NAIST-IS-MT1251207

Master's thesis

A practical benchmarking methodology for IPv4 over IPv6 transition technologies

Marius Liviu Georgescu

August 12, 2014

Department of Information Processing
Graduate School of Information Science
Nara Institute of Science and Technology

A Master's Thesis
submitted to Graduate School of Information Science,
Nara Institute of Science and Technology
in partial fulfillment of the requirements for the degree of
MASTER of ENGINEERING

Marius Liviu Georgescu

Abstract

Years from now, IPv4 will only be an important part of the Internet's history, but for now it remains the dominant Internet Protocol and a pending danger to the Internet's expansion. The transition from IPv4 to IPv6 is an ongoing process which will eventually lead to the end of the IPv4 era. This transition has presented the Internet community with numerous challenges, which led to many opportunities for research. Given the variety and complexity of current production networks, a scenario-based approach is one of the major research directions.

The IETF has introduced multiple transition scenarios, among which there is a transition scenario for enterprise networks, where IPv4-capable hosts exist in the IPv6 Internet and network support is required, to convey IPv4 communications through the IPv6 infrastructure. For simplicity, the scenario can be referred as the enterprise 464 scenario. There is a number of suitable transition technologies for this scenario, which can in turn be referred to as 464 technologies. However, a problem remains unsolved. Considering this scenario, which one of these transition technologies is more suitable than the rest? Moreover, different implementations of the same technology can have different capabilities, further complicating the problem.

To support network operators solve this problem, we are proposing a practical benchmarking methodology, exploring various feasibility dimensions of transition technologies. The methodology is associated with a heterogeneous IPv4 and IPv6 network testbed, which we called the IPv6 Network Evaluation Testbed (IPv6NET). To support this methodology, we have used it to analyze the feasibility of two open source transition implementations, covering multiple transition technologies. The feasibility analysis is based on practical means, employing existing implementations of the transition technologies and empirical measurements. To that end we are showing how network performance results, scalability results and operational capability data can be obtained, analyzed and compared. Furthermore, we are presenting a model for building composite indicators, which can identify the optimal transition solution considering certain feasibility dimensions and the specific feasibility needs of enterprise network operators.

Keywords: IPv6 transition, 464 transition scenario, Enterprise Networks, IPv6NET, benchmarking methodology, asamap, tiny-map-e, MAPe, MAPt, DSLite, 464XLAT[*]

[*]Master's Thesis, Department of Information Processing, Graduate School of Information Science, Nara Institute of Science and Technology, NAIST-IS-MT1251207, August 12, 2014.

Acknowledgments

This work would not have been possible without the help and support of several people. I would like to start by thanking my supervisor, Professor Suguru Yamaguchi, to whom I owe my deepest gratitude for giving me the chance to study here, and for offering his invaluable support and guidance.

I am very grateful to Professor Keiichi Yasumoto and Professor Kazutoshi Fujikawa for kindly agreeing to be members in my thesis committee.

I would like to express my particular appreciation and thanks to Associate Professor Youki Kadobayashi for being a constant source of invaluable constructive feedback and guidance.

I owe a great deal of gratitude to Associate Professor Hiroaki Hazeyama for patiently guiding my research steps and generously supporting each one of them.

I would like to extend my thank you to Associate Professor Takeshi Okuda and Assistant Professor Shigeru Kashihara, for offering their priceless advice and encouragement.

Insightful discussions with my colleagues Doudou Fall, Christopher Michael Yap and Ady Wahyudi Paundu have been a constant source of inspiration, for which I am very grateful. Christopher's generous efforts to improve my English are also worthy of acknowledgment and led to better quality text.

The Internet Engineering Laboratory is not just a great place to conduct research, it also represents a fertile environment for cooperation and friendship. Consequently, I would like to extend my gratitude to all the members of this laboratory.

I would like to thank Mr. Masakazu Asama and Mr. Yukito Ueno, the developers of *vyatta asamap* and *tiny-map-e* respectively, open source implementations, upon which the experimental networks were built. Special thanks should be extended to the teams at NICT StarBED and WIDE for their continuous support.

I wish to also thank the Japanese Ministry of Education, Culture,Sports, Science and Technology for providing me with the the opportunity to study in Japan and for continuously supporting my research.

Last but not least, I would like to thank my family and friends for their love and support throughout the entire process.

Contents

1 Introduction **1**

1.1 Motivation and Problem Statement 1

1.2 Scope 2

1.3 Contribution 2

1.4 Thesis Structure 3

2 Background **5**

2.1 IPv6 transition overview 5

2.2 IPv6 transition challenges 5

2.3 IPv6 transition technologies 6

2.3.1 464 transition technologies 8

2.4 Related research 11

2.4.1 Closed environments 11

2.4.2 Open environments 12

3 Benchmarking methodology **13**

3.1 Feasibility indicators and metrics 13

3.2 Benchmarking approaches 15

3.2.1 Closed environment methodology 15

3.2.2 Open environment methodology 17

3.3 Empirical results summarization 19

3.4 Building the composite indicators 19

4 IPv6NET concept **27**

4.1 Network template 28

4.2 Transition tuples 29

4.3 Network environment 29

4.3.1 Closed network environment 29

4.3.2 Open network environment 32

5 Empirical results **33**

5.1 Closed Experiment results 33

5.1.1 Network performance 33

5.1.2 Scalability 33

5.2 Summarized results 34
5.2.1 Network Performance 35
5.2.2 Scalability 39
5.3 Data collection and repeatability 39
5.4 Open Experiment results 40
5.5 Normalized and composite scores 43
5.5.1 Network performance 43
5.5.2 Scalability 45
5.5.3 Operational capability 48
5.5.4 General Feasibility Considerations 51

6 Discussion of the proposal **53**
6.1 Validity 53
6.2 Limitations 54
6.3 Applicability 54
6.4 Future work 55

7 Conclusion **57**
Appendix.A. 59
Appendix.A.1. 59
Appendix.A.2. 62
Appendix.B. 64
Appendix.B.1. 64
Appendix.B.2. 66

Bibliography **69**

List of Figures

2.1 Basic IPv6 transition technologies 7
2.2 Evolution of transition technologies in the IETF 7

3.1 Feasibility indicators and metrics taxonomy 14
3.2 Test setup 16
3.3 Mean, mode and Median flow chart 19

4.1 IPv6NET Concept 27
4.2 464 Network template 28
4.3 464 Closed network environment 30
4.4 10×1 Closed network environment 30
4.5 464 Open network environment 31

5.1 Delay results 34
5.2 Jitter results 35
5.3 Throughput results 36
5.4 Network performance degradation for asamape 37
5.5 Probability density function for the 10 Mbps Datasets 38
5.6 Probability density function for the 100 Mbps Datasets 38
5.7 Probability density function for the 10CE Scalability Datasets 38
5.8 Repeatability of the DC results 41
5.9 Normalized NP data 43
5.10 NP scores decomposition 46
5.11 Normalized scalability data 46
5.12 SCAL scores decomposition 47
5.13 Normalized OC data 48
5.14 OC scores decomposition 50
5.15 General feasibility considerations 51

List of Tables

3.1 Workload $Framesize \times Framerate$ 16
3.2 Network Performance 26
3.3 Operational capability 26

4.1 464 Transition tuples 29

5.1 10 Mbps Summarized results 36
5.2 100 Mbps Summarized Results 39
5.3 Network Performance Degradation (NPD) results 39
5.4 Relative standard error average 39
5.5 Configuration and Troubleshooting capability results 40
5.6 Applications capability results 42
5.7 Correlation coefficients and correlation significance for Network Performance 44
5.8 Network performance AHP paired comparison Table 45
5.9 Network performance (NP) normalized and composite scores 45
5.10 Correlation coefficients and correlation significance for Scalability 47
5.11 Scalability Normalized scores 47
5.12 Operational Capability AHP paired comparison table 49
5.13 Operational capability 49

Chapter 1

Introduction

Threatened by the limitations of IPv4, the Internet community turned to IPv6 as means to continue the expansion of the Internet. IPv6 uses a 128 bit address, extending the address space to $2^{128} \approx 3.4 \cdot 10^{38}$ unique IP addresses, enough for many years to come. However the appeal of IPv6 has diminished since 1998, mainly because it is not able to communicate directly with its predecessor, IPv4. This introduced the Internet community to a great challenge, namely the transition to IPv6, which is comprised of the stages the Internet will have to undergo until IPv6 will completely replace IPv4.

1.1 Motivation and Problem Statement

Given the complexity of the current IPv4-dominated Internet, the transition to IPv6 will be a long and complex process. So far, only a small number of production networks are IPv6-capable. The APNIC Labs IPv6 deployment report[1] shows that only about 2% of the worldwide users have IPv6 connectivity.

The Internet Engineering Task Force (IETF) has made efforts towards analyzing the different factors influencing the IPv6 adoption process. One of these factors is the complexity of existing production networks. To that end, IPv6 transition network scenarios have been researched within the IETF by the v6ops and Softwire Working Groups. The scenarios were dedicated to four main types of networks: ISP Networks[2], Enterprise Networks[3], 3GPP Networks[4] and Unmanaged Networks[5].

The IETF Next Generation Transition (ngtrans) Working Group has made many efforts to propose and analyze viable transition technologies. Many transition technologies have been proposed. As surveyed by Leng et al. in [6] and by P. Wu et al. in [7], all transition technologies have advantages and disadvantages considering a certain transition scenario, but no transition mechanism can be considered most feasible for all the scenarios. The question of which one of these transition technologies is most feasible for a particular scenario remains open. Given the complexity and the diversity of transition technologies, this leads to a great challenge for network operators faced with the IPv6 transition. Transition implementations, covering one or multiple transition technologies have been proposed as well, further complicating the problem.

1.2 Scope

To solve this problem, we are proposing a benchmarking methodology associated with the IPv6 Network Evaluation Testbed (IPv6NET), an experimental environment dedicated to the feasibility quantification of IPv6 transition mechanisms in a series of practical scenario-based network tests. To support this methodology we are focusing on one specific scenario, introduced by the IETF for enterprise networks in [3]. The scenario targets enterprises using an IPv6-only core network technology, but with IPv4-capable nodes, which need to communicate over the IPv6 infrastructure. To that end IPv4 over IPv6 transition technologies are needed. The scenario can be simply referred as 464 enterprise scenario, while the suitable technologies can be referred as 464 technologies.

1.3 Contribution

The main contribution of this thesis is represented by the benchmarking methodology associated with IPv6NET and the empirical feasibility results, which to the best of our knowledge represent a first in current literature. Another achievement worth mentioning is represented by the detailed building steps of composite indicators, which until now have fallen outside computer science. Last but not least, we have approached a non-formalized aspect of benchmarking, open environment benchmarking, which should assess more practical aspects of a transition implementation, such as interoperability and operational complexity. To that end we have proposed three metrics and an associated measurement methodology. This methodology has made use of practical means, such as existing implementations and empirical measurements, to analyze some core feasibility dimensions of transition technologies.

By analyzing the empirical results, we were able to identify possible performance trends in IPv6 transition technologies benchmarking. Moreover we were able to point out some unexpected behaviors which could have been overlooked if simulators or analytical tools were to be employed. These empirical results can serve as a direct guideline to network operators faced with a similar transition scenario. The detailed feasibility results can also act as feedback for the transition implementation developers. This can lead to further improvement of their products.

The feasibility scores can represent a synthesized version of the empirical results, which can reflect different interest in feasibility of different network operators. The composite indicator charts can ultimately prove to be a starting point for decisions, given the transparency of the feasibility measurement. The alternative of recreating the experiments can be both time and resource consuming. In turn, the model for building composite indicators is, as far as we know, a first in literature dedicated to computer science. The detailed building steps can be useful to fellow computer science researchers interested in benchmarking with composite indicators.

1.4 Thesis Structure

The remainder of the thesis is structured as follows:

Chapter 2 : This chapter presents background information, familiarizing the reader with the IPv6 transition and the challenges it introduced. It also gives an overview of related literature.

Chapter 3 : We present our main contribution in this chapter: the detailed methodology for benchmarking IPv4 over IPv6 transition technologies.

Chapter 4 : This chapter presents details about the IPv6NET concept and how the methodology integrates with the rest of the components.

Chapter 5 : In this chapter the empirical results are introduced and analyzed.

Chapter 6 : This chapter discusses the validity, limitations and applicability of our proposal and the future research directions.

Chapter 7 : In this chapter the conclusions are presented.

Chapter 2

Background

2.1 IPv6 transition overview

The social and business environment represented by the Internet today is close to the biggest turning point in its history. The widely deployed Internet Protocol Version 4 (IPv4) is showing its limitations. The IPv4 address space, which has $2^{32} \approx 4.3$ billion unique IP addresses, is endangering the continual expansion of the Internet. On February 3rd 2011 the Internet Assigned Numbers Authority (IANA) announced the allocation of the last blocks of IPv4 addresses [8]. Also the Asia Pacific Network Information Centre (APNIC), the Regional Internet Registry (RIR) for the Asia Pacific region has announced on April 15th 2012 entering the last stage of IPv4 Exhaustion (the final /8 address block) [9]. The other Regional Internet Registries will surely follow in the future. The answer to the IPv4 addresses exhaustion problem is the deployment of the next generation Internet Protocol, the Internet Protocol Version 6 (IPv6), presented by the Internet Engineering Task Force (IETF) in 1998.

IPv6 uses a 128 bit address, extending the address space to $2^{128} \approx 3.4 \cdot 10^{38}$ unique IP addresses, a significant improvement considering the IPv4 address space. However the appeal of IPv6 has diminished since 1998, mainly because it is not able to communicate directly with its predecessor, IPv4. This introduced the Internet community to a great challenge, namely the transition to IPv6. The transition is an ongoing process and is represented by the stages the Internet will have to undergo until IPv6 will completely replace IPv4.

Given the complexity of the current IPv4-dominated Internet, the transition to IPv6 will be a long and complex process. So far, only a small number of production networks are IPv6-capable. The APNIC Labs IPv6 deployment report[1] shows that only about 2% of the worldwide users have IPv6 connectivity.

2.2 IPv6 transition challenges

From the industry perspective, the book *Global IPv6 strategies*[10] explores some of the obstacles preventing Internet companies from adopting IPv6 so far:

Lack of apparent use can be defined as the lack of a killer application to drive the IPv6 adoption.

Costs of the adoption were considered unjustified. Investments in IPv6 were considered unnecessary, as the return of investment (ROI) was hard to predict.

Technology challenges such as the reliability and security of IPv6 implementations was questioned.

Availability of IPv6-ready products was limited. The lack of commercial IPv6-ready products prohibited transition interested companies from starting the transition process.

Lack of trained staff is still an issue. Many network operation teams lack IPv6 knowledge.

With time, many of these obstacles have been overcome. As shown by the World IPv6 Launch infographic [11] the industry has understood that IPv6 adoption is not a question of if, but a question of when. However, the very low worldwide adoption rate indicates that there are still many open problems.

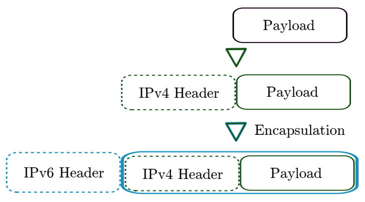
From the academic perspective, the IPv6 transition presented many opportunities for research. The biggest research entity behind the Internet, the IETF has made many efforts to formalize the IPv6 transition, by introducing typical network transition scenarios and proposing transition technologies. As surveyed by X. Leng et al. in [6], and by P. Wu et al. in [7], deciding which transition technology is the most feasible for a specific network scenario, remains one of the biggest challenges. Among other very important challenges we can mention: ensuring security, maintaining network performance and providing applications and operational support. Some more technology-oriented problems are identified in the next section.

2.3 IPv6 transition technologies

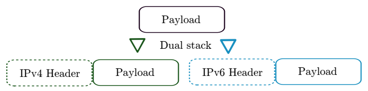
IPv6 was not designed to be backwards compatible. In other words IPv6-only nodes cannot directly communicate with IPv4-only nodes. Consequently, coexistence and transition technologies need to be employed. Initially, three basic transition mechanisms were proposed: dual-stack, translation and tunneling. The associated implementation standards are presented in RFC4213 [12] and RFC6144 [13]. An abstraction of the three is shown in Figure 2.1. Over the years many other transition technologies have been introduced by the ngtrans Working Group of the IETF. Figure 2.2 presents a road-map of the evolution of some of the transition technologies proposed in the IETF.

For dual-stack abstracted in Figure 2.1b, both IPv4 and IPv6 are implemented on the same node . This method is mostly used in host-side nodes and edge nodes. The main challenge introduced by dual-stack is overhead, as it needs two routing tables and routing processes.

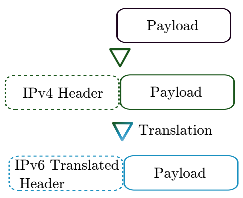
Translation displayed in Figure 2.1c is the only method which achieves direct communication between IPv4 and IPv6, by translating the information and message format between different versions of Internet Protocol. Usually translators are employed at the



(a) Encapasulation



(b) Dual Stack



(c) Translation

Figure 2.1: Basic IPv6 transition technologies

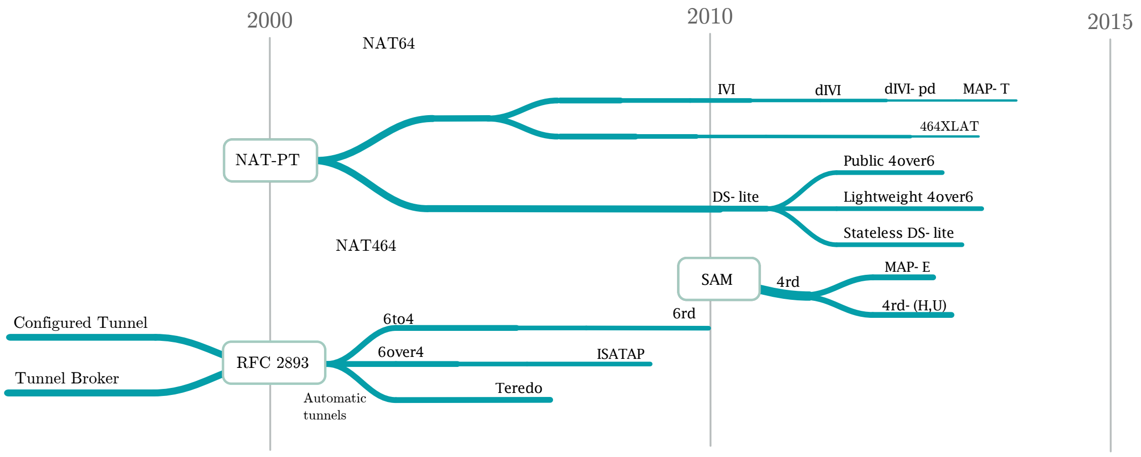


Figure 2.2: Evolution of transition technologies in the IETF

border between an IPv4-only and an IPv6-only site. The main problem with translation is that it breaks something which IPv6 was supposed to bring back: the end-to-end characteristic of the Internet. Aside from that, translation can affect the functionality of secure protocols, such as IPSec or DNSSEC. Modern translation technologies can be classified as stateless technologies (e.g. IVI[14], dIVI[15])and stateful technologies (e.g. NAT64 [16], DSLite [17]). Stateless translation technologies achieve an one-to-one address mapping, translating only the IP and ICMP headers. On the other hand, stateful translation builds a one-to-many IPv4 address mapping, by using the IPv4 address resources as a pool on the translating device, and allocating them at per port granularity. Stateful translators require a great deal of per-state flow maintenance, in other words every incoming packet has to be classified to its corresponding queue, increasing the overhead on the network devices involved. However, stateless translators need one IPv4 address for every IPv6 host, which negates the primary advantage of IPv6, which is the increase in address space.

Tunneling or encapsulation presented in Figure 2.1a is employed to traverse heterogeneous network environments, by encapsulating the IPvX packets into the payload of IPvY packets, were $X, Y \in \{4, 6\}$. At the border of the IPvY and IPvX networks the packets are decapsulated back into IPvX by an edge router. Tunneling technologies, initially introduced in RFC1933 [18], can be classified in three categories: static tunnels, semi-automatic and automatic tunnels. The static tunnels require manual configuration at both ends. Their main advantage is the simplicity of deployment, which makes them cost effective and attractive for some Internet Service providers (e.g. Nippon Telegraph and Telephone Corporation). Semi-automatic tunnels also need manual configuration, but only on the host side, as the provider side is auto-configured. Both static and semi-automatic tunnels are not suitable in large networks because of their low scalability potential and single point of failure considerations. For automatic tunneling (e.g. 6to4 [19], ISATAP [20], TEREDO [21]) the tunnels are created on-demand. These are highly scalable and represent the majority of tunneling mechanisms today. Tunneling mechanisms are confronted with fragmentation and MTU problems because of encapsulation. The encapsulation/decapsulation process will also induce considerable overhead in the network devices involved in the process. Security considerations have to be taken into account as well. Tunnels are especially vulnerable to spoofed encapsulated packet attacks, which can target a normal node or a tunnel end-node. In automatic tunneling mechanisms the security threat can increase by targeting the spoofed packets at the broadcast/multicast address of relay routers. Many of the modern transition technologies use one or more basic transition technologies. For example MAPe [22] and Dual-Stack Lite (DSLite)[17] employ dual-stack and translation at the edge nodes and encapsulation in the core.

Another classification of transition technologies can be achieved by the phases of the IPv6 transition they can be associated with. In RFC6144 [13], three important phases have been identified:

Preparation phase in which IPv6 services are scarce, and few production networks have working IPv6-enabled cores. In this phase, mostly IPv6 over IPv4 technologies (e.g. 6to4 [19], 6rd [23]) are needed.

Transition phase in which IPv6 presence is increasing, hence dual stack support and services should be provided. Although IPv6 use is still very low, many large Internet companies started offering services also over IPv6. This can be considered the current ongoing phase. This phase is expected to increase the number of dual stack and IPv4 over IPv6 technologies (e.g. DSLite[17], MAPe [22]).

post-Transition phase, the last stage of the transition, in which IPv6 will be the dominant protocol. This phase should offer support to IPv4-only islands over IPv6-only infrastructures, and IPv4 over IPv6 technologies will become dominant.

2.3.1 464 transition technologies

The scenario we chose to analyze as support for the benchmarking methodology was proposed by the IETF for enterprise networks in [3]. It targets enterprises using an IPv6-only core, but with IPv4-capable nodes, which need to communicate over the IPv6 infrastructure. For that IPv4 over IPv6 transition technologies are needed, which can simply be referred to as 464 technologies. The scenario is plausible for current production networks, and received a lot of attention in Japan. It also has the potential to become the dominant scenario for enterprise networks, in the last phase of the transition. This is why it represented our first choice. According to the transition technology used for the network core traversal, the 464 transition technologies can be classified into *encapsulation-based* technologies and *translation-based* technologies.

Encapsulation-based technologies

As the name suggests, encapsulation-based transition technologies use encapsulation as method for traversing the IPv6-only core network. In this manuscript two encapsulation-based technologies are analyzed: DSLite [17] and MAPe [22].

MAPe is an automatic tunneling transition mechanism. It allows the transportation of IPv4 packets over an IPv6 backbone network, using IP encapsulation and a mapping mechanism between IPv6 addresses and IPv4 addresses with transport layer ports.

The MAPe environment needs the following building blocks:

- MAP domain: the IPv6 network which interconnects the MAP components. In the same IPv6 networks multiple MAP domains can be employed.
- MAP Border Relay (BR): a MAP-enabled router with at least one IPv6 interface and one IPv4 interface, connected to the native IPv4 network.
- MAP Customer Edge (CE): a customer edge router which serves as a residential site with one IPv6 enabled WAN interface and one or multiple LAN interfaces. It is important to note that the CE router also performs a Network Address Translation (NAT) function.
- MAP Rule: a set of mapping parameters characteristic to a specific MAP domain. For a MAP rule a prefix for both IPv4 and IPv6, and an exact number of Embedded Addresses (EA) bits is required. Additionally, for each customer site, an IPv6 sub-prefix is assigned. Using the EA bits and the customer sub-prefix, the shared IPv4 prefix/IPv4 address and the Port Set Identifier (PSID) are calculated.

One of the advantages of MAPe can be the CE element architecture. The CE is handling the NAT function, relieving the core network of that responsibility. This also eludes the danger of a single point of failure, characteristic to Carrier Grade NAT (CGNAT) [24] architectures.

Perhaps one of the biggest disadvantages of MAPe is represented by the mapping rule, which is complex and can introduce operational issues, when configuring or troubleshooting. To that end, a very useful tool is the MAP simulation tool [25] created by Arthur Lacoste of Cisco Systems.The addressing rule can also create problems, but only for large scale production networks (e.g. ISP Networks) with a low public IPv4 address pool.

DSLite is a stateful tunneling mechanism that relies on an IPv6 backbone network. It employs IPv4-in-IPv6 tunnels to cross the IPv6 network and reach a carrier-grade IPv4-IPv4 Network Address Translation (CGNAT) [24] device, allowing customers to share IPv4 addresses. A DSLite environment is based on the following components:

- Basic Bridging Broad Band (B4) component: represents a function implemented in a dual-stack node, either integrated into a CPE or directly connected, which creates an IPv4-in-IPv6 tunnel to an AFTR.

- Address Family Transition Router (AFTR) component: represents a device which is connected to the native IPv4 network and represents the end-point of the IPv4-in-IPv6 tunnels. The AFTR integrates a carrier-grade NAT function which allows B4 enabled CPEs to share the same IPv4 address pool.
- Shared IPv4 address pool: a public IPv4 prefix/IPv4 address shared among multiple CPEs.

In contrast to MAPe, the provider edge element includes a CGNAT function, which requires per-flow maintenance, increasing the operational complexity. It is also susceptible to the single point of failure issue. However, this can be avoided with a redundant design.

One of the biggest advantages of DSLite is represented by interoperability, as many production networks are already using CGNAT machines.

Translation-based technologies

In the case of translation-based technologies, translation represents the mechanism for traversing the IPv6-only core network. We are analyzing two translation-based technologies: MAPt [26] and 464XLAT[27].

MAPt is an IPv6-IPv4 Network Address Translation (NAT64) solution which provides shared or non-shared IPv4 address connectivity over an an IPv6-only core network.

Similarly to MAPe, the MAPt environment needs the following building blocks:

- MAP domain: the IPv6 core which interconnects the other MAP components. Multiple MAP domains can be employed in the same IPv6 network.
- MAP Border Relay (BR): a MAP-enabled machine connected to the native IPv4 network, at the edge of the MAP domain.
- MAP Customer Edge (CE): a customer edge router used as a residential site. The CE router is performing the Network Address Translation (NAT) function.
- MAP Rule: the mapping parameters specific to a certain MAP domain. Each MAP rule needs an IPv6 prefix, an IPv4 prefix and a specific number of Embedded Address (EA) bits. An IPv6 sub-prefix is assigned for each customer site. From the EA bits and the customer sub-prefix, the shared IPv4 prefix/IPv4 address, and the Port Set Identifier (PSID) are calculated.

The disadvantages of MAPe stand for MAPt as well. The mapping rule can increase the operational complexity for both configuring and troubleshooting. The addressing rule can create problems as well, but mainly for large scale production networks with low a public IPv4 address pool.

The CE element architecture can be one of the main advantages of MAPt as well. By handling the NAT function, the CE relieves the core network of that responsibility. This also can avoid the danger of a single point of failure, characteristic to Carrier Grade NAT (CGNAT) architectures.

464XLAT combines stateful protocol translation with stateless protocol translation to provide IPv4 connectivity across an IPv6-only network. A 464XLAT environment needs the following components:

- PLAT: provider-side translator, which employs stateful translation, N to 1 global IPv6 addresses to global IPv4 addresses, and vice versa.
- CLAT: customer-side translator, employing stateless translation to map 1 to 1 private IPv4 addresses to global IPv6 addresses, and vice versa.

464XLAT also uses a shared IPv4 public address and the stateful translation is realized in the core network. This means it inherits the core network overhead, and single point of failure issues. By combining stateless and stateful translation, 464XLAT is considered easy to deploy and efficient from the public IPv4 pool stand-point. It is also considered suitable for 3GPP transition networks.

All 4 of the above 464 technologies are suitable for the chosen scenario, and although there are structural reasons for choosing one or the other, we contend that a thorough empirical feasibility analysis is needed in order to confirm performance trends or identify interoperability issues and potential pitfalls. To the best of our knowledge, there is no similar initiative, which brings further motivation to our cause.

2.4 Related research

There are a variety of articles dedicated to IPv6 transition experimental environments in current literature. The methodology associated with these experimental environments can be generically classified in two categories: closed and open environments.

2.4.1 Closed environments

Closed environments are usually local environments, which are isolated from production networks or the Internet. For example, I. Raicu et al. have analyzed the performance of two 6-over-4, and IPv6 in IPv4 tunneling implementations in comparison with a homogeneous IPv6-only network in [28]. S. Narayan et al. evaluates the performance of Linux operating systems in relation to an IPv4-v6 configured Tunnel and a 6to4 Tunnel in [29]. Four workstations were employed to build the testbed. S. Sasanus et. al. measures the differences in bandwidth requirements for common network applications like remote login, web browsing, voice communication and database transactions over 3 types of networks: IPv4-only, IPv6-only and a 6to4 tunneling mechanism in [30]. The environment was built using the OPNET simulator, which also served as the basis for the testbed presented by P. Grayeli et. al in [31], which was dedicated to the performance analysis of transition mechanisms over a MPLS backbone. In [32], G. Lencse et al. evaluates the performance of DNS64 implementations, BIND9 and TOTD running on OpenBSD and FreeBSD.

A common trait of the above mentioned closed environments is the thorough performance analysis, which resulted in quantifiable (hard) data such as CPU and memory utilization, throughput, end-to-end delay, jitter and execution time.

However, as P. Wu et al. in [33] have underlined, before transition mechanisms are applied in a large scale environment, a systematic and quantitative performance analysis should be performed. This gets us to the second group of experimental environments, namely open environments.

2.4.2 Open environments

Open environments can be defined as experimental networks connected to a large scale production network or to the Internet. While both types of methodologies can be considered practical, as they usually employ existing implementations, open environments are especially practical as they explore other aspects, less formalized than network performance, such as operational efficiency and interoperability.

In [34], R. Hiromi et al. have identified poor implementation and erroneous operations in a dual-stack environment. A hotel Internet service is presented as a case study. Operational issues such as lack of path/peering, Bad TCP reaction or misbehaving DNS resolution are identified.

H. Babiker et al. in [35] describe the lessons learned from deploying IPv6 in Google's heterogeneous corporate network. The report presents numerous operational troubles: the lack of dual-stack support of the customer-premises equipments (CPE), or the immature IPv6 support of operating systems and applications. One of their conclusions was that the IPv6 transition can affect every operational aspect in a production environment, hence operational considerations have to be made.

In [36], J. Arkko et al. presented experiences with IPv6-only Networks. NAT64 and DNS64 technologies are tested in two open environments: an office and a home environment. Common applications such as web browsing, streaming, instant messaging, VoIP, online gaming, file storage and home control were tested. Application issues in relation to the NAT64/DNS64 technology are identified, for example Skype's limitation to connect to IPv6 destinations, or the lack of network operational diagnostics for certain standalone games.

Experiences with IPv6-only Networks have been also presented by Hazeyama et al. in [37]. A great deal of meaningful interoperability data was presented, such as the IPv6 capability of OSes, applications and network devices. Also many operational issues have been identified. Some examples are long fall-back routine, the low DHCPv6 capability of certain OSes, the lack of IPv6 support in some network devices, DNS64 overload, inappropriate AAAA replies or inappropriate selection of DNS resolvers. Considering these examples, we can conclude that open environments have the potential of exposing interoperability issues, which can otherwise get overlooked.

Combining the advantages of the two benchmarking methodologies can lead to a complete feasibility analysis. Consequently, it represents the goal for our benchmarking methodology.

Chapter 3

Benchmarking methodology

In this chapter, we present details about the proposed benchmarking methodology. We start by explaining some of the semantics associated with the methodology. Next, open and closed environments are used as the two types of benchmarking approaches. Finally, we explain how the results can be normalized and integrated into composite feasibility scores.

3.1 Feasibility indicators and metrics

This section presents some clarifications regarding the semantics for the benchmarking methodology used throughout this thesis. We are using the term *feasibility indicator* as a generic term for composite indicators quantifying the feasibility of transition technologies. The term *feasibility metric* is used for specific systems of measurement, which quantify a specific feasibility aspect. Figure 3.1 shows a taxonomy of the proposed feasibility indicators and metrics. Given that one transition implementation can cover multiple transition technologies, we are using the term *transition tuple* to describe the set of one transition technology and one transition implementation.

The General Feasibility Indicator (GFI) is envisioned as a unique composite score which is associated with a transition tuple. The formula for obtaining this score should be based on the feasibility analysis of network performance, scalability operational capability and security of each transition tuple. Its purpose is to allow an easy comparison among transition tuples. However, currently we do not have a clear solution for quantifying the security of transition technologies. This makes GFI just an idea, and one of the future research landmarks.

For the closed environment methodology, we propose *network performance* and *scalability* as feasibility indicators. Network performance indicates the technical feasibility of each technology in relation to existing computer network standards. To quantify network performance, we have used well-established metrics, such as *round-trip-delay, jitter, throughput* and *frame loss*, as they have been defined in RFC5180 [38] and RFC2544 [39]. Scalability is regarded as the ability of each transition tuple to accommodate topology growth. As it is defined in [40], poor scalability leads to poor performance. Hence, we

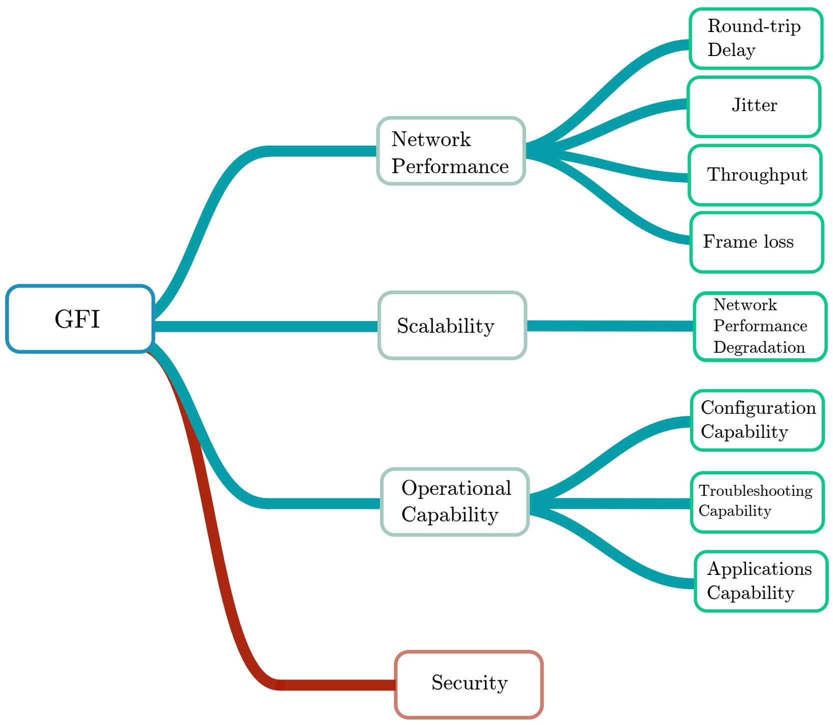


Figure 3.1: Feasibility indicators and metrics taxonomy

are measuring the scalability of transition tuples by analyzing their network performance degradation when the topology of the transition network grows.

For open environments, we are proposing *operational capability* as a feasibility indicator, which shows how a certain technology fits in with the existing environment or how it manages to solve operational problems. To the best of our knowledge, there are no associated metrics for operational feasibility of network devices in current literature. Consequently, we are introducing the following three metrics:

- *configuration capability*: measures how capable a network implementation is in terms of contextual configuration or reconfiguration.
- *troubleshooting capability*: measures how capable a network implementation is at isolating and identifying faults.
- *applications capability*: measures how capable a device is at ensuring compatibility with common user-side protocols.

One of the core challenges of the IPv6 transition is that of security. Quantifying security in a heterogeneous transition environment is one of the aspects we have not approached yet, but represents one of the future research directions.

3.2 Benchmarking approaches

As mentioned prior, we are combining the two major types of benchmarking methods: closed environment benchmarking, for the thoroughness and reliability of the quantified data, and open environment benchmarking, for its potential in identifying interoperability and operational issues.

3.2.1 Closed environment methodology

Inspired by the book *Art of Computer Systems Performance Analysis Techniques For Experimental Design Measurements Simulation And Modeling* [41], we are considering the following steps for our network performance analysis:

The goal of the study and the system boundaries:
The goal of this study is to compare the performance of 464 supporting IPv6 transition tuples. The evaluation study will be conducted so that outside components will have a minimum effect on the evaluation outcome.

System services and possible outcomes:
The service offered by the system is proper management of IPv4 and IPv6 traffic.

Metrics:
The study uses as feasibility metrics: round-trip delay, jitter, throughput and frame loss. Round-trip delay and jitter follow the latency guidelines given in RFC1242 [42]. Both metrics are measured in time with sufficiently fine units in order to distinguish the difference between two events. Considering current network speeds the proposed measurement unit is in millisecond (ms). Throughput and frame loss measurements also follow the guidelines of RFC1242 [42]. The proposed unit of measurement for throughput is in kilobit per second (Kbps). Frame loss is represented as the percentage of frames that are lost. Network performance degradation has no formal definition in current literature. Consequently, as a measurement procedure we are proposing additional network performance experiments under different network topology scales, and quantifying the relative change between results. The proposed unit of measurement is the percentage of relative change in network performance metrics. The scores are calculated using the Formula 3.1.

$$Relative\ change(x, x_{reference}) = \frac{x - x_{reference}}{x_{reference}} \times 100 \tag{3.1}$$

System parameters and factors:
The parameters that affect the network performance of the system are the software and hardware characteristics of the environmental setup, the **workload traffic**, the **IP version**, the **upper layer protocols**, the **IPv6 transition technology**, the **IPv6 transition implementation** and **the topology scale**. From these parameters, we are considering as factors the ones marked in bold font. In other words, we are planning to maintain as a constant only the software and hardware characteristics of the environmental setup.

Table 3.1: Workload *Framesize* × *Framerate*

No	Size	Rate 10 Mbps	Rate 100 Mbps	No	Size	Rate 10Mbps	Rate 100 Mbps
1	64	14,880	148,809	7	1,518	812	8,127
2	128	8,445	84,459	8	1,522	810	8,106
3	256	4,528	45,289	9	2,048	604	6,044
4	512	2,349	23,496	10	4,096	303	3,036
5	1,024	1,197	11,973	11	8,192	152	1,523
6	1,280	961	9,615	12	9,216	135	1,353

Experimental design:
A full factorial design is employed, hence $F1 \times F2 \times F3 \times F4 \times F5 \times F6 = N$ experiments are to be conducted. $F1, F2, F3, F4, F5, F6$ represent the values of each of the above mentioned factors. As suggested in RFC2544 [39], the duration of each experiment is 60 seconds after the first timestamp is sent.

Evaluation technique:
Since our goal is to benchmark working implementations, measurement is employed for the evaluation.

Workload:
The experimental workload is represented by the amount of traffic inserted into the experimental network. We have considered the combinations of frame size and frame rates displayed in Table 3.1. These have been recommended in RFC5180 [38] as maximum frame rates × frame sizes for 10 Mbps and 100 Mbps Ethernet. These workloads represent the benchmarking baseline for now and are used considering technical limitations imposed by the scalability tests.

Test setup Figure 3.2 presents the proposed test setup. The setup follows the recommendations of RFC5180 with a bi-directional traffic exchange between a sender and a receiver element and having the devices under test (DUTs) as forwarding components. To isolate the bottleneck effects of the proposed test setup, a Direct Connection (DC) test between the sender and the receiver is to be performed as well.

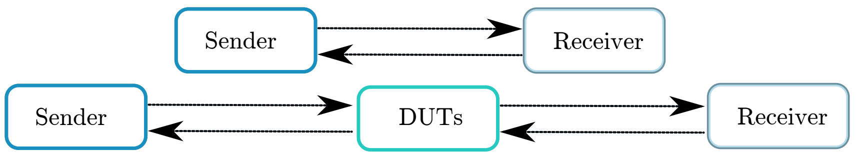


Figure 3.2: Test setup

Data Analysis:
Results will be categorized into categories according to the transition tuple employed. The margin of error is calculated to determine if there is a statistically-significant difference in means between the categories. The values of the margin of error are calculated using the Formula 3.2.

$$\begin{array}{l} Margin\ of\ Error = z_{\alpha/2}\frac{\sigma}{\sqrt{n}} \\ \sigma - standard\ deviation, n - sample\ size \\ z_{\alpha/2} - critical\ value, z_{\alpha/2} = 2.576\ for\ 99\%\ level\ of\ confidence \end{array} \tag{3.2}$$

The repeatability of the experiments will be assessed by calculating the relative standard error, for which we are using the Formula 3.3.

$$\begin{array}{l} Relative\ Standard\ Deviation = \frac{\sigma}{\bar{x}} \\ \sigma - standard\ deviation, \bar{x} - mean \end{array} \tag{3.3}$$

Data presentation:

The final results will be plotted as a function of frame size. The graphs will show comparative results of the tested implementations as much as possible.

3.2.2 Open environment methodology

We are using operational capability as feasibility indicator for measuring feasibility in an open environment, and configuration capability, troubleshooting capability and applications capability as proposed metrics. In our attempt to measure operational capability by using these metrics, we have chosen a task-based non-exhaustive approach. For configuration and troubleshooting capability we have proposed 10 tasks each. These configuration and troubleshooting tasks are designed to verify the existence of basic configuration and troubleshooting means, which should not be missing from any such implementation.

The proposed method for measurement is assisted survey, in which the participants are assisted. This method should isolate the human factor, and potential usability problems as much as possible. The score is calculated as a percentage of successful tasks over the total number of tasks (e.g. 7/10 = 70%), which should easily indicate the most feasible transition tuple. The data collected is organized in Higher is Better (HB) score tables. In the case of applications capability, we are proposing a list of 20 common user-side applications to be tested in relation with the transition tuples.

For configuration capability, we are considering a number of configuration tasks, which have been inspired by the abstracted guidelines presented in [43]. The tasks can be organized in three generic groups, *initial setup*, *reconfiguration* and *confirmation.* For ease of reference, we are associating each task with a task code, in accordance with the respective group association.

1. IinitialSetup1: configure an encapsulation/translation virtual interface using a command line interface or a graphical user interface
2. IinitialSetup2: Save the current temporary configuration commands in a file which can be loaded at start-up
3. IinitialSetup3: Self configuration according to contextual configuration details
4. InitialSetup4: Display warnings in the case of misconfiguration and reject the misconfigured command

5. InitialSetup5: Display warnings in the case of missing command and reject saving the temporary configuration
6. InitialSetup6: Display contextual configuration commands help
7. Reconfiguration1: Convert current configuration settings to configuration commands
8. Reconfiguration2: Back-up and restore the current configuration
9. Confirmation1: Show the current configuration
10. Confirmation2: Show abstracted details for the 464 virtual interface

The configuration capability result is expressed as a percentage of successfully completed configuration tasks from the total number of tasks.

Similarly, for troubleshooting capability, we propose a number of troubleshooting tasks. The tasks follow the fault isolation, fault determination and root cause analysis (RCA) guidelines presented in [43]. Consequently, the tasks can be organized into the three generic categories: *fault isolation*, *fault determination* and *root cause analysis (RCA)*. For ease of reference, these tasks were associated as well with group codes:

1. FaultIsolation1: Capture and analyze IPv4 and IPv6 packets
2. FaultIsolation2: Send and receive contextual ICMP messages
3. FaultDetermination1: Identify a misconfigured contextual route
4. FaultDetermination2: Identify a misconfigured contextual line in the virtual 464 interface configuration
5. FaultDetermination3: Perform self-check troubleshooting sequence
6. RCA1: Log warning and error messages
7. RCA2: Display log
8. RCA3: Display in the user console the critical messages with contextual details
9. RCA4: Log statistical network interface information
10. RCA5: Display detailed statistical network interface information

The troubleshooting capability result is also expressed as a percentage of successful tasks of the total number of troubleshooting tasks.

Inspired by the efforts presented in [36], we are testing a non-exhaustive list of common user applications in relation with the 464 transition technologies in order to measure applications capability. The applications are organized into the following categories: browsing, E-mail, Instant Messaging (IM) and Voice over IP (VoIP), Virtual Private Networks (VPN), File Transfer Protocol (FTP), Cloud and Troubleshooting. For now the list includes 20 popular applications, which are presented in Table 5.6. The measurement result is presented as a percentage of successfully-tested applications from the total number of applications.

3.3 Empirical results summarization

As recommended by RFC2544 [39], the summarizing function for the network performance metrics is *mean* and the data is presented as a function of frame size. To compile the different empirical results into one single score, we need to establish a summarizing function. According to [41] the simple flow chart shown in Figure 3.3 can be used to decide the most suitable function.

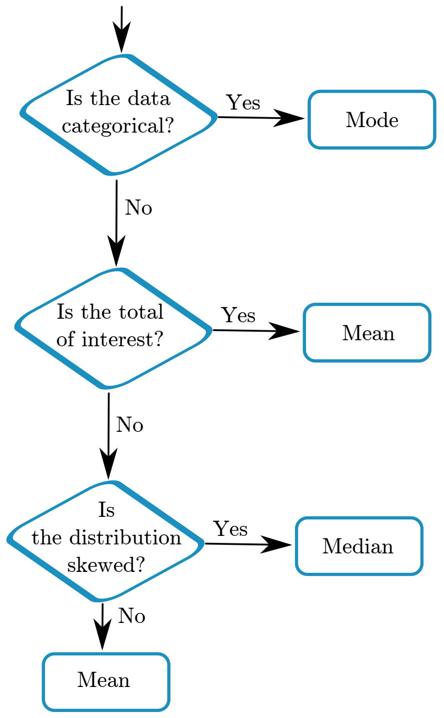


Figure 3.3: Mean, mode and Median flow chart

The type of data measured is the first test. If the data is categorical (e.g. blood types of a person), the most appropriate summarizing function is mean. For network performance we do not have any categorical data, hence this does not apply for now. The second consideration is regarding the total of the observations. If the total is of interest, the mean is recommended. The total is not considered of interest in our case , hence we need to use only the last question, which regards the probability distribution of the data. Therefore, to decide between *mean* and *median* we need to analyze the probability distribution of the empirical results. If the distribution is skewed, we use the *median*. Otherwise the *mean* is used.

3.4 Building the composite indicators

Given the variety of the proposed metrics and units of measurements, the feasibility Indicators are meant to enhance the benchmarking process. However, this is no small task, as poorly-constructed indicators may lead to over-simplistic or misleading conclusions. In [44], M. Nardo et al. have detailed the steps needed to a build a trustworthy composite indicator. Given that the handbook is dedicated to composite indicators meant for social sciences, some of the steps need to be adjusted to our needs. Here are the steps we deem necessary:

The theoretical framework details the purpose of the composite indicator and its substructure. The General Feasibility indicator is designed to measure the feasibility of IPv6 transition tuples in relation to a particular transition scenario. To that end, the indicator is meant to reflect the feasibility of transition tuples considering multiple feasibility dimensions. Figure 3.1 shows the preliminary taxonomy of targeted feasibility indicators and metrics. However, the current taxonomy is far from exhaustive. One major missing dimension is security, which represents a very important aspect of the IPv6 transition, and one of our priorities for future research.

The explored feasibility dimensions are currently network performance, scalability and operational capability. The three feasibility indicators capture important aspects of the feasibility spectrum of transition tuples and are different enough to be represented as sub-indicators. The main reason for combining the three feasibility aspects into a unique score is for the sake of benchmarking, as separate scores may not always identify a Pareto-optimal choice. However, since GFI is supposed to capture the core feasibility aspects of transition tuples, the lack of a security metric renders it premature. Consequently, expressing GFI as a single score remains a future goal. For now, the feasibility of transition tuples is to be assessed through the three feasibility sub-indicators: *network performance, scalability* and *operational capability*, as they were described in Section 3.1.

Regarding *network performance*, the 4 feasibility variables *delay, jitter, throughput* and *frame los* capture different aspects of network performance while being coherent enough to describe the same feasibility dimension. Therefore, we contend the 4 metrics can be integrated into the same sub-indicator. *Scalability*, which is expressed as the relative change of the 4 network performance indicators, considering scale growth, can be obtained by integrating the 4 variables. Similarly, our point of contention is the metrics capture different aspects of feasibility, but can describe the same feasibility dimension. Regarding *operational capability*, the proposed metrics, *configuration, troubleshooting* and *applications* have a very similar approach on measurement, while observing quite different operational aspects of transition tuples. Therefore, we contend they can be expressed using a single composite indicator.

The variables The three sub-indicators: *network performance*, *scalability* and *operational capability* are calculated from two types of variables. On one hand, network performance and scalability, which use as underlying feasibility metrics round-trip delay, jitter, throughput and network performance degradation, have a quantitative (hard) nature. On the other hand, the operational capability data is based on a non-exhaustive survey, which although comparable, can still be considered qualitative (soft) data. The non-exhaustive approach somewhat limits the broad range of potential operational issues, but we contend this can suffice as a first step, and we would like to improve it in the future.

Multivariate analysis is necessary to understand the suitability of the underlaying data and the interrelationships between the sub-indicators. The analysis can be achieved along two general directions: grouping of the data in lower-dimensional sub-indicators and grouping of the data according to the transition tuple. The nested structure of the composite indicator and sub-indicators needs to be balanced. It can be checked with expert opinion and different statistical analysis tools.

One of the very popular statistical multivariate analysis tools is Principal component analysis (PCA), and it is mainly used for dimension reduction through linear combination of the different sub-components of a set. PCA is able to summarize a set of sub-indicators while keeping the maximum proportion in the data set intact. However, the linear correlations might not describe very well the influence of the sub-indicators on the measured phenomenon. Furthermore, it is considered sensitive to the revisions in the data (e.g. new transition tuples) and small-samples. Nevertheless, PCA can prove useful in assessing how balanced the nested structure of composite indicators is. Even so, a minimum sample size should be provided. The minimum sample size is still a subject for debate in current literature. One of the general rules of thumb presented in [44] underlines that the cases-to-variables ratio should be no lower than 3. Another rule dictates that there should be at least 10 cases for each variable. Other presented rules recommend higher values for the number of cases, varying from 51-200 recommended cases. For the proposed sub-indicators the situation is as follows: *5:4* cases-to-variable for network performance, *5:4* for scalability and *5:3* for operational capability. According to the two least demanding sample size rules, PCA does not seem applicable in our case. Considering the small number of cases, Nardo et al. [44] recommend avoiding the use of multivariate analysis, as the results do not have known statistical properties. The analysis can however prove useful in the future, as the number of studied transition tuples should increase.

Grouping of the transition tuples data according to a higher category, should prove useful for a large sample size. However, we are currently targeting only a small number of transition tuples, and would like to clearly distinguish the feasibility of each one of them.

Missing data can affect the robustness and trust-value of a composite indicator. In most cases for the proposed methodology, missing data can be easily re-obtained through repetition of the particular experimental instance. However, when the experiment cannot be repeated following the same characteristics, we are considering a method for data imputation. There are several methods for dealing with missing data: case deletion, single imputation or multiple imputation. Case deletion simply ignores the missing dataset from the analysis. This is not suitable for our methodology, as it leads to the omission of systematic differences which can be critical for the benchmarking process. The other two alternatives consider data imputation models, such as mean/median/mode substitution and regression imputation for single imputation, or Markov Chain and Monte Carlo algorithms for multiple imputation. By using imputation, time and resources can be saved, while retaining an otherwise unusable dataset . The downside of using data imputation is the uncertainty it introduces, which needs to be analyzed. This is the reason, we are considering the use of complete datasets, or experiment recreation as much as possible. In the worse case scenario, given the sample sizes employed and the type of data, we are proposing the use of single imputation, by replacing the missing value with the dataset mean/median, after analyzing the probability distribution.

Normalization is required prior to aggregation, whenever the indicators have different units of measurement. Given the variety of scales and measurement units, normalization is necessary in our case. There are various methods for normalization described in current literature. One of the examples is ranking, which is considered simple and effective. The main drawback of ranking is represented by the inherent loss of performance levels. This makes ranking hard to integrate with our research goal. Another example of normalization is re-scaling, which is used to get the scale the dataset values within the same range (0;1). While easy to use, re-scaling can widen the range of indicators lying within a small interval, making it harder to point out extraordinary behavior.

Another alternative normalization method is that of distance to a reference. Its main disadvantage lies in its sensitivity to extreme values, such as outliers. However, since we plan to use it with summarized data, it appears to be the most suitable option. Given the different scales of measurement, the reference for the data must be discussed on a per-feasibility-metric basis. Similarly, we need to approach the data transformation which may be needed, considering the non-linear behavior of some of the feasibility metrics, such as *frame loss*. A deciding factor for the data transformation function is the probability distribution of the data. If the data is normally-distributed, a linear transformation is advised, while for a skewed-distributed dataset, a logarithmic transformation is more appropriate. Another important aspect is the direction of the scale, which needs to be the same for the variables meant to be part of a composite indicator. Since we would like for GFI to have a higher is better (HB) tendency, it should be the same for the sub-indicators. Taking into consideration these guidelines, the normalization process for each feasibility metric is presented below.

Round-trip delay depends highly on the underlying network infrastructure. The direct connection (DC) between the Sender and Receiver of the generated traffic can be used to quantify the limitations of the experimental infrastructure. Hence, it can be used as reference value. Regarding data transformation, if the data-set has a normal distribution, a linear transformation of the data is advised. In the case of a skewed distribution a logarithmic distribution can be used to correct the skewness. Considering the general skewness of the dataset presented in Figure 5.5a, a logarithmic transformation seems to be the suitable solution. Regarding the lower is better (LB) characteristic of the metric, we are proposing the following normalization formula:

$$Normalized\ D = \log_{10} \tfrac{D_{DC}}{D} \tag{3.4}$$

Using the Formula 3.4 we obtain a HB tendency and 0 as a upper convergence point.

Jitter is limited as well by the benchmarking environment. Similarly, the DC data can be used as reference. In terms of data transformation, the jitter dataset has a skewed distribution as shown in Figure 5.5b. Considering the LB tendency, the proposed formula is:

$$Normalized\ J = \log_{10} \tfrac{J_{DC}}{J} \tag{3.5}$$

After using the formula the normalized data changes to a HB characteristic, with negative values converging onto 0.

Throughput is limited by the amount of workload, expressed as $Framesize \times Framerate$, which should be considered as reference value. The measured data has a skewed distribution as displayed in Figure 5.5c. Consequently, a logarithmic transformation should be performed. Considering the HB characteristic, the proposed formula for normalization is:

$$Normalized\ T = \log_{10} \tfrac{T}{Framesize \times Framerate} \tag{3.6}$$

The normalized data should keep its HB tendency and have 0 as upper convergence point.

Frame loss is expressed as a LB percentage and could be simply normalized by using the opposite of its values. However, frame loss has an exponential impact on network performance, which should be reflected in the normalized scores. Consequently, we are proposing the use of a logarithmic transformation. Considering the LB characteristic and the fact that $log_{10}0$ is not defined, we are proposing the following formula:

$$Normalized\ FL = -\log_{10}(FL + 1) \tag{3.7}$$

The normalized values will have a HB tendency and will converge to 0.

Network Performance Degradation (NPD) is expressed as percentage value with a LB characteristic. Keeping in mind that it is quantified as the relative change of network performance, it should ultimately suffer a similar logarithmic transformation in order to be integrated into the same indicator. Hence, the proposed normalization formula is:

$$Normalized\ NPD = -\log_{10}(NPD + 1) \tag{3.8}$$

The normalization process will lead to negative HB scores converging to 0.

Operational capability results have a similar situation, being expressed as a percentage of successful tasks over total tasks, and having a HB characteristic. Most of the tasks are verifying the existence of essential features, which any transition implementation should not do without. This leads to an exponential impact on operational capability for missing tasks. This can be as well reflected into the scores through the logarithmic transformation. Considering the HB tendency, the proposed formula is:

$$Normalized\, OC = \log_{10}(OC + 1) \tag{3.9}$$

The normalized data will have HB positive values with 0 as lower convergence point.

Weighting and aggregation can have a significant effect on the proposed sub-indicators. This methodology is dedicated to entities involved in the IPv6 transition process, for which the interest in feasibility indicators might fluctuate according to the domain of activity. Keeping that in mind, we would like to propose an open weighting strategy, in which any interest entity can assign customized weights and obtain scores according to their preference. As a showcase, we will present how the open weights can be used considering a hypothetical entity in Section 5.5.

One alternative weighting system is equal weighting, in which the variables are given equal weights. This is the most popular weighting mechanism, but in some cases it can lead to double counting, for example in the case of highly correlated variables which measure similar performance aspects.

Another weighting approach which can be suitable to our goal is the Analytic hierarchically process (AHP). This technique can be complementary to the open weights method by a hierarchical decomposition of the weighting process. However, throughout the process the weights are seen as trade-offs across indicators, not as importance coefficients. The core process for AHP is represented by a pair-wise comparison of the importance of sub-metrics. One question needs to be answered for each sub-metric/sub-indicator pair: *Which of the two do you find more important, on a scale of 1 to 9 ?* A preference value of 1 should indicate equal importance, while 9 means that the specific sub-metric is 9 times more important than the other. The results are presented in a comparison table, and the weights are expressed using an eigenvector. One drawback of AHP is represented by the human factor. Inconsistency in answering the questions can lead to a poor design of the weights. However, the inconsistency is part of human nature. As opposed to simply assigning arbitrarily chosen weights, for AHP the inconsistency can be measured and contained. As recommended by Nardo et al. [44], the value for the *Saaty's inconsistency* should be less than 0.1%. AHP has the disadvantage of being more computationally complex, but it results in weights which are less sensitive to errors of judgment. The way AHP is used for the proposed sub-indicators is shown in Section 5.5.

An important aspect of weighting, which should not be overlooked is represented by double-counting. This can happen when highly correlated variables are given equal weights. The correlation can be tested through statistical mechanisms, such as the Pearson correlation coefficient. As there will almost always be a positive degree of statistical correlation among aggregated variables, double counting should not only be determined by statistical analysis but, also by the analysis of the relation between the indicator itself and the sub-structure, with respect to the described phenomenon.

In terms of aggregation, the use of logarithmic transformations for the normalization process of all feasibility metrics, makes the linear aggregation Formula 3.10, the most appropriate method for obtaining the three sub-indicators: network performance, scalability and operational capability. In the case of strictly positive and normalized data, a suitable alternative is represented by geometric aggregation.

$$\begin{aligned}&Composite\ indicator = \sum_{i=1}^{n} w_i S_i\\&\sum_{i=1}^{n} w_i = 1, 0 < w_i < 1\\&w_i - weight\ assigned\ to\ subindicator\ i\\&S_i - \ score\ of\ subindicator\ i\end{aligned} \tag{3.10}$$

By looking at the normalized empirical results of n feasibility metrics as coordinates in an *n-dimensional* space of *transition tuple vectors*, the Euclidean distance to origin, or the *Euclidean norm* represents a meaningful quantity for the feasibility of each transition tuple. This can be considered an alternative unweighted aggregation strategy, which indicates Pareto-optimal solutions across all measured sub-indicators. Given the nature of the normalized scores of the sub-indicators such as network performance and scalability, which are converging to 0 as an upper bound, to keep a higher is better (HB) tendency, the opposite of Formula 3.11 should be used.

$$\begin{aligned}&Composite\ indicator = \sqrt{\sum_{i=1}^{n} S_i^2}\\&S_i - \ score\ of\ subindicator\ i\end{aligned} \tag{3.11}$$

Uncertainty and Sensitivity are important to the robustness of the composite indicators. An efficient uncertainty or sensitivity analysis needs historical data, as well as a thorough analysis of factors which can generate uncertainty in the composite indicator. That translates in the analysis of different indicator sub-structures, alternative data imputation models, alternative normalization schemes or weighting and aggregation methods. Given the complexity and time consuming nature of such analysis, as well as the lack of historical data, this represents one of our future research goals. The lack of an uncertainty and sensitivity analysis underlines the preliminary nature of the proposed composite sub-indicators, as their robustness is unknown, for now.

De-construction is needed to ensure the transparency of the composite indicator. While GFI and the sub-indicators provide an easy way to compare the results, and can provide an useful summary regarding the feasibility of transition tuples, the contribution of the substructure should remain transparent. Currently the composition of the composite sub-indicators is transparent and we intend to maintain this level of transparency for the future. Graphical attempts of displaying the decomposition of propsed composite indicators are shown in Section 5.5.

Presentation and dissemination is a key aspect of a composite indicator, as it can represent the basis for decisions. A composite indicator, as well as the contribution of its substructure, should be easily interpretable. In order to address this, we are proposing the use of feasibility charts.

The first chart will show the normalized results for the network performance metrics: round-trip delay, jitter, throughput and frame loss with the associated weights. The normalized data is multiplied by 10^4 to have at least 4 significant digits and 2 sub-digits. The last two columns of the chart will be the value of the composite sub-indicator, *network performance (NP)* according to the weighted linear aggregation Formula 3.10, and the Euclidean norm value calculated with the Formula 3.11. A model of the chart is presented in Table 3.2.

Similarly, the network performance degradation (NPD), associated with the *scalability (SCAL)* sub-indicator is presented in the second chart. The chart will follow the network performance model chart shown in Table 3.2.

Finally, the third chart will present the configuration (CC), troubleshooting (TC) and applications capability (AC) normalized results, and the two aggregated values. The model for the chart is presented in Table 3.3. All charts should contain pre-ordered values from best to worse considering the HB tendency of the composite indicators.

Table 3.2: Network Performance

	Delay w_d	Jitter w_j	Throughput w_t	Frame loss w_{fl}	**NP**	**NP E-norm**
transition tuple 1	XXXX.XX	XXXX.XX	XXXX.XX	XXXX.XX	**XXXX.XX**	**XXXX.XX**

Table 3.3: Operational capability

	CC w_{CC}	TC w_{TC}	AC w_{AC}	**OC**	**OC E-norm**
transition tuple 1	XXXX.XX	XXXX.XX	XXXX.XX	**XXXX.XX**	**XXXX.XX**

In terms of dissemination, the feasibility charts will be made public through the project webpage, which for now is www.ipv6net.ro. The scores will be updated as often as necessary and the charts will be identified with number sequence following the pattern $< Composite\ indicator >_{<year>.<month>.<day>}$.

Chapter 4

IPv6NET concept

The benchmarking methodology we are proposing is associated with a heterogeneous IPv6-IPv4 testbed, which we entitled the IPv6 Network Evaluation Testbed (IPv6NET). The IPv6 Network Evaluation Testbed (IPv6NET), introduced in [45], is dedicated to quantifying the feasibility of IPv6 transition implementations in relation to specific network scenarios.

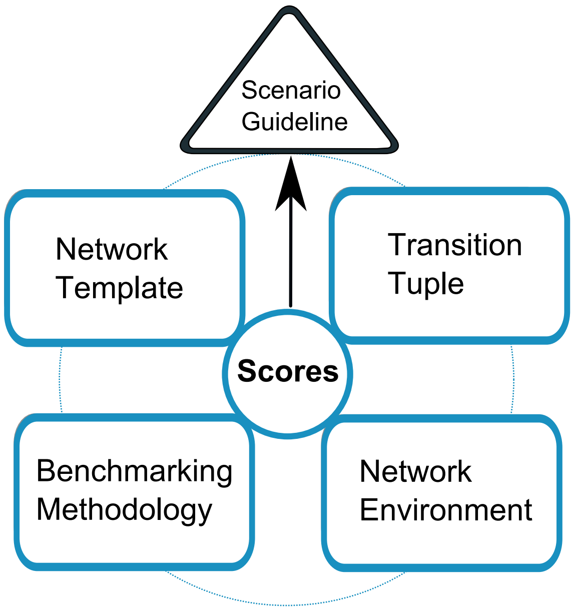


Figure 4.1: IPv6NET Concept

As presented in Figure 4.1, conceptually IPv6NET has four important components:

- **The benchmarking methodology** which dictates the coordinates of the conducted network tests.
- **A network template** associated to a specific transition scenario.
- **A network environment** needed as base for the experimental networks.

- **A transition tuple** represented by at least one transition implementation covering one transition technology.

By combining the four components we will obtain feasibility scores, which can in turn represent the base of a transition scenario guideline.

The proposed benchmarking methodology is presented in Chapter 3. The rest of the IPv6NET components are detailed in the following sections.

4.1 Network template

The network template is associated with a specific network scenario. The scenario targeted in this thesis was introduced by the IETF in RFC4057[3] as Scenario 3. It is dedicated to an enterprise which decided to use IPv6 as the main protocol for network communications. Some applications and nodes, which are IPv4-capable would need to communicate over the IPv6 infrastructure. In order to achieve this, the Enterprise would need to apply an IPv6 transition technology, which would allow both protocols to coexist in the same environment. For simplicity, suitable technologies for this scenario can be referred as *464 technologies*.

The basic, small scale template for 464 technologies shown in Figure 4.2 is composed of a set of network routers: a Customer Edge (CE) router which encapsulates/translates the IPv4 packets in IPv6 packets, and a Provider Edge (PE) router, which handles the decapsulation/translation from IPv6 back to IPv4. The IPv4-only backbone is used for forwarding the IPv4 traffic. The IPv6 traffic would be directly forwarded by the IPv6 backbone.

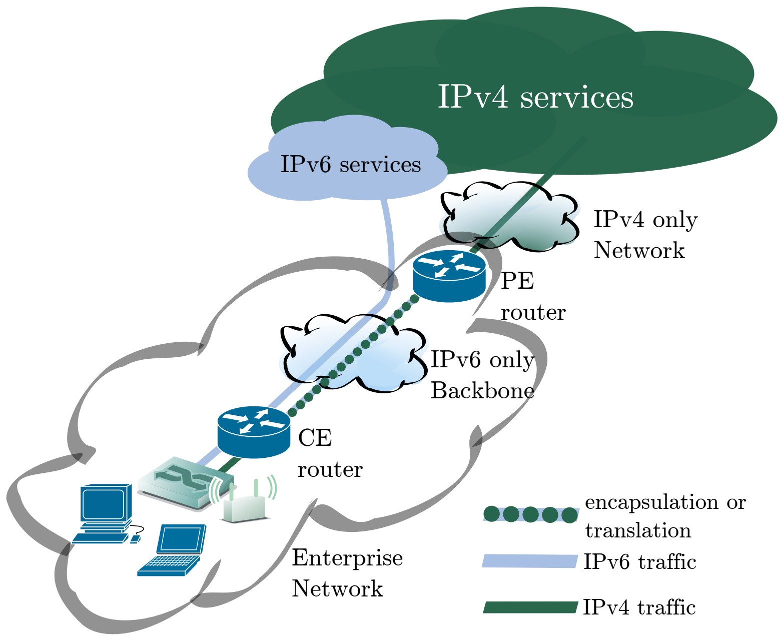


Figure 4.2: 464 Network template

4.2 Transition tuples

Multiple technologies can be considered suitable for the 464 scenario: MAPe [22], MAPt[26], DSLite[17], 464XLAT[27], SA46T[46]. Some implementations supporting these technologies have been proposed. One of those is the *asamap vyatta* distribution[47], an open source transition implementation which covers 4 of those technologies: MAPe, MAPt, DSLite and 464XLAT. Another open source implementation is *tiny-map-e* [48], which covers only MAPe.

We are analyzing the feasibility of five transition tuples, which we encoded as presented in Table 4.1. The table also presents the host Operating System (OS) details.

Table 4.1: 464 Transition tuples

Tuple	Transition technology	Transition implementation	OS
asamape	MAPe	asamap	Vyatta
asamapt	MAPt	asamap	Vyatta
asamapdslite	DSLite	asamap	Vyatta
asamap464xlat	464XLAT	asamap	Vyatta
tinymape	MAPe	tiny-map-e	Ubuntu server

4.3 Network environment

Considering the benchmarking approaches we have proposed, we are using two types of network environments: closed and open network environments. Structural details of the two types are presented in the next subsections.

4.3.1 Closed network environment

The closed experiment's design, presented in Figure 4.3, follows the basic network template, including one Customer Edge (CE) machine and one Provider Edge (PE) machine, which in the context of scalability benchmarking we are calling 1×1 topology.

For the underlying infrastructure of the closed experiments we are using StarBED [49], a large scale general purpose network testbed, administered by the National Institute of Information and Communications Technology (NICT) of Japan. Four computers were used for this experiment: two for the devices under test (DUT), 464 PE and 464 CE, and two for the benchmarking platform. The benchmarking platform computers have used Ubuntu 12.04.3 server as base operating system. The traffic was generated using the Distributed Internet Traffic Generator (D-ITG) [50]. One of the computers performed the ITGSend function, generating the traffic, while the other ran the ITGRecv function, receiving the generated traffic and redirecting it back to the ITGSend machine. The ITGSender was also responsible for reporting the network performance of the traffic flow.

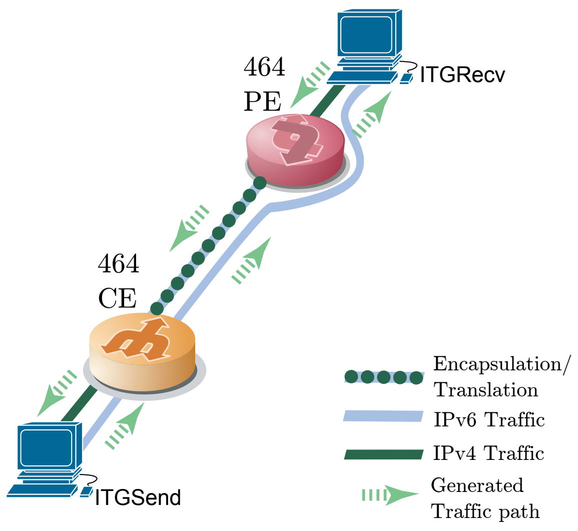


Figure 4.3: 464 Closed network environment

In order to test the scalability of the transition tuples in the context of topology growth we are considering one more topology step for the devices under test: 10 × 1, where the larger number is the number of CE machines connected to the same BR machine. The 10 × 1 setup is presented in Figure 4.4.

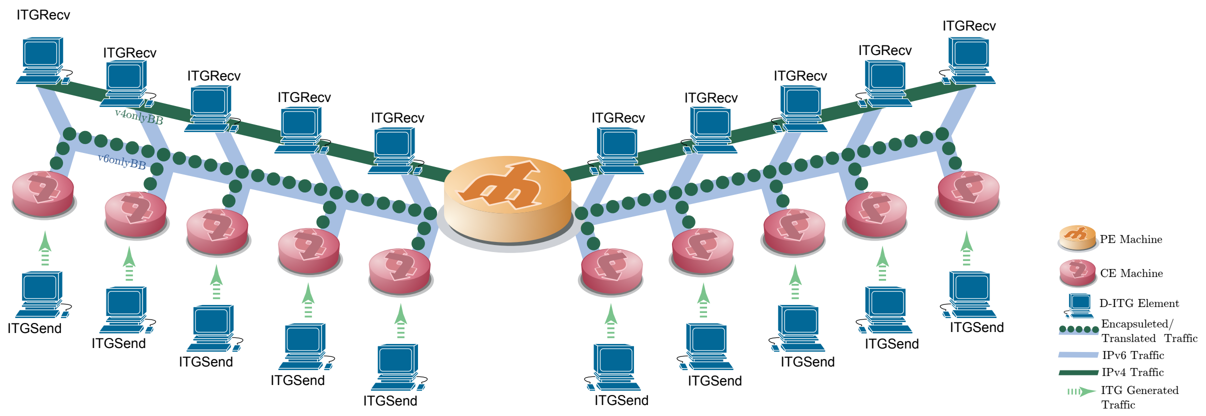


Figure 4.4: 10 × 1 Closed network environment

The setup follows the design principles of the 464 basic network template, and generates the topology growth by increasing the number of CE machines. One of the parameters than can limit the scale of the experimental environment is the mapping rules employed for MAPe or MAPt. In the case of MAPe and MAPt environments we have used the following mapping rule:

- IPv4 prefix: 198.18.1.0/24

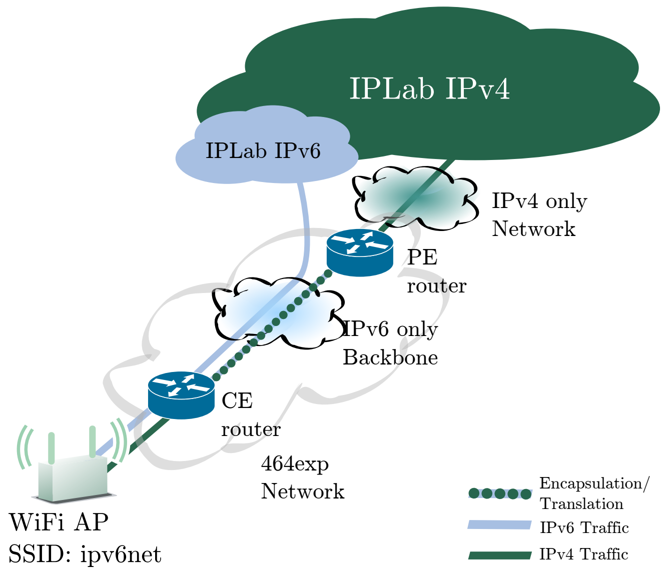


Figure 4.5: 464 Open network environment

- IPv6 prefix: 2001:200:16a::/48
- Embedded Address (EA) bits: 16

From the mapping rule the following limitations can be calculated :

Used IPv4 addresses: 256
This can be derived from the IPv4 prefix's lenghth: $2^{32-[IPv4\, prefix\, length]} = 2^8$.

Maximum number of supported users: 65536
Calculated from the EA bits: $2^{[EA\, bits]} = 2^{16}$

IPv4 address sharing ratio 1 to 256 Each public IPv4 address is shared between 256 users. This is given by the Port Set ID (PSID) length, calculated as:
$[Network\ ID\ length] - [IPv6\ prefix\ length] - [IPv4\ prefix\ length] = 64 - 48 - 8 = 8$.
The IPv4 sharing ratio is: $2^{[PSID]} = 2^8$

Port sharing ratio: 1 to 256
Each user disposes of 256 ports. This is calculated as a difference between the total number of ports, 2^{16} and the number of ports assigned to each CE machine $2^{[PSID]}$, hence: $2^{16} - 2^8 = 256$.

Considering the isolated nature of the experimental environment, these limitations can be adjusted by changing the mapping rule to accommodate the needed network scale. For the targeted scale 10×1, this rule is more than sufficient.

In large network testbeds the network flows can be isolated using Virtual LAN (VLAN) technology. Although it is not mandatory, we are considering a separate VLAN for each of the CE machines. This should help reduce the probability of background traffic affecting the experimental results. In this context, the available number of VLANs in the underlying infrastructure can limit the experimental scale as well. As standardized by [51], the theoretical maximum of usable VLANs is 4096, but in practice at least 2 are reserved. As this is not a requirement, in the event of an insufficient number of available VLANs, the available ones should be reassigned accordingly.

4.3.2 Open network environment

The open experiment topology, presented in Figure 4.5 also follows the basic, small scale 464 network template.

The major difference is that the benchmarking platform is replaced by open up-link and down-link connections. We have built this type of environment as part of a bigger experimental network, which supplies Internet access to the members of the Internet Engineering Laboratory. The 464 network consisted of two virtual machines, the Customer Edge machine (CE) and the Provider Edge machine (PE). The two machines have ran on a virtual environment running Citrix Barebone XenServer 6.0 as hypervisor. Previous experiences with building and analyzing a similar 464 open environment are presented in [52]. On the up-link, the IPv4 and IPv6 traffic was routed by a dual-stack core router. The survey participants were able to connect to the environments through a single SSID, *ipv6net*, handled by a WiFi access point.

Chapter 5

Empirical results

5.1 Closed Experiment results

5.1.1 Network performance

The network performance of the transition tuples is compared with a Direct Connection setup, in which the two test platform servers were connected directly. The results are graphed as a function of frame size. The error bars represent the margin of error for the mean, calculated at a 99% level of confidence.

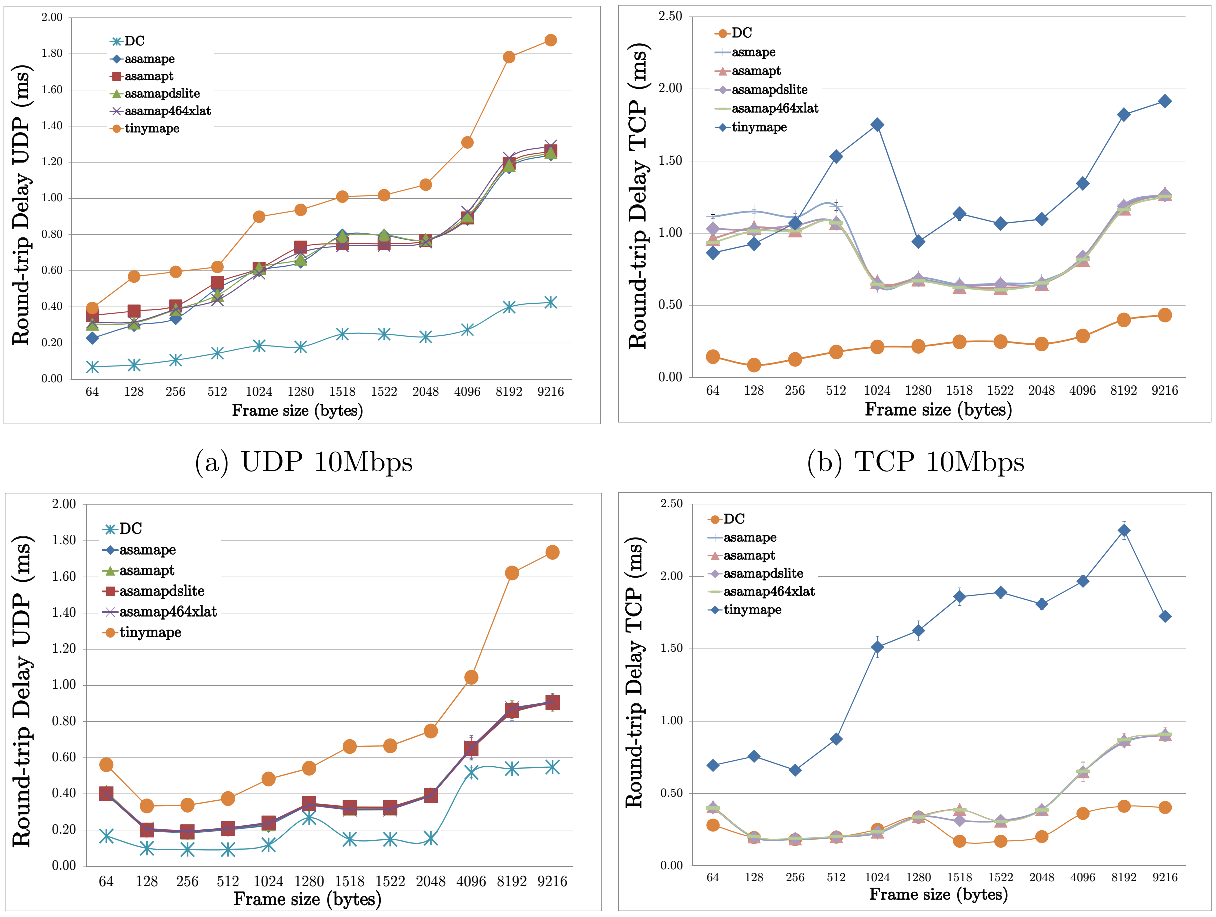
The round-trip delay graphs shown in Figure 5.1 indicate a better general performance for the asamap transition tuples, by comparison with the tinymape tuple. The jitter results are presented in Figure 5.2, while the throughput results are shown in Figure 5.3. In terms of jitter and throughput, the tinymape tuple shows a lower general performance. As it is difficult to observe from the graphed data, other insights can be obtained by analyzing the summarized data.

In the case of the 100 Mbps workload results , presented in Figures 5.1c, 5.1d for delay, in Figures 5.2c, 5.2d for jitter, and Figures 5.3c 5.3d for throughput, the high values of the Margin of Error do not allow us to draw any clear conclusion regarding the difference between the asamap tuples. However, the results help to point out some unexpected behaviors, which are consistent. One example of unexpected behavior is the decrease in throughput for the *1280 frame size*, presented in Figure 5.3c and Figure 5.3d. Another example of unexpected behavior is the lower throughput of the Direct Connection, which can be considered counter-intuitive. The root causes of these behaviors need further analysis.

The loss rates, with the exception of some outliers, are very close to 0. For the outliers, the maximum loss-rate is approximately 0.003%, considered negligible in most cases.

5.1.2 Scalability

Figure 5.4 presents the network performance degradation results of the transition tuple *asamape* tuple, under the two different topology scales: 1×1 and 1×10. The throughput results (5.4a, 5.4b) show a moderate performance degradation, with the results of the



(a) UDP 10Mbps (b) TCP 10Mbps

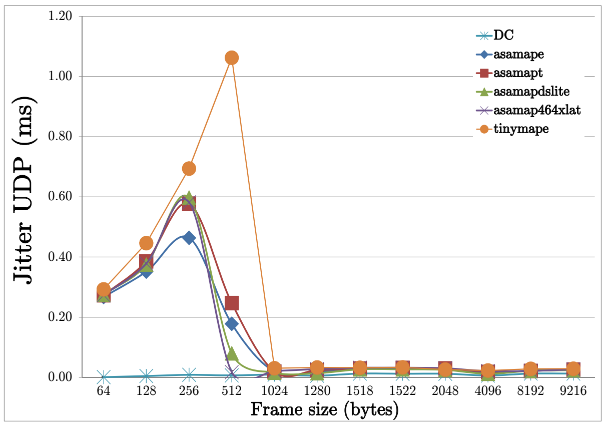
(c) UDP 100Mbps (d) TCP 100Mbps

Figure 5.1: Delay results

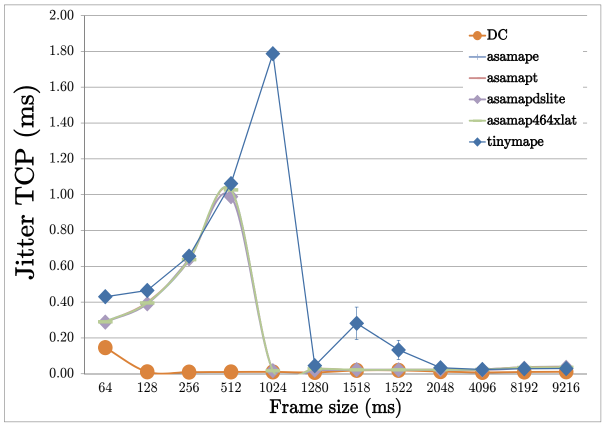
small frame sizes being most affected by the topology growth. The round-trip delay (Figures 5.4c, 5.4d) was dramatically affected in the case of the *64* frame size. The rest of the results show a considerable performance degradation as well. For jitter, the network performance degradation seems to have been equally distributed among the 12 frame sizes. More insights regarding the network performance degradation can be drawn after the results summarization process in the next section.

5.2 Summarized results

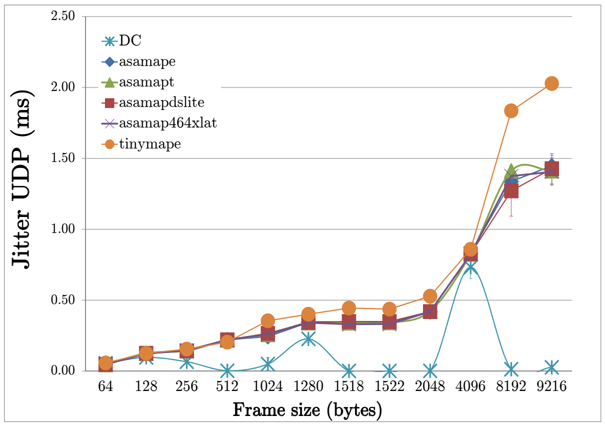
To compile the different frame sizes results into one single score, we need to establish a summarizing function. The probability density function of the gathered datasets is presented in Figure 5.5 for the 10Mbps data, in Figure 5.6 for the 100Mbps results, and in Figure 5.7 for the 10CE network performance degradation data. The density function is calculated respecting the kernel density estimation guidelines presented in [53]. Considering the overall skewness of the datasets it looks like the *median* is the most appropriate choice



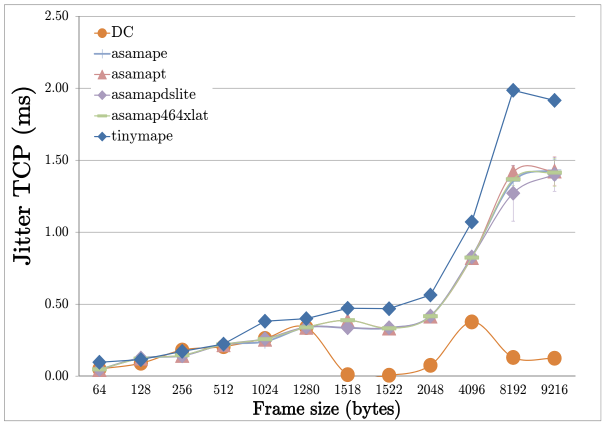
(a) UDP 10Mbps



(b) TCP 10Mbps



(c) UDP 100Mbps



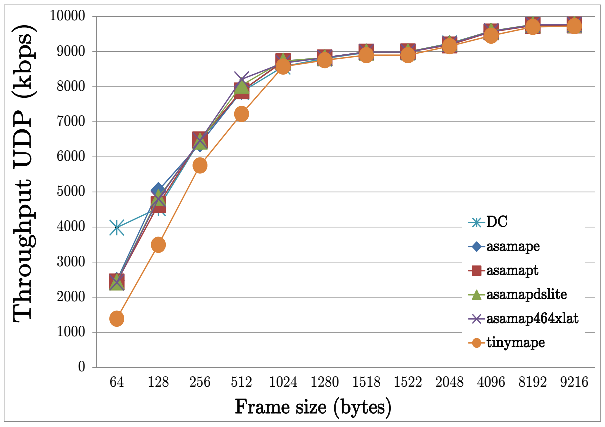
(d) TCP 100Mbps

Figure 5.2: Jitter results

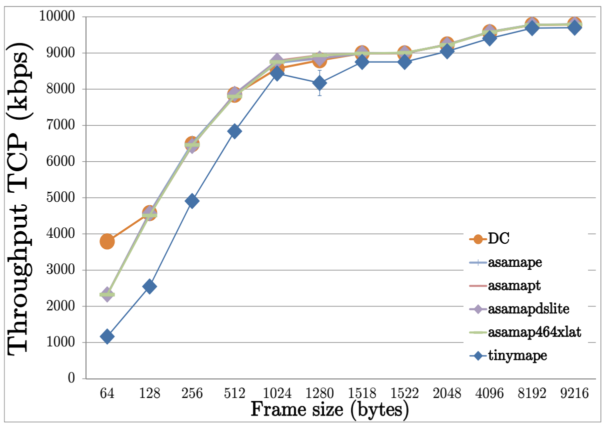
for the summarization process.

5.2.1 Network Performance

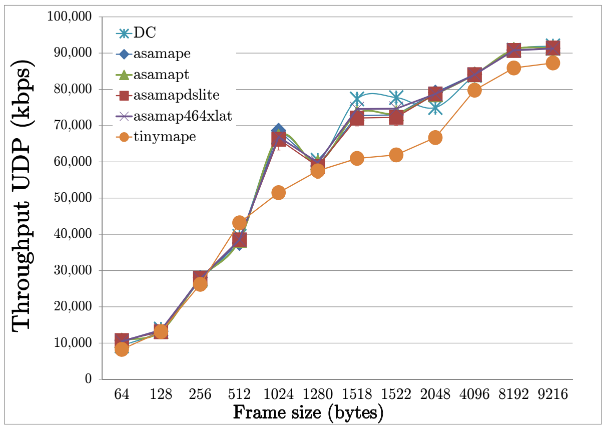
Consequently the network performance results for the 10Mbps and 100Mbps workloads have been summarized in Table 5.1 and Table 5.2 respectively. The 10Mbps summarized results confirm the better performance of the asamap tuples. Furthermore we are able to identify some performance trends like the better delay and jitter results for translation-based transition tuples: *asamapt* and *asamap464xlat*, or the better throughput of encapsulation-based transition tuples: *asamape* and *asamapdslite*. The 100Mbps results confirm the generally lower network performance of the tinymape transition tuple. The throughput performance trend is confirmed as well, with better results for encapsulation-based transition tuples. Unfortunately, the high margin of error values still prevent us from drawing any meaningful conclusions regarding delay and jitter. Given the higher stability of the 10Mbps dataset, it represents our first choice for the calculation of the normalized



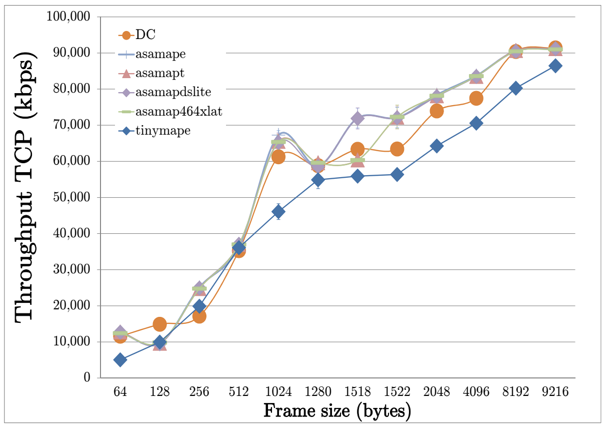
(a) UDP 10Mbps



(b) TCP 10Mbps



(c) UDP 100Mbps



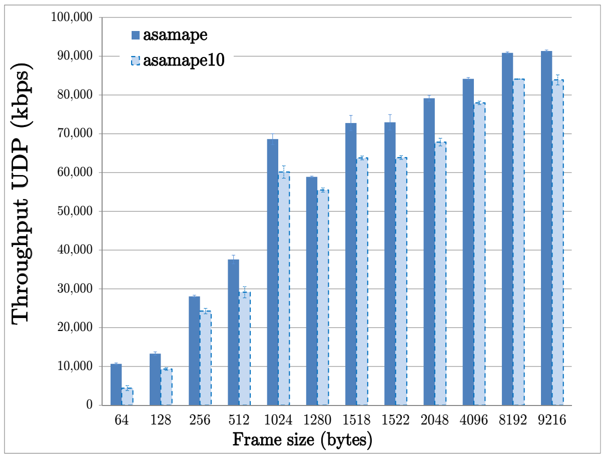
(d) TCP 100Mbps

Figure 5.3: Throughput results

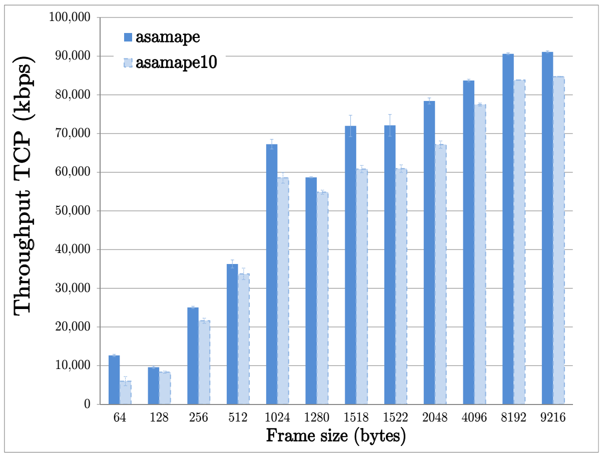
network performance scores.

Table 5.1: 10 Mbps Summarized results

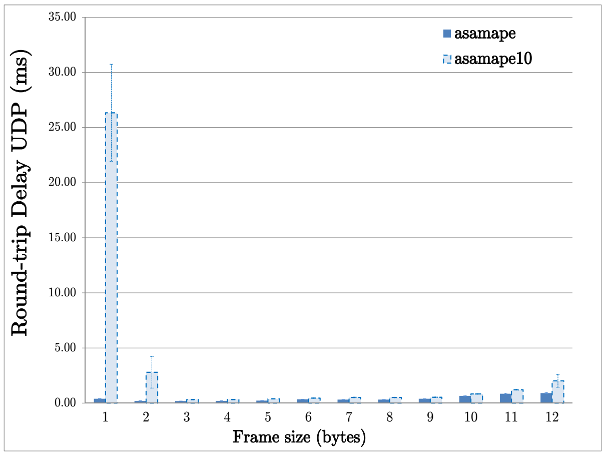
	RT Delay (ms)	+/-	Jitter (ms)	+/-	Throughput (Kbps)	+/-
DC	0.224	0.000	0.010	0.000	9,157.1	0.4
asamape	0.782	0.004	0.032	0.001	8,917.2	3.1
asamapt	0.750	0.003	0.030	0.001	8,862.6	3.0
asamapdslite	0.781	0.003	0.034	0.001	8,917.3	4.0
asamap464xlat	0.738	0.003	0.029	0.001	8,870.9	2.6
tinymape	1.068	0.003	0.040	0.001	8,751.2	3.3



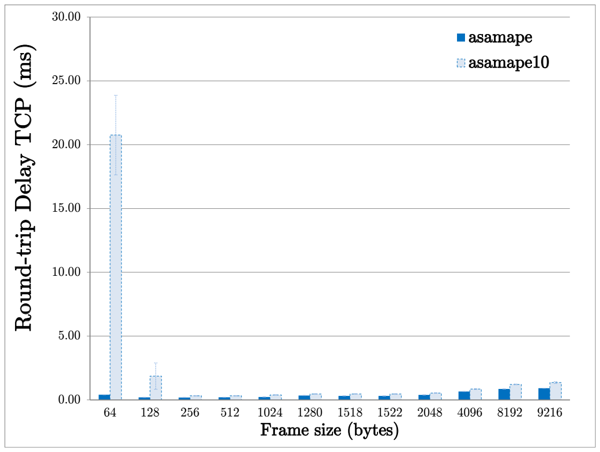
(a) UDP



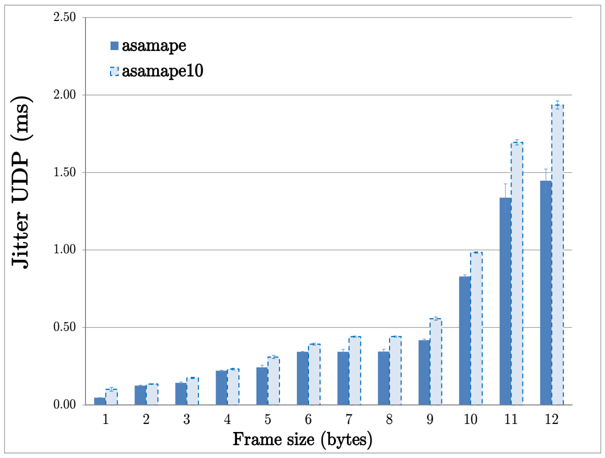
(b) TCP



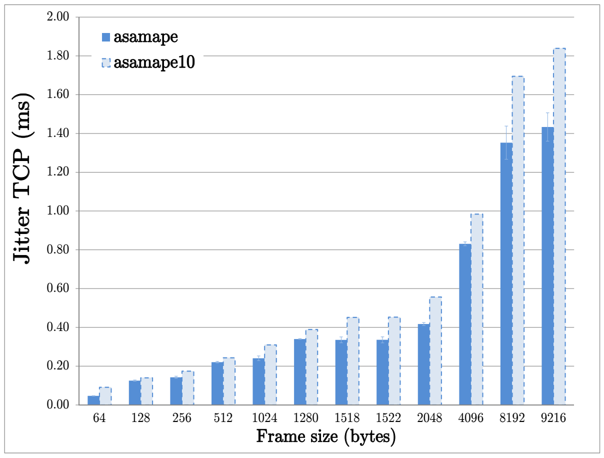
(c) UDP



(d) TCP

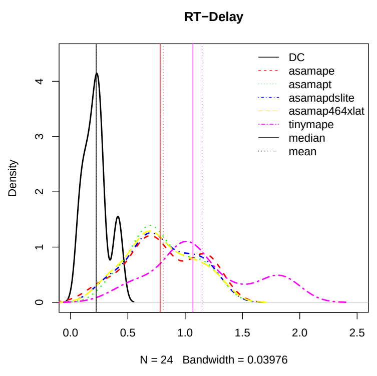


(e) UDP

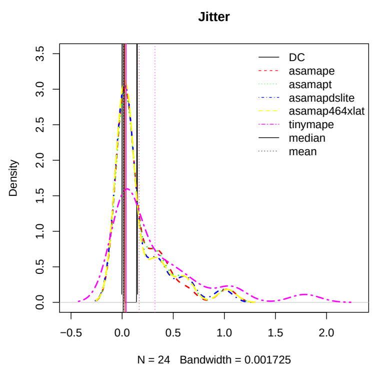


(f) TCP

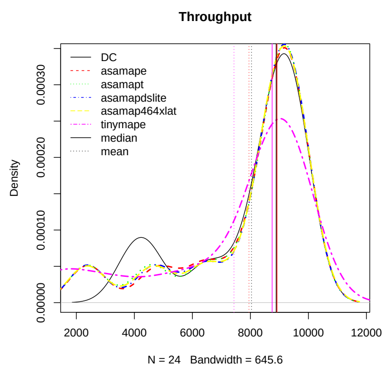
Figure 5.4: Network performance degradation for asamape



(a) Round-Trip Delay

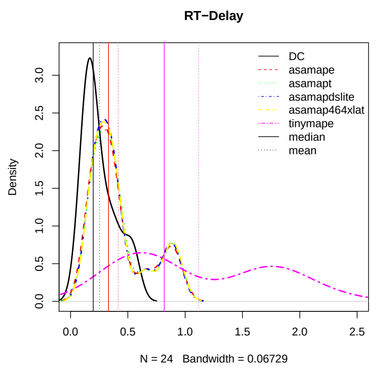


(b) Jitter

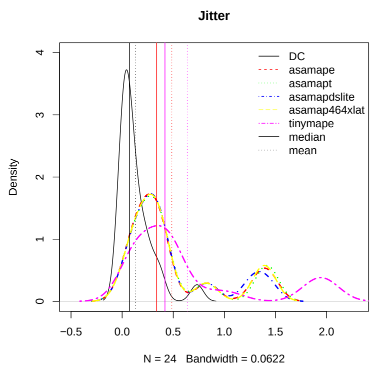


(c) Troughput

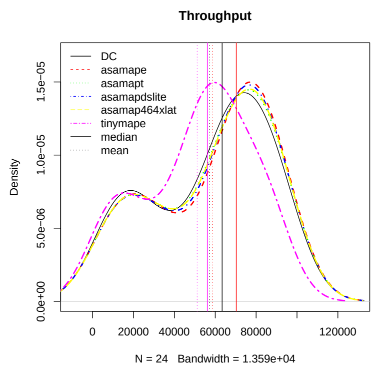
Figure 5.5: Probability density function for the 10 Mbps Datasets



(a) Round-Trip Delay

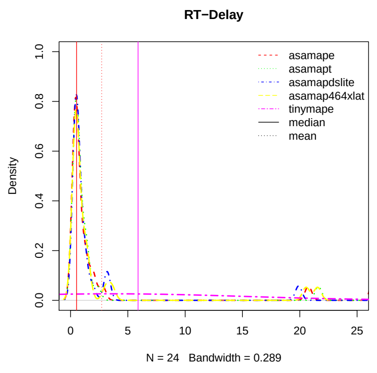


(b) Jitter

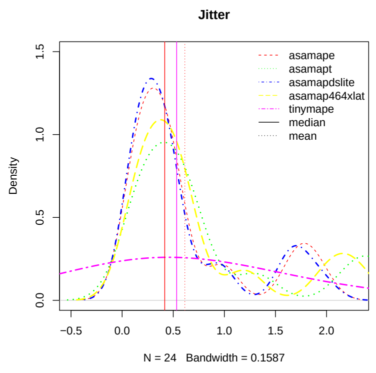


(c) Troughput

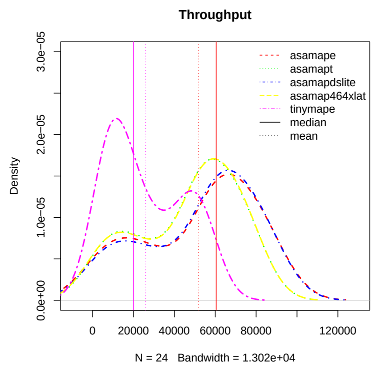
Figure 5.6: Probability density function for the 100 Mbps Datasets



(a) Round-Trip Delay



(b) Jitter



(c) Troughput

Figure 5.7: Probability density function for the 10CE Scalability Datasets

Table 5.2: 100 Mbps Summarized Results

	RT Delay (ms)	+/-	Jitter (ms)	+/-	Throughput (Kbps)	+/-
DC	0.198	0.004	0.071	0.005	63,375.6	622.8
asamape	0.331	0.010	0.338	0.009	70,295.7	339.5
asamapt	0.344	0.012	0.340	0.007	66,277.6	344.9
asamapdslite	0.334	0.016	0.339	0.009	69,070.6	433.0
asamap464xlat	0.338	0.012	0.332	0.007	66,132.4	344.9
tinymape	0.817	0.027	0.419	0.007	56,132.2	392.2

5.2.2 Scalability

The scalability results are summarized in Table 5.3. Considering the asamap covered transition tuples, the translation-based mechanisms (MAPt, 464XLAT) seem to have a better performance considering all aspects of network performance degradation. The most affected feasibility metric for encapsulation-based mechanisms (MAPe, DSLite) is delay, with as much as 84% performance degradation for DSLite. This leads us to believe that, considering the 10 topology, translation-based mechanisms are a better choice in terms of scalability. Regarding the tinymape tuple, the feasibility gap seems to be increasing with a larger scale.

Table 5.3: Network Performance Degradation (NPD) results

	RT Delay (%)	Jitter (%)	Throughput (%)	Frame loss (%)
asamape	58.97	23.34	13.98	0.00
asamapt	33.76	12.35	8.44	0.00
asamapdslite	84.54	16.82	11.57	0.00
asamap464xlat	28.36	3.17	8.56	0.00
tinymape	620.50	27.37	64.24	0.00

5.3 Data collection and repeatability

Table 5.4: Relative standard error average

	RT Delay (%)	Jitter (%)	Throughput (%)
DC	2.21	3.42	0.93
asamape	5.10	3.71	1.04
asamapt	5.90	7.48	1.34
asamapdslite	4.85	2.60	0.99
asamap464xlat	5.67	5.41	0.93
tinymape	6.94	0.96	0.89

For the closed experiment a full factorial design was employed, hence $12(frame\ sizes) \times 2(transport\ layer\ protocols) \times 2(workloads) \times 6(transition\ tuples) = 288$ different experiments were conducted. Each experiment was repeated 20 times. The estimated time for each of the experiments was approximately 70 sec , resulting in a total data collection time of *6720 min*, or *112 h*. For post-processing the raw data we have spent an average of 20 sec for each experimental instance, bringing us to a total of *1920 min* or *32 h*.

The 100 Mbps workload experiment was replicated 17 times on 68 different StartBED nodes to check the repeatability of the experiments. The repeatability results for the Direct Connection have been plotted in Figure 5.8.

Table 5.4 presents the average of the relative standard deviation. The low percentages indicate a low variability among datasets, and by extrapolation, a high repeatability for the experiments.

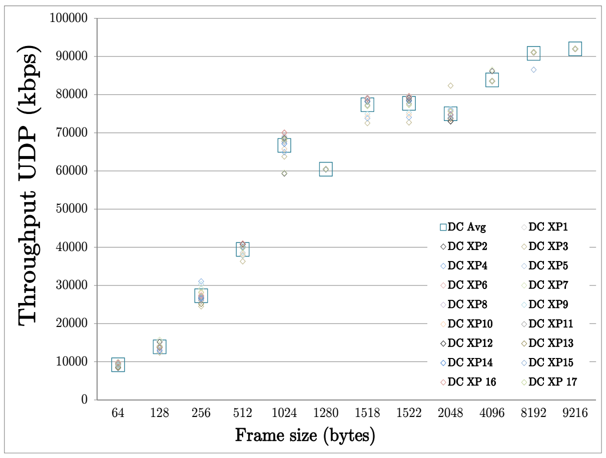
5.4 Open Experiment results

Since the configuration method of each implementation is specific, we needed to prepare a specific survey for each of the two implementations. The detailed tasks of the assisted survey are included in Appendix A for configuration capability and Appendix B for troubleshooting capability. Given the assisted nature of the survey, and our efforts to isolate user skill and usability, only two randomly selected participants were needed to complete the survey. The assisted survey results for configuration and troubleshooting capability have been summarized in Table 5.5. Since the operational tasks are implementation-oriented, the results have been organized according to the transition implementation.

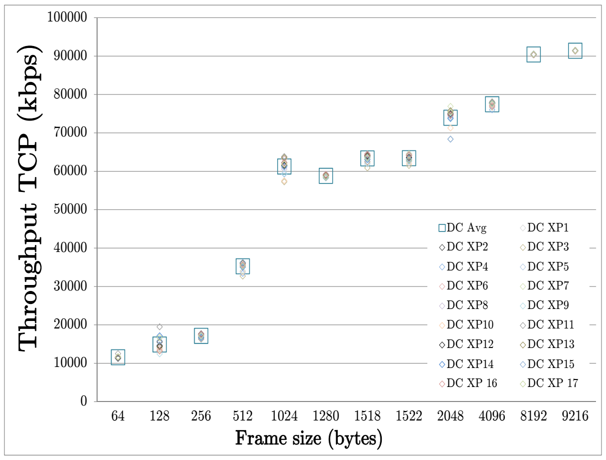
Table 5.5: Configuration and Troubleshooting capability results

Operational Capability		Asamap	Tiny-map-e
Configuration Capability	IinitialSetup1	Pass	Pass
	IinitialSetup2	Pass	Pass
	IinitialSetup3	Fail	Fail
	IinitialSetup4	Pass	Pass
	IinitialSetup5	Pass	Pass
	InitialSetup6	Pass	Fail
	Reconfiguration1	Pass	Fail
	Reconfiguration2	Pass	Pass
	Confirmation1	Pass	Fail
	Confirmation2	Pass	Fail
Configuration capability result		9/10 = 90%	5/10 = 50%
Troubleshooting Capability	FaultIsolation1	Pass	Pass
	FaultIsolation2	Pass	Pass
	FaultDetermination1	Pass	Pass
	FaultDetermination2	Pass	Fail
	FaultDetermination3	Fail	Fail
	RCA1	Pass	Pass
	RCA2	Pass	Pass
	RCA3	Fail	Fail
	RCA4	Pass	Pass
	RCA5	Pass	Pass
Troubleshooting capability result		8/10 = 80%	7/10 = 70%

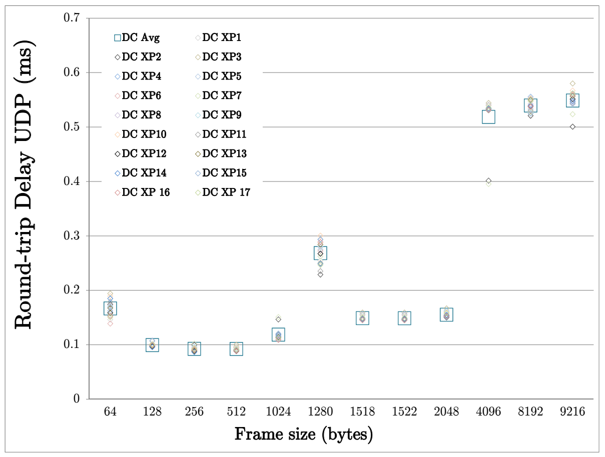
Regarding the configuration capability, most of the tasks have been completed suc-



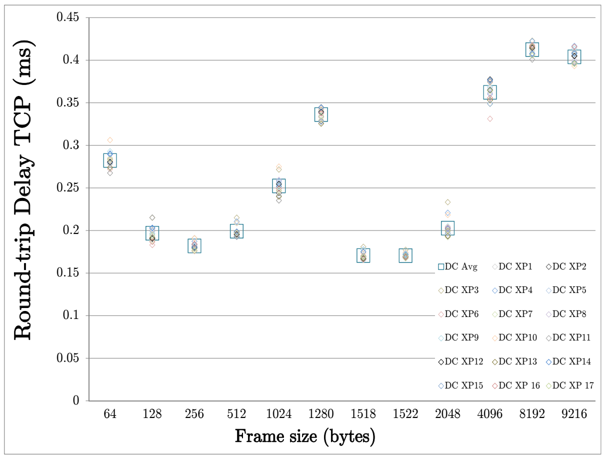
(a) UDP



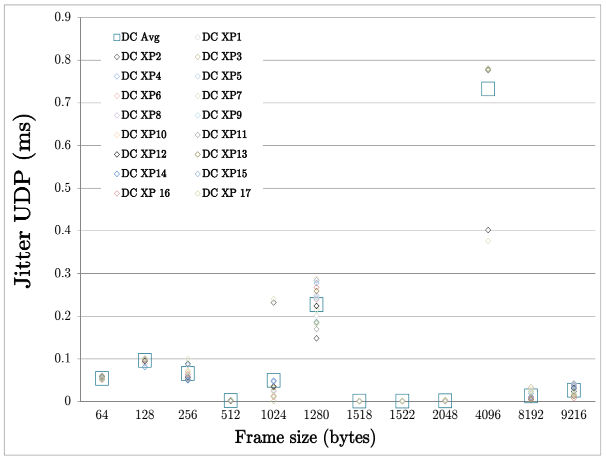
(b) TCP



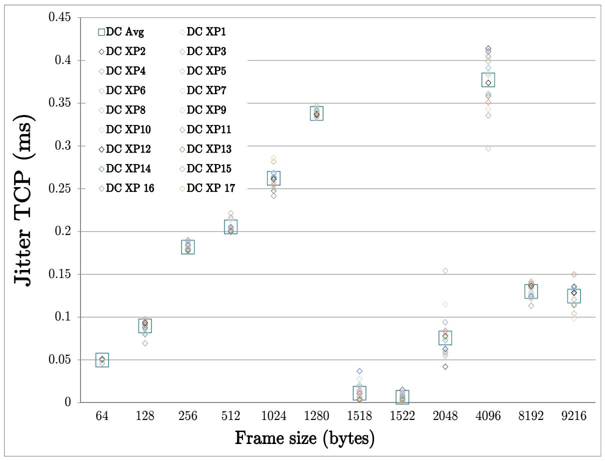
(c) UDP



(d) TCP



(e) UDP



(f) TCP

Figure 5.8: Repeatability of the DC results

cessfully for the asamap implementation. However, a self-configuration setup sequence is not yet available for the asamap implementation. Given the complexity of the transition technologies, a guided self-configuring setup would be a beneficial feature. Regarding the tinymape implementation, 3 other tasks have failed. This can be explained by the *in-development* status of the implementation. In the case of asamap, most of the the troubleshooting tasks have been completed successfully. Two of the troubleshooting tasks could not be completed: FaultDetermination3 (Displaying critical messages with associated details) and RCA3 (self-check sequence). Regarding the first one, some critical messages were displayed in the user console. However, these are hard to interpret and understand. We believe this feature needs improvement. As for the second failed task, a self-check sequence is not available yet. This would represent a substantial improvement of the troubleshooting capability. For tinymape RCA3, FaultDetermination2 and FaultDetermination3 failed, one more than asamap. This confirms the lower operational capability of tinymape and makes asamap the first choice from this standpoint.

In terms of applications capability, we tested a non-exhaustive list of common applications, in accordance with [36]. The full list of applications and the results are presented in Table 5.6. The applications were tested using two machines, one running Windows 7 and the other, a mobile device, running Android 4.2. Both devices were connected to the experimental environment through the prepared WiFi SSID: *ipv6net*. To summarize we did not encounter any application troubles with any of the two implementations.

Table 5.6: Applications capability results

Applications			Asamap	Tiny-map-e
Win 7 / Android 4.2	Browsing	Chrome	Pass	Pass
		Firefox	Pass	Pass
		Dolphin	Pass	Pass
	E-mail	Outlook	Pass	Pass
		Thunderbird	Pass	Pass
		Aquamail	Pass	Pass
	IM&VoIP	Skype	Pass	Pass
		Facebook	Pass	Pass
		Google+	Pass	Pass
		VoIP Buster	Pass	Pass
		Viber	Pass	Pass
		DigiOriunde	Pass	Pass
	VPN	Hideman VPN	Pass	Pass
		Spotflux	Pass	Pass
	Cloud	Dropbox	Pass	Pass
		GDrive	Pass	Pass
	FTP	Filezilla	Pass	Pass
	Troubleshooting	puTTY	Pass	Pass
		WinSCP	Pass	Pass
		ConnectBot	Pass	Pass
Applications capability result			20/20 = 100%	20/20 = 100%

5.5 Normalized and composite scores

5.5.1 Network performance

The network performance (NP) feasibility results have been normalized according to the four normalization formulas 3.4, 3.5, 3.6, 3.7. The normalized values, as well as the initial raw values, are presented in 5.9. It is also important to note that the normalized values have been multiplied by 10^4 for presentation purposes.

In order to avoid double counting, the correlation among the variables of the network performance dataset needs to be analyzed. The simplest method is graphical, by plotting the values in the n-dimensional space. Figure 5.9 presents the NP dataset in the 4-dimensional space created by the 4 feasibility metrics: delay, jitter, throughput and frame loss. From the graph we can observe a correlation among jitter and delay results, but for a clearer image we need to calculate the Pearson correlation coefficients.

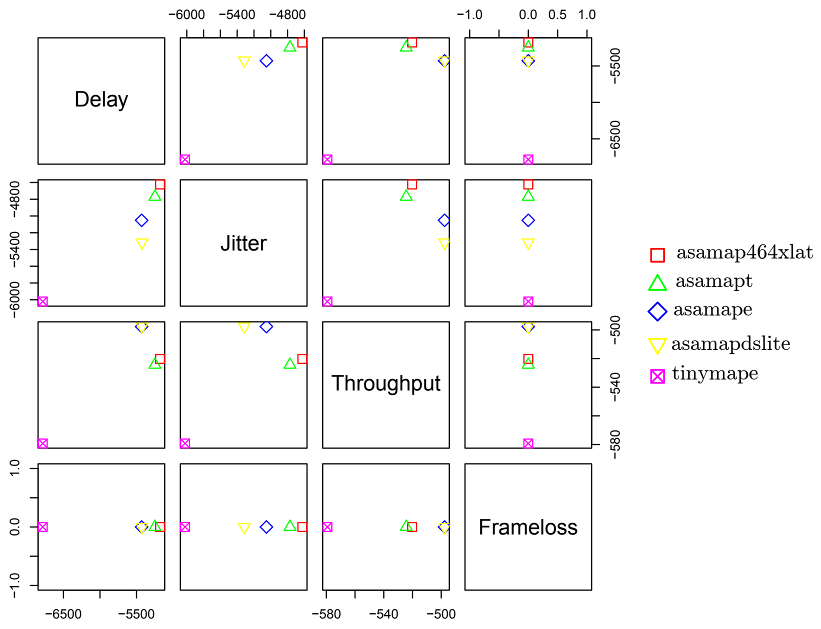


Figure 5.9: Normalized NP data

The calculation procedure for the correlation coefficients and the correlation significance followed the guideline presented by P. Teetor in [54]. The results of the correlation analysis are presented in Table 5.7. As the frame loss data has no variance, the correlation analysis is irrelevant, hence it was omitted. The variables show a strong positive correlation coefficients. However, their significance needs to be assessed as well, in order to avoid naive conclusions. The significance analysis was calculated for a 99% level of confidence, for which the *p-value* should not exceed 0.01. From the analysis we can observe that *p-values* are well above the 0.01 conventional threshold. Moreover the confidence intervals include 0, which means it is possible for the correlation to be 0. Considering these results, we can conclude that the correlation is unlikely to be significant. Therefore, there is a small risk of dealing with double counting in the case of network performance.

Table 5.7: Correlation coefficients and correlation significance for Network Performance

	(delay,jitter)	(delay,throughput)	(jitter,throughput)
Coorelation coeficient	0.938	0.857	0.654
99% Confidence interval	(-0.095 ,0.998)	(-0.490 ,0.995)	(-0.538 ,0.974)
p-value	0.018	0.063	0.230

For the calculation of the the NP scores, we need to assume the hypothetical role of an enterprise network operator called *Ro6Cloud*, which provides cloud services and is currently building an IPv6 transition plan based on 464 transition technologies. In terms of network performance, the priority of the operator is ensuring a *no-packet-loss* environment while keeping delay, jitter throughput within existing standards.

In order to get a clearer image of the importance of each feasibility indicator, an analytic hierarchically process (AHP) should prove useful. For each of feasibility metric pairs the question *Which of the two do you find more important, on a scale of 1 to 9 ?* needs to be answered. A preference value of 1 should indicate equal importance, while 9 means that the specific sub-metric is 9 times more important than the other. Considering the 4 feasibility metrics, a plausible Q&A sequence for the *Ro6Cloud* operational team can be the following:

Q1: from *(Frame loss,Delay)* which one is more important, on a scale of 1 to 9 ?
Frame loss is twice as important. $\rightarrow$ **2**

Q2: from *(Frame loss,Throughput)* which one is more important, on a scale of 1 to 9 ?
Frame loss is 4 times more important. $\rightarrow$ **4**

Q3: from *(Frame loss,Jitter)* which one is more important, on a scale of 1 to 9 ?
Frame loss is 6 times more important. $\rightarrow$ **6**

Q4: from *(Delay,Throughput)* which one is more important, on a scale of 1 to 9 ?
Delay is twice as important. $\rightarrow$ **2**.

Q5: from *(Delay,Jitter)* which one is more important, on a scale of 1 to 9 ?
Delay is 3 times more important. $\rightarrow$ **3**.

Q6: from *(Throughput,Jitter)* which one is more important, on a scale of 1 to 9 ?
Throughput is twice as important. $\rightarrow$ **2**.

The Q&A sequence can be be converted into the following paired comparison Table 5.8.

The weights of the 4 feasibility metrics are calculated using an eigenvector, following the guidelines proposed by T. Saaty in [55]. The resulted weights are:
$w_{frameloss} = 0.51981988$ $w_{delay} = 0.25990994$ $w_{throughput} = 0.13964570$ $w_{jitter} = 0.08062448$
The *Saaty inconsistency* value is *0.004240651*, which is below the conventional threshold of 0.01, hence acceptable.

The network performance (NP) scores for *Ro6Cloud* are finally calculated using the Formula 3.10 and the aforementioned weights. The values are presented in the NP column

Table 5.8: Network performance AHP paired comparison Table

	Frame loss	Delay	Throughput	Jitter
Frame Loss	1	2	4	6
Delay	1/2	1	2	3
Throughput	1/4	1/2	1	2
Jitter	1/6	1/3	1/2	1

of Table 5.9. In the last column, the Euclidean distance to origin, or Euclidean norm is presented, which is calculated with 3.11. In terms of NP *asamap464XLAT* seems to be the best choice in the presented context of the *Ro6cloud* network operator. The Euclidean norm points at *asamap464xlat* as well, identifying this transition tuples as Pareto-optimal choice for network performance.

Table 5.9: Network performance (NP) normalized and composite scores

	RT Delay (ms)	Jitter (ms)	Throughput (Kbps)	Frame loss (%)
asamap464xlat	0.738	0.029	8,870.987	0.00
asamapt	0.750	0.030	8,862.679	0.00
asamape	0.782	0.032	8,917.277	0.00
asamapdslite	0.781	0.034	8,917.396	0.00
tinymape	1.068	0.040	8,751.281	0.00

	Delay 0.26	Jitter 0.08	Throughput 0.14	Frame loss 0.52	**NP**	**NP E-norm**
asamap464xlat	-5,178.08	-4,623.98	-520.28	0.00	**-1,791.30**	**-6,961.64**
asamapt	-5,248.13	-4,771.21	-524.35	0.00	**-1,821.94**	**-7,112.12**
asamape	-5,429.59	-5,051.50	-497.68	0.00	**-1,887.98**	**-7,432.75**
asamapdslite	-5,424.03	-5,314.79	-497.62	0.00	**-1,907.75**	**-7,610.17**
tinymape	-6,783.23	-6,020.60	-579.28	0.00	**-2,329.33**	**-9,088.20**

Figure 5.10 presents how the network performance scores are decomposed in the underlaying feasibility metric scores.

5.5.2 Scalability

For scalability the Network performance degradation (NPD) normalized results together with the raw data are presented in Table 5.11. A 4-dimensional scatter-plot of the scalability data is presented in Figure 5.11. The graph shows some tendencies of correlation. However, for a clear idea we need to check Pearson's correlation coefficients and their significance, given the small sample size. The coefficients are calculated at a 99% level of confidence and are displayed in Table 5.10. Similar to the NP analysis, frame loss was omitted from the analysis, as the data shows no variance. The calculated coefficients show positive correlations. However, the high *p-values* are well above the 0.01 accepted threshold. Moreover, the confidence intervals, which include 0 correlation, indicate that the correlation is likely to be significant. Hence, the risk of double counting is very low.

Considering that scalability is measured as network performance degradation, the scalability (SCAL) scores for the hypothetical network operator *Ro6cloud* were calculated using

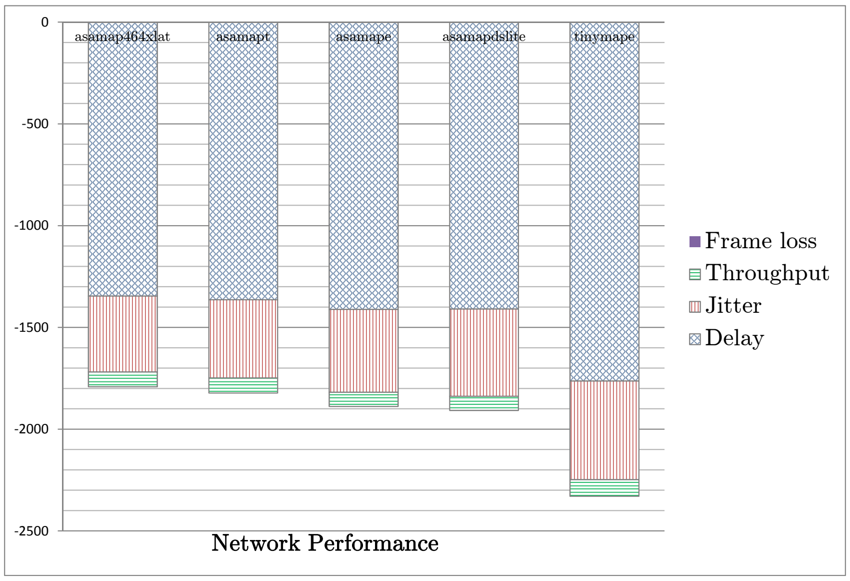


Figure 5.10: NP scores decomposition

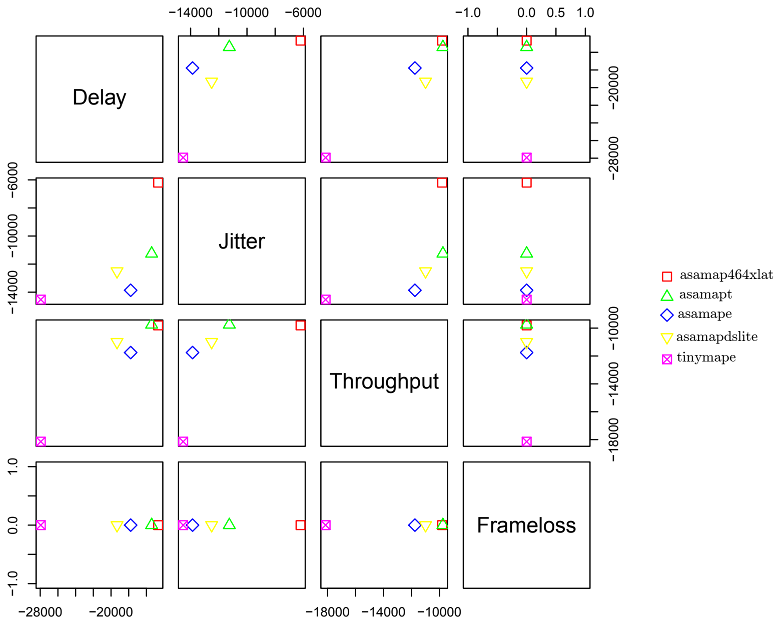


Figure 5.11: Normalized scalability data

the same weights as for network performance, and are shown in Table 5.11. The most feasible transition tuple for *Ro6Cloud* seems to be *asamap464xlat* again. The Euclidean norm scores are presented in the last column of the table. It seems to be the Pareto-optimal solution for scalability as well.

Table 5.10: Correlation coefficients and correlation significance for Scalability

	(delay,jitter)	(delay,throughput)	(jitter,throughput)
Coorelation coeficient	0.686	0.977	0.631
99% Confidence interval	(-0.753 ,0.990)	(-0.379,0.999)	(-0.792 ,0.988)
p-value	0.201	0.042	0.254

Table 5.11: Scalability Normalized scores

	RTDelay (%)	Jitter (%)	Throughput (%)	Frame loss (%)
asamap464xlat	28.36	3.17	8.56	0.00
asamapt	33.76	12.35	8.44	0.00
asamape	58.97	23.34	13.98	0.00
asamapdslite	84.54	16.82	11.57	0.00
tinymape	620.50	27.37	64.24	0.00

	Delay 0.26	Jitter 0.08	Throughput 0.14	Frame loss 0.52	**SCAL**	**SCAL E-norm**
asamap464xlat	-14,676.87	-6,197.53	-9,805.04	0.00	**-5,683.57**	**-18,707.18**
asamapt	-15,410.39	-11,254.25	-9,749.77	0.00	**-6,274.19**	**-21,428.86**
asamape	-17,779.04	-13,863.79	-11,753.80	0.00	**-7,380.08**	**-25,425.40**
asamapdslite	-19,321.49	-12,508.99	-10,992.70	0.00	**-7,565.46**	**-25,507.53**
tinymape	-27,934.41	-14,529.08	-18,145.30	0.00	**-10,965.74**	**-36,341.13**

A decomposition attempt of the scalability score into the underlaying components is presented in Figure 5.12.

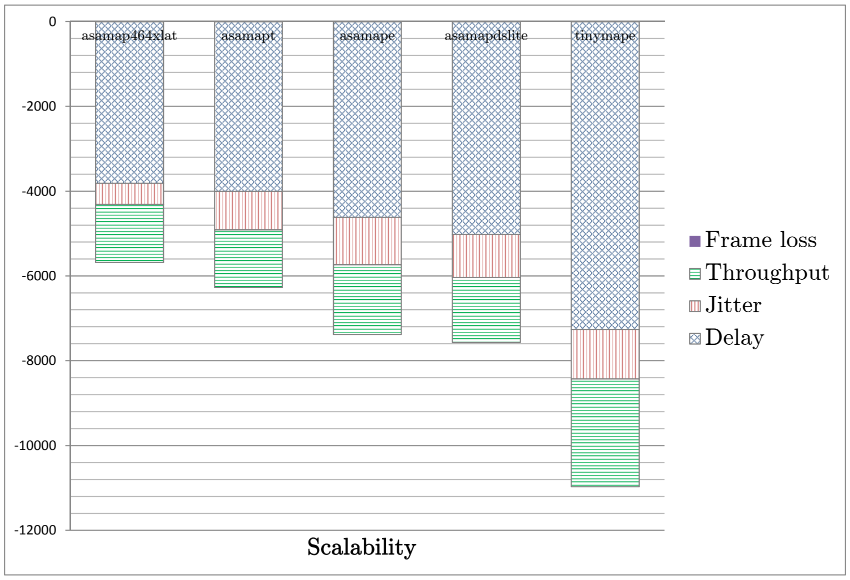


Figure 5.12: SCAL scores decomposition

5.5.3 Operational capability

The normalized operational capability (OC) scores are presented in Table 5.13, and were obtained from the raw data using Formula 3.9. The data is graphed in Figure 5.13. It is evident from the graph that there is a high correlation between the configuration capability (CC) and troubleshooting capability (TC) data, as there is almost no variance. There are only two distinct points, and that is due to the measurement approach. The tasks which dictate the scores are mostly implementation-oriented, hence the asamap tuples have the same scores for both configuration and troubleshooting capability. Moreover, the applications capability data has no variance across the five transition tuples, which means there is perfect correlation among the variables in the dataset. However, the tasks were designed to check completely different operational capabilities, hence the statistical correlation has no significance for this type of data. Consequently, we contend there is no risk of double counting when integrating the three operational variables.

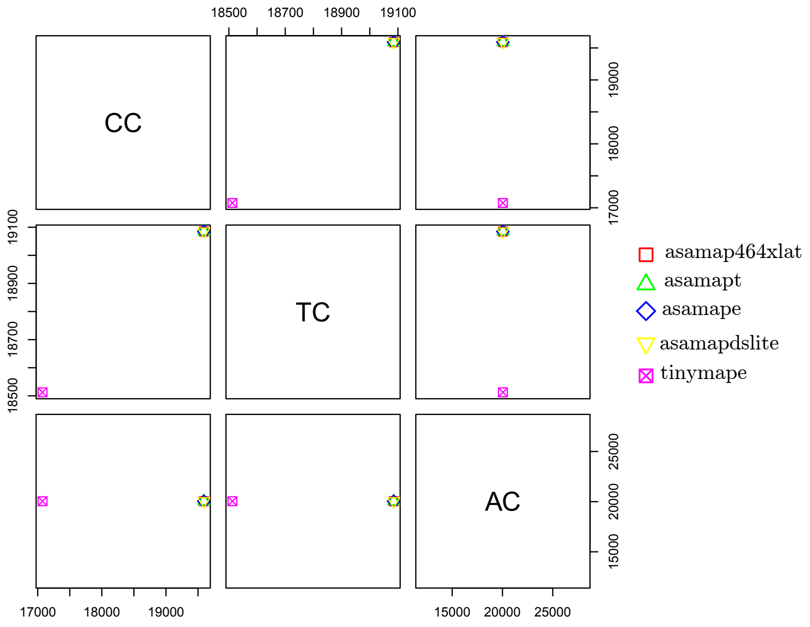


Figure 5.13: Normalized OC data

To compute the customized scores for the hypothetical *Ro6Cloud* network operator, we need to consider their operational priorities. Given the cloud service profile of the operator, their main concern is applications capability. However, the operator is also equally interested in using devices easy to configure and troubleshoot. To compute the assigned weights for the three underlying feasibility metrics we are using the analytic hierarchically process (AHP). The hypothetical Q&A session for operational capability can be the following:

Q1: from *(Applications capability, Configuration capability)* which one is more important, on a scale of 1 to 9 ?

Applications capability is twice as important. $\rightarrow$ **2**

Q2: from *(Applications capability, Troubleshooting capability)* which one is more important, on a scale of 1 to 9 ?
Applications capability twice as important. $\rightarrow$ **2**

Q3: from *(Configuration capability, Troubleshooting capability)* which one is more important, on a scale of 1 to 9 ?
They are equally important. $\rightarrow$ **1**

The operational feasibility preferences of the *Ro6cloud* operator is captured in Table 5.12.

Table 5.12: Operational Capability AHP paired comparison table

	AC	CC	TC
AC	1	2	2
CC	1/2	1	1
TC	1/2	1	1

The resultant weights for each feasibility metric are: $w_{AC} = 0.50$ $w_{CC} = 0.25$ $w_{TC} = 0.25$. The *Saaty inconsistency* value is *0.00*, well below the conventional threshold of 0.01, hence acceptable. The resulting operational capability scores are presented in the OC column of Table 5.13. Given the implementation-oriented nature of the operational capability data, *asamap* can be considered the more feasible implementation for the *Ro6cloud* network operator. The Euclidean norm values are presented in the last column of the table. As expected, the Pareto-optimal choice is still the *asamap* implementation.

Table 5.13: Operational capability

	CC (%)	TC (%)	AC (%)
asamape	90	80	100
asamapt	90	80	100
asamapdslite	90	80	100
asamap464xlat	90	80	100
tinymape	50	70	100

	CC 0.25	TC 0.25	AC 0.5	**OC**	**OC E-norm**
asamape	19,590.41	19,084.85	20,043.21	**19,690.42**	**33,907.91**
asamapt	19,590.41	19,084.85	20,043.21	**19,690.42**	**33,907.91**
asamapdslite	19,590.41	19,084.85	20,043.21	**19,690.42**	**33,907.91**
asamap464xlat	19,590.41	19,084.85	20,043.21	**19,690.42**	**33,907.91**
tinymape	17,075.70	18,512.58	20,043.21	**18,918.68**	**32,187.35**

A decomposition example of the operational capability (OC) scores into the three underlying components is shown in Figure 5.14.

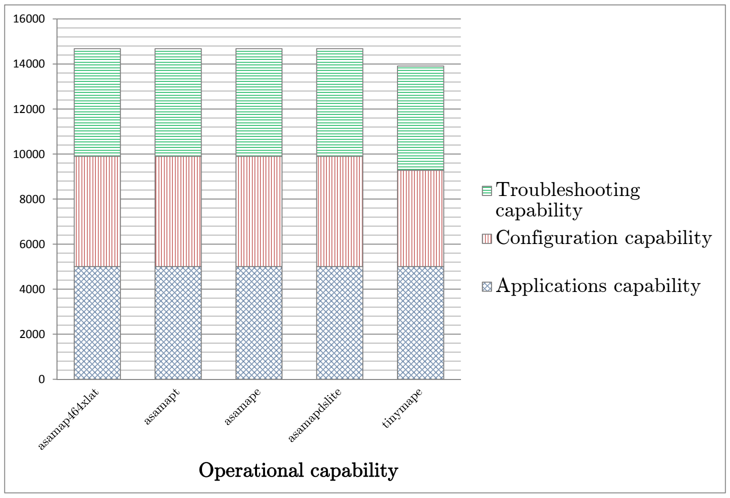


Figure 5.14: OC scores decomposition

5.5.4 General Feasibility Considerations

Security is one of the most important feasibility aspects of the IPv6 transition, but it was not yet approached in our research. Consequently, expressing a general feasibility score can be considered premature. However, that does not prevent us from analyzing the general feasibility of transition tuples through other means than a composite indicator. Figure 5.15 presents a 3-dimensional graphic interpretation of the sub-indicator scores customized for the *Ro6Cloud* hypothetical cloud service operator.

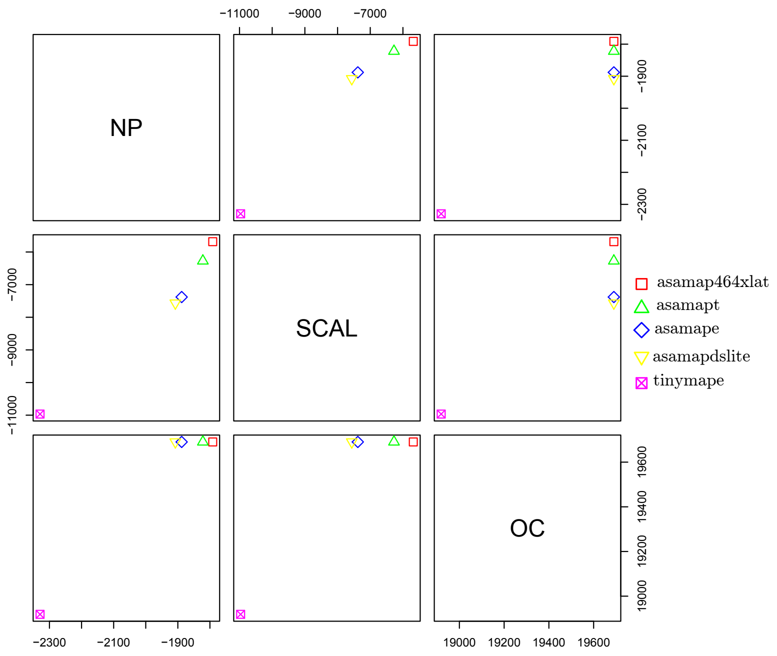


Figure 5.15: General feasibility considerations

The graphical analysis indicates the *asamap464xlat* transition tuple as the best solution for the network operator considering the three feasibility sub-indicators of network performance, scalability and operational capability. The feasibility list is continued by *asamapt, asamape* and *asamapdslite*, covered as well by the *asamap* implementation. The tinymape transition tuple seems to be the least feasible of the transition tuples, which was a somewhat expected result, considering the *in-development* state of the implementation.

Chapter 6

Discussion of the proposal

This chapter discusses the validity, limitations and applicability of the proposed benchmarking methodology and future research directions.

6.1 Validity

IPv6 transition scenarios and IPv6 transition technologies have already been known for some time to the Internet community. However, the worldwide deployment rate of IPv6 is still very low. Given the complexity and the diversity of transition technologies, one of the biggest challenges is understanding which technology to use in a certain network scenario. Various implementations of transition technologies have been introduced, further complicating this problem.

This thesis proposes a solution to that problem in the form of a practical benchmarking methodology associated with a heterogeneous IP4-IPv6 testbed, called IPv6NET. The basis for the feasibility analysis of IPv6 transition implementations is represented by practical means, such as real implementations and empirical measurements. To prove the validity of this methodology we have used it to analyze the feasibility of two suitable transition implementations, covering multiple transition technologies, in relation to the 464 network scenario. In this scenario, IPv4 capable nodes exist in the edge network, but the network core is IPv6-only. Hence IPv4 over IPv6 communication technologies are needed.

By analyzing, integrating and comparing the empirical results, we have reached our goal, finding that one transition tuple is more feasible than the rest, for a hypothetical network operator. Furthermore, we have identified possible performance trends in IPv6 transition technologies benchmarking, for example, encapsulation-based technologies seem to have better throughput performance, while translation-based technologies seem to have better latency performance. We were also able to point out some *unexpected behaviors*, which could have been overlooked if simulators or analytical tools were employed. This underlines the need for a testbed and gives us motivation for a further root cause analysis.

6.2 Limitations

A limitation of the proposed benchmarking methodology is represented by the lack of control data, given that there is no similar alternative system to act as a comparison base for the empirical results. We are planning to solve this by comparing the current open-source-based measurement system with existing commercial network benchmarking tools.

One of the biggest technical limitations is scale. The conducted scalability tests were within the current hardware resources. For now, we have used bare-metal servers, but for further topology growth tests we will need to use virtual technology.

Our approach on operational capability is limited to the proposed non-exhaustive list of tasks and applications. Of course the possibilities are potentially limitless, hence we would like to continue with the best-efforts benchmarking strategy by expanding the list.

Another limitation of this approach is represented by the diversity and complexity of existing production networks by comparison with the presented scenarios. However, by using the detailed methodology any interested party could potentially implement it and obtain customized feasibility data.

The methodology can also serve as guideline for other researchers interested in joining this effort. Coping with a large number of technologies and their future developments may very well be solved by research collaboration, which can transform our project in an exhaustive IPv6 transition resource.

6.3 Applicability

The main contribution of this thesis is represented by the detailed benchmarking methodology associated with IPv6NET and the empirical feasibility results. The empirical results and normalized scores can serve as a direct transition guideline to network operators faced with a similar transition scenario. The high repeatability results indicate that the methodology is also easy to replicate on systems with the same hardware and software characteristics. Furthermore, by assigning different weights when using the proposed composite indicator formula 3.10, network operators can obtain personalized feasibility scores, which can be the starting point in an IPv6 transition plan. The detailed feasibility scores can also act as feedback for the transition implementation developers. This can lead to further improvement of their products.

We have used this methodology to analyze two transition implementations and our contention is we can apply it to numerous others. In terms of scalability, we have shown the methodology can be applied to different network scales and in the unlikely case of limitless resources, our contention is it can be applied to any scale. Although we do not have have supporting data, we contend the benchmarking methodology can be applied to other transition scenarios as well. Our proposed model for building composite indicators is also worth mentioning. The detailed building steps of composite indicators, which until now have eluded computer science, can represent an useful starting point to fellow computer science researchers interested in benchmarking.

By looking at this work from the industry perspective, we have addressed mainly the technological challenges of the IPv6 transition. However, we contend our work can help overcome some other challenges identified in Section 2.2, such as *costs of the adoption*, *availability of IPv6-ready products* and *lack of trained staff*. Regarding the costs of the adoption, the basic network template can help operators and decision makers alike understand the minimum number of transition devices needed to start the IPv6 transition. In turn, the number can express a baseline investment cost. The benchmarking scores of a specific transition implementation can offer insights about its IPv6-readiness. Additionally, more awareness about the feasibility of open-source implementations may fuel the development of other transition implementations. In terms of the lack of IPv6 trained staff, we contend that the detailed operational capability surveys presented in Appendix A and Appendix B can help operators better understand the essential operational features of transition implementations.

6.4 Future work

Our work has a lot potential for expansion. This section presents some of the expansion plans we are currently envisioning.

Operational capability represents the first expansion direction. Currently, we have proposed a limited number of tasks for configuration and troubleshooting capability, which test the existence of core features in the subject implementations. We would like to increase this number in the future and with it raise the complexity of the survey. In terms of applications capability we have only targeted 20 common applications. However, this number is far from being realistic. We plan to increase this number, to better fit a realistic scenario. Furthermore, we would like to take into account other operational variables, such as *operator's skill* or *usability of the implementation*. Quantifying these variables should also lead to a more realistic approach of operational capability.

The uncertainty and sensitivity analysis is one of the aspects of the composite subindicators which remains unfinished for now, given the lack of historical data and the complexity and time-consuming nature of the whole process. It also underlines the preliminary state of the composite indicators, as their robustness is unknown yet. Consequently, it represents one of our future priorities.

Scalability needs to be improved as well. Scalability should also take into account operational factors, which for now were not approached. An example of this is the operational complexity introduced by addressing schemes of different transition technologies. This can mainly affect the scalability of very large networks. Nevertheless, it should be quantified. Another aspect of scalability is represented by virtualization, which is an ever-increasing phenomenon in today's production networks. The capability of implementations to function within a virtual environment, may influence their scalability. One future step towards that goal is to include virtual technology in the benchmarking process.

Security is one of the directions in which we would like to expand our research in. It represents one of the biggest challenges of the IPv6 transition. We would like to include *security* as a feasibility indicator and propose associated metrics. One approach we have envisioned is performing a protocol-oriented penetration test, which should help quantify the exploitability of transition tuples. Another approach we are considering is a risk quantification formula targeting the underlying components of each transition technology.

The quantification of security will lead to a more exhaustive feasibility analysis, as the core dimensions would be covered. This will allow us to express a more robust and trustworthy GFI.

Commercial 464 transition implementations are for now, few and cost prohibitive. As the market evolves, we would like to continue benchmarking with commercial 464 transition implementations.

Other IPv6 transition scenarios represent one of the long terms expansion plans. A potential next step is the IPv6-only scenario. The scenario will bring us closer to the ultimate transition goal, which is the retirement of IPv4.

A permanent IPv6NET is one of the main goals of this research. IPv6NET is the heterogeneous IPv4/IPv6 environment associated with the proposed benchmarking methodology. From the network environment perspective, the testbed has taken form only temporarily using StarBED, IPLab or WIDE network resources. We envision a more stable state for the network environment component of IPv6NET.

Chapter 7

Conclusion

In this thesis we have presented a practical benchmarking approach in analyzing the feasibility of IPv6 transition technologies. The practicality of the approach lies primarily in the means of the analysis, as we have used existing implementations and associated empirical data. To support the validity of our approach, we have used it to analyze the feasibility of IPv4 over IPv6 transition technologies in a heterogeneous IPv4-IPv4 environment, called the IPv6 Network Evaluation Testbed (IPv6NET), which is, for now, just a conceptual testbed dedicated to quantifying the feasibility of IPv6 transition implementations in relation to specific network scenarios, and represents one of the ultimate goals of our research.

We started by explaining the background of the IPv6 transition and have documented the research question we are trying to answer, *which is the most feasible transition technology for a certain transition scenario?* We have cited similar attempts in current literature and identified two trends in feasibility benchmarking, using closed and open environments. The closed, isolated environments are useful for obtaining thorough performance, while open environments are very efficient at identifying more practical aspects, such as interoperability and operational issues. We presented a benchmarking methodology that combines the advantages of the two benchmarking methods to obtain a complete feasibility analysis. To support this methodology, we have used it in relation to a specific network scenario for enterprise networks, the 464 transition scenario. The subjects of the analysis have been two open source transition implementations, covering multiple transition technologies, *asamap* covering MAPe, MAPt, DSLite and 464XLAT, and *tiny-map-e* covering MAPe only.

The analysis targeted three major feasibility dimensions of transition technologies: network performance, scalability and operational capability. Analyzing the empirical results, we were able to identify some performance trends, such as the better latency of translation-based (464XLAT, MAPt) technologies or the better throughput of encapsulation-based mechanisms (MAPe,DSLite). In terms of operational capability, we were able to point out that *asamap* is a more capable transition implementation than *tiny-map-e*, from both configuration and troubleshooting standpoints. By using composite indicators which integrated the empirical data, we were able identify the most feasible transition tuple for a hypothetical cloud service operator *Ro6Cloud* and prove the validity of our method. Finally, we have discussed the limitations and applicability of the proposed methodology, as well as future research directions.

In conclusion we can say that the methodology can support the efforts of network operators finding themselves in a similar transition scenario. To that end, the empirical results can represent a direct IPv6 transition guideline. The complementary composite indicator scores allow any interested party to obtain personalized scores. Furthermore, the Euclidean norm scores can identify Pareto-optimal solutions across the proposed feasibility indicators: *network performance, scalability.* The feasibility scores can also help transition implementation developers, which can use them as feedback and further improve their products. Ultimately we hope that our proposal will contribute to a smoother and faster IPv6 transition for the Internet community.

Appendix A

Configuration capability survey

Appendix A.1

For the asamap vyatta implementation the questions were:

1. IinitialSetup1: Please input the following commands in the console:

```
configure
set interfaces map map0 br-address
'2001:200:16a:2109::a/64'
set interfaces map map0 default-forwarding-mode
'encapsulation'
set interfaces map map0 default-forwarding-rule 'true'
set interfaces map map0 ipv6-fragment-size '1500'
set interfaces map map0 ipv4-fragment-inner false
set interfaces map map0 role 'br'
set interfaces map map0 rule 1 ea-length '8'
set interfaces map map0 rule 1 ipv4-prefix
'163.221.135.16/28'
set interfaces map map0 rule 1 ipv6-prefix
'2001:200:16a:2100::/56'
commit
exit
```

   These commands should create a new 464 virtual interface called map0. To check the existence of the map0 interface please input the following command:

```
show interfaces detail
```

   The command should display details about all interfaces, including the map0 interface. Was the map0 interface created successfully ?

   ☐ Yes

   ☐ No

2. InitialSetup2: Please input the following commands in the console:

```
configure
save
```

   The command should have saved the temporary configuration which should be loaded at start-up.

   Reboot the machine by typing in the console the command:

```
sudo reboot
```

   To check that the setup of the map0 interface was saved use again:

```
show interfaces
```

Was the configuration saved successfully ?

☐ Yes

☐ No

3. InitialSetup3: The non-existence of a self-configuration command can be verified by pressing the *Tab* key while using the console. It should display the existing commands. To check also the configuration mode we must enter it by typing:

   ```
   configure
   ```

   and pressing again the *Tab* key. Is there any self-configuration command available:

   ☐ Yes

   ☐ No

4. InitialSetup4: Please input the following command:

   ```
   shw interfaces
   ```

   The console should display a message warning the user that the command is invalid and should discard it. Was the warning displayed and the command discarded ?

   ☐ Yes

   ☐ No

5. InitialSetup5: Please input the following commands:

   ```
   configure
   set interfaces map map1 default-forwarding-mode
   'encapsulation' set interfaces map map1
   default-forwarding-rule 'true'
   ```

   The console should accept the commands, as they are formally correct. However after trying to commit the temporary configuration:

   ```
   commit
   ```

   the console should display a message warning that additional configuration details are needed and discarding the action. Was a warning message displayed and the commit action discarded ?

   ☐ Yes

   ☐ No

6. InitialSetup6: While typing the command:

   ```
   set interfaces ethernet eth0 address
   ```

   press the *Tab* key.

The console should display information about possible completions for the command or contextual help. Was the contextual help displayed in the console ?

☐ Yes

☐ No

7. Reconfiguration1: Input the following command:

```
show configuration commands
```

The console should display all commands needed to rebuild the current configuration. Was the set of commands displayed?

☐ Yes

☐ No

8. Reconfiguration2: Input the following commands:

```
configure
save backup.config
```

The console should display a message confirming the current was saved and showing the location of the back-up file. To restore the configuration type:

```
load backup.config
```

A message confirming the configuration file was loaded successfully should be displayed.

Were the back-up and restore actions successful ?

☐ Yes

☐ No

9. Confirmation1: Type the command:

```
show configuration
```

The console should display the detailed configuration. Was the detailed configuration displayed ?

☐ Yes

☐ No

10. Confirmation2: Type the command:

```
show interface map map0
```

The console should display the details of the previously configured map0 interface. Was the detailed configuration of the map0 interface displayed ?

☐ Yes

☐ No

Appendix A.2

For the tiny-map-e implementation the questions were:

1. IinitialSetup1: Please input the following commands in the console:

   ```
   sudo ~/tinyMAPe/tiny-map-e-master/map-e -6
   2001:200:16a:2100::/56 -4 163.221.135.32/28 -b
   2001:200:16a:2109::b -l 8 -m br
   ```

   These commands should create a new 464 virtual interface called map-e. To check the existence of the map0 interface please input the following command:

   ```
   ifconfig
   ```

   The command should display details about all interfaces, including the map-e interface. Was the map0 interface created successfully ?

   ☐ Yes

   ☐ No

2. InitialSetup2: Please input the edit the file */etc/rc.local* by running the command:

   ```
   sudo vim /etc/rc.local
   ```

   and add the following line:

   ```
   /home/mariusg/tinyMAPe/tiny-map-e-master/map-e -6
   2001:200:16a:2100::/56 -4 163.221.135.32/28 -b
   2001:200:16a:2109::b -l 8 -m br
   ```

   The command should have saved the temporary 464 interface configuration and should be loaded at start-up.

   Reboot the machine by typing in the console the command:

   ```
   sudo reboot
   ```

   To check that the setup of the map-e interface was saved use again:

   ```
   ifconfig
   ```

   Was the configuration saved successfully ?

   ☐ Yes

   ☐ No

3. InitialSetup3: The non-existence of a self-configuration command can be verified by pressing the *Tab* key while using the console. It should display the existing commands. Is there any self-configuration command available:

   ☐ Yes

   ☐ No

4. InitialSetup4: Please input the following command:

```
mape
```

The console should display a message warning the user that the command is invalid and should discard it. Was the warning displayed and the command discarded ?

☐ Yes

☐ No

5. InitialSetup5: Please input the following command:

```
sudo  /tinyMAPe/tiny-map-e-master/map-e -6
2001:200:16a:2100::/56 -4 163.221.135.32/28 -b
2001:200:16a:2109::b
```

The console should discard the command and explain the usage of the command. Was the command discarded and the usage presented?

☐ Yes

☐ No

6. InitialSetup6: While typing the command:

```
sudo  /tinyMAPe/tiny-map-e-master/map-e -6
2001:200:16a:2100::/56
```

press the *Tab* key.

The console should display information about possible completions for the command or contextual help Was the contextual help displayed in the console ?

☐ Yes

☐ No

7. Reconfiguration1: We need to find a command showing the detailed instructions used to obtain the current configuration. Its non-existence can be verified by pressing the *Tab* key while using the console. The action should display the existing commands. Is the command available ?

☐ Yes

☐ No

8. Reconfiguration2: Input the following commands:

```
sudo cp /etc/rc.local .  sudo cp rc.local /etc/
```

The current configuration file was copied should be copied in the current directory. To restore the configuration type:

```
sudo cp rc.local /etc/
```

The configuration should have been restored. Were the back-up and restore actions successful ?

☐ Yes

☐ No

9. Confirmation1: We need to find a command showing the detailed configuration of the routing device. Its non-existence can be verified by pressing the *Tab* key while using the console. The action should display the existing commands. Is the command available ?

   ☐ Yes

   ☐ No

10. Confirmation2: We need to find a command showing the detailed configuration of the map-e interface. Its non-existence can be verified by pressing the *Tab* key while using the console. The action should display the existing commands. Is the command available ?

    ☐ Yes

    ☐ No

Appendix B

Troubleshooting capability survey

Appendix B.1

The assisted survey, which was used to asses the troubleshooting capability of the asamap vyatta implementation contained the following questions.

1. FaultIsolation1: Type the command:

   ```
   sudo tcpdump -i map0
   ```

   The console should display in a human readable form IPv4 and IPv6 packets captured on the map0 interface. Were there analyzed packets displayed ?

   ☐ Yes

   ☐ No

2. FaultIsolation2: Type the command:

   ```
   ping 192.168.255.1
   ```

   The console should display statistics about the round-trip ICMPv4 packet exchange with the host identified with the IPv4 address 192.168.255.1 .

   Type the command:

   ```
   ping 2001:200:16a:2101::2
   ```

   The console should display statistics about the round-trip ICMPv6 packet exchange with the host identified with the IPv6 address 2001:200:16a:2101::2 .

☐ Yes

☐ No

3. FaultDetermination1: Type the command:
   ```
   show ipv6 route
   show ip route
   ```
   The console should display the IPv4 and IPv6 routing details. This information should be able to help identify a misconfigured IPv4 or IPv6 route. Were the routing details displayed ?

   ☐ Yes

   ☐ No

4. FaultDetermination2: Input the command:
   ```
   show interface map map0
   show interface map map0 rule
   ```
   The console should display detailed information about the map0 interface and the mapping rule it employs. The information should help identify a misconfigured line of the 464 virtual interface, map0. Was the information displayed ?

   ☐ Yes

   ☐ No

5. FaultDetermination3: The non-existence of a self-troubleshooting command can be verified by pressing the *Tab* key while using the console. It should display the existing commands. To check also the configuration mode we must enter it first by typing:
   ```
   configure
   ```
   and pressing again the *Tab* key. Is there any self-troubleshooting command available:

   ☐ Yes

   ☐ No

6. RCA1 and RCA2: Type the commands:
   ```
   show log all | tail
   show log all
   ```
   The first command should confirm that error and warning messages are being logged. The second command should confirm all log information can be displayed. Were the log information displayed ?

   ☐ Yes

   ☐ No

7. RCA3: Create a critical event by intentionally failing to login on a parallel console. The critical events should be displayed in the current console with contextual details. Was any information displayed about these events ?

   ☐ Yes

   ☐ No

8. RCA4 and RCA5: Type the command:

```
show interfaces detail
```

The command should confirm that statistical network information are being logged and can be displayed. Were the network statistics displayed ?

☐ Yes

☐ No

Appendix B.2

The survey employed for quantifying the troubleshooting capability of the tiny-map-e implementation contained the following questions.

1. FaultIsolation1: Type the command:

```
sudo tcpdump -i map-e
```

The console should display in a human readable form IPv4 and IPv6 packets captured on the map0 interface. Were there analyzed packets displayed ?

☐ Yes

☐ No

2. FaultIsolation2: Type the command:

```
ping 192.168.255.1
```

The console should display statistics about the round-trip ICMPv4 packet exchange with the host identified with the IPv4 address 192.168.255.1 .

Type the command:

```
ping 2001:200:16a:2101::2
```

The console should display statistics about the round-trip ICMPv6 packet exchange with the host identified with the IPv6 address 2001:200:16a:2101::2 .

☐ Yes

☐ No

3. FaultDetermination1: Type the command:

```
ip -6 route
ip route
```

The console should display the IPv4 and IPv6 routing details. This information should be able to help identify a misconfigured IPv4 or IPv6 route. Were the routing details displayed ?

☐ Yes

☐ No

4. FaultDetermination2: We need to find a command showing detailed information about the map-e interface and the mapping rule it employs. Its non-existence can be verified by pressing the *Tab* key while using the console. The action should display the existing commands. Is the command available ?

   ☐ Yes

   ☐ No

5. FaultDetermination3: The non-existence of a self-troubleshooting command can be verified by pressing the *Tab* key while using the console. It should display the existing commands. Is there any self-troubleshooting command available:

   ☐ Yes

   ☐ No

6. RCA1 and RCA2: Type the commands:

   ```
   tail /var/log/syslog
   less /var/log/syslog
   ```

   The first command should confirm that error and warning messages are being logged. The second command should confirm all log information can be displayed. Was the log information displayed ?

   ☐ Yes

   ☐ No

7. RCA3: Create a critical event by intentionally failing to login on a parallel console. The critical events should be displayed in the current console with contextual details. Was any information displayed about these events ?

   ☐ Yes

   ☐ No

8. RCA4 and RCA5: Type the command:

   ```
   ifconfig
   ```

   The command should confirm that statistical network information are being logged and can be displayed. Were the network statistics displayed ?

   ☐ Yes

   ☐ No

Bibliography

[1] APNIC, “IPv6 measurements for The World [Online]. Available: ”http://labs.apnic.net/ipv6-measurement/Regions/001%20World/” .” 1.1, 2.1

[2] M. Lind, V. Ksinant, S. Park, A. Baudot, and P. Savola, “Scenarios and Analysis for Introducing IPv6 into ISP Networks.” RFC 4029 (Informational), Mar. 2005. 1.1

[3] J. Bound, “IPv6 Enterprise Network Scenarios.” RFC 4057 (Informational), June 2005. 1.1, 1.2, 2.3.1, 4.1

[4] J. Wiljakka, “Analysis on IPv6 Transition in Third Generation Partnership Project (3GPP) Networks.” RFC 4215 (Informational), Oct. 2005. 1.1

[5] C. Huitema, R. Austein, S. Satapati, and R. van der Pol, “Unmanaged Networks IPv6 Transition Scenarios.” RFC 3750 (Informational), Apr. 2004. 1.1

[6] X. Leng, J. Bi, and M. Zhang, “Study on high performance ipv4/ipv6 transition and access service,” in *Proceedings of the 4th International Conference on Parallel and Distributed Processing and Applications*, ISPA’06, (Berlin, Heidelberg), pp. 183–194, Springer-Verlag, 2006. 1.1, 2.2

[7] P. Wu, Y. Cui, J. Wu, J. Liu, and C. Metz, “Transition from ipv4 to ipv6: A state-of-the-art survey,” *Communications Surveys Tutorials, IEEE*, vol. 15, pp. 1407–1424, Third 2013. 1.1, 2.2

[8] NRO, “Free Pool of IPv4 Address Space Depleted [Online]. Available: ”http://www.nro.net/news/ipv4-free-pool-depleted”,” July 2014. 2.1

[9] APNIC, “Free Pool of IPv4 Address Space Depleted [Online]. Available: ”http://www.apnic.net/publications/news/2011/final-8”,” July 2014. 2.1

[10] P. Grossetete, C. Popoviciu, and F. Wettling, *Global Ipv6 Strategies: From Business Analysis to Operational Planning.* Cisco Press, first ed., 2008. 2.2

[11] WorldIPv6Launch, “Launching the future [Online]. Available: ”http://www.worldipv6launch.org/infographic/”,” July 2014. 2.2

[12] E. Nordmark and R. Gilligan, “Basic Transition Mechanisms for IPv6 Hosts and Routers.” RFC 4213 (Proposed Standard), Oct. 2005. 2.3

[13] F. Baker, X. Li, C. Bao, and K. Yin, “Framework for IPv4/IPv6 Translation.” RFC 6144 (Informational), Apr. 2011. 2.3, 2.3

[14] X. Li, C. Bao, M. Chen, H. Zhang, and J. Wu, “The China Education and Research Network (CERNET) IVI Translation Design and Deployment for the IPv4/IPv6 Co-

existence and Transition." RFC 6219 (Informational), May 2011. 2.3

[15] C. Bao, X. Li, Y. Zhai, and W. Shang, "dIVI: Dual-Stateless IPv4/IPv6 Translation." draft-xli-behave-divi-06, Jan. 2014. 2.3

[16] M. Bagnulo, P. Matthews, and I. van Beijnum, "Stateful NAT64: Network Address and Protocol Translation from IPv6 Clients to IPv4 Servers." RFC 6146 (Proposed Standard), Apr. 2011. 2.3

[17] A. Durand, R. Droms, J. Woodyatt, and Y. Lee, "Dual-Stack Lite Broadband Deployments Following IPv4 Exhaustion." RFC 6333 (Proposed Standard), Aug. 2011. 2.3, 2.3.1, 4.2

[18] R. Gilligan and E. Nordmark, "Transition Mechanisms for IPv6 Hosts and Routers." RFC 1933 (Proposed Standard), Apr. 1996. Obsoleted by RFC 2893. 2.3

[19] B. Carpenter and K. Moore, "Connection of IPv6 Domains via IPv4 Clouds." RFC 3056 (Proposed Standard), Feb. 2001. 2.3

[20] F. Templin, T. Gleeson, M. Talwar, and D. Thaler, "Intra-Site Automatic Tunnel Addressing Protocol (ISATAP)." RFC 4214 (Experimental), Oct. 2005. Obsoleted by RFC 5214. 2.3

[21] C. Huitema, "Teredo: Tunneling IPv6 over UDP through Network Address Translations (NATs)." RFC 4380 (Proposed Standard), Feb. 2006. Updated by RFCs 5991, 6081. 2.3

[22] O. Troan, W. Dec, X. Li, C. Bao, S. Matsushima, T. Murakami, and T. Taylor, "Mapping of Address and Port with Encapsulation (MAP)." draft-ietf-softwire-map-10, Jan. 2014. 2.3, 2.3.1, 4.2

[23] R. Despres, "IPv6 Rapid Deployment on IPv4 Infrastructures (6rd)." RFC 5569 (Informational), Jan. 2010. 2.3

[24] J. Weil, V. Kuarsingh, C. Donley, C. Liljenstolpe, and M. Azinger, "IANA-Reserved IPv4 Prefix for Shared Address Space." RFC 6598 (Best Current Practice), Apr. 2012. 2.3.1

[25] A. Lacoste, "MAP simulation tool [Online]. Available: "http://6lab.cisco.com/map/MAP.php" ," July 2014. 2.3.1

[26] X. Li, C. Bao, W. Dec, O. Troan, S. Matsushima, and T. Murakami, "Mapping of Address and Port using Translation (MAP-T)." draft-ietf-softwire-map-t-05, Sept. 2013. 2.3.1, 4.2

[27] M. Mawatari, M. Kawashima, and C. Byrne, "464XLAT: Combination of Stateful and Stateless Translation." RFC 6877, Apr. 2013. 2.3.1, 4.2

[28] I. Raicu and S. Zeadally, "Evaluating ipv4 to ipv6 transition mechanisms," *IEEE International Conference on Telecommunications 2003*, 2003. 2.4.1

[29] S. Narayan, P. Shang, and N. Fan, "Network performance evaluation of internet protocols ipv4 and ipv6 on operating systems," in *Proceedings of the Sixth international conference on Wireless and Optical Communications Networks*, WOCN'09, (Piscat-

away, NJ, USA), pp. 242–246, IEEE Press, 2009. 2.4.1

[30] S. Sasanus and K. Kaemarungsi, "Differences in bandwidth requirements of various applications due to ipv6 migration," in *Proceedings of the The International Conference on Information Network 2012*, ICOIN '12, (Washington, DC, USA), pp. 462–467, IEEE Computer Society, 2012. 2.4.1

[31] P. Grayeli, S. Sarkani, and T. Mazzuchi, "Performance analysis of ipv6 transition mechanisms over mpls," *International Journal of Communication Networks and Information Security*, vol. 4, no. 2, 2012. 2.4.1

[32] G. Lencse and S. Repas, "Performance analysis and comparison of different dns64 implementations for linux, openbsd and freebsd," in *Proceedings of the 2013 IEEE 27th International Conference on Advanced Information Networking and Applications*, AINA '13, (Washington, DC, USA), pp. 877–884, IEEE Computer Society, 2013. 2.4.1

[33] P. Wu, Y. Cui, J. Wu, J. Liu, and C. Metz, "Transition from ipv4 to ipv6: A state-of-the-art survey," *IEEE Communications Surveys and Tutorials*, vol. 15, no. 3, pp. 1407–1424, 2013. 2.4.1

[34] R. Hiromi and H. Yoshifuji, "Problems on ipv4-ipv6 network transition," in *Proceedings of the International Symposium on Applications on Internet Workshops*, SAINT-W '06, (Washington, DC, USA), pp. 38–42, IEEE Computer Society, 2006. 2.4.2

[35] H. Babiker, I. Nikolova, and K. K. Chittimaneni, "Deploying ipv6 in the google enterprise network lessons learned," in *Proceedings of the 25th international conference on Large Installation System Administration*, LISA'11, (Berkeley, CA, USA), pp. 10–10, USENIX Association, 2011. 2.4.2

[36] J. Arkko and A. Keranen, "Experiences from an IPv6-Only Network." RFC 6586 (Informational), Apr. 2012. 2.4.2, 3.2.2, 5.4

[37] H. Hazeyama, R. Hiromi, T. Ishihara, and O. Nakamura, *Experiences from IPv6-Only Networks with Transition Technologies in the WIDE Camp Spring 2012*, Mar 2012. draft-hazeyama-widecamp-ipv6-only-experience-01.txt. 2.4.2

[38] C. Popoviciu, A. Hamza, G. V. de Velde, and D. Dugatkin, "Ipv6 benchmarking methodology for network interconnect devices," 2008. 3.1, 3.2.1

[39] S. Bradner and J. McQuaid, "Benchmarking methodology for network interconnect devices," 1999. 3.1, 3.2.1, 3.3

[40] A. B. Bondi, "Characteristics of scalability and their impact on performance," in *Proceedings of the 2Nd International Workshop on Software and Performance*, WOSP '00, (New York, NY, USA), pp. 195–203, ACM, 2000. 3.1

[41] R. Jain, *Art of Computer Systems Performance Analysis Techniques For Experimental Design Measurements Simulation And Modeling.* John Wiley & Sons, May 1991. 3.2.1, 3.3

[42] S. Bradner, "Benchmarking Terminology for Network Interconnection Devices." RFC 1242 (Informational), July 1991. Updated by RFC 6201. 3.2.1

[43] D. Harrington, "Guidelines for Considering Operations and Management of New Pro-

tocols and Protocol Extensions.” RFC 5706 (Informational), Nov. 2009. 3.2.2, 3.2.2

[44] M. Nardo, M. Saisana, A. Saltelli, S. Tarantola, A. Hoffman, and E. Giovannini, “Handbook on constructing composite indicators: methodology and user guide,” tech. rep., OECD publishing, 2005. 3.4, 3.4

[45] M. Georgescu, H. Hazeyama, Y. Kadobayashi, and S. Yamaguchi’, “Empirical analysis of ipv6 transition technologies using the ipv6 network evaluation testbed,” in *9th International Conference on Testbeds and Research Infrastructures for the Development of Networks & Communities*, 2014. 4

[46] N. Matsuhira, “ Stateless Automatic IPv4 over IPv6 Encapsulation / Decapsulation.” draft-matsuhira-sa46t-spec-07, July 2013. 4.2

[47] M. Asama, “MAP supported Vyatta [Online]. Available: ”http://enog.jp/ masakazu/vyatta/map/”,” July 2014. 4.2

[48] Y. Ueno, “Tiny-map-e implementation [Online]. Available: ”https://github.com/edenden/tiny-map-e”,” July 2014. 4.2

[49] T. Miyachi, K.-i. Chinen, and Y. Shinoda, “Starbed and springos: large-scale general purpose network testbed and supporting software,” in *Proceedings of the 1st international conference on Performance evaluation methodolgies and tools*, valuetools ’06, (New York, NY, USA), ACM, 2006. 4.3.1

[50] A. Botta, A. Dainotti, and A. Pescapè, “A tool for the generation of realistic network workload for emerging networking scenarios,” *Computer Networks*, vol. 56, no. 15, pp. 3531–3547, 2012. 4.3.1

[51] “Ieee standard for local and metropolitan area networks–media access control (mac) bridges and virtual bridged local area networks–corrigendum 2: Technical and editorial corrections,” *IEEE Std 802.1Q-2011/Cor 2-2012 (Corrigendum to IEEE Std 802.1Q-2011)*, pp. 1–96, Nov 2012. 4.3.1

[52] M. Georgescu, H. Hazeyama, Y. Kadobayashi, S. Yamaguchi, “An empirical study of IPv6 transition in an open environment - experiences from WIDE camp’s Life with IPv6 Workshop,” in *The Fourteenth Workshop on Internet Technology*, June 2013. 4.3.2

[53] M. P. Wand and M. C. Jones, *Kernel smoothing*, vol. 60. Crc Press, 1994. 5.2

[54] P. Teetor, *R cookbook*. ” O’Reilly Media, Inc.”, 2011. 5.5.1

[55] T. L. Saaty, “Relative measurement and its generalization in decision making why pairwise comparisons are central in mathematics for the measurement of intangible factors the analytic hierarchy/network process,” *RACSAM-Revista de la Real Academia de Ciencias Exactas, Fisicas y Naturales. Serie A. Matematicas*, vol. 102, no. 2, pp. 251–318, 2008. 5.5.1