Copyright © IEEE.
Citation for the published paper:

This material is posted here with permission of the IEEE. Such permission of the IEEE does not in any way imply IEEE endorsement of any of BTH's products or services. Internal or personal use of this material is permitted. However, permission to reprint/republish this material for advertising or promotional purposes or for creating new collective works for resale or redistribution must be obtained from the IEEE by sending a blank email message to pubs-permissions@ieee.org.

By choosing to view this document, you agree to all provisions of the copyright laws protecting it.

Evaluation of Delay Performance of Traffic Shapers

Junaid Shaikh, Tahir Nawaz Minhas, Patrik Arlos, Markus Fiedler

Blekinge Institute of Technology

37179 Karlskrona, Sweden

Email: {Junaid.Junaid, Tahir.Nawaz.Minhas, Patrik.Arlos, Markus.Fiedler}@bth.se

Abstract—In network emulation, traffic shapers are used to shape the performance of the network. They are provided with certain inputs in a test environment to vary the network performance accordingly in order to investigate the effects of different network conditions on applications in real yet emulated scenarios. However, it is very important for the shapers to work as supposed in order to successfully realize the desired network conditions. They may make the results of network emulations unrealistic and unreliable if their functioning is not according to the desired specification. In this work, we evaluate the delay shaping of three traffic shapers, NIST Net, Netem and KauNet through the results obtained from a number of experiments.

A comparison of the output of their delay shaping is presented. This comparison can enable us to select the most suitable shaper based on the required shaping. Effects of hardware platforms on the shaping are also filtered out by performing the experiments with shapers installed on Advance Micro Devices (AMD) and Intel platforms separately. Different Protocol Data Unit (PDU) sizes are used in the experiments to test the influence of packet sizes on the shaping. These delay evaluation results are then complemented by the Coefficient of Throughput Variation (CoTV) results.

I. INTRODUCTION

Growth in the real-time applications on the Internet requires better performance from networks [5]. A slight drop in network performance may have a high impact on the tasks being performed due to high sensitivity of the application. Therefore, it is of great importance to test and quantify the network performance and its effects on the respective applications under different conditions. This helps network service providers to avoid situations that may leave undesirable effects on applications and make users abandon networked tasks.

Network emulation is one way to evaluate the network performance in a controlled and repeatable environment [3]. Traffic shapers are usually used in the network emulations to vary the performance parameters such as loss, delay, jitter, bandwidth etc. to realize different network conditions [1]. It helps in testing the behavior of different applications under different network conditions. It is therefore very crucial for traffic shapers to behave according to the given specifications in order to realize the desired network conditions accurately. Inaccuracy in the application of input parameters by the traffic shapers may lead to inefficient and unreliable network emulations. This implies the need for a comprehensive testing for the evaluation of traffic shapers. Traffic shaper evaluations carry an important role in setting a baseline for the reliability of future network emulations that may involve traffic shapers.

On this background, we evaluate the delay implementation of three popular traffic shapers i.e. NIST Net [4], Linux's Network Emulator (Netem) [6] and Dummynet [9] based KauNet traffic shaper [3] in this work [8]. We tested these traffic shapers for different delay values to test if there is any change in their implementation due to the intensity of their input. We also filtered out the effect of different PDU sizes and hardware platforms by using different packet sizes and performing the same tests with shapers installed on Intel and Advance Micro Devices (AMD) platforms.

The remainder of this paper is organized as follows. Section II describes the experiment setup on which the tests were performed for the evaluation of traffic shapers. In Section III, we explain how the analysis was performed on the collected traffic traces. In Section IV, we present and discuss the results related to fixed delay experiments and complement them with the discussion about Coefficient of Throughput Variation (CoTV). In Section IV, we also present a comparison between all three shapers. Finally, Section V concludes the paper.

II. EXPERIMENT SETUP

Figure 1 shows the setup used in the experiments. The Distributed Passive Measurement Infrastructure (DPMI) [7] is used for the measurements. It consists of two distributed Measurement Points (MPs) with the names MP1 and MP2 in this setup. The Sender (S) is responsible for sending the User Datagram Protocol (UDP) packets to the Destination (D). A Full Duplex link of bandwidth 100 Mbps is used from S to D. Wiretaps are used to tap the packet header information of the UDP packets going from S to D. This information is further sent to the MPs. These MPs are equipped with DAG3.5E cards [2]. The shaper here acts as a bridge between S to D. This shaper is provided with one-way delay values as input to shape the one-way delays of packets. NIST Net, Netem and KauNet are used as traffic shapers in this case. These shapers are installed on Intel and then AMD hardware platforms subsequently. There is special script used that controls all the components of this setup including the synchronization of all the nodes. A script also monitors all the tests and reports problems in case of any broken test.

III. ANALYSIS

From the DPMI we collected network traces, and used these to analyze the performance of the shapers. From the traces we extracted all packets, and applied a SHA1 hash on the first 82 bytes of the packet; this covers the network, transport

Fig. 1. Experiment Setup.

and application layer header. Together with the hash we also logged the capture location (physical and logical) and the timestamp. This then allowed us to individually identify where and when the packets passed the wiretaps, hence we could calculate the one-way delay or if that packet was lost. We have used the default settings in terms of the distributions of delays on the packets applied by the shaper. The one-way delay is computed for all the packets successfully transmitted from sender to receiver. Let D_i denote the i^{th} one-way delay value for packet i . From this we then calculate the average delay $\bar{D}_k$ for run k . For each experiment we then calculate the average delay:

$$\bar{D} = \frac{1}{N} \sum_{k=1}^N \bar{D}_k \quad (1)$$

where N is the number of successful runs in this experiment. Standard deviation and Coefficient of Variation (c_V) of one-way delays for whole experiment is then calculated based on the average delays of the runs ($\bar{D}_k$) of an experiment.

$$\sigma = \sqrt{\frac{\sum_{k=1}^N \bar{D}_k^2 - \frac{1}{N} (\sum_{k=1}^N \bar{D}_k)^2}{N-1}} \quad (2)$$

$$c_V = \frac{\sigma}{\bar{D}} \quad (3)$$

Similarly, the minimum and maximum delays are also calculated for all the packets transferred in an experiment. Note that each run consists of 10000 successfully transferred packets between sender and receiver and each experiment consists of 32 runs. Experiments are differentiated from each other based on either the different delay settings or different PDU sizes. The PDU size and the delay setting remains the same throughout the whole experiment.

Now we describe shortly how the CoTV is calculated. CoTV is expressed as:

$$c_{V,j}^P = \frac{s_{R,j}^P}{m_{R,j}^P}. \quad (4)$$

Where $m_{R,j}^P$ is the average bit rate defined as

$$m_{R,j}^P = \frac{1}{n} \sum_{i=j-n+1}^j R_i^P, \quad j \geq n, \quad (5)$$

where n defines the window size. The standard deviation $s_{R,j}^P$ is calculated as:

$$s_{R,j}^P = \sqrt{\frac{1}{n-1} \sum_{i=j-n+1}^j (R_i^P - m_{R,j}^P)^2}, \quad j \geq n. \quad (6)$$

R_i^P defines the bit rate in interval i at specific location, it can be inlet or outlet of network and also involve the layers (Link, network and application), specified by P . The duration of interval i is $\Delta T = 1/F_s$ where F_s is sampling frequency.

For steady state analysis, we let n span the whole session, i.e. $c_V^P = c_{V,n}^P$.

Furthermore, we will also investigate two different ways of relating the input and output coefficient of variation to each other. The first is the *CoTV difference*:

$$\Delta c_{V,j} = c_{V,j}^{\text{out}} - c_{V,j}^{\text{in}}, \quad (7)$$

the second is the *CoTV ratio*:

$$\gamma_{V,j} = \frac{c_{V,j}^{\text{out}}}{c_{V,j}^{\text{in}}}, \quad c_{V,j}^{\text{in}} > 0. \quad (8)$$

IV. RESULTS

This section discusses the results obtained from the tests performed with each of the three traffic shapers. These results are related to the evaluation of fixed delay. Fixed delay experiments here refer to those experiments in which one-way delays of all the packets are supposed to be static without any variation which means shapers are not supposed to vary the packet delays. Fixed delay experiments use the delay settings of 100 ms, 200 ms, 300 ms and 400 ms, respectively. The following subsections discuss the results of fixed delays performed with each of the shapers on AMD and Intel platforms. We are only showing the results with one delay setting, i.e. 100 ms. The reason for not discussing all the four settings is that the trends of the plots are quite similar and not vary with the change of settings. We then present the curves of CoTV obtained from the same experiments at inlet and outlet of the shapers. Such results are discussed in the paper [10]. The idea is to see if there are any indications about the one-way delay accuracy obtained from CoTV results. Finally, we also compare these shapers in terms of their applications of fixed delays by presenting some of their one-way delay and CoTV statistics.

A. NIST Net

Figure 2 shows the distribution of delay (supposed to be fixed to 100 ms) on a logarithmic y -axis. The delay was set to 100 ms, and the AMD platform was used. Different plots in the figure identify different PDU sizes i.e. 64, 256, 512, 1024 and 1470 B. Results show that the one-way delays of around 98 % of the packets are found between 100.0 ms and 100.3 ms. Furthermore, we also observe 99.9 ms of one-way delay with PDU size of 64 B. However we do not see any packets with one-way delay less than the nominal delays in any of the other experiments (other PDU sizes) with NIST

Fig. 2. One-way delay for the NIST Net Shaper on the AMD platform, nominal delay 100 ms.

Fig. 3. One-way delay for the NIST Net Shaper on the Intel platform, nominal delay 100 ms.

Net. On the other hand, there is a small but not negligible share of the total number of packets with one-way delays above 100.3 ms. This tail comprises around 2% of the total number of packets and their one-way delays lie between 100.3 to 108 ms. The one-way delays of packets with size 1470 B are more dispersed as compared to those belonging to other PDU sizes. The trend of distribution plots above 100.3 ms can be described by oscillations. Hence the tail doesn't really show a pronounced decreasing trend with an increasing value of one-way delay.

In figure 3, we show results of the same experiments performed with shapers on Intel platform. In these plots, we do not observe the tail that we see in case of the AMD platform. The one-way delays are distributed only between 100 ms to around 100.5 ms. As it became evident from these plots, we may say that the longer tail observed in the plots of previous figure are not due to the NIST Net shaping application itself but due to the AMD hardware.

Figures 4 and 5 show the *Coefficient of Throughput Variation* (CoTV) of inlet and outlet of NIST Net shaper on AMD

Fig. 4. Fixed Delay CoTV for 64 Byte PDU NIST Shaper AMD Platform

platform with PDU size 64 B and 1470 B, respectively. The CoTV is shown along y -axis, while the underlying sample intervals ΔT are shown along the x -axis. Delay was set to 100 ms but we see that inlet and outlet CoTVs are not identical for all the time scales especially for $\Delta T \geq 0.1$ s. The difference between the outlet and inlet CoTV is more visible as ΔT decreases. Similarly the Figures 6 and 7 have the same setting as discussed above, the only difference is that the Intel platform was used instead of AMD.

By comparing the figures {4, 6} and {5, 7} with each other, we can see that the difference between outlet and inlet CoTVs for a given sample time is increasing with decrease in ΔT . Moreover the difference between inlet and outlet of CoTV is bigger in case of AMD as compare to Intel platform. According to [10], CoTV can be used to quantify the packet delay variation. This explains the bigger difference on inlet and outlet CoTV in case of AMD as there is delay variation from 100 ms to 108 ms, see Figure 2. Hence it is obvious from these results that difference between CoTV's inlet and outlet values can be a good reflection of the one-way delay distributions of packets. We can say that both results are complementing each other.

B. Netem

In this subsection, we present Netem's one-way delay shaping analysis. In figure 8, we observe that less than 1 % of the packets are delayed more than 100.3 ms for the experiments performed with Netem on AMD platform. The rest of them consists of one-way delays between between 100 ms and 100.2 ms. We observe a decreasing trend in the plots above 100.2 ms. Although this part holds only 1 % of total packets, it still forms a tail that stretches to around 106 ms. In the Netem case, we do not observe any one-way delays less than the nominal delay.

Figure 9 illustrates the shaping of one-way delays done by Netem on Intel platform. In this case, once again the one-way delays are not spread on a larger scale. They are rather more accurate and do not extend 100.5 ms. This further strengthens

Fig. 5. Fixed Delay CoTV for 1470 Byte PDU NIST Shaper AMD Platform

Fig. 6. Fixed Delay CoTV for 64 Byte PDU NIST Shaper Intel Platform

Fig. 7. Fixed Delay CoTV for 1470 Byte PDU NIST Shaper Intel Platform

Fig. 8. One-way delay for the Netem Shaper on the AMD platform, nominal delay 100 ms.

Fig. 9. One-way fixed delay for the Netem Shaper on the Intel Platform, nominal delay 100 ms.

the idea that difference in the resulting one-way delay above the nominal delay is mainly due to the platform used but not due to the shaper itself.

The CoTV was also calculated in case of the Netem Shaper on Intel and AMD platform, respectively. Figures 10 and 11 show CoTV versus sample interval ΔT for AMD platform, while Figures 12 and 13 show the corresponding results for the Intel platform. In case of the AMD platform, the inlet CoTV is slightly smaller as compared to inlet CoTV observed in case of Intel than Intel for $\Delta T \geq 0.1$ s. For all sample intervals, the outlet CoTV of AMD platform is bigger as compared to the outlet CoTV of Intel, except for 64 B and $\Delta T = 0.001$ s. The greater values of the outlet CoTV of the AMD-based experiment are due to delay variation from 100 ms to 105.5 ms shown in Figure 8

C. KauNet

Plots in figure 14 show distribution of one-way delays as shaped by KauNet with set delay of 100 ms and different PDU sizes with AMD platform. One-way delays are spread from

Fig. 10. CoTV for 64Byte PDUs via the Netem Shaper on the AMD Platform, nominal delay 100 ms.

Fig. 13. CoTV for 1470Byte PDUs via the Netem Shaper on the Intel Platform, nominal delay 100 ms.

Fig. 11. CoTV for 1470Byte PDUs via the Netem Shaper on the AMD Platform, nominal delay 100 ms.

Fig. 14. Fixed Delay KauNet Shaper AMD Platform

Fig. 12. CoTV for 64Byte PDUs via the Netem Shaper on the Intel Platform, nominal delay 100 ms.

93 ms to around 108 ms. This result illustrates that KauNet shapes one-way delays of packets both above and below the set delay. Trends of distributions above and below the set delay are quite similar. However, 99 % of the packet delays are still uniformly distributed between 99.1 ms and 100.1 ms despite of the tails on both sides of set delay.

Figure 15 shows the results obtained by KauNet's shaping on Intel platform. In these results, we notice that one-way delays as applied by KauNet are distributed between 97 ms to around 101.5 ms. It confirms that one-way delays lie both above and below the set delay with KauNet's shaping. Once again, we do not see any tails on both sides. However, we observe two regions of one-way delays; one ranging from 98.2 ms to 99.2 ms while other from 99.3 ms to 100.2 ms with considerably high PDFs of around 0.03 and 0.07 respectively. We don't observe such two regions in case of AMD platform and we see only one region (99.1-100.1 ms) with high PDF of around 0.1. Hence, it becomes quite evident that hardware plays an important role in accuracy of shaping. Shaping software alone can not guarantee the accuracy of desired

Fig. 15. Fixed Delay KauNet Shaper Intel Platform

output.

In case of KauNet Shaper, the CoTV results for AMD and Intel platform are quite interesting. Figures 16 and 17 are illustrating the CoTV for AMD platform while Figures 18 and 19 show the CoTV versus sample Interval ΔT for Intel platform. On Intel platform, the inlet CoTV is slightly smaller as compared to the Inlet CoTV we get on AMD. Outlet CoTV on Intel comes out to be much higher as compared to AMD platform. For example in case of $\Delta T = 0.001$ s the outlet CoTV on Intel platform is 0.35 while on AMD platform, the CoTV is 0.14, see Figures 16 and 18. It is an interesting behavior apparently looks different from NIST Net and Netem shapers. If we turn our attention towards the distribution of Delay in case of KauNet, see Figures 14 and 15. We will find the answer of this behavior. In case of AMD platform delay distribution is [93 ms 108 ms] while in case of Intel platform the distribution is [97 ms 101 ms]. As we discussed earlier, in case of Intel, major portion of delay is varying in two intervals [98 ms 99 ms] and [99 ms 100 ms] which is the cause of higher outlet CoTV that we get on Intel platform.

D. Comparison of shapers

In previous sections, we discussed about delay shaping performance of each of the shapers individually with both AMD and Intel platforms. In this section, we compare these shapers through some of the statistics of their one-way delays and CoTVs. Table I shows the inlet CoTV ($\hat{c}_V^{\text{in}}$) and outlet CoTV ($\hat{c}_V^{\text{out}}$) with the corresponding confidence intervals in the third and fourth column for both shapers. The fifth column Δc_V shows the difference of outlet and inlet CoTV calculated by using Equation 7, and the last column γ_V represents the ratio of outlet and inlet CoTV, calculated by using Equation 8. By comparing the results of both shapers Netem and NIST Net, we can see that the Netem is more accurate as compared to NIST Net. For example, Δc_V is 0.0049 for PDU size of 1470 B for Netem with sample interval 10 ms at Intel platform and Δc_V is 0.0065 for NIST Net with same condition. Moreover, if we cross-compare the platforms with each other for the same

Fig. 16. CoTV for 64Byte PDUs via the KauNet Shaper on the AMD Platform, nominal delay 100 ms.

Fig. 17. CoTV for 1470 Byte PDUs via the KauNet Shaper on the AMD Platform, nominal delay 100 ms.

Fig. 18. CoTV for 64Byte PDUs via the KauNet Shaper on the Intel Platform, nominal delay 100 ms.

Fig. 19. CoTV for 1470Byte PDUs via the KauNet Shaper on the Intel Platform, nominal delay 100 ms.

shaper, we can see that Δc_V is always greater for AMD except for the 64B-PDU with $\Delta T = 1$ s in the Netem-AMD case. As discussed above, AMD has more packet delay variation as compared to the Intel platform. Obviously, this delay variation can nicely be discovered from CoTV measurements, see also [10]. Furthermore KauNet is showing the normal behavior like Netem and NIST net in case of AMD platform while for Intel platform values of outlet CoTV are much higher as compare to other two cases, details are discussed in Section IV-C.

Tables II and III list mean, minimum, maximum, standard deviation and Coefficient of Variation (CoV) of one-way delays in the fourth to the eighth column. These statistics are based on all the 32 runs of each experiment. Tables II and III correspond to the statistics of the experiments with PDU sizes 64 and 1470 B, respectively. We choose 64 and 1470 B as they represent the smallest and biggest possible PDU sizes used in our experiments. We observe some difference in the maximum one-way delays of all the shapers due to the difference in platform. On Intel, the maximum one-way delay is closer to the mean; the deviation is less than 2 ms. On AMD, the maximum value exceeds the mean by around 4 to 8 ms, a reason for the tail that we observed in plots in previous sections. Mean and minimum delays are almost the same and we do not observe much difference in case of NIST Net and Netem. However, the minimum delay values with KauNet on AMD platform are less than the values we get on Intel platform. KauNet also applies delays less than the set delays that we do not observe in case of Netem and NIST Net. We do not observe much difference in statistics among the shapers on Intel platform. However, we observe relatively higher maximum delays in case of NIST Net and KauNet as compared to Netem. In general, the variation in the one-way delays of Netem is smaller than that of NIST Net. Hence the one-way delay shaping with Netem is more accurate as the applied delays are closer to set delay.

NetEm, PDU size 64 Bytes					
Platform	ΔT [ms]	$\hat{c}_V^{\text{in}}$	$\hat{c}_V^{\text{out}}$	Δc_V	γ_V
Intel	10	0.026 ± 0.0001	0.030 ± 0.0008	0.0041	1.15
AMD	10	0.024 ± 0.0011	0.029 ± 0.0010	0.0054	1.23
Intel	2	0.035 ± 0.0011	0.070 ± 0.0008	0.0341	1.96
AMD	2	0.031 ± 0.0013	0.066 ± 0.0017	0.0348	2.11
Intel	1	0.037 ± 0.0010	0.109 ± 0.0012	0.0714	2.91
AMD	1	0.033 ± 0.0013	0.094 ± 0.0028	0.0607	2.83

NetEm, PDU size 1470 Bytes					
Platform	ΔT [ms]	$\hat{c}_V^{\text{in}}$	$\hat{c}_V^{\text{out}}$	Δc_V	γ_V
Intel	10	0.015 ± 0.0016	0.019 ± 0.0012	0.0049	1.34
AMD	10	0.011 ± 0.0013	0.017 ± 0.0011	0.0052	1.46
Intel	2	0.021 ± 0.0018	0.039 ± 0.0010	0.0175	1.82
AMD	2	0.020 ± 0.0017	0.042 ± 0.0027	0.0217	2.07
Intel	1	0.022 ± 0.0017	0.050 ± 0.0009	0.0280	2.27
AMD	1	0.021 ± 0.0016	0.054 ± 0.0026	0.0324	2.54

NIST net, PDU size 64 Bytes					
Platform	ΔT [ms]	$\hat{c}_V^{\text{in}}$	$\hat{c}_V^{\text{out}}$	Δc_V	γ_V
Intel	10	0.025 ± 0.0005	0.030 ± 0.0005	0.0053	1.22
AMD	10	0.020 ± 0.0005	0.031 ± 0.0009	0.0107	1.53
Intel	2	0.033 ± 0.0008	0.111 ± 0.0005	0.0776	3.34
AMD	2	0.026 ± 0.0009	0.111 ± 0.0012	0.0850	4.28
Intel	1	0.036 ± 0.0008	0.166 ± 0.0005	0.1300	4.63
AMD	1	0.029 ± 0.0008	0.165 ± 0.0014	0.1360	5.75

NIST net, PDU size 1470 Bytes					
Platform	ΔT [ms]	$\hat{c}_V^{\text{in}}$	$\hat{c}_V^{\text{out}}$	Δc_V	γ_V
Intel	10	0.009 ± 0.0011	0.016 ± 0.0007	0.0065	1.71
AMD	10	0.008 ± 0.0010	0.021 ± 0.0015	0.0127	2.59
Intel	2	0.016 ± 0.0014	0.046 ± 0.0006	0.0299	2.93
AMD	2	0.014 ± 0.0015	0.050 ± 0.0017	0.0357	3.58
Intel	1	0.016 ± 0.0013	0.062 ± 0.0005	0.0460	3.80
AMD	1	0.017 ± 0.0014	0.067 ± 0.0021	0.0502	4.01

KauNet, PDU size 64 Bytes					
Platform	ΔT [ms]	$\hat{c}_V^{\text{in}}$	$\hat{c}_V^{\text{out}}$	Δc_V	γ_V
Intel	10	0.026 ± 0.0009	0.097 ± 0.0064	0.0715	3.76
AMD	10	0.027 ± 0.0010	0.035 ± 0.0013	0.0077	1.29
Intel	2	0.035 ± 0.0011	0.247 ± 0.0058	0.2120	7.09
AMD	2	0.036 ± 0.0013	0.093 ± 0.0029	0.0570	2.59
Intel	1	0.038 ± 0.0011	0.348 ± 0.0039	0.31000	9.27
AMD	1	0.039 ± 0.0012	0.139 ± 0.0038	0.1010	3.62

KauNet, PDU size 1470 Bytes					
Platform	ΔT [ms]	$\hat{c}_V^{\text{in}}$	$\hat{c}_V^{\text{out}}$	Δc_V	γ_V
Intel	10	0.013 ± 0.0014	0.085 ± 0.0010	0.0729	6.85
AMD	10	0.014 ± 0.0015	0.034 ± 0.0017	0.0199	2.44
Intel	2	0.020 ± 0.0016	0.203 ± 0.0015	0.1830	10.4
AMD	2	0.021 ± 0.0017	0.083 ± 0.0033	0.0623	4.01
Intel	1	0.020 ± 0.0014	0.254 ± 0.0036	0.2340	12.9
AMD	1	0.022 ± 0.0015	0.106 ± 0.0031	0.0842	4.91

TABLE I
ESTIMATED CoTV AT INLET AND OUTLET OF NIST NET, NETEM AND KAUNET SHAPERS.

V. CONCLUSION

In this paper, we presented the fixed one-way delay evaluation results of three traffic shapers, NIST Net, Netem and KauNet. We discussed how each of these shapers implement one-way delays on the packets when instructed with certain nominal values of one-way delays. We also filtered out the effects of hardware platforms by doing experiments with shapers installed on Intel and AMD platforms. According

TABLE II
ONE-WAY DELAY STATISTICS FOR EXPERIMENTS WITH PDU SIZE 64 B.

Shaper	Platform	Mean [ms]	Min [ms]	Max [ms]	σ [ms]	c_V
NIST Net	Intel	100.2	100.0	100.6	5.3e-04	5.3e-06
	AMD	100.2	99.9	106.7	2.0e-02	2.0e-04
Netem	Intel	100.2	100.0	100.6	1.1e-03	1.1e-05
	AMD	100.2	100.0	105.3	8.1e-04	8.1e-06
KauNet	Intel	99.6	97.3	101.4	1.3e-06	1.3e-05
	AMD	99.6	93.5	108.2	6.2e-06	6.2e-05

TABLE III
ONE-WAY DELAY STATISTICS FOR EXPERIMENTS WITH PDU SIZE 1470 B.

Shaper	Platform	Mean [ms]	Min [ms]	Max [ms]	σ [ms]	c_V
NIST Net	Intel	100.3	100.1	100.5	1.17e-03	1.17e-05
	AMD	100.4	100.1	108.0	1.92e-02	1.91e-04
Netem	Intel	100.3	100.1	101.1	9.49e-04	9.47e-06
	AMD	100.3	100.2	105.5	1.0e-03	1.0e-05
KauNet	AMD	100.2	100.0	105.3	8.1e-04	8.1e-06
	Intel	99.7	97.5	101.6	1.8e-06	1.8e-05
	AMD	99.7	93.5	107.3	6.3e-06	6.3e-05

to the distributions of one-way delays extracted from these tests, we conclude that the major effect of variation from the applied delays was due to the hardware platform, while the shaper software was playing a minor role. In the Intel case, we observed that the applied one-way delays are much closer to the desired delays. One-way delays of the packets are more dispersed above the desired delays in case of AMD, which becomes evident through a pronounced tail. We observed that Netem applies delays more accurately as compared to other two shapers. However the difference between Netem and NIST Net is not too high. KauNet applies delays both below and above the set delay values therefore choice of nominal delay values must be made keeping this in mind.

We also complemented the one-way delay measurements with measurements of the Coefficient of Throughput Variation (CoTV), where the deviation between inlet and outlet CoTV illustrates the inaccuracy of one-way delays applied by the shapers. We observed that this difference between inlet and outlet of CoTV gets bigger in case of AMD as compared to Intel with NIST Net and Netem. With KauNet, the outlet CoTV on Intel is much higher as compared to the outlet CoTV on AMD platform due to the two major regions of delay values. This again underlines that the hardware platform has the major role in the inaccuracy of shaper implementation and also the feasibility of the alternative indicator CoTV.

Therefore, we conclude that hardware for the traffic shaper should be chosen carefully for better network emulations with less artifacts introduced by the systems.

REFERENCES

- [1] E. Conchon, J. Garcia, T.Prennou, M. Diaz, *Improved IP-level Emulation for Mobile and Wireless Systems*, In Proceedings of the IEEE Wireless Communications and Networking Conference (IEEE WCNC), 2007.
- [2] Endace. *Endace Measurement Systems*, [Online]. Available: <http://www.endace.com/>.
- [3] J. Garcia, P. Hurtig, A. Brunström, *KauNet: Design and Usage*, Technical report, Karlstad University, August 2009.

- [4] M. Carson and D. Santay, *Nistnet: a Linux-based network emulation tool*, ACM SIGCOMM Computer Communication Review, vol. 33, no. 3, pp. 111–126, July 2003.
- [5] M. El-Gendy, A. Bose, K.G. Shin, *Evolution of the Internet QoS and support for soft real-time applications*, Proceedings of the IEEE, 2003.
- [6] *The Netem Network Emulator*, <http://www.linuxfoundation.org> (last seen: March 5, 2010).
- [7] P. Arlos, *On the Quality of Computer Network Measurements*, PhD thesis, Blekinge Institute of Technology, Karlskrona, Sweden, 2005.
- [8] P. Vicat-Blanc Primet, R. Takano, Y. Kodama, T. Kudoh, O. Gluck, C. Otal, *Large Scale Gigabit Emulated Testbed for Grid Transport Evaluation*, In Proceedings of The Fourth International Workshop on Protocols for Fast Long-Distance Networks (PFLDnet'2006), 2006.
- [9] R. Luigi, *Dummysnet: a simple approach to the evaluation of network protocols*, SIGCOMM Computer Communication Review, 1997.
- [10] Tahir N. Minhas, M. Fiedler and P. Arlos, *Quantification of Packet Delay Variation through the Coefficient of Throughput Variation*. In Proceeding of TRAFFIC Analysis and Classification (TRAC) Workshop, Caen, France, 2010.