Measurement of One-Way Internet Packet Delay

Doru Constantinescu, Patrik Carlsson, Adrian Popescu and Arne A. Nilsson

Dept. of Telecommunication Systems
School of Engineering
Blekinge Institute of Technology
371 79 Karlskrona, Sweden
{doru.constantinescu, patrik.carlsson, adrian.popescu, arne.nilsson}@bth.se

Abstract

The paper reports on a dedicated measurement system for delay measurements in IP routers, which follows specifications of the IETF RFC 2679. The system uses both passive and active measurements. Dedicated application-layer software is used to generate traffic. Pareto traffic models are used to generate self-similar traffic in the link. Both packet inter-arrival times and packet sizes are matching real traffic models. A passive measurement system is used for data collection that is based on using several so-called Measurement Points, each of them equipped with DAG monitoring cards. Hashing is used for the identification and matching of packets. The combination of passive measurements and active probing, together with using the DAG monitoring system, gives us an unique possibility to perform precise traffic measurements as well as the flexibility needed to compensate for the lack of analytic solutions.

1 Introduction

Today network capacities are being deliberately overengineered in the Internet so that the packet loss rate is very low and the end-to-end (e2e) delay is minimized. However, given the heterogeneity of the network and the fact that the overengineering solution is not adopted everywhere, especially not by backbone teleoperators in developing countries, the question arises as to how the delay performance impacts on e2e performance. There are several important parameters that may impact on delay performance in the link, e.g., traffic self-similarity, routing flaps and link utilization.

Several papers have been published so far that report on e2e delay performance, and both Round-Trip Time (RTT) and One-Way Transit Time (OWTT) are considered

[2, 6, 10, 11]. Generally, RTT measurements are simpler but the analysis is more complex due to problems related to clock synchronization, packet timestamping, protocol complexity, asymmetries in direct and return paths as well as path variations. Other problems related to difficulties in measuring queueing delays in operational routers and switches further complicates the picture.

Our paper is a contribution towards a better understanding of traffic measurements associated with e2e delays occurring in best-effort networks. We describe problems and solutions associated with OWTT delay measurements, and give examples of such measurements. A dedicated measurement system is reported for delay measurements in IP routers, which follows specifications of the IETF RFC 2679. The system uses both passive measurements and active probing.

The rest of the paper is organized as follows. In Section 2 we describe the delay components associated with One-Way Transit Time measurements. Section 3 describes the measurement system and the technology used for traffic collection as well as traffic generation and packet identification. In Section 4 we describe several experiments done and the associated details. Section 5 is dedicated to reporting several results obtained on one-way delay performance. Finally, Section 6 concludes this paper.

2 Delay Components

One-Way Transit Time (OWTT) is measured by timestamping a specific packet at the sender, sending the packet into the network, and comparing the timestamp with the timestamp generated at the receiver. Packet timestamping can be done either by software (for the case of delay measurements at the application level) or by hardware (for the case of delay measurements in the network), and in this case special hardware is used. Clock synchronization between the sender and the receiver nodes is important for the precision of one-way delay measurements [1]. On top of that, delay measurements at the application level are sensitive to possible uncertainties related to the difference between the "wire time" and the "host time" as well.

OWTT has several components:

$$OWTT = D_{prop} + \sum_{i=0}^{N} D_{n,i} \tag{1}$$

where the delay per node i, $D_{n,i}$ is given by:

$$D_{n,i} = D_{tr,i} + D_{proc,i} + D_{q,i} \tag{2}$$

The different components in (1) and (2) can be summarized as follows:

- D_{prop} is the total propagation delay along the physical links that make up the Internet path between the sender and the receiver. This time is solely determined by the properties of the communication channel as well as the distance. It is independent of traffic conditions on the links.

- N is the number of nodes between the sender and the receiver.

- $D_{tr,i}$ is the transmission time for node i. This is the time it takes for the node i to copy the packet into the first buffer as well as to serialize the packet over the communication link. It depends on the packet length and it is inversely proportional to the link speed.

- $D_{proc,i}$ is the processing delay at the node i. This is the time needed to process an incoming packet (e.g., to decode the packet header, to check for bit errors, to lookup routes in a routing table) as well as the time needed to prepare the packet for further transmission, on another link. This delay depends on parameters like network protocol, computational power at node, and efficiency of network interface cards.

- $D_{q,i}$ is the queueing delay in the node i. This delay refers to the waiting times in buffers, and depends upon traffic characteristics, link conditions (e.g., link utilization ρ, interference with other IP packets) as well as implementation details of the node.

Several statistics, e.g., mean, median, maximum, minimum, variance, peakedness, and probability distribution function, are usually used in the description of delay for non-corrupted packets. Typical values obtained for OWTT range from tens of μs to hundreds of ms [3].

For a general discussion, the OWTT delay can be partitioned into two components, a deterministic delay D_d and a stochastic delay D_s:

$$OWTT = D_d + D_s \tag{3}$$

D_{prop}, D_{tr} and (partly) D_{proc} are contributing to the deterministic delay D_d, whereas the stochastic delay D_s is created by D_q and, at some extent, D_{proc}. The stochastic part of the router processing delay can be observed especially in the case of low and very low link utilization, i.e., when the queueing delays are minor.

3 Delay Measurements

A dedicated measurement system has been developed to do delay measurements in IP routers, which follows specifications of the IETF RFC 2679 [1]. The system uses both passive measurements and active probing. Dedicated application-layer software is used to generate UDP traffic with TCP-like characteristics. The well-known interactions between TCP sources and network are thus avoided. UDP is not aware of any network congestion, and this gives us the choice of doing experiments where the focus is on the network only. The software consists of a client and a server running on two different hosts, which are separated by a number of routers. Pareto traffic models are used to generate self-similar traffic in the link. Both packet inter-arrival times and packet sizes are matching real traffic models. A passive measurement system is used for data collection that is based on using several so-called Measurement Points (MPs) [4], each of them equipped with DAG monitoring cards [7].

To estimate the delay that a packet experiences one needs to first identify the packets as they pass the MPs on their way through the routers. Hashing is used for the

identification and matching of packets. The hashing function is implemented with the SHA-1 Secure Hash Algorithm [12]. All captured packets are masked before hashing. The identification and matching software provides timestamp readings for every two adjacent MPs.

3.1 Measurement Setup

Figure 1 shows the measurement setup used for our experiments. The key component in the system is the MP, the device that does the actual packet capturing. The capabilities of an MP are decided by the capture hardware that is installed in the MP, and in our experiments we use the DAG 3.5E network monitoring card [7]. The MPs are capable of collecting and timestamping frames with an accuracy of less than 100 ns. Data analysis is performed off-line.

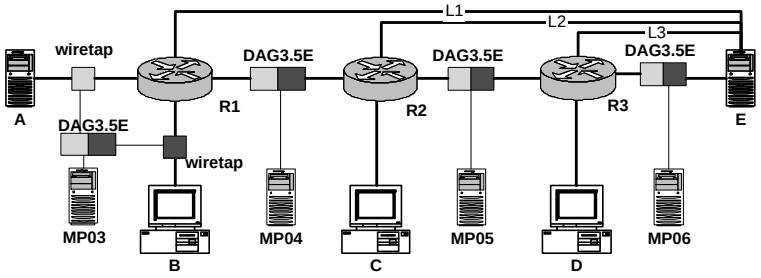

Figure 1: Measurement setup

The system collects and timestamps traces consisting of the first 96 bytes of every frame captured on Ethernet links (10Base-T). The Ethernet, IP and transport headers are thus collected as well as part of the payload. Packets smaller than 96 bytes are zero-padded. The clocks of the DAG cards, which generate the timestamps, are locally synchronised in the sense that all clocks are synchronised to one MP's DAG clock which, in turn, is synchronised to a NTP server. A high accuracy of the timestamps is thus obtained (less than 100 ns), as compared with the computer timestamps of about 10 μs. This offers the advantage of fairly accurate delay measurements that are suitable for our experiments.

The networks that we are measuring are 10 Mbps full duplex Ethernets. On a 10 Mbps Ethernet the maximum frame rate is 14881 frames/s, and this equals a frame inter-arrival time of 67.2 μs. This time is significantly larger than the timestamp accuracy of the MP.

The routers R1, R2 and R3 are all of the same type (Cisco 3620). The source host A, the sink host E and the hosts that generate cross traffic B, C, and D are all identical with regards to hardware and software configuration.

3.2 Traffic Generation

The objective is to generate data at the application layer while using UDP as the transport protocol. Data lengths are selected from a truncated Pareto distribution with

parameters that match traffic models observed in real networks. Truncation is done in such a way that the maximum allowed IP payload is not violated, i.e., 65515 bytes. The inter-packet times are selected from an exponential distribution. Furthermore, since the OWTT estimation software uses parts of the IP payload field for identification, a specific data structure is used to create unique payloads for the generated UDP packets. Each payload contains a packet identification number, a timestamp and a block of ASCII data. By using this specific data structure and the way the *send* function is implemented, one can easily transmit payloads with different sizes with no need to generate new data.

Two generators are used for the traffic generation. The first generator is used to emulate a single traffic source while the second aims at emulating an arbitrary number of traffic sources. Further, the generating software has three operating modes: *packet*, *time* and *infinite* [5]. In the *packet* mode, the software generates a fixed number of packets and terminates. In the *time* mode, the traffic is generated over a finite amount of time. Finally, in the *infinite* mode the traffic generation terminates when CTRL-C is pressed. In addition to the operating modes, the software also considers parameters that determine traffic characteristics in link, e.g., the link utilization and the Hurst parameter.

The traffic generated between the source A and the sink E follows a Pareto distribution for the packet length with the shape parameter α (which determines the mean and the variance) and the location parameter β (which determines the minimum value). An exponential distribution with parameter λ is used for inter-packet times and the (measured) link utilization ρ depends on λ as well as α and β. This model matches traffic models observed for the World Wide Web, which is one of the most important contributors to the traffic in Internet [8]. High traffic intensities are considered in our experiments as well, with the consequence of higher loads on routers, and thus we obtain better delay models.

The cross traffic generated by the traffic generators B, C, D has a form that approaches fractional Brownian motion (fBm) traffic with self-similar characteristics, which is typical for Ethernet [9]. This traffic is generated in a similar way, i.e., Pareto distribution for packet sizes and exponential distribution for inter-packet times. The difference however is that a large number of processes are used in this case to generate a large number of Pareto traffic flows in every traffic generator. This gives us the choice of doing experiments where we can control diverse parameters of the traffic mixture in the link, especially the Hurst parameter H and the link utilization ρ.

3.3 Packet Identification

A meaningful analysis of delay measurements requires identification of packets present on multiple traffic traces. Hashing is used to uniquely represent each captured packet from the traffic traces. Before hashing, each packet is bit-masked (bitwise AND) with a mask previously defined [5].

As an IP packet travels through the routers the payload remains unchanged and some fields in the IP header (i.e., TTL, Header Checksum) are changed. Therefore, by masking these fields a correct and unique identification of each packet can be obtained. Furthermore, as only uncorrupted IP packets carrying UDP payload are processed, any

other traffic that may appear in a collected traffic trace (e.g., ARP, ICMP, HELLO) is discarded.

The SHA-1 hashing algorithm is used to correctly identify target packets in the link traffic. SHA-1 is a cryptographic one-way hash function with an output size of 20 bytes [12]. It accepts as input a message of length L (of maximum 2^{64} bits) and outputs a 160-bit message digest. The SHA-1 algorithm provides a very low collision probability as it takes 2^{64} attempts to find two messages with the same hash value, provided that the hashed items are independent and equally likely. Hashing has also the advantage that it increases the computational speed as only 20 bytes (hash value) are checked, compared to the 96 bytes representing the total length of each captured packet. The use of both masking and hashing ensures so that every packet is uniquely identified by the hash value.

The packets are sequentially read from the trace file, copied into a packet vector and then hashed. The SHA-1 hash value is stored into a hash vector together with the unique identification of each packet, i.e., the index in the packet vector. The timestamp field is masked in the frame header because it changes at every MP. The masking ensures that only 54 bytes are hashed, starting from the IP header. The hash covers selected fields in the IP header including the source and destination IP addresses, IP header Identification field and other fields (including possible IP options). The only fields in the IP header that are not hashed are the Time-To-Live (TTL) and the Header Checksum fields, as they are changed at every router. 37 bytes of the IP payload (including IP options and eventual padding) are included in the hash as well. The hash vector is sorted (based upon the hash values) before packet matching, in order to improve the computational performance [5].

3.4 Sources of Errors

A number of parameters are used in the OWTT analysis, which are the timestamping information T_i for a given packet, the place for timestamping (i.e., the DAG interface on which the packet was captured) and the size of the packet payload. For a specific packet n, the OWTT is estimated by taking the time difference between two consecutive timestamps readings:

$$OWTT_{i,j}(n) = T_i(n) - T_j(n) \tag{4}$$

$OWTT_{i,j}(n)$ is the estimated one-way transit time between points i and j for the n^{th} packet, $T_i(n)$ represents the timestamp reading taken at the measurement point i and $T_j(n)$ is the timestamp reading taken at the measurement point j. Accurate samples of the $T_i(n)$ and $T_j(n)$ are requested for a correct estimation of OWTT.

There are several sources of errors that are possible in OWTT estimation. For instance, one of the most common sources of error in obtaining accurate timestamps are the presence of duplicate packets in the traffic traces [10, 11]. For the duplicate packets, all 54 bytes hashed are identical giving thus the same SHA-1 digest. Since the measurement experiments are done in a strictly controlled environment the probability of duplicate occurrence is extremely low. In fact, no duplicate IP packets have been observed in our experiments [5].

Another source of errors is given by possible presence of unmatched packets in the collected traffic traces. Unmatched packets may occur because of several reasons, e.g., local cross traffic at Ethernet hubs, packets sent to, or originating from one of the IP addresses of routers, or fragmentation required in a router for an outgoing link. We have observed a very low probability of occurrence of unmatched packets in our experiments, which is between 0.01 % and 5 % for more than one million packets processed. The conclusion therefore is that it is likely that unmatched packets are created by other interferring traffic, e.g., ARP and inter-router traffic as well as possible discarded datagrams due to congestion avoidance in routers during heavy-load traffic conditions.

4 Experiments

We have done a number of experiments to measure OWTT in IP routers under different load and traffic conditions [5].

A first set of experiments has been done to measure the processing delay of a router for IP packets containing ICMP and UDP payloads. It has been observed for instance that the processing delay of the Cisco 3620 router can be approximated by a Gaussian density and the mean processing delay is about 100 μsec [5].

Another set of experiments is regarding the router delay for a single data flow and more data flows as well as to measure the e2e delay for a chain of routers [5]. Different conditions for the link utilization have been used ($\rho = 0.1$ to 0.9) as well as for the Hurst parameter of the traffic in link ($H = 0.5$ to 0.9). Table 1 shows the summary of experiments and the associated traffic parameters.

In experiment 1, only the source computer A generates traffic and R1 is receiving traffic from the computer A only. One process is used in this case to generate Pareto traffic with α parameter shown in table and a traffic intensity λ that corresponds to the (measured) ρ values shown in table. Further, the parameter $\beta = 40$. The traffic generated in link has no fBm-like characteristics. Only MP03 and MP06 are used to capture the traffic. The routers R2 and R3 are not used (direct link between R1 and E) and the MP04 and MP05 are not connected either. The traces generated correspond to different values for burstiness and traffic intensities as shown in table 1. Each trace is quite big and it has about one million packets. The reason for that is because of the well-known problems related to the slow convergence to steady-state in the case of heavy-tailed workloads [9]. Nine traces have been generated in this case, which are shown as **1-1**, **1-2**, to **1-9**.

In experiment 2, the router R1 is receiving traffic from both computers A and B. The routers R2 and R3 are still not used and the MP04 and MP05 are not connected either. In this case, computer A is generating non-fBm-like traffic with the same characteristics as in experiment 1 while computer B is generating cross traffic with fBm-like characteristics with $H \approx 0.9$ and $\rho \approx 0.1$ in all experiments. One hundred processes are used for traffic generation in computer B. Every process generates UDP traffic with Pareto distribution for the packet lengths with $\beta = 40$ and α of different values, i.e., $\alpha = 2, 1.6, 1.2$. The traffic generated by the 100 processes are mixed at the UDP level, creating so the fBm-like traffic observed in the link. As a result, nine

Table 1: Summary of experiments and traffic generation parameters

Exp[1]	A α, ρ (1)	Exp[2]	A α, ρ (1)	B α, ρ (100)	Exp[3]	A α, ρ (1)	B α, ρ (50+50)	C α, ρ (50+50)	D α, ρ (50+50)
1-1	2, 0.2	**2-1**	2, 0.2	1.2, 0.1	**3-1**	2, 0.2	1.2, 0.1	1.2, 0.1	1.2, 0.1
1-2	2, 0.4	**2-2**	2, 0.4	1.2, 0.1	**3-2**	2, 0.4	1.2, 0.1	1.2, 0.1	1.2, 0.1
1-3	2, 0.6	**2-3**	2, 0.6	1.2, 0.1	**3-3**	2, 0.6	1.2, 0.1	1.2, 0.1	1.2, 0.1
1-4	1.6, 0.2	**2-4**	1.6, 0.2	1.2, 0.1	**3-4**	1.6, 0.2	1.2, 0.1	1.2, 0.1	1.2, 0.1
1-5	1.6, 0.4	**2-5**	1.6, 0.4	1.2, 0.1	**3-5**	1.6, 0.4	1.2, 0.1	1.2, 0.1	1.2, 0.1
1-6	1.6, 0.6	**2-6**	1.6, 0.6	1.2, 0.1	**3-6**	1.6, 0.6	1.2, 0.1	1.2, 0.1	1.2, 0.1
1-7	1.2, 0.2	**2-7**	1.2, 0.2	1.2, 0.1	**3-7**	1.2, 0.2	1.2, 0.1	1.2, 0.1	1.2, 0.1
1-8	1.2, 0.4	**2-8**	1.2, 0.4	1.2, 0.1	**3-8**	1.2, 0.4	1.2, 0.1	1.2, 0.1	1.2, 0.1
1-9	1.2, 0.6	**2-9**	1.2, 0.6	1.2, 0.1	**3-9**	1.2, 0.6	1.2, 0.1	1.2, 0.1	1.2, 0.1

traces have been generated, with different ρ for the traffic observed at the sink E. Every trace is about one million packets long. The Hurst parameter of the bit rate has been measured to be $H = 0.9$ for most of traces. There are however several traces showing larger values for H (e.g., $H = 1.09$) for large ρ and low α. These are traces corresponding to traffic with infinite mean. We are aware of this problem, we wanted however to test the router under very severe conditions. We therefore did not exclude these traces from our experiments. Nine traces have been generated in experiment 2, which are shown as **2-1**, **2-2**, to **2-9**.

For experiment 3, the computer A is still generating traffic with the same characteristics as in experiment 1. The difference is that now all three computers B, C and D are generating fBm-like traffic flows, with $H \approx 0.9$ and $\rho \approx 0.1$. These computers are generating both merging traffic and crossing traffic. One hundred processes are used for traffic generation in every computer. The generated traffic flows are broken up in the routers R1, R2 and R3 in a way that 50 % of every flow is merging the traffic coming from computer A and the rest of 50 % (for every flow) is crossing the routers only. Nine traces have been generated in experiment 3 as well, which are shown as **3-1**, **3-2**, to **3-9**.

5 Results

The complete set of results obtained in all experiments is quite large and it is reported in [5]. In the following we report an example of results obtained in experiment 3 only.

Table 2 reports the main statistics of the OWTT measured between the generator A and the sink E. The main observation in this case is regarding the large disparity for all statistics collected, except for the minimum. We also observe a large number of samples with large delays, and we believe that most of them are due to queueing delays in routers. We do not observe any sample with extremely large delay, the conclusion of which is that all three routers seem to work properly. The associated histograms have been observed to have a Gamma-like shape with a heavy tail. The tail has been observed to look similar for most of traces and it is dependent on α and ρ.

Figure 2 shows examples of throughput (Mbps) measured in experiment **3-8** at the output of computer A (upper left) and the associated histogram (upper right).

Table 2: Summary of OWTT results for experiment 3

Experiment	Mean	95%-Conf.	Variance	Max	Min	Peakedness	Dups[4] / Unm[5]
3-1	2.59	± 0.008419	18.32	70	0.40	7.1	0 / 7180
3-2	4.88	± 0.012140	37.36	92.2	0.41	7.6	0 / 25657
3-3	7.56	± 0.013240	43.56	77.7	0.41	5.8	0 / 44961
3-4	2.58	± 0.008468	18.56	77.3	0.41	7.2	0 / 5880
3-5	3.02	± 0.009749	24.53	80.9	0.41	8.1	0 / 8496
3-6	3.02	± 0.009749	24.53	80.9	0.41	8.1	0 / 51138
3-7	3.13	± 0.010160	26.69	102	0.41	8.5	0 / 6199
3-8	5.1	± 0.014930	56.96	107	0.41	11	0 / 18238
3-9	16	± 0.020870	100.2	110	0.42	6.3	0 / 115763

The computer A generates in this case traffic with $\alpha = 1.2$ and $\rho = 0.4$. On the contrary, computers B, C and D generate (each of them) traffic with $\alpha = 1.2$ and $\rho = 0.1$. Their traffic is broken up in the routers R1, R2 and R3 into merging traffic (50 %) and cross traffic (50 %). Below these plots it is also shown the throughput (Mbps) measured at the destination E and the associated histogram. Underneath it is shown the OWTT measured in this experiment together with the associated histogram. OWTT is observed to have spikes as high as 100 ms. Furthermore, it is also observed that the OWTT histogram has a heavy tail.

Finally, figure 3 shows the measured mean and variance of OWTT obtained for all three experiments. It is observed the strong dependence of the mean and the variance on traffic characteristics and link utilizations. Further, it is also observed that it is the α parameter of the generated traffic that mostly influences the router behavior at large ρ. This is of course the consequence of queueing behavior, which typically shows heavy-tailed delay performance (e.g., Weibull distribution) in the case of traffic with Long-Range Dependence.

6 Conclusion

A dedicated measurement system for collecting high-quality traffic traces and measuring e2e delay has been reported. The measurement infrastructure deals with problems like synchronization, high-accuracy in timestamping and processing of multiple traces collected at various measurement points. A measurement study of delays through IP routers has been reported. The measurement setup is reported in a detailed way together with the set of experiments done. Several important statistics have been reported about the delay for a router and for a chain of routers. Our results confirm other results reported earlier that the delay in IP routers is generally influenced by traffic characteristics, link conditions and, to some extent, details in hardware implementation and software releases. Our future work will be about modeling of the obtained results, to include possible correlations existent between queue lengths at adjacent nodes. Our goal is to find a formula for the e2e delay in a chain of routers.

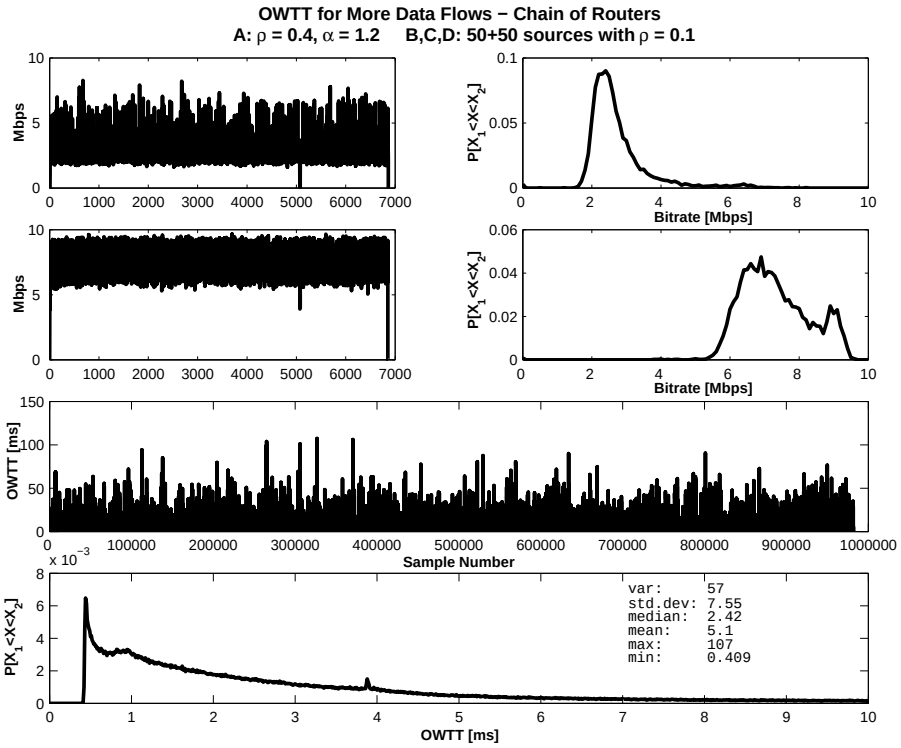

Figure 2: Throughput measured at computer A (upper left) and at computer E (below) together with the associated histograms (right side). Underneath: OWTT measured for more data flows in a chain of routers (experiment 3-8) with associated histogram

References

[1] Almes G., Kalidindi S. and Zekauskas M., *A One-way Delay Metric for IPPM*, IETF RFC 2679, 1999.

[2] Bovy C. J., Mertodimedjo H. T., Hooghiemstra G., Uijterwaal H. and Van Mieghem P., *Analysis of End-to-End Delay Measurements in Internet*, ACM PAM, Fort Collins, Colorado, USA, 2002.

[3] Caida 2001 Network Measurement Metrics WG., http://www.caida.org/outreach/metricswg/faq.xml [checked June 2004]

[4] Carlsson P., Ekberg A. and Fiedler M., *On an Implementation of a Distributed Passive Measurement Infrastructure*, COST279 TD(03)042, 2003.

[5] Constantinescu D., Carlsson P., Popescu A., *One-Way Transit Time Measurements*, Technical Report, Blekinge Institute of Technology, 2004, ISSN:1103-1581.

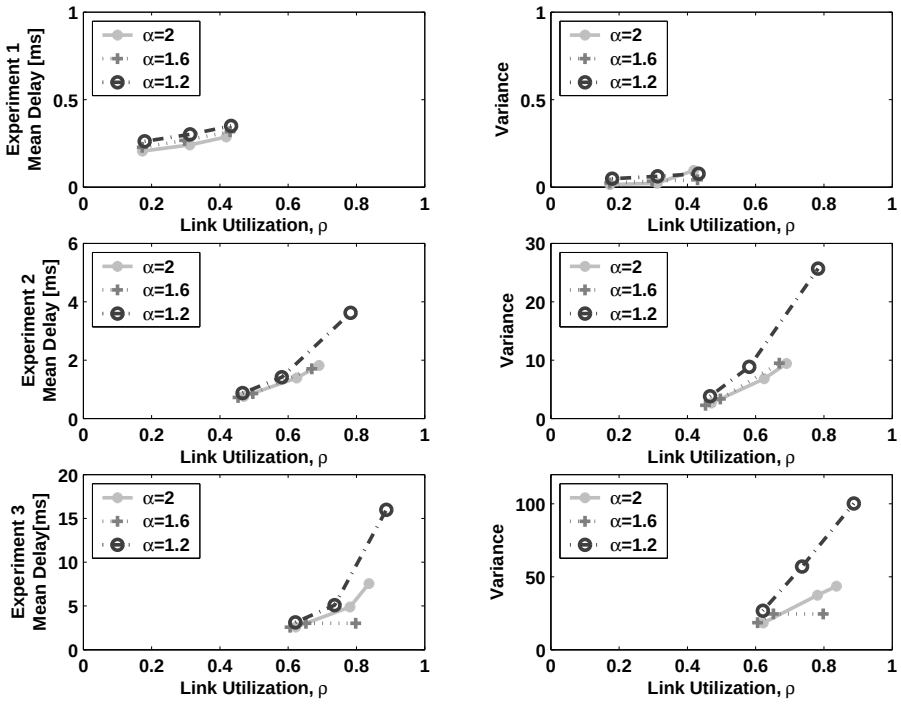

Figure 3: Summary of measured OWTT performance

[6] Claffy K. C., Polyzos G. C. and Braun H. W., *Measurement Considerations for Assessing Unidirectional Latencies*, Journal of Internetworking, Vol. 4, No. 3, 1993.

[7] *Endace Measurement Systems*, http://www.endace.com

[8] Jena A. K., Popescu A. and Nilsson A. A., *Modeling and Evaluation of Internet Applications*, International Teletraffic Congress ITC-18, Berlin, Germany, 2003.

[9] Leland W. E., Taqqu M. S., Willinger W. and Wilson D. V., *On the Self-Similar Nature of Ethernet Traffic (Extended Version)*, IEEE/ACM Transactions on Networking, Vol. 2, No. 1, 1994.

[10] Papagiannaki K., Moon S., Fraleigh C., Thiran P. and Diot C., *Measurement and Analysis of Single-Hop Delay on an IP Backbone Network*, IEEE Journal on Selected Areas in Communications, Vol. 21, No. 5, August 2003.

[11] Paxson V., *Measurements and Analysis of End-to-End Internet Dynamics*, PhD Dissertation, University of California at Berkeley, 1997.

[12] Secure Hash Algorithm, *Announcement of Weakness in the Secure Hash Standard*, National Institute of Standards and Technology (NIST), 1994