 3-10 of T-Nova Deliverable 4.32 [del b])

Figure 5.22 : Packet loss for: no-chain, Chain1 and Chain2 scenarios

Figure 5.23 : Packet loss for: 2 chains, 4 chains and 6 chains scenarios

5.4. CONTRIBUTIONS SUMMARY AND CONCLUSIONS

Figure 5.24 : Latency for: no-chain, Chain1 and Chain2 scenarios

Figure 5.25 : Latency for: 2 chains, 4 chains and 6 chains scenarios

Figure 5.26 : Jitter for: no-chain, Chain1 and Chain2 scenarios

Figure 5.27 : Jitter for: 2 chains, 4 chains and 6 chains scenarios

Figure 5.28 : Video transcoding and watermark embedding by the vMT. Left: ffmpeg video processing output on User2. Right: final video result.

Chapter 6

Validation and Results

6.1 Validation in European Projects

Some parts of the contributions described in this thesis took part of the following three European projects: FIWARE and T-NOVA as representatives from the FP7 program and SESAME as a member of the Horizon 2020 project initiative. We will reflect briefly the adoption of the component and the respective validation of the results within each of them.

6.1.1 FIWARE - A Core Platform of the Future Internet

The FIWARE project marked multiple involvements in this research during the entire cycle. Initially the author worked on the Cloud Hosting task which has the objective to design and advance Portal for the entire FIWARE framework, based on the OpenStack Dashboard. Although this contribution is not directly related to this thesis, it is worth to be acknowledged due to the longest development cycle it took during the thesis work and because it has also inspired several ideas and side contributions to main research line. The results in terms of open-source implementation and the forthcoming community engagements to this task has marked a successful transfer of results to the community and the FIWARE project in general.

Another work that took part and was materialized within this project is the QoS library within the KIARA middleware, developed within the I2ND task for network interface management. The role of the QoS component in KIARA, was to define APIs that enable end-to-end QoS provision between two endpoints using KIARA client/server and/or KIARA endpoint and a web server. This was performed via SDN plugin on the network layer that offered the following set of parameters/actions:

- QoS parameters (bandwidth, latency, jitter and packet loss)
- Well arranged P2P multicast overlay
- Optimization of the datapath through the network
- Topology information i.e., the type of the underlying network

As described in Section 5.2, we created QoS application as a proof of concept for the SDN based QoS retrieval and control. The application combined and provisioned the desired parameters for latency and bandwidth between a new node and the existing node in the network. The code of the application was included in the SDN module of the KIARA transport component.

6.1.2 T-NOVA - Network Functions-as-a-Service over virtualized Infrastructures

The T-Nova project focused on the design and implementation of an integrated end-to-end architecture for NFV services dedicated to automated provisioning, management, monitoring and optimization of Network Functions-as-a-Service (NFaaS) over virtualized Network/IT infrastructures. The aim was to address the key challenges of deploying VNFs in carrier grade environments by covering all layers of the technical framework: from the Marketplace down to the Infrastructure (NFVI). Having this target in mind, T-Nova presented a complete functional solution, which can be elevated to pre-operational status with minimal additional development.

Netfloc, NETwork FLOws for Clouds, was developed as an open source SDK for datacenter network programming. It is comprised of set of tools and libraries packed as Java bundles that interoperate with the OpenDaylight controller. Netfloc exposes REST API abstractions and Java interfaces for network programmers to enable optimal integration in cloud datacenters and fully SDN-enabled end-to-end management of OpenFlow enabled switches.

Providing a solution for Service Function Chaining was one of the target use cases inside the T-Nova domain. This capability was included as a reference library implementation inside Netfloc, to demonstrate a first end-to-end scenario that was enabled by directly employing its dedicated traffic steering library. Moreover the product of the SFC implementation has delivered a successful integration of several T-Nova components into a holistic chaining solution, demonstrates successfully during the second, and the third and last year in the project resulting in multi-PoP demonstration of the SFC service [tno]. Netfloc was a core component for the network management in T-Nova, delivering the the integral chain scenario what has proved and validated among the others, the video transcoding chain as the most prominent to this thesis. The details of the validation has been described in Section 5.3.3.

6.1.3 SESAME - Small cELLS coordinAtion for Multi-tenancy and Edge services

The SESAME project has as objective creating multi-tenant cloud enabled (C-RAN) architecture included under the H2020 5GPPP initiative. This effort includes evolving commercial Small Cells (SC) towards a so-called Cloud Enabled Small Cell (CESC). A CESC integrates a virtualized execution platform (micro server) equipped with, e.g., IT resources (RAM, CPU, storage) and Hardware Accelerators (HWA) such as: Graphics Processing Units (GPU), Digital Signal Processors (DSP), and/or Field-Programmable Gate Arrays (FPGA). SESAME also proposes CESC's clustering for creating distributed data center, denominated Light Data Centre (Light DC). This component in essence places the network intelligence and applications in the network edge with the help of NFV while preserving the QoS and 5G performance requirements such as latency and bandwidth.

Within the scope of this project Netfloc was validated as a component to steer the traffic between different VNFs deployed in a LightDC. One of the demonstrations using the SFC component was held between the SESAME and COHERENT projects, in the area of

6.1. VALIDATION IN EUROPEAN PROJECTS

Figure 6.1 : Joint SESAME-COHERENT demo architecture shown in EUCNC'16

multi-domain orchestration and service function chaining on the EUCNC 2016 conference [Trajkovska]. Figure 6.1 shows the end-to-end schema. The objective of this demo was to show a synergy among the two projects under the frame of 5GPPP; SESAME and COHERENT as a first step towards bridging the gap between the telco and cloud world, by creating a network service based on the services and technologies coming from both worlds. With this in mind, the demonstration tackled the following two objectives: (1) orchestration and automatic deployment of virtual network functions (VNFs) into the CESC cloud infrastructure (2) providing on the top, an SDN-enabled SFC between network services involving both, Cloud-based and Telco-based VNFs.

The process included end-to-end orchestration and composition of a service that delivers optimized and tailored video to an end-user through their user equipment, a mobile device. The demonstration showed the delivery of the following individual services:

- Cloud-RAN deployment of a RAN and EPC services of the OpenAirInterface project [ope b]. This part is supplied by the COHERENT 5GPPP project depicted in the Coherent PoP, Figure 6.1 (left).
- SDN-based Service Function Chaining between the cloud-deployed VNFs (vTC and vMT) depicted in the Sesame PoP, Figure 6.1 (right).

The part of the demo in focus of the COHERENT project demonstrated how to build, deploy, provision, and dispose various service topologies for LTE-LTE-A using the OpenAirInterface 5G

Figure 6.2 : LTE/LTE-A network deployed with Juju and OpenAir interface

platform orchestrated by Canonical JuJu framework [juj]. Service instances can be efficiently scaled in/out overtime to meet the workload demand. In particular we demonstrated the ability to deploy a real word LTE/LTE-A network shown in Figure 6.2.

In the SESAME part of the demo, the ICCLab-developed orchestrator (Hurtle) [hur] was used as a main initiator of both, the VNF services and the Netfloc services. Hurtle first deploys the VNF VMs on the OpenStack infrastructure and then starts their services automatically via the Heat orchestration engine. Afterwards it establishes the underlying chaining graph by invoking the Netfloc's Create Chain API, creating with this, a mapping of the Neutron port IDs from the VNFs "to be chained", to their respective OVS endpoints of the service chain graph. The SFC traffic has the following order: the end-user request that comes from the telco endpoint hosted by the EUCNC PoP, is orchestrated by Heat to follow up into the SESAME Cloud PoP. A VNF holding the requested video is triggered to start its service and send out the video for enhanced service management inside the previously defined chain graph. From the video source VNF, the traffic is steered to the vTC that performs a basic traffic analysis, and sends out the video traffic to the vMT being the following VNF in the chain. The vMT performs media traffic transcoding and adds up a watermark to the video before sending it out of the Cloud-PoP and back to the end-user who requested the enhanced video service. Finally the result is shown on the end user's terminal via VLC.

The project SESAME is ongoing at current state and there are several contributions in preparation that involve the SFC component. Overall we can conclude that the SFC use case extrapolated to a larger scale scenario among several service operators, is a way to confirm the importance of SDN for delivering value-added services and functionalities in the network. Integrating SDN as integral building-block in a 5G telco network as the one described in SESAME, shows the feasibility of delivering end-to-end video service to users in different geo-locations. It moreover validates the ability to offer additional services on the edge, like the QoS adjustment and control QoS, monitoring, load-balancing etc. In a network as this, a

6.2. DISSEMINATION OF RESULTS

video-service provider e.g. Netflix, would be able to deliver video services by contracting network slices and VNFs, for the purpose of providing a targeted streaming with specific SLAs and granulated price models to their users.

6.2 Dissemination of Results

6.2.1 Publications

International Indexed Journals

- P. S. Khodashenas, B. Blanco, I. Taboada, M.A. Kourtis, G. Xilouris, I. Giannoulakis, E. Jimeno, I. Trajkovska, J. O. Fajardo, E. Kafetzakis, J. Garcia, F. Liberal, A. Whitehead, M. Wilson. *Service Management and Orchestration over Multi-Tenant Cloud-Enabled RAN*. IEEE Transactions on Network and Service Management, 2017. [submitted].
- Irena Trajkovska, Michail-Alexandros Kourtis, Christos Sakkas, Denis Baudinot, João Silva, Piyush Harsh, George Xylouris, Thomas Michael Bohnert, Harilaos Koumaras. *SDN-based service function chaining mechanism and service prototype implementation in NFV scenario*. Computer Standards & Interfaces, 2017.
- Bego Blanco, Jose Oscar Fajardo, Ioannis Giannoulakis, Emmanouil Kafetzakis, Shuping Peng, Jordi Pérez-Romero, Irena Trajkovska, Pouria S. Khodashenas, Leonardo Goratti, Michele Paolino, Evangelos Sfakianakis, Fidel Liberal, George Xilouris. *Technology pillars in the architecture of future 5G mobile networks: NFV, MEC and SDN*. Computer Standards & Interfaces , 2017.
- I. Giannoulakis, E. Kafetzakis, I. Trajkovska., P. S. Khodashenas, I. Chochliouros, C. Costa, I. Neokosmidis, P. Bliznakov. *The emergence of operator-neutral small cells as a strong case for cloud computing at the mobile edge*. Transactions on Emerging Telecommunications Technologies, July 2016. ett.3077.
- Fajardo, Jose Oscar, Fidel, Liberal, Ioannis, Giannoulakis, Emmanouil, Kafetzakis, Vincenzo, Pii, Irena, Trajkovska, Thomas Michael Bohnert,, Leonardo, Goratti, Roberto, Riggio, Javier Garcia Lloreda, Pouria Sayyad Khodashenas, Michele Paolino, Pavel Bliznakov, Jordi Perez-Romero, Claudio Meani, Ioannis Chochliouros,, Maria Belesioti. *Introducing Mobile Edge Computing Capabilities through Distributed 5G Cloud Enabled Small Cells*. Mobile Networks and Applications, 2016.
- J. Cerviño, P. Rodríguez, I. Trajkovska, F. Escribano, J. Salvachúa. *A Cost-Effective Methodology Applied to Videoconference Services over Hybrid Clouds*. Mobile Networks and Applications, Vol. 18, Num. 1, pages 103–109, 2013.

International Non-(JCR) Indexed Journals

- I. Trajkovska, P. Rodríguez, J. Cerviño, J. Salvachúa. *A Novel Monetary Incentive for Commercial P2P Streaming*. Journal of Multimedia Processing and Technologies, Vol. 3, Num. 3, pages 124–135, 2012.

Book Chapters

- J. Cerviño, F. Escribano and P. Rodríguez, I. Trajkovska, J. Salvachúa. *Leasing Videoconference Resources on Hybrid Clouds*. Cloud Computing: Methodology, Systems, and Applications, pages 511–527, CRC Press, 2011.

International Conferences

- P. S Khodashenas, A. Betzler, J. G. Lloreda, E. Jimeno, I. Trajkovska, L. D. Vecchio, A. Whitehead. *Ensuring Quality of Service in a Multi-Tenant Cloud-Enabled RAN Environment*. European Conference on Networks and Communications (EUCNC), 2017.
- I. Chochliouros, E. Sfakianakis, M. Belesioti, D. Arvanitiozisis, A. Spiliopoulou, I. Trajkovska, V. Pii, I. Neokosmidis, T. Rokkas, A. Dardamanis, H. Mouratidis, C. Turyagyenda, S. Peng. *Defining a Proper Methodology for Effective SESAME-based Scenarios and Potential Use-Cases to create Market Impact and Promote Modern Services*. EUCNC Special Session: "Designing and Developing a Cloud-enabled "Small Cell as a Service" concept, for Multi-Tenancy and Edge Services in the forthcoming 5G Framework.", June 2016.
- I. Trajkovska, P. Harsh, T. M. Bohnert, J. Salvachúa. *Real-time streaming in the cloud can increase the provider's profit and the P2P users' benefit*. Proceedings of the IEEE International Conference on Consumer Electronics - Taiwan (ICCE-TW), pages 35–36, May 2014.
- I. Trajkovska, P. Aeschlimann, C. Marti, T. M. Bohnert, J. Salvachúa. *SDN enabled QoS provision for online streaming services in residential ISP networks*. Proceedings of the IEEE International Conference on Consumer Electronics - Taiwan (ICCE-TW), pages 33–34, May 2014.
- I. Trajkovska, P. Rodríguez, J. Cerviño, P. Harsh, J. Salvachúa. *P2P incentive model for QoS based streaming systems*. Proceedings of the 11th IEEE Consumer Communications and Networking Conference (CCNC), pages 281–286, January 2014.
- P. Harsh, K. Benz, I. Trajkovska, A. Edmonds, P. Comi, T. Bohnert. *A highly available generic billing architecture for heterogenous mobile cloud services*. Proceedings of the International Conference on Grid Computing and Applications (GCA), The Steering

6.2. DISSEMINATION OF RESULTS

Committee of The World Congress in Computer Science, Computer Engineering and Applied Computing (WorldComp), 2014.

- I. Trajkovska, P. Rodríguez, J. Cerviño, J. Salvachúa. *Billing model with discount incentive in a P2P-cloud multimedia streaming system*. Proceedings of the World Congress on Computer and Information Technology (WCCIT), pages 1–6, June 2013.
- I. Trajkovska, P. Rodríguez, J. Cerviño, J. Salvachúa. *Opportunities and Challenges of Implementing P2P Streaming Applications in the Web*. Proceedings of the IADIS Conference on Collaborative Technologies, July 2012.
- P. Rodríguez, J. Cerviño, I. Trajkovska, J. Salvachúa. *Advanced Videoconferencing Services Based on WebRTC*. Proceedings of the IADIS Conference on Collaborative Technologies, July 2012.
- J. Cerviño, P. Rodríguez, I. Trajkovska, F. Escribano, J. Salvachúa. *A Cost Effective Methodology Applied to Videoconference Services on Hybrid Clouds*. Proceedings of the FTRA Workshop on Human centric computing, P2P, Grid and Cloud computing (HPGC), December 2011.
- J. Cerviño, P. Rodríguez, I. Trajkovska, A. Mozo, J. Salvachúa. *Testing a Cloud Provider Network for Hybrid P2P and Cloud Streaming Architectures*. Proceedings of the IEEE International Conference on Cloud Computing (CLOUD), pages 356–363, July 2011.
- J. Cerviño, F. Escribano, P. Rodríguez, I. Trajkovska, J. Salvachúa. *Videoconference Capacity Leasing on Hybrid Clouds*. Proceedings of the IEEE International Conference on Cloud Computing (CLOUD), pages 340–347, July 2011.
- I. Trajkovska, J. Salvachúa, P. Rodríguez, A. Mozo Velasco. *A Novel P2P and Cloud Computing Hybrid Architecture for Multimedia Streaming with QoS Cost Functions*. Proceedings of the 18th ACM International Conference on Multimedia, MM '10, pages 1227–1230, ACM, 2010.

Demonstrations

- I. Trajkovska, A. Edmonds, N. Nikaein, K. Katsalis. *Joint demonstration between the 5G-PPP SESAME and COHERENT projects, in the area of multi-domain orchestration and service function chaining*. June 2016. European Conference on Networks and Communications (EUCNC), Athens Greece.

6.2.2 Standards and open-source communities

The implementations of both, the QoS library and the chain mechanism described in the thesis are made available publicly as GitHub projects, delivered under the Apache license. It was a primary

objective since the creation of these solutions to offer them as open software to the communities, targeting not only an extensive adoption by the developers and the users, but also the receiving of feedback towards the process of continuous integration. Following this objective, we also want to contribute the solution to the relevant groups that will result on further project engagement and improvement of the idea.

The joint research efforts between the NFV and the SDN working groups, have received a substantial amount of attention lately. In the span of one year alone, several SFC draft proposals has been submitted; header specification for SFC has been announced and adopted in the the new version of OVS; new standards in progress are tracing the way to a joint SDN-NFV community. It is undoubtedly the best growing era for these communities, with quantity of tools and shared support coming from developers, researchers and companies. The presented chain mechanism leverages the OpenFlow standard and builds upon the OpenDaylight controller that is currently holding a leading position among the SDN community [odl a]. Netfloc moreover uses the ML2 plugin of Neutron to establish a full SDN control of the OpenStack network, and therefore it is fully compliant to the joint OpenStack-ODL projects.

In more details, Netfloc's idea of chaining neutron ports in OpenStack, relates to the recently introduced *neutron-sfc* project (Neutron ports list as an ordered set of connection points in the chain). This OpenStack's approach is similar in the description of the chain as the one presented here, but the overlay topology is different. Having an alternative (L2-based) approach coming from T-Nova as a collaborative European project, would be relevant contribution to the OpenStack and OpenDaylight projects. The OpenStack Neutron community could be a solid ground to offer our non-tunneling approach to build a simplified overlay, as alternative to their native encapsulation mechanisms. Solutions like the VNT-based chaining by NEC [NEC] (tackled in the Section 2 of this paper), that were contributed to the Hydrogen release of ODL, are also based on a dedicated overlay. They build OpenFlow solution based on flow rewriting similar to ours, being accepted as alternative to the SFC standardized approach within the open source community.

In a recent work, the authors have created an OpenStack Heat based support for automatic chain deployment and service description [net e]. Moreover a solution based on the ETSI NFV MANO specification for network service (NS) and VNF description (VNFFGD) is enclosed among the latest advancements. The authors will proceed to provide a support for Tosca-based service description, in order to make the SFC solution compatible to NFV standards and enable automatic service deployment from Tosca-based orchestrators, like for instance in OpenStack. From OPNFV point of view, the vTC component used in this contribution is forming part of Yardstick [OPNFV] – enclosed in the list of OPNFV projects, therefore following the same approach could be another option for promoting adoption and contribution of the Netfloc's SFC solution.

Finally the *odl-sfc* project is growing larger and currently enables multi-protocol renderers. Netfloc's SFC project may potentially fit in the OpenFlow specific render of the odl-sfc project.

6.2. DISSEMINATION OF RESULTS

With the above said, the authors believe that the presented solution can contribute a good value to the community. Therefore it is our current objective to emphasize the promotion of the Netfloc's SFC mechanism among the relevant open source groups. At current stage the author is involved in a process of promoting the Netfloc SFC mechanism among the OpenDaylight community and adapting the project to best fit in the integral framework.

6.2.3 Research visit

The author joined the ICCLab at the Zurich University of Applied Sciences [icc b] in Switzerland during three months research stay as part of the PhD program. The idea was to exchange knowledge and leverage the doctoral research in joint efforts with external colleagues that work on a similar problematic in the field.

During that period the author encountered for the first time with the SDN technology, which was in the initial steps of development at the time. Prior to selecting a the best matching contribution that is closest to the author's topic research, she revised the State of the Art in the SDN field. Afterwards she was involved in a cooperative work related to quality of service, i.e., the author worked on a proof of concept application for QoS control using SDN and OpenFlow version 1.3. The work was tested in Mininet simulation environment, considering that the provisioning of physical SDN testbed at the lab was yet under construction at the time. Nonetheless the results of the application were validated and included in the SDN component within the KIARA middleware project. This contribution is elaborated in details in the Section 5.2.1. In period following the research visit, this component forming part of KIARA was used by other projects in the FIWARE platform [fiw] that aimed on providing QoS for end-user IoT services like streaming and realtime multi-user 3D applications among others. With this work the author had the chance to acknowledge the benefits of the emerging SDN technology for controlling the quality of service, which was a principal characteristic of the proposed P2P-cloud architecture. In this work it was demonstrated the feasibility of controlling the set of QoS parameters for real-time communication on a network layer, using an SDN controller with OpenFlow-enabled capabilities and APIs.

In parallel to the research and development, that author had the chance to perform education activities and knowledge transfer to the students of the annually organized summer school at the ZHAW [icc a], where together with Piyush Harsh she held a course "Introduction to OpenStack" to a joint group of Swiss and American students. Some of the other lab involvements include: submission of several publications in cooperation with the lab members (all enclosed in the Section 6.2.1), joint discussions and meetings, attending talks and new candidate interviews, create student project proposal, deployment and testing of OpenStack infrastructure, etc. For the purpose of work promotion and dissemination the author attended a COST project meeting in Ohrid, Macedonia where she gave a talk entitled "P2P Multicast in KIARA with OpenFlow based QoS" [icc c]. The leading objective of this project was to support European research and

innovation by encouraging scientific developments and integration of research communities. This talk was related to the research work and results during the ICCLab stay and served as a potential enabler for mutual discussions, ideas and a way to establish future networking activities.

6.2.4 Teaching and talks

- *Introduction to OpenStack* [icc a] - Lecture at ICCLab (ZHAW) Summer School on Cloud Computing, July 2013 Winterthur, Switzerland.
- *P2P Multicast in KIARA with OpenFlow based QoS* - Talk at COST IC0906 WiNeMO. 24-25 September, Ohrid, Macedonia.
- *Cloud Networking I and II* - Lectures and Labs at ZHAW for Master and Bachelors courses. 2015-2016, Winterthur and Zurich, Switzerland.
- Co-organized with SWITCH and held three talks on SDN workshop/meetup [sdn b]. 2015-2016, Bern, Zurich and Winterthur, Switzerland.

Chapter 7

Conclusions

The immersive technology for real-time streaming got its momentum during the promotion of new access devices on the market, such as: smart phones, tables, and leading edge TV receivers. This trend was bound to the improvements in traffic quality of the last mile in order to offer a full range of available services from content providers to end users on the Internet. This process has been challenged by the academia and the industry together who acted in cooperative manner towards the creation and deployment of novel solutions for real time multimedia delivery. Providing seamless system that is able to address all the requirements with a good scaling performance has been a great challenge even to companies specialized in these technologies. This dissertation took the opportunity of the prospective P2P research combined with the commercial cloud offers in order to offer an extensive research framework as a baseline to understand, experiment and offer solutions aligned to the initially defined problem statement.

The author has focused on the entire spectrum of P2P applications and algorithms for streaming in order to analyze the problems and find potential answers to relevant questions with respect to algorithms for differentiated multimedia delivery expressed via QoS policies. The lack of solutions that unify all types of profiles - from commercial clients to classic P2P users was a main motivation for proposing an architecture that will be suitable to host them together. The perception was that such a solution with proper business model and QoS guarantees will be attractive to many users. Consequently, an increased service popularity tends towards great success to service providers and cloud providers involved in hosting such services. The whole idea for such a service leaned on the assumption that if P2P became and remained popular standing on the shoulders of altruism, it will well fit in a system that is moreover enhanced with a business model based on QoS, cooperation and monetary remuneration incentives.

Multiple doubts were raised along the process of the research, each coupled to a specific task, its results and followed by reasoning for the adjacent targets to pursue. Interpretation of overlay topologies and their role into the service design, P2P algorithms and their impact to specific parts of the service, media transcoding for targeted video delivery quality etc. are only few of the research activities along the process. Several legacy solutions for P2P were tested in order to analyze the effect of the proposal in a functional system. Life-scale scenarios were provisioned spanning across multiple domains and involving several modern technologies, all having one goal in common - to offer a testbed environment and tools to facilitate implementation, deployment and validation of the proposals. QoS APIs were designed and implemented in coherence with the SDN principles and extrapolated to cloud, residential, NFV and 5G multi-domains to validate the

solution and demonstrate the performance of the use-case services. Analytical studies involved: a comprehensive analysis of a cloud infrastructure to support real-time streaming, feasibility of the streaming service topology in large-scale networks, theoretical evaluation of the discount incentive mechanism, and eventually it was wrapped up with the the SFC proof of concept demonstration and characterization of the common service QoS parameters.

As a result of the previous, this dissertation has made multiple contributions in the domain of multimedia streaming over hybrid cloud -P2P topology, interfaces for quality of service control, and video quality adjustment via SDN and NFV. Therefore the entire work can be divided in three dimensions: (1) Research related to steaming technologies, algorithms and protocols, including tests of the famous streaming mechanisms for multimedia video content, which resulted in a proposal of a novel P2P-Cloud streaming architecture; (2) Research study related to the incentives of the proposed architecture. An incentive mechanism is proposed based on discounts and empirical study that validated the incentive mechanism of the P2P-Cloud architecture, as a trade-off for the multimedia content provider and the users. (3) QoS control of multimedia streams based on SDN technology is implemented and tested in a prototype application. The idea of SDN-gearred QoS provision for residential streaming serviced is bounded to the model, and finally full implementation of a chaining methodology based on VNFs to support QoS control on network level is implemented and tested. The following section revisits the objectives defined in the Section 1 giving summary on the respective contributions.

7.1 Contributions

- *Propose architecture for streaming as a foundation of the research.*

A novel architecture for P2P multimedia streaming was suggested, as a potential solution towards the ISP conflicting issues with the P2P applications regarding traffic dominance on the Internet. The architecture was based on two technological specifiers, cloud infrastructure and P2P overlay topology. The motivation for this approach was to grasp the upcoming opportunities promised by the cloud providers in their early stage and exploit the potentials of the cloud to serve in the creation of streaming architectures based on P2P streaming algorithms. At the time of defining the scopes of the research and setting up the problem statement, we identified potential directions to take in the process of expanding the P2P use-cases. In the times of increased popularity of streaming services supported by content delivery networks, Internet television, interactive online gaming, video conferencing etc., the classical file-share P2P applications were facing important issues and limitations for disseminating real-time contents. The problems escalated as a consequence of the immense increase in the number of service users who solicited higher resource demands form the entire eco-system. The streaming service was a representation of opportunistic use-case as a core for the further research ideas and initiatives.

Under these terms, this conceptual architecture was one of the pioneers at the time to

7.1. CONTRIBUTIONS

present a use-case on the top of the two converged technologies, P2P and cloud. The cloud was chosen as important asset for the architecture scenario, primarily guided by the dominant role it started to play in many technological domains, and secondary inspired by its novel concepts, such as: distributed, managed servers that offload the hosting responsibility from the customers, highly scalable and available resources, in-time and on-demand responsibility, economically attractive price models etc. This approach was intended to serve as catalyst to reinforce the interaction between the service providers and the end users and an inception for similar applications in the cloud.

The proposed architecture contributed in several ways to the emerging cloud and P2P research: (1) it employed the cloud principles as a base of the P2P architecture in order to further exploit and demonstrate the economical and technical achievements that it could bring to the de-facto P2P applications; (2) leveraged partially in a client-server fashion, it was shown that the streaming service could take the best of both worlds such as: increased bandwidth resources, controlled quality of service, targeted video instance replication at certain geo-location; (3) exposed APIs to users in order to tune down the connections and the desired QoS; (4) presented a business model bound to the QoS levels as aggregate cost functions that determine the price policy; (5) elaborated an idea on service that can be deployed in a manner that combines commercial and free of charge users together, according to their video quality preference, excess network resources and altruistic intentions.

We discussed how the QoS APIs offered by the providers facilitate to the users a clear representation of the streaming service's current status. The aim was to enable negotiation of relevant QoS such as bandwidth, jitter and latency, between a streaming service provider hosted in the cloud, and the clients willing to join the service. Based on their preferences, the users should be able to choose the best matching price plan or alternatively join the overlay swarm as a peer - completely free of charge. This idea was a natural approach towards solving the enduring P2P-ISP argument, by offering a transparent solution equally to the service providers and the end users. This solution however was an alternative to the CDN principles because it followed the P2P approach on building service over both, public and private cloud infrastructures. It moreover set the stone for scalable cloud SaaS based on QoS guarantees to be deployed by streaming service providers.

- *Emulate the proposed architecture in a real-life scenario.*

After establishing the architecture and the hypothesis of the potential cloud benefit, the research progressed towards emulating a real test cases of streaming video in a known cloud provider's network among geographically distributed users, composed of cloud virtual machines and regular Internet users. In this process, a through analysis of the cloud network was pursued in order to measure essential QoS parameters in the context of

streaming video across multiple distributed cloud instances. The idea was to examine the assumption and answer the question if the cloud was ready to support the deployment of real-time services. With the the tests it was demonstrated the behavior of the network and the quality of service perceived by the users. Since the video streaming is supported on application level and it is dependent mainly on compute resources, a streaming architecture was created by deploying cloud instanced and PlanetLab nodes across different locations on the planet representing a large-scale demonstration scenario.

We further designed a series of practical test scenarios in order to represent the P2P-Cloud architecture in a realistic large-scale setup supported by a famous cloud provider in the nucleus. With combined placement – part of the nodes in the cloud, and part – as PlanetLab hosts (peers), the tests aimed on determining the following: (1) benefit of the cloud in multimedia streaming and (2) quality of cloud for multimedia streaming. The measurements considered the most common QoS for multimedia services: latency, bandwidth, packet loss and jitter.

- *Analyze the cloud ability to support real-time services.*

This work is tightly coupled with the study performed in the previous objective. After the architecture setup, the goal behind the first series of test scenarios was to represent the ability of the cloud to support real-time services. For these measurements using tools like ping and iperf between different Amazon datacenters were in charge of characterizing the commonly considered QoS in real-time streaming. The conclusion included the following: (a) jitter degradation is nearly eliminated if both, the source and destination node of each intercontinental traverse are placed in the cloud, (b) throughput degradation is alleviated by placing the source node of each intercontinental traversal in the cloud, and (c) incrementally moving nodes to the cloud, decreases packet drop.

The second analytical task was to depict the video quality changes bound to the diverse positioning of the nodes participating in the streaming. A component called "irouter" was run in each of the nodes in order to allow packet forwarding and reception. It worked in a way that it received data from the video daemon, displayed it and then forward it to other irouters (users). This component was borrowed from a related videoconferencing system and modified in order to gather data and process statistics from all the incoming packets. A real video traffic was used for the tests in order to better approximate a realistic scenario. This enabled to authentically compare the received QoS results, to the ones required in multimedia transmission. As previously stated, we used PlanetLab to geographically spread the distribution network nodes, and Amazon as cloud provider. The conclusion retrieved from this part states that a video service provider can take advantage of reduced packet loss and low latency that exists among the datacenters of a cloud provider, and shape up accordingly the service topology. As a result of controlled QoS within the cloud

7.1. CONTRIBUTIONS

infrastructure, we managed to reduce the jitter in the end points, and we improved the available bandwidth. We also minimized the packet loss in each of the peers involved in the streaming scenarios.

The contribution of this work was twofold: (1) it provided a preliminary and new study for characterization of a cloud provider's network and answered the question on whether the cloud is suitable to support real-time multimedia streaming; (2) offered an overview on the usability of the cloud in a multi-tree streaming scenario as depicted with the architecture, and established the threshold on the QoS bounds in function of the topology of the streaming distribution network. Presuming the location of the peers is familiar, comprehending this behavior of the cloud is essential for the design of a robust P2P overlay. As a takeaway, the results proved feasibility for deployment of a streaming service over mixed P2P-cloud infrastructure, and validated the assumption for the cloud performance. Moreover it confirmed the possibility of the cloud to host value-added services, given that other cloud providers exhibit a similar solid ground to tune down QoS parameters on individual links and nodes in the network.

- *Elaborate a methodology for business model of the streaming service.*

One of the important characteristics of a P2P is the incentive model bound to the streaming system. This goal was dedicated to the understanding the effect of user behavior in a multi-tree P2P overlay, and suggested a business model idea based on monetary discount as incentive aimed for the P2P-Cloud multimedia streaming system service described in the first part. The exponential growth of online video traffic has raised the demand for better video quality. In a search for increasing revenues, several content providers have refined their flat-rate billing models with new premium offers for high definition (HD) video delivery. Deploying a P2P streaming service in a centrally controlled system opens as many possibilities for the streaming content providers as for the users. During the time of this research, we did not find any commercial streaming service that based the video QoS as a fundamental of their business model.

The study focused on providing formal representation of the benefit from the discount from the perspective of both the provider and the users in the service. To reach that goal we referred to related work and extended the utility functions to render the model of our system. This representation was inferred from the linear taxation studies that are derived from the economic studies in order to ensure budget balance within a service. In addition, a methodology was described on how to apply discount-based incentive mechanism in mixed P2P-cloud streaming services. In particular, we modeled an economic projection of the cost-profit tendency of a P2P-cloud service that tackles a tiered, quality of service (QoS) based billing model. With this we aimed to: (1) assess the designers of streaming services and the content providers with fundamental understanding of the billing model's

dependence to the user profiles and (2) offer insights and set the bounds for the provider's costs and profits when P2P system is deployed in the cloud. Performing scalability prediction analysis is a vital study for the robustness and the market evolution of the real-time streaming services. Furthermore we defined a method that estimates the number of cloud instances required to host the currently connected users in the multicast.

To promote innovative multimedia services, it is fundamental for a content provider to observe and examine the users' behavior based on the streaming service history. Gathering historical data for scientific purposes was difficult due to the closed nature of the commercial services that are relying on private streaming protocols. Furthermore, the lack of empirical data from content providers like Netflix, Zattoo, and Octoshape prevented the economic-related academic studies from performing accurate analysis using realistic traffic patterns. To circumvent this handicap we did theoretical evaluations based on heuristic parameters. We finally represented the impact of the three factors: users' forwarding capacity (high, low), cooperation level and user fractions on the system's welfare to the users' and provider's welfare. Our case study showed that a streaming system provider can establish or adapt his business model by applying the described bounds to achieve a good discount-revenue trade-off and promote the system to the users. In conclusion the evaluations diagnose benefit increase to the users and the provider. A billing model based on remuneration, encourages the users for increased cooperation and attracts more of them to join the system knowing they could save money if they cooperate.

The key contribution here was the general methodology for discount incentive modeling in P2P-cloud based streaming services, bringing out the idea of open, tiered, QoS-based billing. Moreover a comprehensive analysis was shown for different user profiles classified according to three varying degrees: forwarding bandwidth capacity (high, low), cooperation level and user fractions. The experimentations evaluated the economic impact of the billing model, represented through the users' welfare and the provider's profit. The results suggested the following: (1) a suitable price-discount trade-off encourages users' cooperation; (2) increased users' participation and cooperation, improves their own welfare; (3) this behavior causes insignificant impact to the provider's profit. This concept potentially decreases the dynamics in the swarm, by discouraging intermittent churn caused by ungraceful departures during the streaming session. The constraints can be instructive for content providers to achieve a balance among their own profit, the required cooperation and the maximum allowed discount. Finally these results hold a pivotal role for dynamic price shaping in similar billing models.

- *Present interfaces for managing QoS.*

Unlike the case of file share or video on demand, QoS is deterministic factor for the participants in a real-time streaming. Therefore finding a ways to ensure seamless

7.1. CONTRIBUTIONS

end-to-end quality has been a great challenge in the research for years. The consensus has always been that if the lower level does not provide QoS control, then it can not be expected from the higher level to provide QoS guarantees. In this research task, the intention was to find a potential solutions to control the QoS for the described streaming service and propose APIs that can be offered as a service to the content provider and the users (clients) of the service. We elaborated on the multicast comeback in the network by programmatically defining the constraints for enforcing the QoS rules defined by SDN application - directly into the networking stack.

After considering the possibilities that the emerging SDN technology was offering at the time, we proposed QoS control algorithm based on SDN ability to provide primitives for shaping the bandwidth intervals in certain traffic types. This algorithm was applied over P2P multicast running on the top of the network with QoS parameters injected by an SDN application. This contribution complemented the studies related to the previously described P2P-cloud streaming service, where the QoS levels define the overlay topology and the billing model together. Following the algorithm, a PoC was implement as a test-case for QoS control as a contribution in the KIARA project. We demonstrated the novelty in controlling a set of QoS parameters for real-time communication using SDN controller with OpenFlow-enabled capabilities and APIs. The role of the QoS component in KIARA was to define APIs that enable end-to-end QoS provision between two endpoints using KIARA client/server and/or KIARA endpoint and a web server. This SDN plugin was dedicated to provide the following set of parameters/actions:

- QoS parameters (bandwidth, latency, jitter and packet loss)
- Well arranged P2P multicast overlay
- Optimization of the datapath through the network
- Topology Information i.e., the type of the underlying network

The contributions of this work boils down to: (1) implemented prototype application for adjustment and retrieval of aggregated QoS; (2) tested the application in SDN experimental setup with OpenFlow compatible switch, using Mininet network emulator and Ryu controller. In these experiments we validated the feasibility of the the idea for QoS provisioning on network level. (3) extrapolated the idea to SDN management of QoS for online streaming services deployed over proprietary residential networks; (4) proposed APIs in the context of the described use-case service.

- *Implement a mechanism that validates end-to-end service delivery and video quality adjustment within and among larger cloud DCs.*

The final part of the thesis advanced the research in the direction of using SDN for exploring augmented functionality for applications and services in the network. The idea was to abide

to the focus of video services and quality of service adjustment as the main objective through the PhD research cycle. The proposal that came out of this work is a novel traffic steering solution based on Netfloc, an SDK for SDN management that was developed within the frames of the research projects described in Section 6. Within the vision of SDN and NFV, the goal of traffic steering is to enable the deployment of service function chains composed of several virtual network functions, each providing complementary functionality to the overall service traffic. From a standpoint of this thesis the most far-fetching contribution is the service chain PoC related to video traffic classification and transcoding delivered to end users in the network. Besides Netfloc, the essential components to this functionality are the virtual Traffic Classifier, virtual Media Transcoder among others.

For proof case of the implementation, several coexisting service chains were deployed in a realistic cloud-NFV scenario. The one dedicated to video traffic consisted of the vTC and the vMT as VNFs placed along the traffic chain. The vTC is a common starting VNF for all chains because it was used to classify different traffic types, in this case, segregate the UDP video from the rest. The separated video traffic is then steered to the next VNF in the chain - vMT for transcoding and quality shaping. The overall experimental results included performance comparison of the following QoS: throughput, packet loss, latency and jitter for all chain services. For the video traffic in particular, we evaluated the SFC effect to end user application by measuring the video quality signal transmitted between the source and the destination endpoints and afterwards, comparing the original to the video sequence stored at the destination using recognized video quality assessment method. The conclusions stated satisfactory video quality in the case of lower throughput rates, confirming acceptable impact from the service chain and the vMT traffic processing to the video quality degradation. Scalability tests were performed additionally, by replicating the scenarios to a total of 4 and 6 chains running in parallel, showing almost zero difference to the quality impact caused by the singular chain scenarios.

These findings interpreted in the context of the original streaming architecture of this thesis involving user groups and cloud instances, confirm the feasibility of such deployments, relying on SDN and NFV as technologies with great potential to support advanced and isolated services dedicated to particular user groups. For example, a streaming service can be deployed over infrastructure similar as the described, composed of multiple PoP (ISPs networks), where the traffic from users of different quality groups can be streamed over segregated path composed of service chains and VNFs that intervene in the required traffic shaping (differentiation, transcoding, quality adjustment etc.). Virtual load balancers could also be considered by a content provider, in order to control flash crowd situations and divide the load between different video instances placed in the cloud. Moreover it is worth mentioning that the security related issues existing in classical P2P networks, could be easily eliminated in such environment by simply including VNFs dedicated to intrusion

7.2. FUTURE RESEARCH

detection and firewall at the edge of network, mitigating in a transparent way the malicious traffic. Finally to provide a support for improved video quality perception in a service chain environment, one option is to deploy services over more powerful fabrics, including switches and machines with lower impact to the performance. An actual example are acceleration frameworks such as Intel's DPDK in the current architecture. Of course emulating an industry-scale service solution in SDN-NFV environment is still a challenging and costly task within the academic ambit. The process of extrapolating individual and incremental solutions to real networks however in an ongoing and cutting edge progress that will eventually set its ground based on the research contributions in the area.

As a summary, the contributions of this work are: (1) design of OpenFlow-based SFC mechanism and its implementation in PoC, targeting effective and robust end-to-end traffic delivery for cloud-NFV services; (2) integral service chain open source reference solution, from design to implementation, deployment and testing; (3) extensive testing on ICMP/TCP/UDP and video traffic patterns in multiple chain scenarios as a characterization study that validated the correctness, effectiveness and scalability of the proposed solution in a real-life scenario. This work complements the SDN-based QoS model delivered in the previous contribution, by employing an alternative (SDN-NFV) approach for video quality adaptation. As discussed above, the delivered PoC is compliant with a streaming scenario build over P2P-cloud infrastructure, or even in a hybrid solution that combines the best of both worlds to facilitate application level QoS control on the top of SDN-based infrastructure assisted by specialized NFV components and OpenFlow primitives.

7.2 Future research

Possible advancements in the work of this thesis can be classified in several dimensions, depending on the technology and the end-user application. Hence, we identify the following future initiatives as prospective research topics:

- *Leveraging P2P concepts for value-added services in 5G-NFV-SDN scenarios:* This thesis has set the stone for some of the most prominent use-cases in SDN and P2P together. P2P is already a mature technology, with a huge exploitation success in RTC services. The challenge at current stage is to apply the P2P concepts on deploying seamless overlay service in managed-SDN networks. The possibility to control the entire networking stack brought by SDN would allow replication of the same streaming services, or creation of new ones, this time with improved and controlled quality of service. SDN would allow moreover research related to fostering security, improved routing, and combining multiple features in order to allow composition of services in straightforward manner. One way to enforce this is to allow a fine-granular network adaptation and control on a lower level by outsourcing

to the SDN controller the application multicast routing issues, while assigning the overlay topology management and the business model to the P2P client and third party applications. A follow-up research in this line could be inspired from our latest know-how [Blanco 2017, Giannoulakis 2016, Fajardo 2016, P. S Khodashenas 2017b]. Current and future trends will be also based on the adapting the streaming solutions to the NFV concept by using P2P and SDN technologies to achieve increased revenues and improved QoS and QoE, as it was discussed through the example research works in Section 2.3.1.

- *Service optimization, orchestration, composition and chaining:* There are several problems that should be resolved on a network level to provide automatic end-to-end service delivery across multiple provider networks. Some of those relate to the cumbersome architecture of the underlying stack, which coupled with software component running in mixed - virtual and bare-metal environment, create a complex and error-prone service. Each segment of such architecture introduces additional complexity and load in the system. The results however show an increased demand for such services and their rapid adoption by the telco and the service providers [P. S. Khodashenas 2017a].
- *In-time analysis on protocols and solutions:* Currently it is very important to align all the new technologies (versioning, components, etc.) with the new service developments to test the benefits and quantify the advantages of the offered solutions. As it was described in the thesis, a proliferation of SDN-based research exist and different approaches to the same solution exist. Although these services are targeted for different customer profiles, the VNF services for instance, we should remember that continuous evaluation and comparison of approaches and solutions is required, in order to point out the advancements in the field. Currently there is a solid ground for these types of studies, as most of the tool and frameworks are in a mature state (OpenStack, OpenDaylight, etc.), hence a focus can be turned towards improving the services that were created as proof of concepts use case applications. A thorough research on business achievements (a part from the CAPEX/OPEX) in SDN/NFV/5G environment is an optional future stream of this thesis, in order to foster adoption of the newly created services by the industry sector and third-parties. Scalability is a parallel concept that has to be encountered especially in cross-provider and large telco 5G networks. Such an analysis and evaluation has been pioneered with the contribution described in Chapter 5, [Trajkovska 2017].

Bibliography

- [Aggarwal 2007] Vinay Aggarwal, Anja Feldmann and Christian Scheideler. *Can ISPS and P2P users cooperate for improved performance?* SIGCOMM Comput. Commun. Rev., vol. 37, no. 3, pages 29–40, 2007.
- [Agrawal 2015] Gaurav Agrawal. *Performance Measurement Architecture for SFC*. Internet-Draft draft-sfc-agv-performance-measurement-architecture-00, Internet Engineering Task Force, December 2015. Work in Progress.
- [Aka] Akamai. <http://akamai.com>. Accessed: 03-2017.
- [Akyildiz 2015] Ian F. Akyildiz, Pu Wang and Shih-Chun Lin. *SoftAir: A software defined networking architecture for 5G wireless systems*. Computer Networks, vol. 85, pages 1–18, 2015.
- [Altunbasak 2005] Hayriye Altunbasak, Sven Krasser, Henry L. Owen, Jochen Grimmering, Hans-Peter Huth and Joachim Sokol. *Securing Layer 2 in Local Area Networks*. In Proceedings of the 4th International Conference on Networking - Volume Part II, ICN’05, pages 699–706, Berlin, Heidelberg, 2005. Springer-Verlag.
- [Anwer 2015] Bilal Anwer, Theophilus Benson, Nick Feamster and Dave Levin. *Programming Slick Network Functions*. In Proceedings of the 1st ACM SIGCOMM Symposium on Software Defined Networking Research, SOSR ’15, pages 14:1–14:13, New York, NY, USA, 2015. ACM.
- [Asaduzzaman 2008] Shah Asaduzzaman, Gregor Bochmann and Ying Qiao. *CliqueStream: An Efficient and Fault-Resilient Live Streaming Network on a Clustered Peer-to-Peer Overlay*. 2008 Eighth International Conference on Peer-to-Peer Computing (P2P), vol. 00, pages 269–278, 2008.
- [Banerjee 2002] Suman Banerjee, Bobby Bhattacharjee and Christopher Kommareddy. *Scalable Application Layer Multicast*. SIGCOMM Comput. Commun. Rev., vol. 32, no. 4, pages 205–217, August 2002.
- [Baset 2006] S. A. Baset and H. G. Schulzrinne. *An Analysis of the Skype Peer-to-Peer Internet Telephony Protocol*. In INFOCOM 2006. 25th IEEE International Conference on Computer Communications. Proceedings, pages 1–11, 2006.

- [Baun 2011] Christian Baun and Marcel Kunze. *The KOALA cloud management service: a modern approach for cloud infrastructure management*. In Proceedings of the First International Workshop on Cloud Computing Platforms, CloudCP 2011, pages 1:1–1:6, New York, NY, USA, 2011. ACM.
- [Bellavista 2015] Paolo Bellavista, Franco Callegati, Walter Cerroni, Chiara Contoli, Antonio Corradi, Luca Foschini, Alessandro Pernaflini and Giuliano Santandrea. *Virtual network function embedding in real cloud environments*. Computer Networks, vol. 93, pages 506–517, 2015.
- [Bernardini 2010] Riccardo Bernardini, Roberto Cesco Fabbro and Roberto Rinaldo. *Peer-to-peer Streaming Based on Network Coding Decreases Packet Jitter*. In Proceedings of the 2010 ACM Workshop on Advanced Video Streaming Techniques for Peer-to-peer Networks and Social Networking, AVSTP2P '10, pages 13–18, New York, NY, USA, 2010. ACM.
- [Bhamare 2016] Deval Bhamare, Raj Jain, Mohammed Samaka and Aiman Erbad. *A survey on service function chaining*. Journal of Network and Computer Applications, vol. 75, pages 138 – 155, 2016.
- [Bit a] *BitTorrent Protocol Client*. <http://www.bittorrent.com>. Accessed: 03-2017.
- [Bit b] *BitTorrent TV*. <https://btlive.tv>. Accessed: 03-2017.
- [Blanco 2017] Bego Blanco, Jose Oscar Fajardo, Ioannis Giannoulakis, Emmanouil Kafetzakis, Shuping Peng, Jordi PÃlrez-Romero, Irena Trajkovska, Pouria S. Khodashenas, Leonardo Goratti, Michele Paolino, Evangelos Sfakianakis, Fidel Liberal and George Xilouris. *Technology pillars in the architecture of future 5G mobile networks: NFV, {MEC} and {SDN}*. Computer Standards & Interfaces, pages –, 2017.
- [blo] *Techtarget Blog - Vint Cerf: At Google, OpenFlow now runs in all data center networks*. <http://searchsdn.techtarget.com/news/2240181909/Vint-Cerf-At-Google-OpenFlow-now-runs-in-all-data-center-networks>. Accessed: 03-2017.
- [Borcoci 2017] J. Bruneau-Queyreix; M. Lacaud; D. Ngru; J. M. Batalla; E. Borcoci. *MS-Stream: A multiple-source adaptive streaming solution enhancing consumer’s perceived quality*. In The 14th Annual IEEE Consumer Communications & Networking Conference, January 2017.
- [Botta 2012] Alessio Botta, Alberto Dainotti and Antonio Pescap. *A tool for the generation of realistic network workload for emerging networking scenarios*. Computer Networks, vol. 56, no. 15, pages 3531–3547, 2012.

BIBLIOGRAPHY

- [Braden 1994] R. Braden, D. Clark and S. Shenker. *Integrated Services in the Internet Architecture: an Overview*. RFC 1633 (Informational), June 1994.
- [Brogle 2010a] M. Brogle, A. Rüttimann and T. Braun. *Quality of Service for Overlay Multicast in Chord*. In The IEEE symposium on Computers and Communications, pages 734–737, June 2010.
- [Brogle 2010b] Marc Brogle, Sebastian Barthlomé and Torsten Braun. *Quality of Service for Multicasting Using NICE*. In Proceedings of the 2010 ACM Symposium on Applied Computing, SAC '10, pages 670–677, New York, NY, USA, 2010. ACM.
- [Bueno 2013] I. Bueno, J.I. Aznar, E. Escalona, J. Ferrer and J. Antoni Garcia-Espin. *An OpenNaaS Based SDN Framework for Dynamic QoS Control*. In Future Networks and Services, 2013 IEEE SDN for, pages 1–7, Nov 2013.
- [Callegati 2015] F. Callegati, W. Cerroni, C. Contoli and G. Santandrea. *Implementing dynamic chaining of Virtual Network Functions in OpenStack platform*. In Transparent Optical Networks (ICTON), 2015 17th International Conference on, pages 1–4, July 2015.
- [Carlini 2010] Emanuele Carlini, Massimo Coppola and Laura Ricci. *Integration of P2P and clouds to support massively multiuser virtual environments*. In Proceedings of the 9th Annual Workshop on Network and Systems Support for Games, NetGames 2010, pages 17:1–17:6, Piscataway, NJ, USA, 2010. IEEE Press.
- [Castro 2002] M. Castro, P. Druschel, A.-M. Kermarrec and A. I T Rowstron. *Scribe: a large-scale and decentralized application-level multicast infrastructure*. Selected Areas in Communications, IEEE Journal on, vol. 20, no. 8, pages 1489–1499, 2002.
- [Castro 2003a] Miguel Castro, Peter Druschel, Anne-Marie Kermarrec, Animesh Nandi, Antony Rowstron and Atul Singh. *SplitStream: high-bandwidth multicast in cooperative environments*. In SOSR '03: Proceedings of the nineteenth ACM symposium on Operating systems principles, pages 298–313, New York, NY, USA, 2003. ACM.
- [Castro 2003b] Miguel Castro, Peter Druschel, Anne-Marie Kermarrec, Animesh Nandi, Antony Rowstron and Atul Singh. *SplitStream: high-bandwidth multicast in cooperative environments*. SIGOPS Oper. Syst. Rev., vol. 37, no. 5, pages 298–313, October 2003.
- [Cerviño 2011] J Cerviño, P Rodríguez, I Trajkovska, A Mozo and J Salvachúa. *Testing a Cloud Provider Network for Hybrid P2P and Cloud Streaming Architectures*. In IEEE CLOUD 2011, The 4th International Conference on Cloud Computing, Washington, 2011.
- [Cervino 2011a] J. Cervino, F. Escribano, P. Rodriguez, I. Trajkovska and J. Salvachua. *Videoconference Capacity Leasing on Hybrid Clouds*. In Proceedings of the IEEE International Conference on Cloud Computing (CLOUD), pages 340–347, July 2011.

- [Cervino 2011b] J. Cervino, P. Rodriguez, I. Trajkovska, A. Mozo and J. Salvachua. *Testing a Cloud Provider Network for Hybrid P2P and Cloud Streaming Architectures*. In Cloud Computing (CLOUD), 2011 IEEE International Conference on, pages 356–363, july 2011.
- [Cervino 2011c] J. Cervino, P. Rodriguez, I. Trajkovska, A. Mozo and J. Salvachua. *Testing a Cloud Provider Network for Hybrid P2P and Cloud Streaming Architectures*. In Proceedings of the IEEE International Conference on Cloud Computing (CLOUD), pages 356–363, July 2011.
- [Cervino 2011d] J. Cervino, P. Rodriguez, I. Trajkovska, A. Mozo and J. Salvachua. *Testing a Cloud Provider Network for Hybrid P2P and Cloud Streaming Architectures*. In Cloud Computing (CLOUD), 2011 IEEE International Conference on, pages 356–363, 2011.
- [Cerviño 2013] Javier Cerviño, Pedro Rodríguez, Irena Trajkovska, Fernando Escribano and Joaquín Salvachúa. *A cost-effective methodology applied to videoconference services over hybrid clouds*. Mobile Networks and Applications, vol. 18, no. 1, pages 103–109, 2013.
- [Challa] Madhu Challa. *OpenVswitch Performance measurements & analysis*. openvswitch.org/support/ovscon2014/18/1600-ovs_perf.pptx. Accessed: 10-2016.
- [Chan 2009] W.K. Chan, Lijun Mei and Zhenyu Zhang. *Modeling and testing of cloud applications*. In Services Computing Conference, 2009. APSCC 2009. IEEE Asia-Pacific, pages 111–118, 2009.
- [Chang 2009] Hyunseok Chang, Sugih Jamin and Wenjie Wang. *Live Streaming Performance of the Zattoo Network*. In Proc. of the 9th ACM SIGCOMM Conf. on Internet Measurement Conference, IMC '09, pages 417–429, New York, NY, USA, 2009. ACM.
- [Chang 2012] Hong-Yi Chang, Ya-Yueh Shih and Yuan-Wei Lin. *CloudPP: A novel cloud-based P2P live video streaming platform with SVC technology*. In 2012 8th International Conference on Computing Technology and Information Management (NCM and ICNIT), volume 1, pages 64–68, April 2012.
- [Chu 2000a] Yang-hua Chu, Sanjay G. Rao and Hui Zhang. *A Case for End System Multicast (Keynote Address)*. SIGMETRICS Perform. Eval. Rev., vol. 28, no. 1, pages 1–12, June 2000.
- [Chu 2000b] Yang-hua Chu, Sanjay G. Rao and Hui Zhang. *A case for end system multicast (keynote address)*. In SIGMETRICS '00: Proceedings of the 2000 ACM SIGMETRICS

BIBLIOGRAPHY

- international conference on Measurement and modeling of computer systems, pages 1–12, New York, NY, USA, 2000. ACM.
- [Chu 2004] Yang-hua Chu, John Chuang and Hui Zhang. *A case for taxation in peer-to-peer streaming broadcast*. In Proceedings of the ACM SIGCOMM workshop on Practice and theory of incentives in networked systems, PINS '04, pages 205–212, New York, NY, USA, 2004. ACM.
- [Chun 2003] Brent Chun, David Culler, Timothy Roscoe, Andy Bavier, Larry Peterson, Mike Wawrzoniak and Mic Bowman. *PlanetLab: An Overlay Testbed for Broad-Coverage Services*. ACM SIGCOMM Computer Communication Review, vol. 33, no. 3, pages 00–00, July 2003.
- [Civanlar 2010] S. Civanlar, M. Parlakisik, A. M. Tekalp, B. Gorkemli, B. Kaytaz and E. Onem. *A QoS-enabled OpenFlow environment for Scalable Video streaming*. In 2010 IEEE Globecom Workshops, pages 351–356, Dec 2010.
- [clo] *Amazon CloudFront*. <http://aws.amazon.com/cloudfront>. Accessed: 03-2017.
- [Consortium 2012] Concord Consortium. *FI-PPP Program*. <http://www.fi-ppp.eu/about/>, 2012. Accessed: 2013-06-24.
- [Csoma 2014] Attila Csoma, Balázs Sonkoly, Levente Csikor, Felician Németh, András Gulyás, Wouter Tavernier and Sahel Sahhaf. *ESCAPE: extensible service chain prototyping environment using mininet, click, NETCONF and POX*. In ACM SIGCOMM 2014 Conference, SIGCOMM'14, Chicago, IL, USA, August 17-22, 2014, pages 125–126, 2014.
- [Dagher 2008] Rabih Dagher, Cristian Gadea, Bogdan Ionescu, Dan Ionescu and Robin Tropper. *A SIP Based P2P Architecture for Social Networking Multimedia*. In Proceedings of the 2008 12th IEEE/ACM International Symposium on Distributed Simulation and Real-Time Applications, DS-RT '08, pages 187–193, Washington, DC, USA, 2008. IEEE Computer Society.
- [dds] *OMG Standard for RTI-DDS*. <https://www.rti.com/products>. Accessed: 03-2017.
- [Deelman 2008] Ewa Deelman, Gurmeet Singh, Miron Livny, Bruce Berriman and John Good. *The cost of doing science on the cloud: the Montage example*. In Proceedings of the 2008 ACM/IEEE conference on Supercomputing, SC '08, pages 50:1–50:12, Piscataway, NJ, USA, 2008. IEEE Press.

- [Deering 1990] Stephen E. Deering and David R. Cheriton. *Multicast Routing in Datagram Internetworks and Extended LANs*. ACM Trans. Comput. Syst., vol. 8, no. 2, pages 85–110, May 1990.
- [del a] *T-Nova Deliverable 4.31: SDK for SDN - Interim*. <http://cordis.europa.eu/docs/projects/cnect/0/619520/080/deliverables/001-TNOVAD431SDKforSDNInterimv10Ares20162347437.pdf>. Accessed: 01-2017.
- [del b] *T-Nova Deliverable 4.32: SDK for SDN*. <http://cordis.europa.eu/docs/projects/cnect/0/619520/080/deliverables/001-TNOVAD432SDKforSDNv10Ares20162347437.pdf>. Accessed: 10-2016.
- [del c] *T-Nova Deliverable 7.1: Early Pilot Site Deployment*. <http://cordis.europa.eu/docs/projects/cnect/0/619520/080/deliverables/001-TNOVAD71EarlyPilotSiteDeploymentv10Ares20162344864.pdf>. Accessed: 01-2017.
- [den Bossche 2010] Ruben Van den Bossche, Kurt Vanmechelen and Jan Broeckhove. *Cost-Optimal Scheduling in Hybrid IaaS Clouds for Deadline Constrained Workloads*. Cloud Computing, IEEE International Conference on, pages 228–235, 2010.
- [Diot 2000] C. Diot, B. N. Levine, B. Lyles, H. Kassem and D. Balensiefen. *Deployment Issues for the IP Multicast Service and Architecture*. Netw. Mag. of Global Internetwkg., vol. 14, no. 1, pages 78–88, January 2000.
- [Dong 2016] Meng Dong, Huanqiang Zeng, Jing Chen, Canhui Cai and Kai-Kuang Ma. *Multiple description video coding based on adaptive data reuse*. Journal of Visual Communication and Image Representation, vol. 38, pages 378 – 385, 2016.
- [dpd] *Intel Data Plane Development Kit - DPDK*, *howpublished = "http://dpdk.org/"*, *note = Accessed: 04-2016*.
- [Druschel 2001] Peter Druschel and Antony Rowstron. *PAST: A large-scale, persistent peer-to-peer storage utility*. In In HotOS VIII, pages 75–80, 2001.
- [ec2] *Amazon EC2*. <http://aws.amazon.com/ec2/>. Accessed: 03-2017.
- [Efthymiopoulou 2016] M. Efthymiopoulou, N. Efthymiopoulos and A. Christakidis. *Enabling QoS in peer to peer live streaming through dynamic bandwidth or playback rate control*. In NOMS 2016 - 2016 IEEE/IFIP Network Operations and Management Symposium, pages 672–678, April 2016.

BIBLIOGRAPHY

- [Egilmez 2011] H. E. Egilmez, B. Gorkemli, A. M. Tekalp and S. Civanlar. *Scalable video streaming over OpenFlow networks: An optimization framework for QoS routing*. In 2011 18th IEEE International Conference on Image Processing, pages 2241–2244, Sept 2011.
- [Egilmez 2012] H.E. Egilmez, S.T. Dane, K.T. Bagci and A.M. Tekalp. *OpenQoS: An OpenFlow controller design for multimedia delivery with end-to-end Quality of Service over Software-Defined Networks*. In Signal Information Processing Association Annual Summit and Conference (APSIPA ASC), 2012 Asia-Pacific, pages 1–8, 2012.
- [Enns 2011] Rob Enns, Martin Bjorklund, Andy Bierman and Jürgen Schönwälder. *Network Configuration Protocol (NETCONF)*. RFC 6241, June 2011.
- [eri] *Ericsson SFC*. https://wiki.opendaylight.org/images/8/89/Ericsson-Kumbhare_Joshi-OpenDaylight_Service_Function_Chaining.pdf. Accessed: 10-2016.
- [ets] *ETSI MANO NFV Architectural Framework*. http://www.etsi.org/deliver/etsi_gs/NFV-MAN/001_099/001/01.01.01_60/gs_nfv-man001v010101p.pdf. Accessed: 10-2016.
- [Fajardo 2016] Jose Oscar Fajardo, Fidel Liberal, Ioannis Giannoulakis, Emmanouil Kafetzakis, Vincenzo Pii, Irena Trajkovska, Thomas Michael Bohnert, Leonardo Goratti, Roberto Riggio, Javier Garcia Lloreda, Pouria Sayyad Khodashenas, Michele Paolino, Pavel Bliznakov, Jordi Perez-Romero, Claudio Meani, Ioannis Chochliouros and Maria Belesioti. *Introducing Mobile Edge Computing Capabilities through Distributed 5G Cloud Enabled Small Cells*. Mobile Networks and Applications, pages 1–11, 2016.
- [Ferretti 2010] S. Ferretti, V. Ghini, F. Panzieri, M. Pellegrini and E. Turrini. *QoS-Aware Clouds*. In Cloud Computing (CLOUD), 2010 IEEE 3rd International Conference on, pages 321–328, 2010.
- [ffm] *FFMPEG*. <https://www.ffmpeg.org/>. Accessed: 03-2016.
- [fiw] *FIWARE Project and Community*. <https://www.fiware.org>. Accessed: 03-2017.
- [Flo 2013] *QoS in Floodlight*. <http://www.openflowhub.org/display/floodlightcontroller/>, 2013. Accessed: 2013-08-24.
- [Fouquet 2009] Marc Fouquet, Heiko Niedermayer and Georg Carle. *Cloud Computing for the Masses*. In Proceedings of the 1st ACM Workshop on User-provided Networking: Challenges and Opportunities, U-NET '09, pages 31–36, New York, NY, USA, 2009. ACM.

- [Garfinkel 2007] Simson L. Garfinkel. *An Evaluation of Amazon's Grid Computing Services: EC2, S3 and SQS*. Rapport technique, Center for, 2007.
- [Ghaznavi 2016] Milad Ghaznavi, Nashid Shahriar, Reaz Ahmed and Raouf Boutaba. *Service Function Chaining Simplified*. CoRR, vol. abs/1601.00751, 2016.
- [Giannoulakis 2016] I. Giannoulakis, E. Kafetzakis, I. Trajkovska, P. S. Khodashenas, I. Chochliouros, C. Costa, I. Neokosmidis and P. Bliznakov. *The emergence of operator-neutral small cells as a strong case for cloud computing at the mobile edge*. Transactions on Emerging Telecommunications Technologies, pages n/a–n/a, 2016. ett.3077.
- [gnu] *Gnutella*. <http://rfc-gnutella.sourceforge.net/>. Accessed: 01-2017.
- [Goh 2013] Chin Yong Goh, Hui Shyong Yeo, Hyotaek Lim, Poo Kuan Hoong, Jay W. Y. Lim and Ian K.T Tan. *A comparative study of tree-based and mesh-based overlay P2P media streaming*. International Journal of Multimedia and Ubiquitous Engineering (IJMUE), vol. 8, no. 4, pages 97–106, July 2013.
- [Goyal 2010] Pankaj Goyal. *Enterprise Usability of Cloud Computing Environments: Issues and Challenges*. Enabling Technologies 2010, IEEE International Workshops on, pages 54–59, 2010.
- [Guha] Saikat Guha, Neil Daswani and Ravi Jain. *An Experimental Study of the Skype Peer-to-Peer VoIP System*. In IPTPS'06: The 5th International Workshop on Peer-to-Peer Systems. Microsoft Research.
- [H. Yang 2013] R. Hu H. Yang. *Taxation-based Neighbor Selection for Layered Peer-to-Peer Streaming*. Int. Journal of Multimedia and Ubiquitous Engineering, vol. 8, no. 2, 2013.
- [Hajjat 2010] Mohammad Hajjat, Xin Sun, Yu wei Eric Sung, David Maltz, Sanjay Rao, Kunwadee Sripanidkulchai and Mohit Tawarmalani. *Cloudward bound: Planning for beneficial migration of enterprise applications to the cloud*. In In Proceedings of 2010 SIGCOMM, 2010.
- [Han 2015] Bo Han, Vijay Gopalakrishnan, Lusheng Ji and Seungjoon Lee. *Network function virtualization: Challenges and opportunities for innovations*. IEEE Communications Magazine, vol. 53, no. 2, pages 90–97, 2015.
- [Harsh 2014] P Harsh, K Benz, I Trajkovska, A Edmonds, P Comi and T Bohnert. *A highly available generic billing architecture for heterogenous mobile cloud services*. In Proceedings of the International Conference on Grid Computing and Applications (GCA), page 1. The Steering Committee of The World Congress in Computer Science, Computer Engineering and Applied Computing (WorldComp), 2014.

BIBLIOGRAPHY

- [He 2014a] J. He, Y. Yang, X. Wang, Y. Zeng and C. Tang. *PeerSorter: Classifying Generic P2P Traffic in Real-Time*. In 2014 IEEE 17th International Conference on Computational Science and Engineering, pages 605–613, Dec 2014.
- [He 2014b] Jian He, Yonggang Wen, Jianwei Huang and Di Wu. *On the Cost-QoE Tradeoff for Cloud-Based Video Streaming Under Amazon EC2's Pricing Models*. IEEE Trans. Circuits Syst. Video Techn., vol. 24, no. 4, pages 669–680, 2014.
- [Hefeeda 2003] Mohamed Hefeeda, Ahsan Habib, Boyan Botev, Dongyan Xu and Bharat Bhargava. *PROMISE: Peer-to-Peer Media Streaming Using CollectCast*, 2003.
- [Herbaut 2016a] N. Herbaut, D. Negru, Y. Chen, P. A. Frangoudis and A. Ksentini. *Content Delivery Networks as a Virtual Network Function: A Win-Win ISP-CDN Collaboration*. In 2016 IEEE Global Communications Conference (GLOBECOM), pages 1–6, Dec 2016.
- [Herbaut 2016b] N. Herbaut, D. Negru, D. Magoni and P. A. Frangoudis. *Deploying a content delivery service function chain on an SDN-NFV operator infrastructure*. In 2016 International Conference on Telecommunications and Multimedia (TEMU), pages 1–7, July 2016.
- [hua Chu 2003] Yang hua Chu, Aditya Ganjam, T. S. Eugene Ng, Sanjay G. Rao, Kunwadee Sripanidkulchai, Jibin Zhan and Hui Zhang. *Early Experience with an Internet Broadcast System Based on Overlay Multicast*, 2003.
- [Huang 2011] Z. Huang, C. Mei, L. E. Li and T. Woo. *CloudStream: Delivering high-quality streaming videos through a cloud-based SVC proxy*. In 2011 Proceedings IEEE INFOCOM, pages 201–205, April 2011.
- [hur] *Hurtle Orchestrator, ICCLab*. <https://github.com/icclab/hurtle>. Accessed: 1-2017.
- [icc a] *ICCLab 2013 Summer School on Cloud Computing*. <https://blog.zhaw.ch/icclab/icclab-2013-summer-school-on-cloud-computing/>. Accessed: 02-2017.
- [icc b] *ICCLab at the Zurich University of Applied Sciences*. <https://blog.zhaw.ch/icclab/>. Accessed: 03-2017.
- [icc c] *ICCLab WiNeMO Blog*. <https://blog.zhaw.ch/icclab/cost-ic0906-winemo-24-25-september-ohrid-macedonia/>. Accessed: 02-2017.
- [IETF] IETF. *The Addition of Explicit Congestion Notification (ECN) to IP*. <https://tools.ietf.org/html/rfc3168>. Accessed: 10-2016.

- [II 2015] Harold Owens II and Arjan Durresi. *Video over Software-Defined Networking (VSDN)*. Computer Networks, vol. 92, Part 2, pages 341 – 356, 2015. Software Defined Networks and Virtualization.
- [Imran 2007a] K. Imran, M. Mellia and M. Meo. *Measurements of Multicast Television over IP*. In Local Metropolitan Area Networks, 2007. LANMAN 2007. 15th IEEE Workshop on, pages 170–175, 2007.
- [Imran 2007b] Kashif Imran, Marco Mellia and Michela Meo. *Measurements of Multicast Television over IP*. 2007 15th IEEE Workshop on Local & Metropolitan Area Networks, no. October, pages 170–175, June 2007.
- [Institute 2011] China Mobile Research Institute. *C-RAN: The Road Towards Green RAN*. White paper. version 2.5., October 2011.
- [Isa] *Isabel - real time collaboration tool for the Internet*. <http://isabel.dit.upm.es/>. Accessed: 03-2017.
- [J. Cerviño 2011] P. Rodríguez I. Trajkovska J. Salvachúa J. Cerviño F. Escribano. *Leasing Videoconference Resources on Hybrid Clouds*. In Cloud Computing: Methodology, Systems, and Applications, chapter 10, pages 511–527. CRC Press, 6000 Broken Sound Parkway NW, Suite 300. Boca Raton, FL 33487., 2011.
- [Jain 2013] Sushant Jain, Alok Kumar, Subhasree Mandal, Joon Ong, Leon Poutievski, Arjun Singh, Subbaiah Venkata, Jim Wanderer, Junlan Zhou, Min Zhu, Jon Zolla, Urs Hölzle, Stephen Stuart and Amin Vahdat. *B4: Experience with a Globally-deployed Software Defined Wan*. SIGCOMM Comput. Commun. Rev., vol. 43, no. 4, pages 3–14, August 2013.
- [Jannotti 2000] John Jannotti, David K. Gifford, Kirk L. Johnson, M. Frans Kaashoek and James W. O’Toole Jr. *Overcast: reliable multicasting with on overlay network*. In OSDI’00: Proceedings of the 4th conference on Symposium on Operating System Design & Implementation, pages 14–14, Berkeley, CA, USA, 2000. USENIX Association.
- [Jeong 2012] Kwangtae Jeong, Jinwook Kim and Young-Tak Kim. *QoS-aware Network Operating System for software defined networking with Generalized OpenFlows*. In NOMS, pages 1167–1174. IEEE, 2012.
- [Jia Zhao 2013] Changqiao Xu Wei Su Jia Zhao Jianfeng Guan and Hongke Zhang. *Peering Strategic Game Models for Interdependent ISPs in Content Centric Internet*. The Scientific World Journal, vol. 2013, no. 380265, 2013.
- [Jin 2016] Xianshu Jin, Hwiyun Ju, Sungchol Cho, Boyeong Mun, Cheongbin Kim and Sunyoung Han. *QoS routing design for adaptive streaming in Software Defined Network*. In 2016

BIBLIOGRAPHY

- International Symposium on Intelligent Signal Processing and Communication Systems (ISPACS), pages 1–6, Oct 2016.
- [juj] *Canonical JuJu framework*. <https://jujucharms.com/q/oai>. Accessed: 02-2017.
- [Kamvar 2003] Sepandar D. Kamvar, Mario T. Schlosser and Hector Garcia-Molina. *The Eigentrust algorithm for reputation management in P2P networks*. In Proceedings of the 12th international conference on World Wide Web, WWW '03, pages 640–651, New York, NY, USA, 2003. ACM.
- [Karakus 2017] Murat Karakus and Arjan Durresi. *Quality of Service (QoS) in Software Defined Networking (SDN): A survey*. Journal of Network and Computer Applications, vol. 80, pages 200 – 218, 2017.
- [Karlsson 1996] G. Karlsson. *Asynchronous transfer of video*. Communications Magazine, IEEE, vol. 34, no. 8, pages 118 –126, August 1996.
- [Kawashima 2013] Ryota Kawashima and Hiroshi Matsuo. *Non-tunneling Edge-Overlay Model Using OpenFlow for Cloud Datacenter Networks*. In CloudCom (2), volume 2, pages 176–181, Bristol, UK, 2013. IEEE Computer Society.
- [kia a] *Kiara Advanced Middleware*. <https://catalogue.fiware.org/enablers/kiara-advanced-middleware>. Accessed: 02-2017.
- [kia b] *KIARA InfiniBand Functionality Overview*. <https://blog.zhaw.ch/icclab/kiara-infiniband-functionality-overview/>. Accessed: 02-2017.
- [kia c] *Kiara transport - ktransport*. <https://github.com/icclab/kiara-transport>. Accessed: 02-2017.
- [Kim 2010] Hyunjoo Kim, Yaakoub el Khamra, Shantenu Jha and Manish Parashar. *Exploring application and infrastructure adaptation on hybrid grid-cloud infrastructure*. In Proceedings of the 19th ACM International Symposium on High Performance Distributed Computing, HPDC 2010, pages 402–412, New York, NY, USA, 2010. ACM.
- [Kim 2016] E. D. Kim, S. Lee, Y. Choi, M. K. Shin and H. J. Kim. *Evaluations of a multiple flow tables construction scheme for service function chaining*. In 2016 International Conference on Information Networking (ICOIN), pages 424–426, Jan 2016.
- [Klepec 2001] B. Klepec and a. Kos. *Performance of VoIP applications in a simple differentiated services network architecture*. EUROCON'2001. International Conference on Trends in Communications. Technical Program, Proceedings (Cat. No.01EX439), pages 214–217, 2001.

- [Kostic 2003] D. Kostic, A. Rodriguez, J. Albrecht and A. Vahdat. *Bullet: high bandwidth data dissemination using an overlay mesh*. In Proceedings of the 19th ACM Symposium on Operating System Principles (SOSP), 2003. Dept. of Comput. Sci., Duke Univ., Durham, NC, USA.
- [Kourtis 2015] M. A. Kourtis, G. Xilouris, V. Riccobene, M. J. McGrath, G. Petralia, H. Koumaras, G. Gardikis and F. Liberal. *Enhancing VNF performance by exploiting SR-IOV and DPDK packet processing acceleration*. In 2015 IEEE Conference on Network Function Virtualization and Software Defined Network (NFV-SDN), pages 74–78, San Francisco, CA, USA, Nov 2015.
- [Kurochkin 2016] Ilya Kurochkin and Anatoliy Saeveskiy. *{BOINC} Forks, Issues and Directions of Development I*. Procedia Computer Science, vol. 101, pages 369 – 378, 2016. 5th International Young Scientist Conference on Computational Science, {YSC} 2016, 26-28 October 2016, Krakow, Poland.
- [Leitao 2007] J. Leitao, J. Pereira and L. Rodrigues. *Epidemic Broadcast Trees*. In 2007 26th IEEE International Symposium on Reliable Distributed Systems (SRDS 2007), pages 301–310, Oct 2007.
- [Li 2005] J. Li, Chou PA and Zhang C. *Mutualcast: an efficient mechanism for content distribution in a P2P network*. In Acm Sigcomm Asia Workshop, Beijing, China. Apr. 10-12, 2005.
- [Li 2007] Bo Li and Hao Yin. *Peer-to-peer live video streaming on the internet: issues, existing approaches, and challenges [Peer-to-Peer Multimedia Streaming]*. Comm. Mag., vol. 45, no. 6, pages 94–99, June 2007.
- [Li 2009] Jim Li, John Chinneck, Murray Woodside, Marin Litoiu and Gabriel Iszlai. *Performance model driven QoS guarantees and optimization in clouds*. In Proceedings of the 2009 ICSE Workshop on Software Engineering Challenges of Cloud Computing, CLOUD 2009, pages 15–22, Washington, DC, USA, 2009. IEEE Computer Society.
- [Li 2010] Ang Li, Xiaowei Yang, Srikanth Kandula and Ming Zhang. *CloudCmp: comparing public cloud providers*. In Proceedings of the 10th annual conference on Internet measurement, IMC '10, pages 1–14. ACM, 2010.
- [Li 2011] Zhenyu Li, Rong Gu and Gaogang Xie. *Measuring and Enhancing the Social Connectivity of UGC Video Systems: A Case Study of YouKu*. In Proc. of the 9th Int. Workshop on Quality of Service, IWQoS '11, pages 36:1–36:9, Piscataway, NJ, USA, 2011. IEEE Press.

BIBLIOGRAPHY

- [Liang 2005] Jin Liang, Steven Y. Ko, Indranil Gupta and Klara Nahrstedt. *MON: On-demand Overlays for Distributed System Management*. In Proceedings of the 2Nd Conference on Real, Large Distributed Systems - Volume 2, WORLDS'05, pages 13–18, Berkeley, CA, USA, 2005. USENIX Association.
- [Lin 2009] W. Sabrina Lin, H. Vicky Zhao and K. J. Ray Liu. *Incentive cooperation strategies for peer-to-peer live multimedia streaming social networks*. Trans. Multi., vol. 11, no. 3, pages 396–412, April 2009.
- [Lin 2015] Ying-Dar Lin, Po-Ching Lin, Chih-Hung Yeh, Yao-Chun Wang and Yuan-Cheng Lai. *An extended SDN architecture for network function virtualization with a case study on intrusion prevention*. IEEE Network, vol. 29, no. 3, pages 48–53, 2015.
- [Liu 2007a] Jiangchuan Liu, Sanjay G. Rao, Bo Li and Hui Zhang. *Opportunities and Challenges of Peer-to-Peer Internet Video Broadcast*. In In (invited) Proceedings of the IEEE, Special Issue on Recent Advances in Distributed Multimedia Communications, 2007.
- [Liu 2007b] Zhengye Liu, Yanming Shen, Shivendra S. Panwar, Keith W. Ross and Yao Wang. *Using layered video to provide incentives in P2P live streaming*. In Proc. of the 2007 workshop on Peer-to-peer streaming and IP-TV, P2P-TV '07, pages 311–316, New York, NY, USA, 2007. ACM.
- [Liu 2008] Yong Liu, Yang Guo and Chao Liang. *A survey on peer-to-peer video streaming systems*. volume 1, pages 18–28, March 2008.
- [Lua 2005] Eng Keong Lua, J. Crowcroft, M. Pias, R. Sharma and S. Lim. *A survey and comparison of peer-to-peer overlay network schemes*. IEEE Communications Surveys Tutorials, vol. 7, no. 2, pages 72–93, Second 2005.
- [López-Fuentes 2015] F.A. López-Fuentes. *{P2P} Video Streaming Strategies based on Scalable Video Coding*. Journal of Applied Research and Technology, vol. 13, no. 1, pages 113 – 124, 2015.
- [Ma 2013] Richard T.B. Ma, John C.S. Lui and Vishal Misra. *On the Evolution of the Internet Economic Ecosystem*. In Proc. of the 22Nd International Conference on World Wide Web, WWW '13, pages 849–860, Republic and Canton of Geneva, Switzerland, 2013. International World Wide Web Conferences Steering Committee.
- [Magharei 2006] Nazanin Magharei and Reza Rejaie. *Understanding mesh-based peer-to-peer streaming*. In NOSSDAV '06: Proceedings of the 2006 international workshop on Network and operating systems support for digital audio and video, pages 1–6, New York, NY, USA, 2006. ACM.

- [Magharei 2007] Nazanin Magharei and Reza Rejaie. *Mesh or multiple-tree: A comparative study of live p2p streaming approaches*. In INFOCOM 2007, Anchorage, Alaska, 6-12, pages 1424–1432, 2007.
- [Magharei 2009] N. Magharei and R. Rejaie. *PRIME: Peer-to-Peer Receiver-Driven Mesh-Based Streaming*. IEEE/ACM Transactions on Networking, vol. 17, no. 4, pages 1052–1065, Aug 2009.
- [Maheswari 2016] B. U. Maheswari and T. S. B. Sudarshan. *Reputation based mesh-tree-mesh cluster hybrid architecture for P2P live streaming*. In 2016 3rd International Conference on Devices, Circuits and Systems (ICDCS), pages 240–243, March 2016.
- [Manjunatha 2010] A. Manjunatha, A. Ranabahu, A. Sheth and K. Thirunarayan. *Power of Clouds in Your Pocket: An Efficient Approach for Cloud Mobile Hybrid Application Development*. In Cloud Computing Technology and Science (CloudCom), 2010 IEEE Second International Conference on, pages 496–503, 30 2010-dec. 3 2010.
- [Maymounkov 2002] Petar Maymounkov and David Mazières. *Kademlia: A Peer-to-Peer Information System Based on the XOR Metric*. In Revised Papers from the First International Workshop on Peer-to-Peer Systems, IPTPS '01, pages 53–65, London, UK, UK, 2002. Springer-Verlag.
- [McKeown 2008] Nick McKeown, Tom Anderson, Hari Balakrishnan, Guru Parulkar, Larry Peterson, Jennifer Rexford, Scott Shenker and Jonathan Turner. *OpenFlow: enabling innovation in campus networks*. SIGCOMM Comput. Commun. Rev., vol. 38, no. 2, pages 69–74, March 2008.
- [McLellan 2010] Alastair McLellan. *White paper. Conflicting messages from the hint at growing resistance*. The Health service journal, vol. 120, no. 6224, page 3, September 2010.
- [Mei 2010] Yiduo Mei, Ling Liu, Xing Pu and S. Sivathanu. *Performance Measurements and Analysis of Network I/O Applications in Virtualized Cloud*. In Cloud Computing (CLOUD), 2010 IEEE 3rd International Conference on, pages 59–66, 2010.
- [min] *Mininet - virtual network emulator*. <http://mininet.org>. Accessed: 03-2017.
- [Moshref 2010] M. Moshref, R. Motamedi, H. R. Rabiee and M. Khansari. *LayeredCast - a hybrid Peer-to-Peer live layered video streaming protocol*. In 2010 5th International Symposium on Telecommunications, pages 663–668, Dec 2010.
- [mss] *Multiple-Source Streaming over HTTP*. <http://msstream.viotech.net/>. Accessed: 04-2017.

BIBLIOGRAPHY

- [Mushtaq 2006] Mubashar Mushtaq, Toufik Ahmed and Djamal-Eddine Meddour. *Adaptive packet video streaming over P2P networks*. In Proceedings of the 1st international conference on Scalable information systems, InfoScale '06, New York, NY, USA, 2006. ACM.
- [Nadeau 2015] Thomas Nadeau and Paul Quinn. *Problem Statement for Service Function Chaining*. RFC 7498, April 2015.
- [nap] *Napster*. <http://gb.napster.com/>. Accessed: 01-2017.
- [ndp] *ntop: nDPI - Open and Extensible LGPLv3 Deep Packet Inspection Library*. <http://www.ntop.org/products/deep-packet-inspection/ndpi>. Accessed: 04-2016.
- [NEC] NEC. *VTN Open Flow*. https://wiki.opendaylight.org/images/d/da/NEC_VTN_Demo_0722.pdf. Accessed: 04-2016.
- [net a] *Netflix current pricing model*. <http://www.digitaltrends.com/movies/netflix-cost-pricing-plan-breakdown/>. Accessed: 03-2017.
- [net b] *Netflix tiered pricing model introduced in 2014*. <http://gigaom.com/2014/01/22/netflix-wants-to-introduce-three-pricing-tiers-for-new-members-in-coming-months>. Accessed: 03-2017.
- [net c] *Netfloc - NETWORK FLOws for your Cloud datacenters*. <http://icclab.github.io/netfloc>. Accessed: 03-2017.
- [net d] *Netfloc Patterns*. http://icclab.github.io/netfloc/docs/netfloc_api_spec/netfloc.html#action_update_flow_pattern_by_id_examples. Accessed: 03-2016.
- [net e] *OpenStack Heat plugin for Netfloc*. <https://github.com/icclab/netfloc-heat>. Accessed: 10-2016.
- [neu] *OpenStack Neutron and OpenFlow*. <http://dischord.org/2015/03/09/troubleshooting-openstack-neutron-networking-part-one>. Accessed: 03-2016.
- [NFV 2014] ETSI Industry Specification Group (ISG) NFV. *Network Functions Virtualisation (NFV) Architectural Framework*. Rapport technique V1.2.1, ETSI GS NFV 002, December 2014.
- [Ngan 2004] Tsuen-Wan Johnny Ngan, Dan S. Wallach and Peter Druschel. *Incentives-Compatible Peer-to-Peer Multicast*, 2004.

BIBLIOGRAPHY

- [Nic] *Nicira Extension Structures*. http://ryu.readthedocs.io/en/latest/nicira_ext_ref.html. Accessed: 03-2017.
- [Niu 2012] Di Niu, Chen Feng and Baochun Li. *A theory of cloud bandwidth pricing for video-on-demand providers*. In INFOCOM, 2012 Proceedings IEEE, pages 711–719, March 2012.
- [nsh] *IETF Data Tracker for Network Service Header (NSH)*. <https://datatracker.ietf.org/doc/draft-ietf-sfc-nsh/>. Accessed: 03-2017.
- [ntop] ntop. *ntop: PFRING - High-speed packet capture, filtering and analysis*. http://www.ntop.org/products/packet-capture/pf_ring. Accessed: 04-2016.
- [odl a] *OpenDaylight Controller Github repository*. <https://github.com/.opendaylight>. Accessed: 10-2016.
- [odl b] *OpenDaylight: Open Source SDN Platform*. <https://www.opendaylight.org>. Accessed: 03-2016.
- [OF-] *OpenFlow-config–OpenFlow Management and Configuration Protocol*. <https://www.opennetworking.org/images/stories/downloads/sdn-resources/onf-specifications/openflow-config/of-config-1.2.pdf>. Accessed: 03-2017.
- [OF] *Open Networking Foundation (ONF) - OpenFlow*. <http://archive.openflow.org/>. Accessed: 03-2017.
- [ofs a] *OpenFlow Software Switch open-source implementation*. <https://github.com/CPqD/ofsoftswitch13>. Accessed: 03-2017.
- [OFs b] *OpenFlow Switch Specification v. 1.3.2*. <https://www.opennetworking.org/images/stories/downloads/sdn-resources/onf-specifications/openflow/openflow-spec-v1.3.2.pdf>. Accessed: 03-2017.
- [ono a] *ONOS-OpenStack*. <https://github.com/openstack/networking-onos>. Accessed: 10-2016.
- [ono b] *ONOS-OPNFV*. <https://wiki.opnfv.org/display/onosfw/ONOS+Framework+Home>. Accessed: 10-2016.
- [ono c] *ONOS Project*. <http://onosproject.org>. Accessed: 10-2016.
- [ono d] *ONOS-SFC*. <https://wiki.onosproject.org/pages/viewpage.action?pageId=4163192>. Accessed: 10-2016.

BIBLIOGRAPHY

- [ope a] *Noop firewall driver*. <https://gist.github.com/djoreilly/db9c2d32a473c6643551>. Accessed: 03-2016.
- [ope b] *OpenAirInterface*. <http://www.openairinterface.org/>. Accessed: 02-2017.
- [ope c] *OpenContrail*. <http://www.juniper.net/us/en/local/pdf/whitepapers/2000535-en.pdf>. Accessed: 10-2016.
- [ope d] *OpenContrail-SFC*. https://www.juniper.net/techpubs/en_US/contrail2.0/topics/task/configuration/service-chaining-vnc.html. Accessed: 10-2016.
- [ope e] *OpenStack networking details*. <https://www.rdoproject.org/networking/networking-in-too-much-detail/>. Accessed: 04-2016.
- [ope f] *OpenStack SFC*. <http://docs.openstack.org/developer/networking-sfc>. Accessed: 03-2016.
- [ope g] *OpenStack user's survey 2016*. <https://www.openstack.org/assets/survey/April-2016-User-Survey-Report.pdf>. Accessed: 04-2016.
- [ope h] *Overlay encapsulation in OpenStack DC network*. <https://gist.github.com/djoreilly/db9c2d32a473c6643551>. Accessed: 03-2016.
- [opn a] *OPNFV Open Platform for NFV Project*. <https://wiki.opnfv.org/>. Accessed: 04-2016.
- [opn b] *OPNFV Service Function Chaining*. <https://wiki.opnfv.org/display/sfc/Service+Function+Chaining+Home>. Accessed: 04-2016.
- [opn c] *OPNFV Brahmaputra SFC release*. <http://artifacts.opnfv.org/sfc/brahmaputra/docs/release-notes/index.html>. Accessed: 10-2016.
- [opn d] *OPNFV Colorado SFC release*. <http://artifacts.opnfv.org/sfc/colorado/docs/releasenotes/index.html>. Accessed: 10-2016.
- [OPNFV] *OPNFV. Yardstick project OPNFV*. <https://wiki.opnfv.org/display/yardstick/Yardstick>. Accessed: 04-2016.
- [organizations 2012] Various organizations and authors. *Network Functions Virtualisation: An Introduction, Benefits, Enablers, Challenges and Call for Action*. Introductory white paper, October 2012.
- [P. S. Khodashenas 2017a] I. Taboada M.A. Kourtis G. Xilouris I. Giannoulakis E. Jimeno I. Trajkovska J. O. Fajardo E. Kafetzakis J. Garcia F. Liberal A. Whitehead

- P. S. Khodashenas B. Blanco and M. Wilson. *Service Management and Orchestration over Multi-Tenant Cloud-Enabled RAN*. IEEE Transactions on Network and Service Management, 2017. [submitted].
- [P. S Khodashenas 2017b] J. G. Lloreda E. Jimeno I. Trajkovska L. D. Vecchio A. Whitehead P. S Khodashenas A. Betzler. *Ensuring Quality of Service in a Multi-Tenant Cloud-Enabled RAN Environment*. In European Conference on Networks and Communications (EUCNC), 2017. [submitted].
- [p2p a] *History of P2P*. <http://iml.jou.ufl.edu/projects/Fall02/Moody/history.html>. Accessed: 02-2017.
- [p2p b] *Taxonomy P2P Overlays*. <http://www.cs.uml.edu/~buford/irtf-p2prg/JBufordKRoss-IETF-Overlay-Systems-v4.pdf>. Accessed: 01-2017.
- [Padmanabhan 2002] Venkata N. Padmanabhan, Helen J. Wang, Philip A. Chou and Kunwadee Sripanidkulchai. *Distributing Streaming Media Content Using Cooperative Networking*. In Proceedings of the 12th International Workshop on Network and Operating Systems Support for Digital Audio and Video, NOSSDAV '02, pages 177–186, New York, NY, USA, 2002. ACM.
- [Pai 2005] Vinay S. Pai, Kapil Kumar, Karthik Tamilmani, Vinay Sambamurthy and Alexander E. Mohr. *Chainsaw: Eliminating Trees from Overlay Multicast*. In Peer-to-Peer Systems IV, 4th International Workshop, IPTPS 2005, Ithaca, NY, USA, February 24-25, 2005, Revised Selected Papers, pages 127–140, 2005.
- [Pathan 2014] Mukaddim Pathan, Ramesh K. Sitaraman and Dom Robinson. *Overlay networks: An akamai perspective*, pages 710–. Wiley-IEEE Press, 2014.
- [Payberah 2010] Amir H. Payberah, Fatemeh Rahimian, Seif Haridi and Jim Dowling. *Sepidar: Incentivized market-based P2P live-streaming on the Gradient overlay network*. In Proceedings - 2010 IEEE International Symposium on Multimedia, ISM 2010, pages 1–8, 2010. QC 20110517.
- [Payberah 2012] A. H. Payberah, H. Kavalionak, V. Kumaresan, A. Montresor and S. Haridi. *CLive: Cloud-assisted P2P live streaming*. In 2012 IEEE 12th International Conference on Peer-to-Peer Computing (P2P), pages 79–90, Sept 2012.
- [Peltotalo 2008] J. Peltotalo, J. Harju, A. Jantunen, M. Saukko, L. Vaatamoinen, I. Curcio, I. Bouazizi and M. Hannuksela. *Peer-to-Peer Streaming Technology Survey*. In Networking, 2008. ICN 2008. 7th Int. Conf. on, pages 342–350, April 2008.

BIBLIOGRAPHY

- [Pereira 2010] R. Pereira, M. Azambuja, K. Breitman and M. Endler. *An Architecture for Distributed High Performance Video Processing in the Cloud*. In Cloud Computing (CLOUD), 2010 IEEE 3rd International Conference on, pages 482–489, 2010.
- [Pianese 2007] F. Pianese, D. Perino, J. Keller and E. W. Biersack. *PULSE: An Adaptive, Incentive-Based, Unstructured P2P Live Streaming System*. Trans. Multi., vol. 9, no. 8, pages 1645–1660, December 2007.
- [Piatek 2007] Michael Piatek, Tomas Isdal, Thomas Anderson, Arvind Krishnamurthy and Arun Venkataramani. *Do Incentives Build Robustness in Bit Torrent*. In Proceedings of the 4th USENIX Conference on Networked Systems Design & Implementation, NSDI’07, pages 1–1, Berkeley, CA, USA, 2007. USENIX Association.
- [Piatek 2010] Michael Piatek, Arvind Krishnamurthy, Arun Venkataramani, Richard Yang, David Zhang and Alexander Jaffe. *Contracts: Practical Contribution Incentives for P2P Live Streaming*. In Proc. of the 7th USENIX Conference on Networked Systems Design and Implementation, NSDI’10, pages 6–6, Berkeley, CA, USA, 2010. USENIX Association.
- [ppl] *PPLive*. <http://www.pplive.com>. Accessed: 03-2017.
- [Quemada 2005] J. Quemada, T. de Miguel, S. Pavon, G. Huecas, T. Robles, J. Salvachua, D.A.A. Ortiz, V. Sirvent, F. Escribano and J. Sedano. *Isabel: an application for real time collaboration with a flexible floor control*. In Collaborative Computing: Networking, Applications and Worksharing, 2005 International Conference on, 0 2005.
- [Ratnasamy 2001] Sylvia Ratnasamy, Paul Francis, Mark Handley, Richard Karp and Scott Shenker. *A Scalable Content-addressable Network*. SIGCOMM Comput. Commun. Rev., vol. 31, no. 4, pages 161–172, August 2001.
- [res] *Amazon EC2 Reserved Instances*. <http://aws.amazon.com/ec2/reserved-instances>. Accessed: 03-2017.
- [Riggio 2016] Roberto Riggio, Imen Grida Ben Yahya, Tinku Rasheed and Steven Latre. *Orchestrating Virtual Network Functions with Scylla*. In IEEE/IFIP Network Operations and Management Symposium (NOMS), Istanbul, April 2016.
- [Rodrigues 2010] Rodrigo Rodrigues and Peter Druschel. *Peer-to-peer Systems*. Commun. ACM, vol. 53, no. 10, pages 72–82, October 2010.
- [Rodriguez 2009] Pedro Rodriguez, Daniel Gallego, Javier Cervino, Fernando Escribano, Juan Quemada and Joaquín Salvachúa. *VaaS: Videoconference as a service*. In CollaborateCom, pages 1–11, 2009.

BIBLIOGRAPHY

- [Rodríguez 2012] P. Rodríguez, J. Cerviño, I. Trajkovska and J. Salvachúa. *Advanced Videoconferencing Services Based on WebRTC*. In Proceedings of the IADIS Conference on Collaborative Technologies, July 2012.
- [Rosen 2001] E. Rosen, A. Viswanathan and R. Callon. *Multiprotocol Label Switching Architecture*. RFC 3031 (Proposed Standard), January 2001.
- [Rosenberg 2010] Jonathan Rosenberg. *Interactive Connectivity Establishment (ICE): A Protocol for Network Address Translator (NAT) Traversal for Offer/Answer Protocols*. RFC 5245, April 2010.
- [Rowstron 2001] Antony I. T. Rowstron and Peter Druschel. *Pastry: Scalable, Decentralized Object Location, and Routing for Large-Scale Peer-to-Peer Systems*. In Proceedings of the IFIP/ACM International Conference on Distributed Systems Platforms Heidelberg, Middleware '01, pages 329–350, London, UK, UK, 2001. Springer-Verlag.
- [rtc] *IETF rtcweb Working Group*. <https://tools.ietf.org/wg/rtcweb/>. Accessed: 03-2017.
- [ryu] *Ryu SDN controller*. <http://osrg.github.io/ryu/>. Accessed: 03-2017.
- [Schulzrinne 2003] H. Schulzrinne, S. Casner, R. Frederick and V. Jacobson. *RTP: A Transport Protocol for Real-Time Applications*. RFC 3550 (INTERNET STANDARD), July 2003. Updated by RFCs 5506, 5761, 6051, 6222, 7022, 7160, 7164.
- [SDN a] *SDN definition - Open Networking Foundation*. <https://www.opennetworking.org/sdn-resources/sdn-definition>. Accessed: 03-2017.
- [sdn b] *SDN Meetup Switzerland*. <https://www.meetup.com/SDN-Switzerland/>. Accessed: 02-2017.
- [Sen 2013] Soumya Sen, Carlee Joe-Wong, Sangtae Ha and Mung Chiang. *A Survey of Smart Data Pricing: Past Proposals, Current Plans, and Future Trends*. ACM Comput. Surv., vol. 46, no. 2, pages 15:1–15:37, November 2013.
- [ses] *SESAME*. <https://5g-ppp.eu/sesame/>. Accessed: 03-2017.
- [Sezer 2013] S. Sezer, S. Scott-Hayward, P.K. Chouhan, B. Fraser, D. Lake, J. Finnegan, N. Viljoen, M. Miller and N. Rao. *Are we ready for SDN? Implementation challenges for software-defined networks*. Communications Magazine, IEEE, vol. 51, no. 7, pages 36–43, 2013.
- [sfc 2016] *Service Function Chaining (SFC)*. Internet-Draft draft-sfc-agv-performance-measurement-architecture-00, IETF, April 2016.

BIBLIOGRAPHY

- [Shen 2009] Zhijie Shen and Roger Zimmermann. *ISP-friendly peer selection in P2P networks*. In MM '09: Proc. of the seventeen ACM international conference on Multimedia, pages 869–872. ACM, 2009.
- [Shrivastava 2005] Vivek Shrivastava. *Natural Selection in Peer-to-Peer Streaming: From the Cathedral to the Bazaar*. In In Proceedings of the International Workshop on Network and Operating Systems Support for Digital Audio and Video, pages 93–98. ACM Press, 2005.
- [Sinha 2007] Rishi Sinha, Christos Papadopoulos and John Heidemann. *Internet Packet Size Distributions: Some Observations*. Rapport technique ISI-TR-2007-643, USC/Information Sciences Institute, May 2007. Originally released October 2005 as web page <http://netweb.usc.edu/%7ersinha/pkt-sizes/>.
- [Sivathanu 2010] S. Sivathanu, Ling Liu, Mei Yiduo and Xing Pu. *Storage Management in Virtualized Cloud Environment*. In Cloud Computing (CLOUD), 2010 IEEE 3rd International Conference on, pages 204–211, 2010.
- [Sotomayor 2009] B. Sotomayor, R.S. Montero, I.M. Llorente and I. Foster. *Virtual infrastructure management in private and hybrid clouds*. IEEE Internet Computing, vol. 13, no. 5, pages 14–22, 2009.
- [str a] *History of live streaming*. <http://blog.streamingmedia.com/2016/03/history-of-the-streaming-media-industry.html>. Accessed: 03-2017.
- [str b] *A history of media streaming and the future of connected TV*. <https://www.theguardian.com/media-network/media-network-blog/2013/mar/01/history-streaming-future-connected-tv>. Accessed: 03-2017.
- [Su 2010] Xiao Su and Suchreet K. Dhaliwal. *Incentive Mechanisms in P2P Media Streaming Systems*. IEEE Internet Computing, vol. 14, no. 5, pages 74–81, 2010.
- [Sweha 2012] Raymond Sweha, Vatche Ishakian and Azer Bestavros. *AngelCast: cloud-based peer-assisted live streaming using optimized multi-tree construction*. In Proc. of the 3rd Multimedia Systems Conference, MMSys '12, pages 191–202, New York, NY, USA, 2012. ACM.
- [t-n] *T-NOVA - European FP7 Project for Network Function-as-a-Service over Virtualized Infrastructures*. <http://www.t-nova.eu/>. Accessed: 03-2017.
- [tac a] *Tacker - OpenStack NFV Orchestration*. <https://wiki.openstack.org/wiki/Tacker>. Accessed: 10-2016.

- [tac b] *Tacker SFC*. <https://github.com/openstack/tacker-specs/blob/master/specs/archive/liberty/tacker-sfc.rst>. Accessed: 10-2016.
- [ten] *Tenor Orchestrator, T-NOVA*. <https://github.com/T-NOVA/TeNOR>. Accessed: 1-2017.
- [tno] *T-Nova final year demo and project review outcome*. <https://blog.zhaw.ch/icclab/t-nova-final-year-demo-and-project-review-aftermath/>. Accessed: 03-2017.
- [Tomovic 2014] S. Tomovic, N. Prasad and I. Radusinovic. *SDN control framework for QoS provisioning*. In 2014 22nd Telecommunications Forum Telfor (TELFOR), pages 111–114, Nov 2014.
- [tos] *OASIS Topology and Orchestration Specification for Cloud Applications (TOSCA) TC*. https://www.oasis-open.org/committees/tc_home.php?wg_abbrev=tosca. Accessed: 10-2016.
- [Trajkovska] I. Trajkovska, A. Edmonds, N. Nikaein and K. Katsalis. *Joint demonstration between the 5G-PPP SESAME and COHERENT projects, in the area of multi-domain orchestration and service function chaining*. <http://www.eucnc.eu/?q=node/134/>. European Conference on Networks and Communications (EUCNC), Athens Greece, June 2016.
- [Trajkovska 2010a] Irena Trajkovska, Joaquin Salvachua Rodriguez and Alberto Mozo Velasco. *A novel P2P and cloud computing hybrid architecture for multimedia streaming with QoS cost functions*. In Proceedings of the international conference on Multimedia, MM '10, pages 1227–1230. ACM, 2010.
- [Trajkovska 2010b] Irena Trajkovska, Joaquin Salvachua Rodriguez and Alberto Mozo Velasco. *A novel P2P and cloud computing hybrid architecture for multimedia streaming with QoS cost functions*. In Proc. of the international conference on Multimedia, MM '10, pages 1227–1230, New York, NY, USA, 2010. ACM.
- [Trajkovska 2012a] I. Trajkovska, P. Rodríguez, J. Cerviño and J. Salvachúa. *Opportunities and Challenges of Implementing P2P Streaming Applications in the Web*. In Proceedings of the IADIS Conference on Collaborative Technologies, July 2012.
- [Trajkovska 2012b] Irena Trajkovska, Pedro Rodríguez Pérez, Javier Cerviño Arriba and Joaquín Salvachúa Rodríguez. *A novel monetary incentive for commercial P2P streaming*. Journal of Multimedia Processing and Technologies, vol. 3, no. 3, pages 124–135, September 2012.

BIBLIOGRAPHY

- [Trajkovska 2013] I. Trajkovska, P. Rodríguez, J. Cerviño and J. Salvachúa. *Billing model with discount incentive in a P2P-cloud multimedia streaming system*. In Proceedings of the World Congress on Computer and Information Technology (WCCIT), pages 1–6, June 2013.
- [Trajkovska 2014a] I. Trajkovska, P. Aeschlimann, C. Marti, T. M. Bohnert and J. Salvachúa. *SDN enabled QoS provision for online streaming services in residential ISP networks*. In Proceedings of the IEEE International Conference on Consumer Electronics - Taiwan (ICCE-TW), pages 33–34, May 2014.
- [Trajkovska 2014b] I. Trajkovska, P. Aeschlimann, C. Marti, T. M. Bohnert and J. Salvachúa. *SDN enabled QoS provision for online streaming services in residential ISP networks*. In 2014 IEEE International Conference on Consumer Electronics - Taiwan, pages 33–34, May 2014.
- [Trajkovska 2014c] I. Trajkovska, P. Harsh, T. M. Bohnert and J. Salvachúa. *Real-time streaming in the cloud can increase the provider's profit and the P2P users' benefit*. In Proceedings of the IEEE International Conference on Consumer Electronics - Taiwan (ICCE-TW), pages 35–36, May 2014.
- [Trajkovska 2014d] I. Trajkovska, P. Rodríguez, J. Cerviño, P. Harsh and J. Salvachúa. *P2P Incentive Model for QoS based Streaming Systems*. In Consumer Communications and Networking Conference (CCNC), 2014 IEEE, pages 1–1, Jan 2014.
- [Trajkovska 2014e] I. Trajkovska, P. Rodríguez, J. Cerviño, P. Harsh and J. Salvachúa. *P2P incentive model for QoS based streaming systems*. In Proceedings of the 11th IEEE Consumer Communications and Networking Conference (CCNC), pages 281–286, Jan 2014.
- [Trajkovska 2017] Irena Trajkovska, Michail-Alexandros Kourtis, Christos Sakkas, Denis Baudinot, JoÃcho Silva, Piyush Harsh, George Xylouris, Thomas Michael Bohnert and Harilaos Koumaras. *SDN-based service function chaining mechanism and service prototype implementation in {NFV} scenario*. Computer Standards & Interfaces, pages –, 2017.
- [Tran 2003] Duc A. Tran and et al. *ZIGZAG: An efficient peer-to-peer scheme for media streaming*. In IN PROC. OF IEEE INFOCOM, 2003.
- [Ullah 2012] I. Ullah, G. Doyen, G. Bonnet and D. Gaiti. *An autonomous topology management framework for QoS enabled P2P video streaming systems*. In Network and service management, 2012 8th int. conf. and workshop on systems virtualization management, pages 126–134, Oct 2012.

- [Varvello 2012] Matteo Varvello, Moritz Steiner and Koen Laevens. *Understanding BitTorrent: A Reality Check from the ISP's Perspective*. Comput. Netw., vol. 56, no. 3, pages 1054–1065, February 2012.
- [Venkataraman 2006] V. Venkataraman, K. Yoshida and P. Francis. *Chunkyspread: Heterogeneous Unstructured Tree-Based Peer-to-Peer Multicast*. In Proceedings of the 2006 IEEE International Conference on Network Protocols, pages 2–11, Nov 2006.
- [vid] *Big Buck Bunny Reference Video*. <https://peach.blender.org/>. Accessed: 03-2016.
- [Vieira 2013] Alex Borges Vieira, Ana Paula Couto da Silva, Francisco Henrique, Glauber Goncalves and Pedro de Carvalho Gomes. *SopCast P2P Live Streaming: Live Session Traces and Analysis*. In Proc. of the 4th ACM Multimedia Systems Conference, MMSys '13, pages 125–130, New York, NY, USA, 2013. ACM.
- [Vu 2010] Long Vu, Indranil Gupta, Klara Nahrstedt and Jin Liang. *Understanding Overlay Characteristics of a Large-scale p2p IPTV System*. ACM Trans. Multimedia Comput. Commun. Appl., vol. 6, no. 4, pages 31:1–31:24, November 2010.
- [Wang 1998] Zheng Wang, Mark A. Carlson, Walter Weiss, Elwyn B. Davies and Steven L. Blake. *An Architecture for Differentiated Services*. RFC 2475, December 1998.
- [Wang 2004] Zhou Wang, A. C. Bovik, H. R. Sheikh and E. P. Simoncelli. *Image Quality Assessment: From Error Visibility to Structural Similarity*. Trans. Img. Proc., vol. 13, no. 4, pages 600–612, April 2004.
- [Wang 2008] Jiajun Wang, Cheng Huang and Jin Li. *On ISP-friendly rate allocation for peer-assisted VoD*. In MM '08: Proc. of the 16th ACM international conference on Multimedia, pages 279–288. ACM, 2008.
- [Wang 2010a] F. Wang, Y. Xiong and J. Liu. *mTreebone: A Collaborative Tree-Mesh Overlay Network for Multicast Video Streaming*. IEEE Transactions on Parallel and Distributed Systems, vol. 21, no. 3, pages 379–392, March 2010.
- [Wang 2010b] Guohui Wang and T. S. Eugene Ng. *The Impact of Virtualization on Network Performance of Amazon EC2 Data Center*. 2010 Proceedings IEEE INFOCOM, pages 1–9, March 2010.
- [Wang 2012] Tzu-Ming Wang, Wei-Tsong Lee, Tin-Yu Wu, Hsin-Wen Wei and Yu-San Lin. *New P2P Sharing Incentive Mechanism Based on Social Network and Game Theory*. In Proc. of the 2012 26th Int. Conf. AINA Workshops, WAINA '12, pages 915–919, Washington, DC, USA, 2012. IEEE Computer Society.

BIBLIOGRAPHY

- [web] *WebRTC - Web Real-Time Communication*. <https://webrtc.org/>. Accessed: 03-2017.
- [Weissman] Jon Weissman and Siddharth Ramakrishnan. *Using Proxies to Accelerate Cloud Applications*.
- [wic] *WICM - WAN Infrastructure and Connectivity Manager*. <https://github.com/T-NOVA/WICM>. Accessed: 04-2016.
- [wow] *Wowza - Media streaming with Amazon EC2*. <https://www.wowza.com>. Accessed: 03-2017.
- [Wu 2011] Y. Wu, C. Wu, B. Li, X. Qiu and F. C. M. Lau. *CloudMedia: When Cloud on Demand Meets Video on Demand*. In 2011 31st International Conference on Distributed Computing Systems, pages 268–277, June 2011.
- [Wu 2015] Jun Wu, Zhifeng Zhang, Yu Hong and Yonggang Wen. *Cloud radio access network (C-RAN): a primer*. IEEE Network, vol. 29, no. 1, pages 35–41, 2015.
- [Xiong 2003] Li Xiong and Ling Liu. *A reputation-based trust model for peer-to-peer e-commerce communities*. In E-Commerce, 2003. CEC 2003. IEEE International Conference on, pages 275–284, 2003.
- [Xu 2013] Hong Xu and Baochun Li. *A Study of Pricing for Cloud Resources*. SIGMETRICS Perform. Eval. Rev., vol. 40, no. 4, pages 3–12, April 2013.
- [Yan 2015] J. Yan, H. Zhang, Q. Shuai, B. Liu and X. Guo. *HiQoS: An SDN-based multipath QoS solution*. China Communications, vol. 12, no. 5, pages 123–133, May 2015.
- [Yiakoumis 2012] Yiannis Yiakoumis, Sachin Katti, Te-Yuan Huang, Nick McKeown, Kok-Kiong Yap and Ramesh Johari. *Putting Home Users in Charge of Their Network*. In Proc. of 2012 ACM Conference on Ubiquitous Computing, UbiComp '12, pages 1114–1119, New York, NY, USA, 2012. ACM.
- [Yilmaz 2015] S. Yilmaz, A. M. Tekalp and B. D. Unluturk. *Video streaming over software defined networks with server load balancing*. In 2015 International Conference on Computing, Networking and Communications (ICNC), pages 722–726, Feb 2015.
- [Yu 2015] Tsung-Feng Yu, Kuochen Wang and Yi-Huai Hsu. *Adaptive routing for video streaming with QoS support over SDN networks*. In 2015 International Conference on Information Networking (ICOIN), pages 318–323, Jan 2015.
- [zer] *Zero MQ*. <http://zeromq.org/>. Accessed: 03-2017.

- [Zhang 2005] Xinyan Zhang, Jiangchuan Liu, Bo Li and Tak-Shing Peter Yum. *CoolStreaming/DONet: a data-driven overlay network for peer-to-peer live media streaming*. In INFOCOM, pages 2102–2111. IEEE, 2005.
- [Zhang 2009] Hui Zhang, Guofei Jiang, Kenji Yoshihira, Haifeng Chen and Akhilesh Saxena. *Intelligent Workload Factoring for a Hybrid Cloud Computing Model*. In Proceedings of the 2009 Congress on Services - I, pages 701–708, Washington, DC, USA, 2009. IEEE Computer Society.
- [Zhao 2005] Li Zhao, Jian-Guang Luo, Meng Zhang, Wen-Jie Fu, Ji Luo, Yi-Fei Zhang and Shi-Qiang Yang. *Gridmedia: A Practical Peer-to-Peer Based Live Video Streaming System*. In IEEE 7th Workshop on Multimedia Signal Processing, MMSP 2005, October 30 2005 - November 2 2005, Shanghai, China, pages 1–4, 2005.
- [Zhao 2006] B. Y. Zhao, Ling Huang, J. Stribling, S. C. Rhea, A. D. Joseph and J. D. Kubiatowicz. *Tapestry: A Resilient Global-scale Overlay for Service Deployment*. IEEE J.Sel. A. Commun., vol. 22, no. 1, pages 41–53, September 2006.
- [Zhou 2007] Runfang Zhou and Kai Hwang. *PowerTrust: A Robust and Scalable Reputation System for Trusted Peer-to-Peer Computing*. IEEE Trans. Parallel Distrib. Syst., vol. 18, no. 4, pages 460–473, April 2007.
- [Zhu 2011] W. Zhu, C. Luo, J. Wang and S. Li. *Multimedia Cloud Computing*. IEEE Signal Processing Magazine, vol. 28, no. 3, pages 59–69, May 2011.
- [Zhuang 2001] Shelley Q. Zhuang, Ben Y. Zhao, Anthony D. Joseph, Randy H. Katz and John D. Kubiatowicz. *Bayeux: an architecture for scalable and fault-tolerant wide-area data dissemination*. In Proceedings of the 11th international workshop on Network and operating systems support for digital audio and video, pages 11–20. ACM, 2001.