Paper

Performance Tests of Xen-based Node for Future Internet IIP Initiative

Grzegorz Rzym and Krzysztof Wajda

Department of Telecommunications, AGH University of Science and Technology, Kraków, Poland

Abstract—In this paper the authors intend to report results for performance evaluation of Xen-based node providing both transport and computational functionalities. This node design is proposed as the implementation platform developed for the Polish Initiative of Future Internet called System IIP. In particular, we search for mutual dependence among transport and computational performance parameters of the node. Our investigations show that there is significant dependence among performance indices, such as virtual link bandwidth and node's processing power, strongly depending on frame size. The tests and measurements were done according to fundamental methodology designed for network devices, described in RFC 2544. The goal of those investigations is to design a provisioning module allocating both transport and computational resources.

Keywords—benchmarking, Future Internet, measurements, virtualization, Xen.

1. Introduction

We are witnessing growing efforts in recent years aiming at designing a new architecture of the Internet, not based exlusively on classical IP protocols family. The general ideal is to design new networking environment being strongly oriented towards emerging applications and networking paradigms, such as Internet of Things (IoT) and Content Aware Networking (CAN). Besides mentioned above goals are also modern mechanisms such as virtualization and parallelization of networks, complemented by new (or revisited) approaches to data, control and management planes.

Most promising and pioneering initiatives towards Next Generation Internet comprise Japanese Akari [1], US-based GENI [2] and European project 4WARD [3]. Involvement of EU-promoted Future Internet Assembly (FIA), supported by US-based National Scence Foundation (NFS), also ETSI, and ITU-T has given a significant and fruitful impact on research and experimental works in this area.

This paper is organized as follows: Section 2 presents Future Internet initiative, Section 3 outlines the important aspects of Xen relative to virtual forwarding, bridging and routing, in Section 4 implemented node model is invited, Section 5 presents the experimental setup that authors use to perform our evaluation, and Section 6 presents results. The paper is concluded in Section 7.

2. Related Work

Evaluating of transport features and capabilities of Xen nodes can be done experimentally due to lack of theoretical models.

Initial tests of Xen 1.0 presented in [4] have shown that performance of Xen platform is practically equivalent to the performance of native Linux system. The developers from University of Cambridge have shown that Xen can be widely used to proceed transmitted data. Running simultaneously 128 virtual machines caused only 7.5% loss of total throughput compared to Linux with 5 ms maximum scheduling "slice". When scheduling "slice" were set to 50 ms (the default value used by ESX Server) the throughput curve was very close to the native Linux. Mean response time in such configured system was 5.4 ms.

N. Egi *et al*. in [5] presented the forwarding performance of driver domain (dom0) and virtual machines (domU). They configured Xen network with tree types of mechanisms to transmit packets between physical network devices and virtual interfaces and inside VMs: bridging, routing and hybrid method (combination of routing with bridging). Observations included in [5] have shown that the performance of hybrid connection inside Xen node is 30% lower than using only bridging or routing. Since CPU is reported as the bottleneck of PC-based virtual router the authors suggested that domU should only host the control path and the forwarding path should be located and served in dom0. This solution leads to avoidance of context switching overheads. The authors also compared network performance for driver domain and native Linux system. Their results have shown very close performance in forwarding of short frames.

In [6] three techniques for network optimization in the Xen were proposed:

– redefinition of the virtual network interfaces which allows guest domains to incorporate high-level network offload features (i.e., TCP segmentation offloading, scatter-gather I/O and TCP checksum offloading) in physical NICs,

– new data path between dom0 and domU avoids data remapping operations and proposal of usage of superpages and global page memory mapping.

All these techniques provide 35% growth of data transmission performance in driver domain and 18% growth in guest's domains.
Pujolle *et al*. presented in [7] evaluation of Xen transmission performance. The authors believed that forwarding should be accomplished inside virtual machines (virtual routers). Since the forwarding performance of dom0 is better than domU they propose several optimization methods. They provide a mechanism to guarantee the required system throughput and latency by:

– optimized CPU allocation, prioritizing packets inside dom0 before switching them to the target virtual machine,

– appropriate configuration of the Xen Credit scheduler and finally dedicated virtual routers to carry flows with different priorities.

This allows to forward quality-sensitive flows with acceptable delay and throughput.
Further experimental results applicable to IIP project are recently investigated and analyzed in [8]. Adamczyk and Chydziński studied isolation between virtual machines across virtual network adapters. Their investigations have shown lack of proper performance isolation among virtual machines. Xen Netback driver that is responsible for the scheduling work uses simple round-robin algorithm. In addition several network adapters can be mapped to the same Netback kernel thread. The authors propose additional two parameters (priority and min rate) for every virtual network adapter to improve the network performance isolation.
Recent works focus only on throughput and forwarding performance of PC-based virtual routers under different configurations. The mutual dependence among transport and node performance parameters have not been investigated in any work yet. The obtained results can be used to design a provisioning module combining both transport and processing resources.

3. Future Internet Initiative

Designing and experimenting with novel architecture of Future Internet are ongoing as a target of a Polish initiative called System IIP. The structure of the System IIP is based on implementation of four levels of the architecture (bottom-up description): physical infrastructure (L1), virtualization (L2), Parallel Internets (PIs, L3) and virtual networks (L4).
The Future Internet Engineering project assumes existence of three Parallel Internets (PIs): IPv6 Quality of Service (IPv6 QoS), Content Aware Network (CAN) and Data Stream Switching (DSS). More details can be found in [9] and [10].

Xen is one of three system virtualization environment used in IIP project. Others are NetFPGA and EZchip NP-3 [11]. The advantage of Xen is that it can be run on each computer that supports virtualization, it is an open source and fully programmable environment.

4. Xen as Virtualization Platform with Bridging/Routing Features

Xen [4] is an open source virtualization platform. It consist of the *Hypervisor* and *Domains*.
Hypervisor is an abstraction software layer running directly above the hardware. This layer is an interface between operations running on the hardware (CPU, I/O devices, etc.) and the guest's operating system. Its main tasks are CPU scheduling, memory management and forwarding of interrupts (hypercalls in Xen).
Xen runs virtual machines in separate environments known as domains. There are two kinds of domains: privileged Domain0 and unprivileged DomainU. The former, the driver Domain0 is responsible for the communication between guest domains DomainU and the devices. It implements all device drivers and has direct connection through *event channel* with these devices. DomainU is an unprivileged domain started by Domain0 and runs guest operating systems.
Xen implements two types of drivers: back-end and front-end. Back-end drivers are used by Domain0 to communicate from one side with hardware, from the other – with front-end implementation in DomainU. This concept of distributed drivers are used by block and networks devices.
There are three types of mechanisms to transmit frames between physical network devices and virtual interfaces: bridging, routing and additionally Network Address Translation (NAT) mechanism.

4.1. Bridging in Xen

Bridging is the default network configuration in Xen. It uses standard Linux bridging mechanism for packets forwarding. All guest domains can share the same Network Interface Card (NIC). Back-end devices are connected to the physical NIC and are represented as a pethX devices in the system.
When packet arrives to NIC Hypervisor it is notified by physical interrupt. Then Hypervisor forwards that information to the Domain0 through the event channel and copies the packet to its address space. When guest domain receives its scheduled time slot it sees the notification and looks for the packet in the common memory page. Next, the guest domain can start processing the packet. The return route is similar. We can observe two way communication: the communication between the Hypervisior and the driver domain and the communication between the driver domain and the guest domain.

4.2. Routing in Xen

In a routed network configuration in Xen a point-to-point link is created between the driver domain Domain0 and each guest virtual network interface. Is it necessary for each guest domain have an IP address. Packets forwarded between physical interfaces and other virtual network cards are routed like in native Linux.
The use of NAT allows us to create many private Local Areas Networks (LANs) inside one physical computer. System configuration is the same as in the case of network routing. In order to address translation it is necessary to use standard Linux tools such as iptables (also used to traffic filtering).

5. The Node Model

The IIP node consists of following elements: frame classifier, scheduler and three virtual machines corresponding to three various Parallel Internets presented in the Fig. 1.

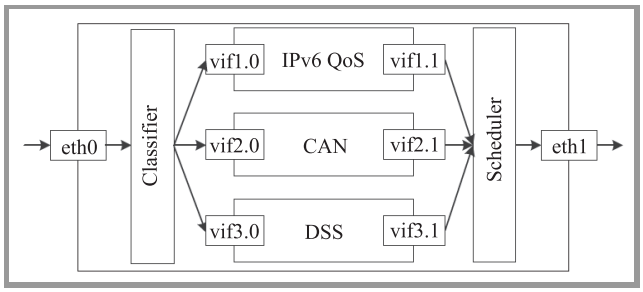

Fig. 1. The node model.

5.1. Frames Classifier

The IIP System Classifier is a mechanism that allows for demultiplexing of incoming traffic to the one of three virtual machines. Therefore, the IIP System defines a new Parallel Internet Header (PIH). It has 8 bits and is located next to the Ethernet frame header.
The Classifier is located in the main virtual machine – Domain0. This enables the PDU classification by using modified *ebtables* rules [12].

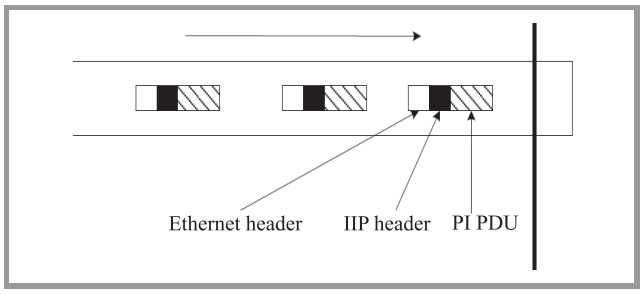

Fig. 2. IIP incoming data stream.

In the Fig. 2 an incoming data stream with distinguished order of the headers can be observed [13].

5.2. Virtual Node

As a virtual router we use a separated virtual machines. As guest operating system we run Gentoo Linux with software specified to perform different tasks for each PI. Such solution significantly simplifies the implementation and allows for future expansion without major modifications of the existing system.

5.3. Scheduler

The main goal of the scheduling mechanism is the division of physical link resources (its capacity C) between each Parallel Internets [13]. In the case of three previously mentioned PIs the following condition 1 should be satisfied:

$$C_{IPv6QoS} + C_{CAN} + C_{DSS} \leqslant C \tag{1}$$

In the first level the scheduling mechanism is based on a cycle. A duration o each phase of the cycle should be set in the provisioning (i.e. dimensioning) phase. The cycle is composed of fixed number of phases. It is worth mentioning that as a result of using Scheduler the link data rate is lost – it can not be used by other virtual machines (*non-work conserving* algorithm). Scheduling mechanism for the second level is different and suitable for each PI. In the Fig. 3 the two-level scheduling mechanism is shown [13].
In the case of the Xen platform the *Netback* driver is responsible for scheduling process of outgoing traffic. Used standard FIFO queue and *Credit Scheduler* allows us to specify the maximum outgoing throughput.

6. Experimental Settings

Evaluating of Xen-based node was performed by using HP ProLiant DL360 G6 server with 2 Intel Xeon X5660 2.80 GHz processors, 6×4 GB DDR3 1333 MHz RAM memory, $4 \times$ Intel 82571EB NIC and 500 GB HP SCSI hard drive.

6.1. Test Topology

General scheme of network performance testing Xen-based node is presented in the Fig. 4.
As a Traffic Generator and Traffic Sink a Spirent TestCenter STC-2000 platform was used. This hybrid (hardware and software) device provides wide variety of network test configurations for functions ranging from 2nd to 7th OSI-ISO layers. It also offers high performance traffic generation and measurements.
Network tests were performed according to recommendations defined in RFC2544 [14] and RFC5180 [15]. Generated bandwidth was in the range from 10 Mbit/s to 330 Mbit/s for each StreamBlock – three simultaneous streams from three Parallel Internets (that gives altogether 990 Mbit/s of maximum throughput). Tests were

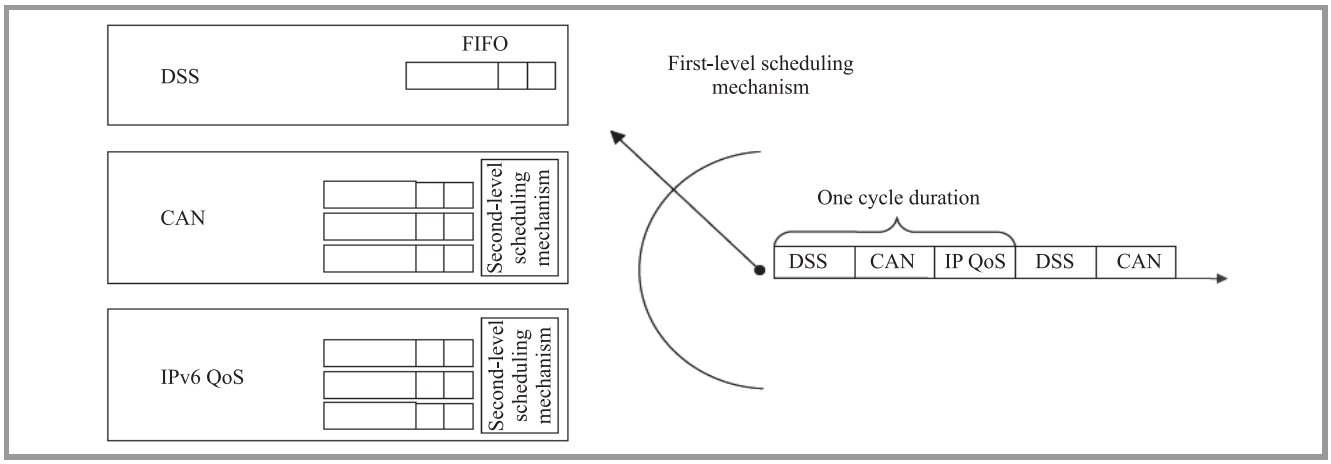

Fig. 3. Scheduling mechanism.

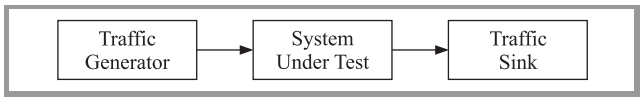

Fig. 4. Network topology used during tests.

run with the following frame sizes: 79, 128, 256, 512, 1024, 1280, and 1518 bytes since this parameter has significant impact on the performance. One measurement run took 110 seconds.

6.2. Test Description

Since we did not find any recommendation for virtual machines performance measurement we proposed the following algorithm:

1. Start the server.
2. Run simultaneously 3 IIP virtual machines.
3. Wait 5 minutes for system stabilization.
4. Run test on Spirent TestCenter platform according to scenarios proposed in recommendations.
5. Save results.
6. Restart server and go back to the first point.

First tests to evaluate the native performance of our system is run. These results will help in evaluation of the overhead imposed by the use of virtualization and IIP System implementation.
Next, we run simultaneously 3 virtual machines providing data stream transmission within the different Parallel Internets. Each guest domain had allocated (pinned) 2 processors, 4 GB RAM memory, 5 GB virtual hard drive as an image file. The driver domain had pinned 8 cores. When network performance was measured we run a script that monitors the usage of computational power of processors. Each test were performed with 100 μs phase size for each virtual machine.

7. Results

In this section the results of experiments carried in two settings (conditions) are presented. For the first test set there were carried experiments for fixed frame sizes, according to RFC 2544 (i.e., each run for fixed and different size of frame: 79, 128, 256, 512, 1024, 1280 and 1518 bytes). Next, in second test set we make measurements with random frame size. Results are presented in two versions: X axis is expressed either in Mbit/s (suitable for general purposes of dimensioning) or kpps (kilo packets per second), reflecting transport efficiency of the node.

7.1. Fixed Frame Size

We can observe a strong relation between maximum throughput and frame size. For a given throughput maximum number of small frames (i.e., 128 bytes) that can be processed by the node is much lower than for the larger frames (e.g., 1024 bytes) because number of headers that the node must analyze in the latter case is lower. In the Fig. 5a node throughput for each tested frame size is presented. As we can observe in the Fig. 5b maximum number of handled packets for small frames is variable.
As can be observed in the Fig. 6, the average frame latency is strongly correlated with maximum throughput. The latency is measured by Spirent device as the difference in time between the instant of receiving frame and sending the frame. Until the frames are not dropped, the mean latency remains low (about 150 μs). Above the maximum throughput the average frame delay increases significantly and then remains at a constant level – for frames from 79 to 512 bytes of size the average latency above the maximum throughput tends to 20 ms, for bigger frames latency can be even higher.
Our investigations show that for the required bandwidth the processing power utilization for each virtual machine is larger when handling smaller frames, i.e., for 150 Mbit/s about 50% for 128 bytes and 16% for 1024 bytes frames. In the Figs. 7–8 the impact of traffic load on the utilization

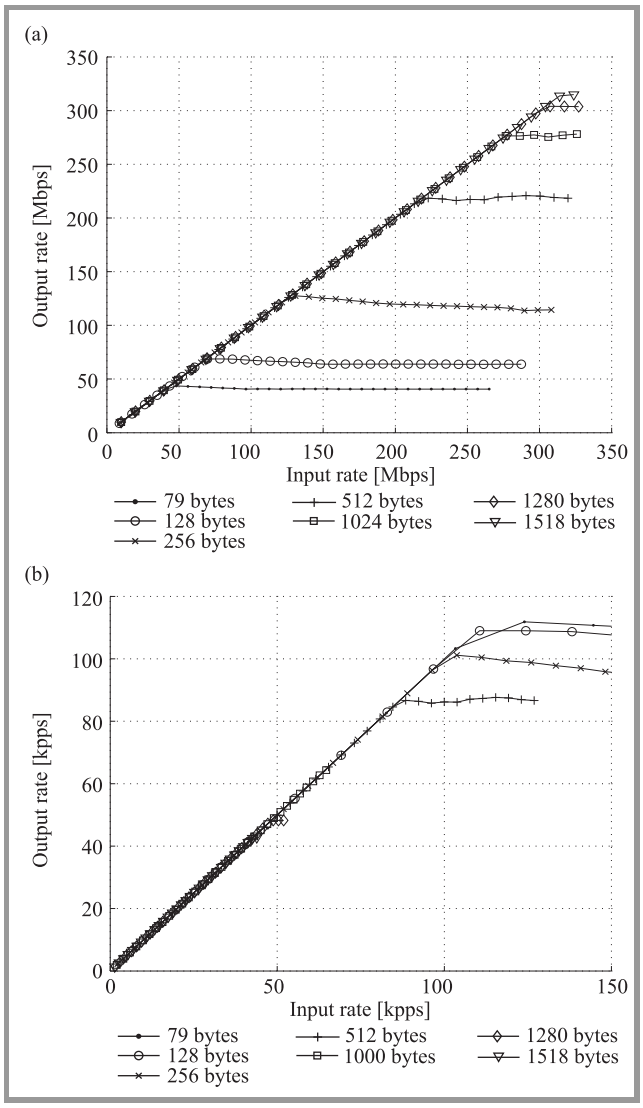

Fig. 5. Node throughput for a specified frame size.

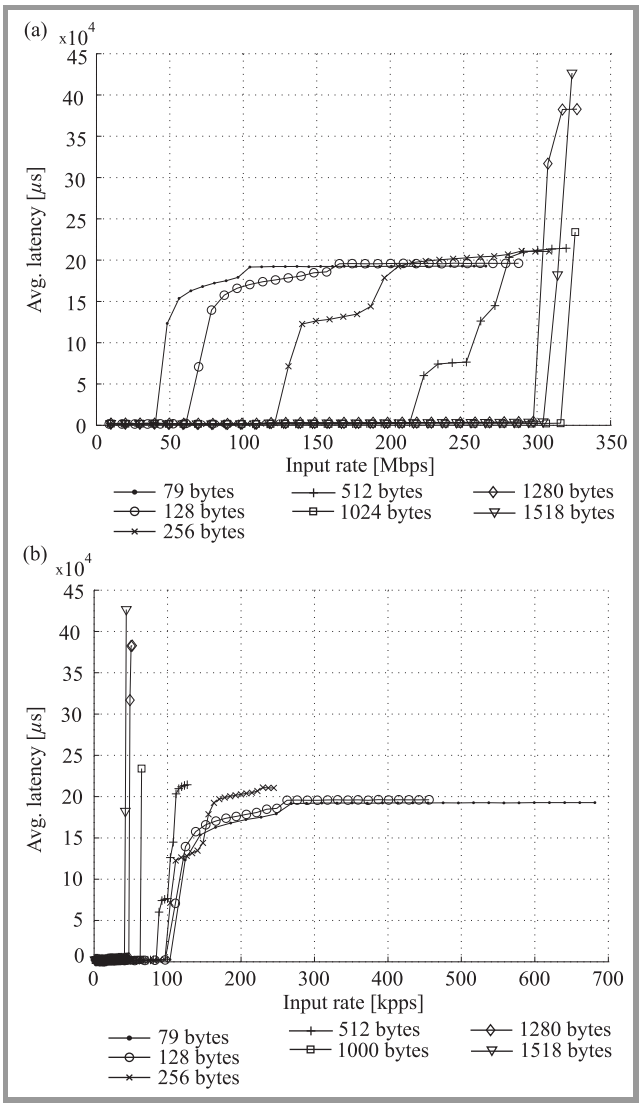

Fig. 6. Node latency for a specified frame size.

of processing power is shown, allocated to each virtual machines for 128 and 1024 bytes frame size, respectively. Presented results show large differences in the use of node processing power depending on the frame size. For bigger frames (i.e., 1024 bytes) system load increases almost linearly with the size of incoming traffic rate in the whole range. For smaller frames system load is linear until frames are lost, then system utilization is maintained at a constant level.

As we expected, the processors assigned to the CAN virtual machine are most utilized. This result is caused by necessary analysis of data contained in the PDU frame by machine kernel after removing of PIH header according to the specifications presented in [13]. Virtual machine serving traffic from DSS Parallel Internet uses the least amount of the server computational power.

It is also visible uniformity of load for each processor assigned to virtual machines. First processors are used much harder than the other in each virtual machine. This difference increases with the increase of the bandwidth. Also

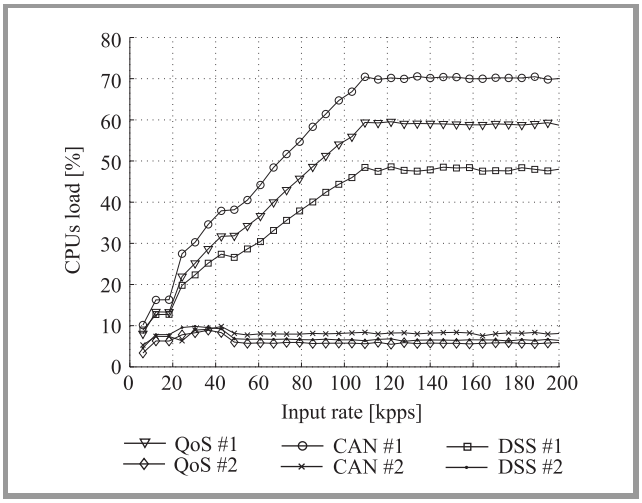

Fig. 7. System load for 128 bytes frame size.

important is the fact that when frame size increases, the difference in the use of processing power in each machine

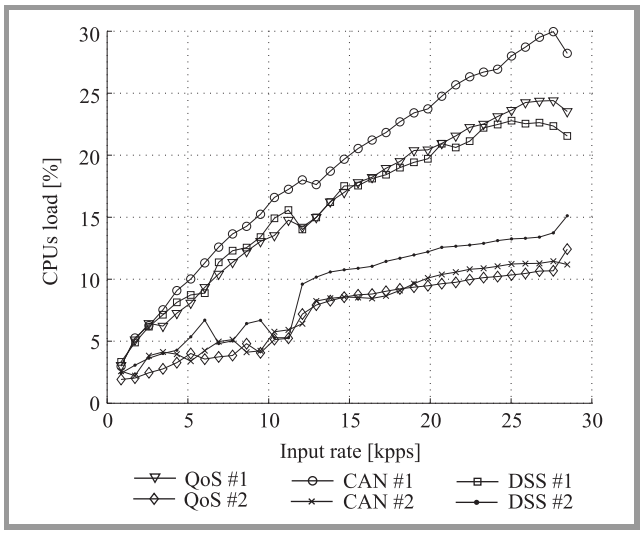

Fig. 8. System load for 1024 bytes frame size.

becomes lower. Larger fluctuations in load are noticeable for larger incoming throughput.

7.2. Random Frame Size

Next, we tested our system with random frame size. We used continuous linear distribution in the range from 79 to 1518 bytes to generate random frames (the only possible distribution in the Spirent TestCenter application). Tests were repeated 10 times and average results are presented. Performance characteristics suggest a linear increase in computational load for each processor with increasing throughput. Only for the largest bandwidth the breakdown in the plots can be observed suggesting problems with handling so high volume of traffic (Fig. 9).

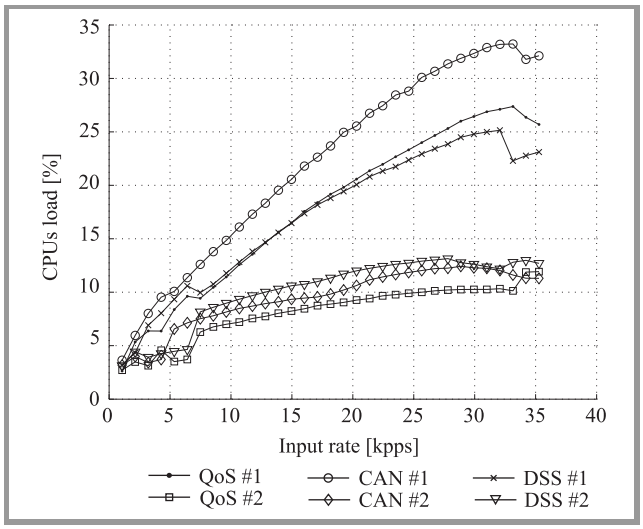

Fig. 9. System load for a random frame size.

As before, the greatest demand for computing power is imposed on VM handled traffic from CAN Parallel Internet. This machine performs the largest number of operations on the received header. There is no significant difference between the use of processor power for QoS IPv6 and DSS machines.

In the Figs. 10–11 capabilities of traffic passing through the system can be observed. These charts confirm relations that can be seen in the Fig. 9. When frames are dropped the average latency increases strongly from several microseconds to tens of milliseconds.

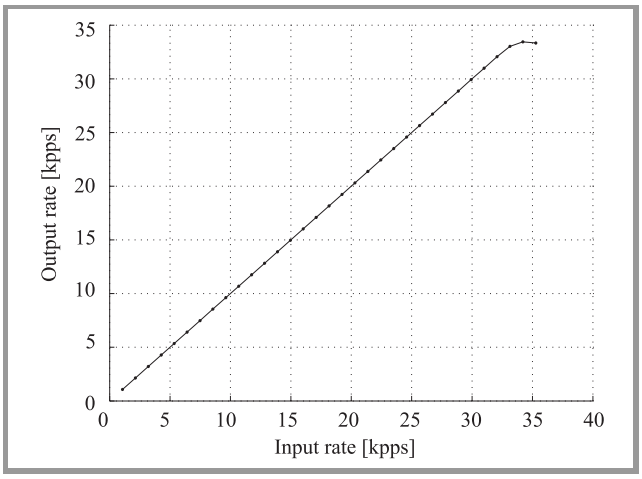

Fig. 10. Node throughput for a random frame size.

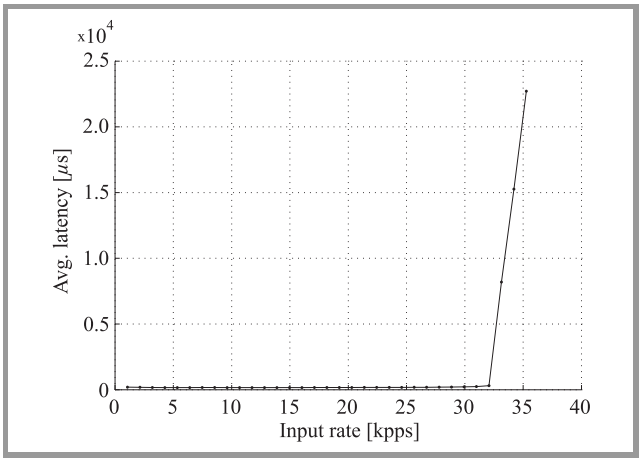

Fig. 11. Node latency for a random frame size.

Continuous uniform distribution is not the best to approximate the actual Internet traffic characterization. Unfortunately, only the linear distribution of traffic generation was possible in the Spirent TestCenter device. However, occurring randomness can in some way simulate Internet traffic. It should be noted that each Parallel Internet has individual traffic specification that it supports. In a case of CAN it will be a lot of queries searching for published content, and in IPv6 QoS the traffic will be served with accordance with specified classes.

7.3. Fitting of Linear Coefficients

Our investigations have shown strong correlation between indices, such as virtual link bandwidth and node's pro-

Table 1
Coefficients of linear equation

Frame size	QoS			CAN			DSS		
	a	b	R-square	a	b	R-square	a	b	R-square
79	3.2580	17.4400	0.9859	3.8780	20.7800	0.9827	2.5440	15.9600	0.9885
128	2.6810	8.4820	0.9859	3.1700	10.7200	0.9850	2.0680	9.1430	0.9832
256	1.4670	9.5110	0.9832	1.7810	12.1500	0.9882	1.1080	10.5500	0.9626
512	0.9354	7.4550	0.9561	1.0680	8.9720	0.9614	0.7271	9.0230	0.9182
1024	0.6521	4.4250	0.9842	0.7717	5.1020	0.9808	0.5987	5.2490	0.9637
1280	0.3863	4.8230	0.9646	0.4427	4.9360	0.9117	0.4262	4.5550	0.8367
1518	0.3717	3.9690	0.9614	0.3769	4.7130	0.8977	0.4678	3.3700	0.9452

cessing power, being strongly depending on frame size. This relation is linear, so we propose to find best coefficients of the linear Eq. 2. From Eq. 2 it is possible to calculate required processing power (the percentage of processor usage). Since first processor in each virtual node during our tests were more utilized than the second one, we focus on the calculation of coefficients only the first CPU.

$$CPU_{usage} = a \cdot x + b \tag{2}$$

where: a, b – coefficients of linear equation, CPU_{usage} – observed processor utilization, x – given throughput in kpps.

We have calculated best coefficients of this linear equation for each tested frame size. Values of these coefficients and coefficients of determination (R-square) [16] are shown in the Table 1. Sample fit for the CAN virtual machine handling 256 bytes size of frames is shown in the Fig. 12.

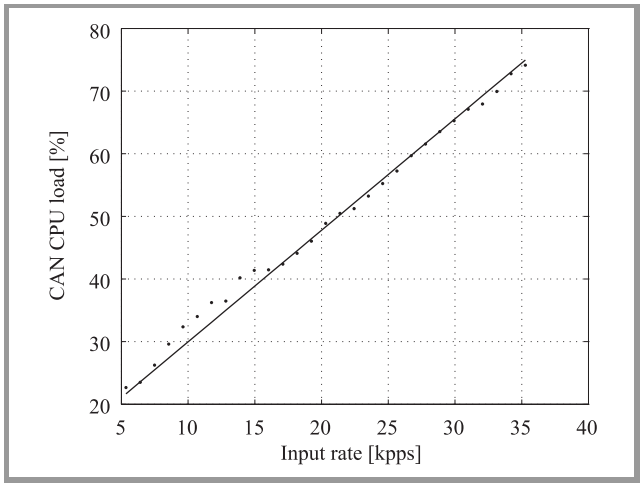

Fig. 12. Sample fit.

As we can observe in Table 1 there is strong dependance of adjusted coefficients from the frame size. Calculated coefficients can be used to estimate required computational power of the virtual node for given throughput value in provisioning phase of IIP System [17].

8. Conclusion

In this paper the preliminary results for performance evaluation of Xen-based node possessing both transport and computational functionalities were presented.
Performed tests indicate the rightness of the Xen implementation for the IIP System. Maximum use of CPU computing power for random frame sizes for each virtual machines does not exceed 35%. This allows for another or more complex program implementations associated with the processing of network traffic being performed inside each virtual machine. It is also possible to run programs not directly related with such procedures.
It is worth mentioning that the use of cyclic scheduler imposes a significant reduction in network performance compared the solution without it. This is due to the fact that the division of the total time slots causes larger or smaller periods of inactivity, in which the server can not send data. For example, when the frame 1518 comes to the scheduler it may be not possible to send it, because the remaining time slot can be too short. Then this frame must be queued and can be sent only after obtaining access to the link in the next time slot.
Our investigations provide linear model that can be used to estimate required processing power of the virtual machine for required throughput in dimensioning phase of IIP System.
Also it should be noted that the tested node constitutes the transport part of larger IIP System node.
The IIP project is now within implementation phase and this enables for running tests showing behavior of transport platforms and also to infer into inside the system performance in order to obtain measures quantifying both transport and processing aspects of served streams.

Acknowledgments

This work has been supported in part by the Polish Ministry of Science and Higher Education under the European

Regional Development Fund, Grant no. POIG.01.01.02-00-045/09-00 Future Internet Engineering. The authors would like to thank Błażej Adamczyk for valuable discussions.

References

[1] Akari project [Online]. Available: http://akari-project.nict.go.jp/eng/index2.htm/

[2] GENI project [Online]. Available: http://www.geni.net/

[3] 4WARD project [Online]. Available: http://www.4ward-project.eu/

[4] P. Barham *et al.*, "Xen and the art of virtualization", in *Proc. 19th ACM Symp. Oper. Sys. Princip. SOSP 2003*, New York, USA, 2003, pp. 164–177.

[5] N. Egi *et al.*, "Evaluating Xen for router virtualization", *Proc. 16th Int. Conf. Comp. Commun. Netw. ICCCN 2007*, Honolulu, Hawaii USA, 2007, pp. 1256–1261.

[6] A. Menon, A. L. Cox, and W. Zwaenepoel, "Optimizing network virtualization in xen", *in Proc. Ann. Tech. Conf. USENIX'06*, Boston, MA, USA, 2006, pp. 15–28.

[7] M. Bourguiba, K. Haddadou, and G. Pujolle, "Evaluating and enhancing Xen-based virtual routers to support real-time applications", in *Proc. IEEE Consum. Commun. Netw. Conf.*, Las Vegas, USA, 2010, pp. 1–5.

[8] B. Adamczyk and A. Chydziński, "On the performance isolation across virtual network adapters in Xen", in *Proc. 2nd Int. Conf. Cloud Comput. GRIDs Virtual. CLOUD COMPUTING 2011*, Rome, Italy, 2011, pp. 222–227.

[9] W. Burakowski *et al.*, "Architektura Systemu IIP", *Krajowe Sympozjum Telekomunikacji i Teleinformatyki*, Łódź, Poland, 2011, pp. 720–722 (in Polish).

[10] IIP project [Online]. Available: http://www.iip.net.pl/

[11] P. Zwierko *et al.*, "Platformy wirtualizacji dla Systemu IIP", *Krajowe Sympozjum Telekomunikacji i Teleinformatyki*, Łódź, Poland, 2011, pp. 824–831 (in Polish).

[12] B. De Schuymer *et al.*, "Ebtables" [Online]. Available: http://ebtables.sourceforge.net/

[13] W. Burakowski *et al.*, "Idealne urządzenie umożliwiające wirtualizację infrastruktury sieciowej w Systemie IIP", *Krajowe Sympozjum Telekomunikacji i Teleinformatyki*, Łódź, Poland, 2011, pp. 818–823 (in Polish).

[14] S. Bradner and J. McQuaid, "Benchmarking Methodology for Network Interconnect Devices", Request for Comments: RFC 2544, 1999.

[15] C. Popoviciu, A. Hamza, G. Van de Velde, and D. Dugatkin, "IPv6 Benchmarking Methodology for Network Interconnect Devices", Request for Comments: RFC 5180, 2008.

[16] C. A. Colin *et al.*, "An R-squared measure of goodness of fit for some common nonlinear regression models", *J. Econometrics*, vol. 77, no. 2, pp. 1790–1792, 1997.

[17] J. Gozdecki, M. Kantor, K. Wajda, and J. Rak, "A flexible provisioning module optimizing utilization of resources for the Future Internet IIP initiative", in *Proc. XVth Int. Telecommun. Netw. Strat. Plan. Symp. NETWORKS 2012*, Rome, Italy, 2012, pp. 1–6.

Grzegorz Rzym received his M.Sc. in Electronics and Telecommunications in 2012 and B.Sc. in Acoustic Engineering in 2013, both from AGH University of Science and Technology, Poland. Currently, he is a Ph.D. student at the Department of Telecommunications. His research interests cover management system design and implementation and virtualization.
E-mail: rzym@kt.agh.edu.pl
Department of Telecommunications
AGH University of Science and Technology
A. Mickiewicza av. 30
30-059 Kraków, Poland

Krzysztof Wajda received his M.Sc. in Telecommunications in 1982 and Ph.D. in 1990, both from AGH University of Science and Technology, Kraków, Poland. In 1982 he joined AGH-UST, where he was responsible for laboratory of switching technology. He spent a year at Kyoto University and half year in CNET (France). Dr. Wajda is currently an Assistant Professor at AGH University of Science and Technology. He was involved in a few international projects: COST 242, Leonardo da Vinci (JOINT and ET-NET), Copernicus ISMAN, ACTS 038 BBL, TEMPUS JEP No. 0971, IST LION, IP NOBEL, NoE e-photon/ONe(+), BONE, SmoothIT. At present he works in Future Internet Engineering project, Polish initiative towards NG Internet. He participated also in a few grants supported by National Science Foundation. He has been a consultant to private telecommunication companies. Main research interests: traffic management for broadband networks, performance evaluation, network reliability, control plane, management systems. K. Wajda is the author (or coauthor) of 6 books (in Polish) and over 100 technical papers. He is a member of IEEE.
E-mail: wajda@kt.agh.edu.pl
Department of Telecommunications
AGH University of Science and Technology
A. Mickiewicza av. 30
30-059 Kraków, Poland