EAI Endorsed Transactions
on on Industrial Networks and Intelligent Systems
Research Article ICST.ORG

Empirical analysis of IPv6 transition technologies using the IPv6 Network Evaluation Testbed

Marius Georgescu*, Hiroaki Hazeyama, Youki Kadobayashi, Suguru Yamaguchi

Nara Institute of Science and Technology, 8916-5 Takayama, Ikoma, Nara 630-0192, JAPAN

Abstract

IPv6 has yet to become more than a worthy successor of IPv4, which remains, for now, the dominant Internet Protocol. This is due to the complicated transition period through which the Internet will have to go, until IPv6 will completely replace IPv4. One of the challenges introduced by this transition is to decide which technology is more feasible for a particular network scenario. To that end, this article proposes the IPv6 Network Evaluation Testbed (IPv6NET), a research project whose ultimate goal is to obtain feasibility data in order to formulate a coherent, scenario-based IPv6 transition strategy. The paper presents the overview of IPv6NET, the testing methodology and empirical results for a specific network scenario. The presented empirical feasibility data includes network performance data such as latency, throughput, packet loss, and operational capability data, such as configuration, troubleshooting and applications capability.

Keywords: IPv6 transition, IETF IPv6 scenario, 464 scenario, Enterprise Networks, IPv6NET, Asamap, MAPe, MAPt, DSLite, 464XLAT

Received on 20 May 2014, accepted on 11 November 2014, published on 25 February 2015

doi: 10.4108/inis.2.2.e1

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 with a great challenge, namely the transition to IPv6, which 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. IPv6 transition 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 mechanisms. Many transition mechanisms have been proposed and implemented. All have advantages and disadvantages considering a certain transition scenario, but no transition mechanism can be considered most feasible for all the scenarios. This opens many research opportunities. One of which is a scenario-based analysis of IPv6 transition implementations, and represents the ultimate goal of our research.

In this article, we are proposing the IPv6 Network Evaluation Testbed (IPv6NET), which is dedicated to measuring the feasibility of transition mechanisms in a series of scenario-based network tests. As a study case, the article focuses on one of the scenarios, introduced by the IETF for Enteprise 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.

The paper is organized as follows: section 2 presents related literature, section 3 introduces the IPv6NET concept and the testing methodology, in section 4 the empirical results are introduced, and the feasibility of the tested implementation is analyzed in relation to the specific scenario, section 5 discusses our approach and lastly section 6 states the conclusions and future work.

2. Related Work

There are a variety of articles dedicated to IPv6 transition experimental environments in current literature. They can be generally classified into closed environments and open environments. The closed environments are usually small scale, local environments,

*Corresponding author. liviumarius-g@is.naist.jp

which are isolated from production networks or the Internet. In [6], two 6-over-4, and IPv6 in IPv4 tunneling implementations are tested and experimental performance results are analyzed, in comparison with a homogeneous IPv6-only network. In [7], the performance of Linux operating systems is evaluated in relation to an IPv4-v6 Configured Tunnel and a 6to4 Tunnel. Four workstations were employed to build the testbed. In [8], differences in bandwidth requirements for common network applications like remote login, web browsing, voice communication, database transaction, and video streaming are analyzed over 3 types of networks: IPv4-only, IPv6-only and a 6to4 tunneling mechanism. The environment was built using the OPNET simulator, which also served as the basis for the testbed presented in [9], dedicated to the performance analysis of transition mechanisms over a MPLS backbone. [10] 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 data such as CPU and memory utilization, throughput, end-to-end delay, jitter and execution time.

However, as [11] also underlines, 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. They can be defined as experimental networks connected to a large scale production network or to the Internet. In [12], poor implementation and erroneous operations are identified in an 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. [13] describes 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 interoperability considerations have to be made. In [14], experiences with IPv6-only Networks are presented. 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 from previous WIDE Camp events in [15] present a great deal of meaningful interoperability data such as IPv6 capability of OSes, applications and network devices. Many operational issues have been identified. Some examples are long fall-back routine, low DHCPv6 capability of certain OSes, 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 environment testing has the potential of exposing interoperability issues, which can otherwise get overlooked.

Combing the advantages of the two testing methods can lead to a complete feasibility analysis. Hence we are considering both methods for testing.

3. Testing Methodology of IPv6NET

The IPv6 Network Evaluation Testbed (IPv6NET) is dedicated to quantifying the feasibility of IPv6 transition implementations in relation to a specific network scenarios. IPv6NET has two main components: the testing component and the infrastructure component. The testing component has the following building blocks: a specific network scenario, an associated network template and a test methodology. The infrastructure component is represented by the implementations under testing and the network test environment. As mentioned prior, we are considering building both closed and open environments.

The scenario targeted in this article was introduced by the IETF in [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, the technologies suitable for this specific scenario will be referred to as *464 technologies*.

3.1. IPv6NET Feasibility indicators and metrics

This subsection presents some clarifications regarding the semantics used for the methodology associated with IPv6NET throughout this paper. For the empirical feasibility analysis presented in this article, we are using the term *feasibility indicator* as a generic classifier for performance metrics. For closed environment testing, the proposed feasibility indicator was *network performance*. 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 *packet loss*. For open-environment testing, we have proposed *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 have introduced the following three metrics:

- *configuration capability*: measures how capable a network implementations 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

Details about the measurement process for these three metrics, as well as other methodology and infrastructure details, are presented in the following subsections.

3.2. Closed environment

Infrastructure. The basic, small scale template for 464 technologies 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. The closed experiment's design, presented in Fig. 1a, follows the basic network template, including one Customer Edge (CE) machine and one Provider Edge (PE) machine.

Multiple technologies can be considered suitable for the 464 scenario: MAPe[16], MAPt[17], DSLite[18], 464XLAT[19], SA46T[20]. Some implementations supporting these technologies have been proposed. One of those is the Asamap vyatta distribution[21], which covers 4 of those technologies: MAPe, MAPt, DSLite and 464XLAT. Both 464 PE and 464CE machines have used as Operating System the Asamap vyatta distribution.

For the underlaying infrastructure, the closed experiment uses StarBED [22], a large scale general purpose network testbed, administered by the National Institute of Information and Communications Technology (NICT) of Japan. Four Cisco UCS C200 M2 servers were used for this experiment: two for the devices under test (DUT), 464 PE and 464 CE, and two for the testing platform. As hardware details, each computer used a dual Intel Xeon X5670 CPU and 49.152 GB of RAM. The testing 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) [23]. One of the computers performed the ITGSend function, generating the traffic, while the other ran the ITGRecv function, receiving the generated traffic.

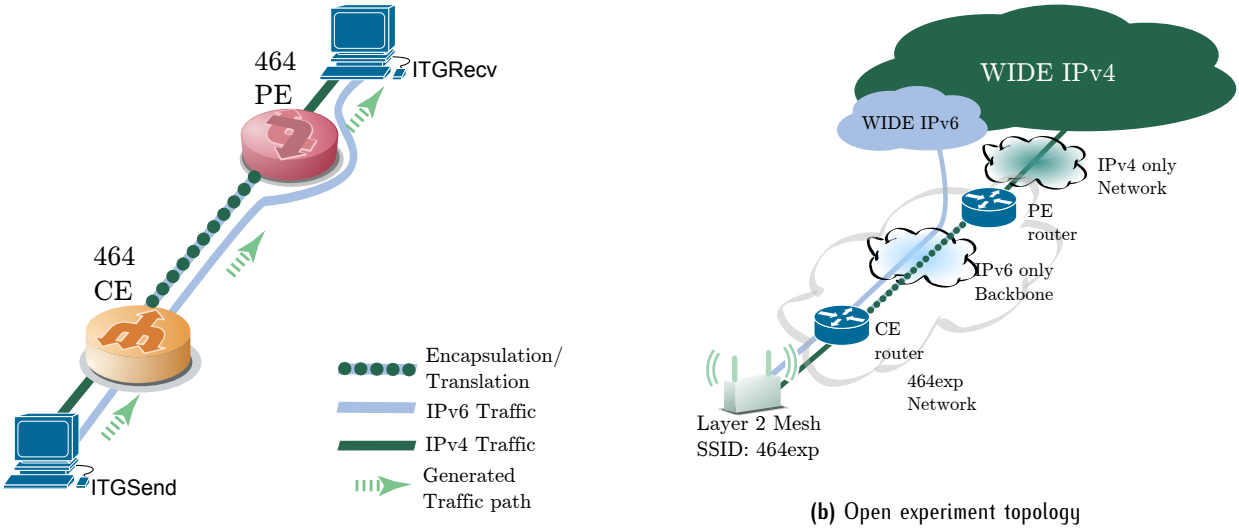
Methodology. The experimental workload is represented by the amount of traffic inserted into the experimental network. We have considered the combinations of frame size and frame rates displayed in Table 1. These have been recommended in RFC5180, IPv6 Benchmarking Methodology for Network Interconnect Devices [24], as maximum frame rates × frame sizes for 10 Mbps and 100 Mbps Ethernet. For future tests we intend to expand to 1Gbps as well.

We have considered the following parameters as potentially affecting the network performance: the IP version, IPv4 and IPv6, the upper layers protocols, UDP and TCP, the IPv6 transition technology and the IPv6 transition implementation. A full factorial design was employed. As recommended by RFC2544 [25], the duration of each experiment was 60 seconds after the first timestamp is sent. Each test was repeated 20 times and the average of the recorded values was reported.

3.3. Open environment

Infrastructure. The open experiment topology, presented in Fig. 1b also follows the basic, small scale 464 network template. The major difference is that the testing 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 supplied Internet access to participants at the WIDE Camp 1309, a networking event, held between September 10 and September 13 2013, at Shinsu-Matsushiro Royal Hotel, Nagano, Japan. 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 consisting of a Dell PowerEdge R805, with the following hardware description: Six-Core AMD Opteron 2400 CPU and 8GB of RAM. For the hypervisor a Citrix XenServer 6.0 distribution was employed. Previous experiences with building and analyzing a similar 464 open environment are presented in [26]. The base implementation for all four tested transition technologies, MAPe, MAPt, DSLite, 464XLAT has been the Asamap vyatta distribution. On the up-link, the IPv4 and IPv6 traffic was routed by a dual-stack core router. WIDE Camp participants were able to connect to the environments trough a single SSID, *464exp*, handled by the Layer 2 Cisco WiFi Mesh.

Methodology. For operational capability the proposed metrics are: *configuration capability*, *troubleshooting capability* and *applications capability*. As measurement method for configuration capability, we are considering



(a) Closed experiment topology

(b) Open experiment topology

Figure 1. Experimental setup

Table 1. Workload $Framesize \times Framerate$

No	Size	Rate 10 Mbps	Rate 100 Mbps	No	Size	Rate 10Mbps	Rate 100 Mbps
1	64	14880	148809	7	1518	812	8127
2	128	8445	84459	8	1522	810	8106
3	256	4528	45289	9	2048	604	6044
4	512	2349	23496	10	4096	303	3036
5	1024	1197	11973	11	8192	152	1523
6	1280	961	9615	12	9216	135	1353

a number of configuration tasks, which have been inspired by the abstracted guidelines presented in [27]. The tasks can be organized in three generic groups, *initial setup*, *reconfiguration* and *confirmation*. For ease of reference we have associated 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 can be expressed as a ratio between the number of successfully completed configuration tasks and the total number of tasks. Similarly, for troubleshooting capability we are proposing a number of troubleshooting tasks. The

tasks follow the fault isolation, fault determination and root cause analysis (RCA) guidelines presented in [27]. 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 can also be expressed as a ratio of successful tasks over total number of troubleshooting tasks.

To measure applications capability, inspired by the efforts presented in [14], we have tested a non-exhaustive list of common user applications in relation with the 464 transition technologies. The measurement result can be presented as a ratio between the number of successfully-tested applications and the total number of applications.

4. Empirical results

4.1. Closed Experiment results

The network performance of the devices under testing (DUTs) was compared with a Direct Connection setup, in which the two test platform servers were connected directly. The results have been graphed as a function of frame size. The error bars present the margin of error for the mean, calculated at a 99% level of confidence, using the formula 1

$$\begin{aligned} &moe = z_{\alpha/2}\frac{\sigma}{\sqrt{n}} \\ &\sigma - standard\ deviation, n - sample\ size \\ &z_{\alpha/2} = 2.575 \end{aligned} \tag{1}$$

The average of the results for the 10Mbps and 100Mbps workloads have been summarized in Table 2 and Table 3.

The latency results for the 10Mbps workload, composed of end-to-end delay (Fig. 2a, 2b) and jitter (Fig. 3a, 3b) indicate a better performance for 464XLAT, by comparison with the rest of the technologies. Also, in terms of average, translation-based technologies (MAPt, 464XLAT) had a better performance than encapsulation-based technologies (MAPe, DSLite).

The average throughput results, presented in Fig. 4, show a similar performance for the four technologies. The overall average shows a small lead for DSLite and encapsulation-based technologies in the case of the 10 Mbps workload.

In the case of the 100 Mbps workload results , presented in Fig. 2c, 2d for delay, 3c 3d for jitter, and 4c 4d for throughput, the high values of the Margin of Error do not allow us to draw any clear overall conclusion. However, the results help to point out some *unexpected behaviours*, which are consistent. One example of this is the decrease in throughput for the *1280 frame size*, presented in Fig. 4c and 4d. Another example is the lower throughput of the Direct Connection, which is counter-intuitive. The root causes of these behaviors need further analysis.

The loss rates, with the exception of some outliers for translation-based technologies over UDP (MAPt and 464XLAT), are very close to 0. For the outliers, the maximum loss-rate is approximately 0.003%, considered negligible in most cases.

Considering the overall average of these measurements, the best performance was achieved by MAPe, followed closely by DSLite, MAPt and 464XLAT.

4.2. Data collection and repeatability

For the closed experiment a full factorial design was employed, hence $12(frame\ sizes) \times 2(transport\ layer\ protocols) \times 2(workloads) \times 5(transition\ technologies) \times 1(implementation) = 240$ 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 *5600 min*, or *93 h*. For post-processing the raw data we have spent an average of 20 sec for each experiment, bringing us to a total of *1600 min* or *26 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 Fig. 5.

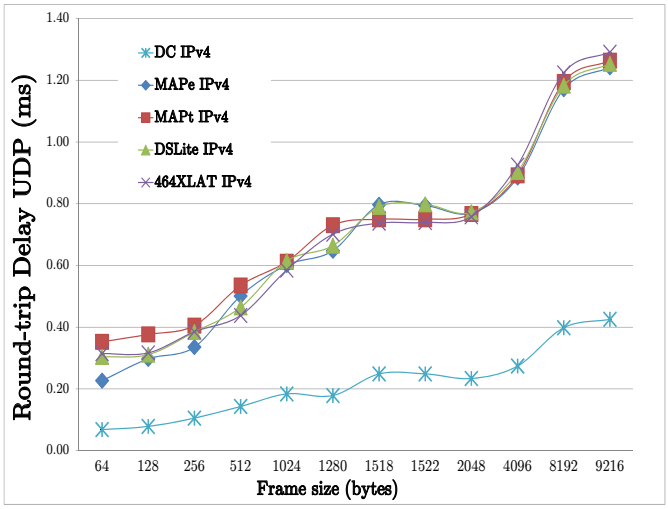
Table 4 presents the average of the relative standard deviation calculated with the formula 2.

Table 2. 10 Mbps Results Averages

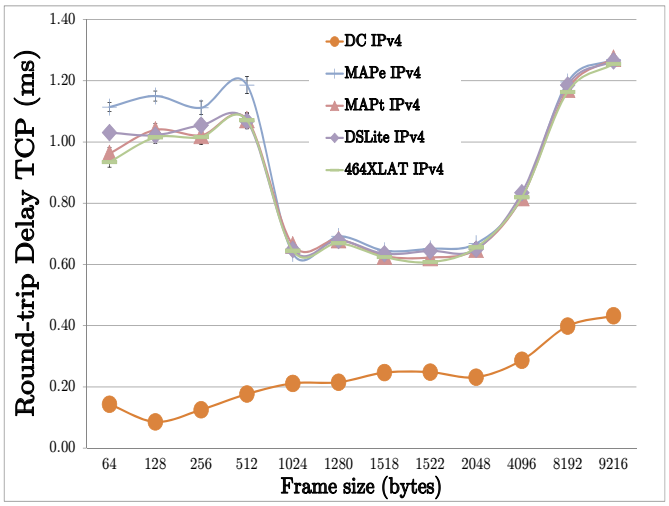
	RT Delay (ms)	+/-	Jitter (ms)	+/-	Throughput (Kbps)	+/-
DC	0.225	0.000	0.016	0.000	8039.0	0.4
MAPe	0.809	0.001	0.167	0.000	7951.8	1.4
MAPt	0.802	0.001	0.177	0.001	7934.6	1.7
DSLite	0.810	0.001	0.167	0.001	7953.5	1.4
464XLAT	0.787	0.001	0.167	0.000	7810.4	1.5

Table 3. 100 Mbps Results Averages

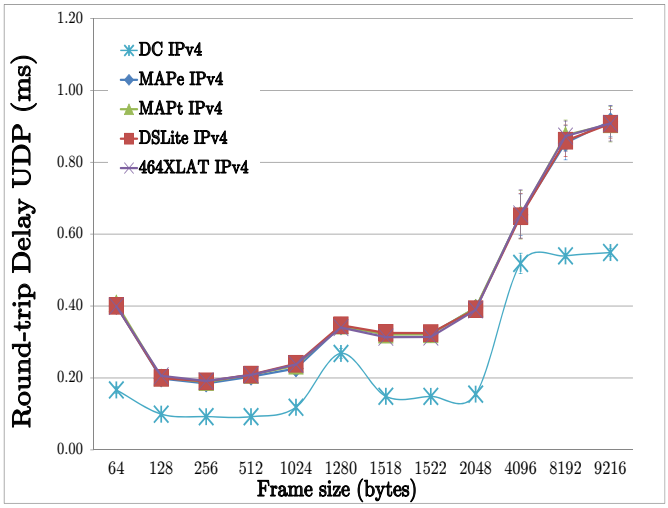
	RT Delay (ms)	+/-	Jitter (ms)	+/-	Throughput (Kbps)	+/-
DC	0.253	0.003	0.130	0.003	57196.1	333.4
MAPe	0.417	0.013	0.486	0.011	58575.2	380.9
MAPt	0.419	0.015	0.480	0.022	58309.4	490.8
DSLite	0.423	0.013	0.493	0.008	58089.2	361.7
464XLAT	0.422	0.013	0.487	0.008	58158.5	361.7



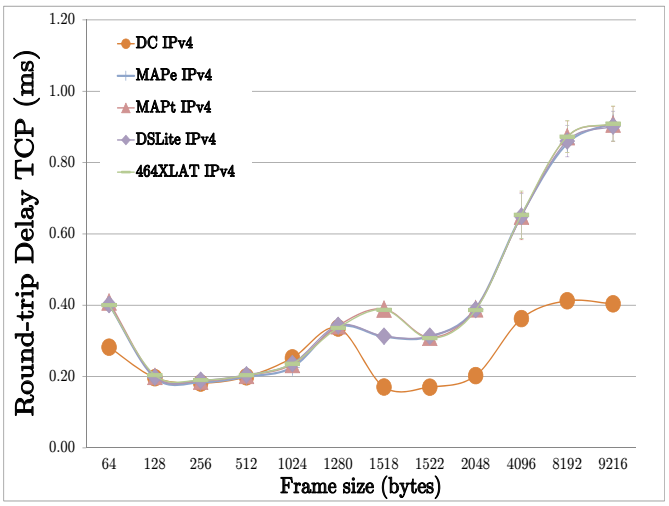
(a) UDP 10Mbps



(b) TCP 10Mbps

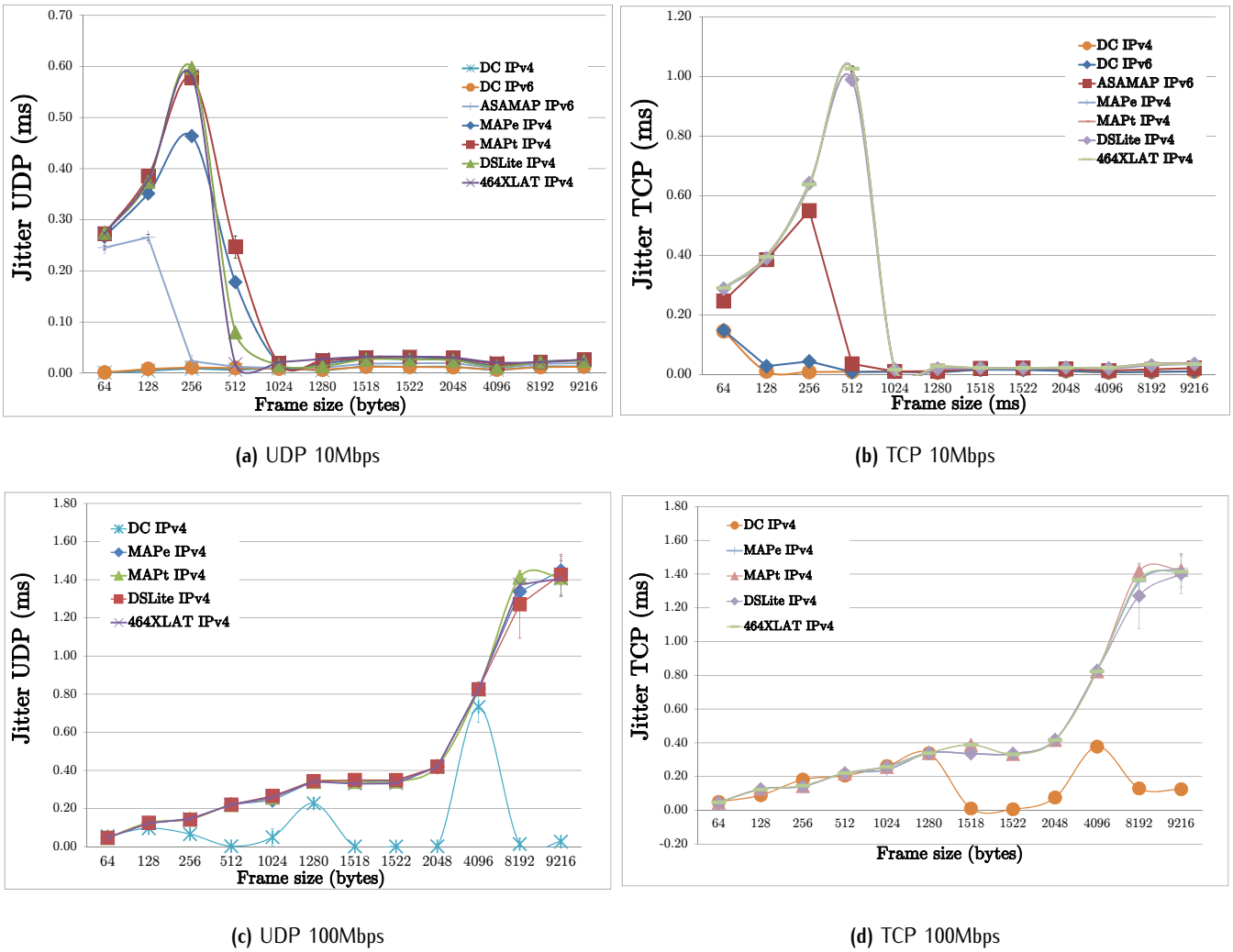


(c) UDP 100Mbps



(d) TCP 100Mbps

Figure 2. Delay results



(a) UDP 10Mbps

(b) TCP 10Mbps

(c) UDP 100Mbps

(d) TCP 100Mbps

Figure 3. Jitter results

Table 4. Relative standard deviation average

	RT Delay (%)	Jitter (%)	Throughput (%)
DC	2.21	3.42	0.93
MAPe	5.10	3.71	1.04
MAPt	5.90	7.48	1.34
DSLite	4.85	2.60	0.99
464XLAT	5.67	5.41	0.93

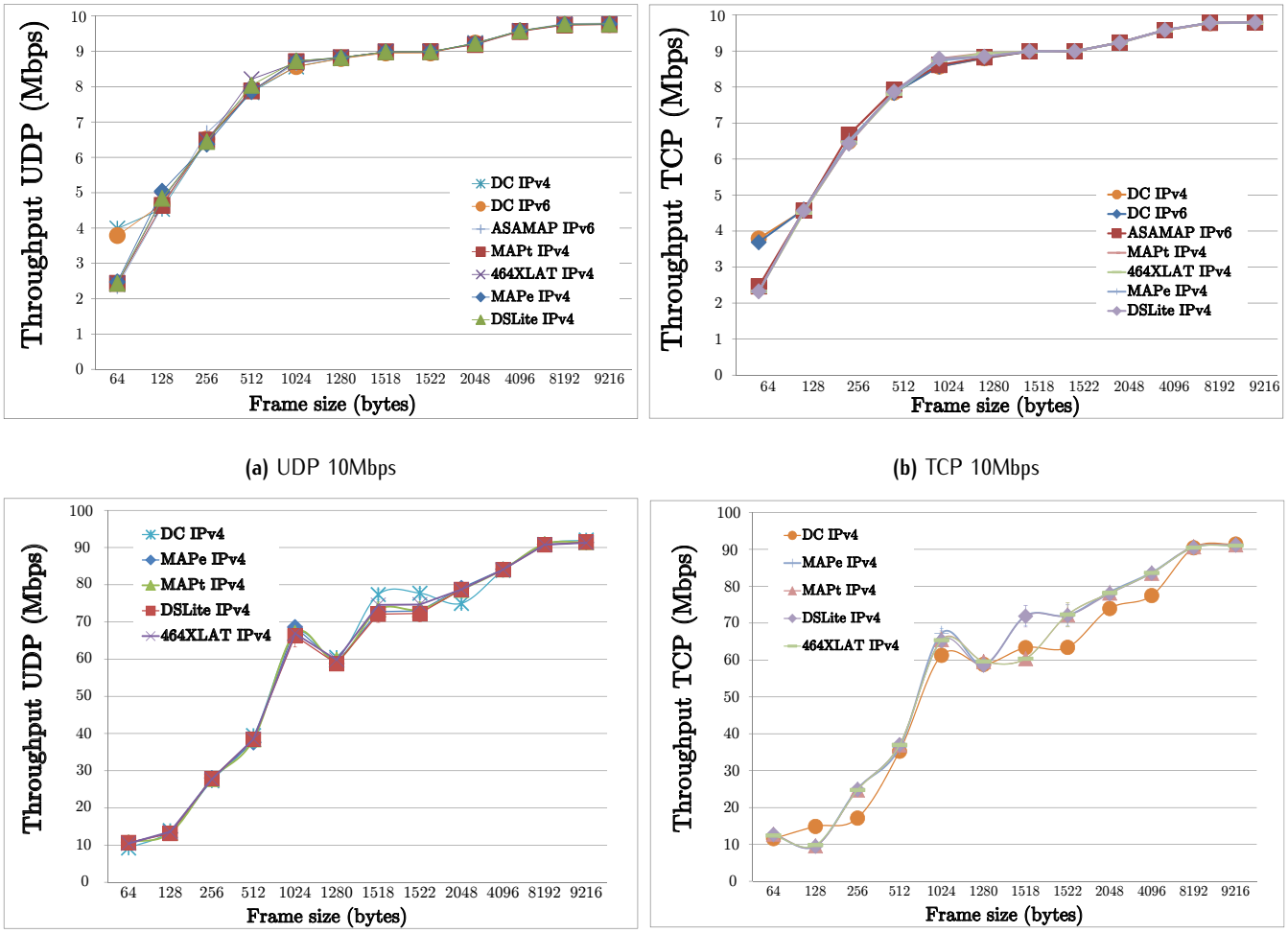
$$\%rsd = \frac{\sigma}{\bar{x}} \times 100$$
$$\sigma - standard\ deviation, \bar{x} - mean \quad (2)$$

The low percentages indicate a low variability among datasets, and by extrapolation a high repeatability for the experiments.

4.3. Open Experiment results

During the four days of the WIDE Camp 1309 event, we had the chance to test the operational capability of the Asamap implementation. However the results are only limited to our experiences. The detailed tasks are included in Appendix A for configuration capability and Appendix B for troubleshooting capability. The results for configuration and troubleshooting capability have been summarized in table 5.

Regarding the configuration capability, most of the tasks have been completed successfully. 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



(a) UDP 10Mbps

(b) TCP 10Mbps

(c) UDP 100Mbps

(d) TCP 100Mbps

Figure 4. Throughput results

would be a beneficial feature. For the troubleshooting capability, most of the 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 one, a self-check sequence is not available yet. This would represent a substantial improvement of the troubleshooting capability.

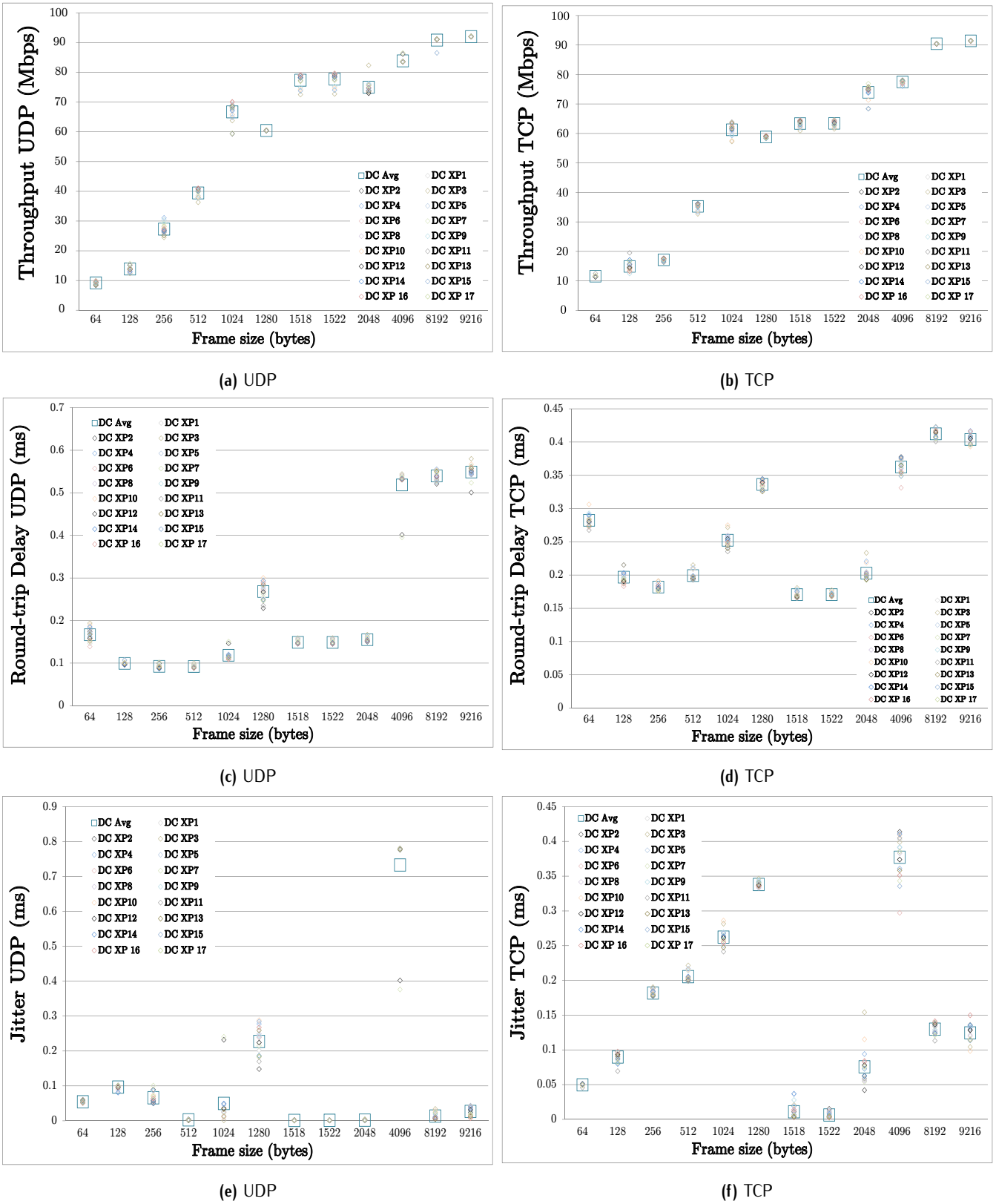
In terms of applications capability, we tested a non-exhaustive list of common applications, in accordance with [14]. The full list of applications and the results are presented in table 6. To summarize we did not encounter any applications troubles for any of the four technologies.

5. Discussion

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.

This article is proposing an answer to that challenge in the form of a network evaluation testbed, called IPv6NET. The contribution of this paper is the detailed testing methodology associated with IPv6NET and the empirical feasibility results, which to the best of our knowledge represent a first in current literature.

Analyzing the empirical results we found that one transition technology is *more feasible* than the rest,



(a) UDP (b) TCP

(c) UDP (d) TCP

(e) UDP (f) TCP

Figure 5. Repeatability of the DC results

Table 5. Operational capability results

Operational Capability		Asamap			
		MAPe	MAPt	464XLAT	DSLite
Configuration Capability	IinitialSetup1	Pass	Pass	Pass	Pass
	IinitialSetup2	Pass	Pass	Pass	Pass
	IinitialSetup3	Fail	Fail	Fail	Fail
	IinitialSetup4	Pass	Pass	Pass	Pass
	IinitialSetup5	Pass	Pass	Pass	Pass
	InitialSetup6	Pass	Pass	Pass	Pass
	Reconfiguration1	Pass	Pass	Pass	Pass
	Reconfiguration2	Pass	Pass	Pass	Pass
	Confirmation1	Pass	Pass	Pass	Pass
	Confirmation2	Pass	Pass	Pass	Pass
Configuration capability result		9/10 = 0.9	9/10 = 0.9	9/10 = 0.9	9/10 = 0.9
Troubleshooting Capability	FaultIsolation1	Pass	Pass	Pass	Pass
	FaultIsolation2	Pass	Pass	Pass	Pass
	FaultDetermination1	Pass	Pass	Pass	Pass
	FaultDetermination2	Pass	Pass	Pass	Pass
	FaultDetermination3	Fail	Fail	Fail	Fail
	RCA1	Pass	Pass	Pass	Pass
	RCA2	Pass	Pass	Pass	Pass
	RCA3	Fail	Fail	Fail	Fail
	RCA4	Pass	Pass	Pass	Pass
	RCA5	Pass	Pass	Pass	Pass
Troubleshooting capability result		8/10 = 0.8	8/10 = 0.8	8/10 = 0.8	8/10 = 0.8

Table 6. Applications capability results

Applications			Asamap			
			MAPe	MAPt	464XLAT	DSLite
Win 7 / Win 8 / Ubuntu 12.04 / Android 2.3	Browsing	Chrome	Pass	Pass	Pass	Pass
		Firefox	Pass	Pass	Pass	Pass
		Dolphin	Pass	Pass	Pass	Pass
	E-mail	Outlook	Pass	Pass	Pass	Pass
		Thunderbird	Pass	Pass	Pass	Pass
		Aquamail	Pass	Pass	Pass	Pass
	IM&VoIP	Skype	Pass	Pass	Pass	Pass
		Facebook	Pass	Pass	Pass	Pass
		Google+	Pass	Pass	Pass	Pass
		VoIP Buster	Pass	Pass	Pass	Pass
		Viber	Pass	Pass	Pass	Pass
		DigiOriunde	Pass	Pass	Pass	Pass
	VPN	OpenVPN	Pass	Pass	Pass	Pass
		Spotflux	Pass	Pass	Pass	Pass
	Cloud	Dropbox	Pass	Pass	Pass	Pass
		GDrive	Pass	Pass	Pass	Pass
	FTP	Filezilla	Pass	Pass	Pass	Pass
	Troubleshooting	puTTY	Pass	Pass	Pass	Pass
		WinSCP	Pass	Pass	Pass	Pass
		ConnectBot	Pass	Pass	Pass	Pass
Applications capability result			20/20 = 1	20/20 = 1	20/20 = 1	20/20 = 1

namely *MAPe*. We have also identified possible performance trends in IPv6 transition technologies benchmarking, for example, encapsulation-based technologies seem to have better throughput performance and translation-based technologies better latency performance. However, we must note that the empirical results are highly dependent on the quality of the implementation. In other words, the same transition technology can perform differently under different implementations. This is why we decided to test the IPv6 transition technologies on a per-implementation basis.

We were also able to point out some *unexpected behaviors*, which could have been overlooked if simulators or analytical tools are employed. This underlines the need for a testbed and gives us motivation for a further root cause analysis. The high repeatability results indicate that the methodology is also easy to replicate on systems with the same hardware and software characteristics.

A limitation of this method is represented by the lack of control data, given 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.

The empirical results can serve as a direct guideline to enterprise network operators faced with a similar transition scenario. Many enterprise networks nowadays include industrial segments, which in many cases run over IP networks. The performance and operational aspects of the underlaying networks can have a critical impact on industrial applications. In this context, the guidelines and empirical data can serve as a rough impact analysis of the IPv6 transition on the industrial network segments.

Another limitation of this approach is represented by the diversity and complexity of existing production networks by comparison with the presented scenario. 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 this project in an exhaustive IPv6 transition resource.

6. Conclusion

In this article we have introduced IPv6NET, a project aiming to empirically analyze the feasibility of IPv6 transition technologies in relation with specific network scenarios. From the methodology standpoint, IPv6NET combines two types of testing environments: closed environments for thorough network performance data, and open environments for operational data. By using the proposed IPv6NET and the associated methodology we were able to indicate MAPe as having the best network performance, followed closely by DSLite, MAPt and 464XLAT. We were also able to identify some performance general guidelines, e.g. for latency, the translation-based technologies (464XLAT, MAPt) had a better performance. For throughput, the results were in favor of encapsulation-based technologies (MAPe, DSLite). However, we must note that the empirical results are highly dependent on the quality of the used implementation. Consequently, this results should be interpreted in association with the Asamap implementation.

Some of the empirical results also pointed out *unexpected behaviors*, which further underline the need for a testbed. By replicating the experiments 17 times, on 68 different nodes, we have shown that the proposed methodology has a a high level of repeatability. In terms of operational capability, we have proposed a task-based methodology. However, the data we have so far is limited to our experiences. As a future plan, we would like to replicate the proposed system and the associated methodology, and organize a survey with people of different network operating skills. Also as future work, we intend to increase the scale of the network template, and propose an associated metric for scalability. Another future step is proposing a unique general feasibility indicator (GFI), associated with each transition technology, which would help to better centralize and compare the the results.

Acknowledgments

The authors would like to thank Mr. Masakazu Asama for providing the vyatta Asamap distribution, upon which the experimental networks were implemented. Special thanks should be given to the teams at NICT StarBED and WIDE Project for their continuous support.

Appendix A. Configuration capability tasks

For the Asamap vyatta implementation the tasks 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 B. Troubleshooting capability tasks

The troubleshooting capability test of the Asamap vyatta implementation contained the following tasks.

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

References

[1] APNIC, "IPv6 measurements for The World," Apr. 2014.

[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.

[3] J. Bound, "IPv6 Enterprise Network Scenarios." RFC 4057 (Informational), June 2005.

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

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

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

[7] 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, (Piscataway, NJ, USA), pp. 242–246, IEEE Press, 2009.

[8] 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.

[9] 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.

[10] 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.

[11] 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.

[12] 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.

[13] 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.

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

[15] 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.

[16] 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-08, Aug. 2013.

[17] 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-04, Sept. 2013.

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

[19] M. Mawatari, M. Kawashima, and C. Byrne, "464XLAT: Combination of Stateful and Stateless Translation." RFC 6877, Apr. 2013.
[20] N. Matsuhira, " Stateless Automatic IPv4 over IPv6 Encapsulation / Decapsulation." draft-matsuhira-sa46t-spec-07, July 2013.
[21] M. Asama, "MAP supported Vyatta. Online available: http://enog.jp/ masakazu/vyatta/map/," Mar. 2014.
[22] T. Miyachi, K. 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.
[23] 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.
[24] C. Popoviciu, A. Hamza, G. V. de Velde, and D. Dugatkin, "Ipv6 benchmarking methodology for network interconnect devices," 2008.
[25] S. Bradner and J. McQuaid, "Benchmarking methodology for network interconnect devices," 1999.
[26] 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.
[27] D. Harrington, "Guidelines for Considering Operations and Management of New Protocols and Protocol Extensions." RFC 5706 (Informational), Nov. 2009.