On the Feasibility of a Large Scale Distributed Testbed for Measuring Quality of Path Characteristics in the Internet

Miriam Allalouf
IBM - Haifa Research Labs

Elad Kaplan
Tel-Aviv University

Yuval Shavitt
Tel-Aviv University

***Abstract*—There is a long line of research on measuring the Quality of Service (QoS) path characteristics of the Internet, such as available bandwidth, path capacity, packets reordering, delay and jitter. Most of the measurement techniques are based on active probing using pairs or trains of packets. The packets are either transmitted back-to-back or at a desired spacing (e.g., to achieve a certain rate). In most cases, one-way active probing techniques are preferred over round trip measurements as they gather less measurement noise.**

However, a large scale study of the Internet using such techniques was not feasible due to the need to deploy and manage a large number of packet emitters and sinks. In this paper, we present the design of a system for conducting large scale QoPC measurements. Our novel design is based on the ability to emit packets either back to back or at desired rates using off the shelf MS Windows hosts, thus achieving the ability to use a volunteer community as measurement hosts. We demonstrate experimentally and explain how this can be done, and discuss the system aspects of such a solution.

I. Introduction

The application variety in the Internet is constantly increasing, with different applications requiring different networking capabilities. Therefore, the characteristics of the path which the application packets traverse through, are of great importance. They determine the suitability of the path to the application requirements, such as: available bandwidth, jitter, packets reordering, delay and so forth. Future trends for the quality of service are to advise better management through enhanced monitoring tools. These tools should retrieve more information on the characteristics of the path with regard to the application requirements as described above. Hence, a new large scale measurement infrastructure is essential.

There is a long thread of research on both path capacity [4] and path available bandwidth estimation [8], which use packet pairs and packet trains are the key probing tools in active techniques. The transmitter node emits the probing packets with known (possible 'zero') inter-packet gaps. The receiver node may be located either at the far end of the path (measuring one way path), or may be the transmitter node itself (measuring round trip path). In the round trip case the measurement is based on ICMP packets, which are generated by the far node as a response to the original transmitted packets. In both methods the estimation of the path characteristics is based on the analysis of the receiver packet interarrival times. In general, one way methods are preferred over round trip methods since they are less susceptible to measurement noise. Many studies aiming to understand and quantify reordering of the Internet [11], also prefer one-way measurements.

However, one way measurement testbeds are hard to set up, and thus most of the experimental work in this field is limited to a few emitters (at most a few tens) and a few receivers. Many tools were designed to measure end to end available bandwidth [8], and some experiments were done using these tool over a large deployment of well tuned Linux servers [6], while other techniques were designed to measure reordering [11]. However, we are not aware of a large scale attempt to gather multiple profile characteristics over the Internet using open large scale testbed.

The Inter-packet Delay Measurement (IDM) system was designed and developed to be a powerful tool for researchers, enabling them to perform one way measurements using packet trains emitters and sinks for determining the QoPC. In order to achieve rapid deployment and simple management, IDM was designed as an extension to the DIMES [9] infrastructure. DIMES is a highly distributed Internet measurement project, aimed at studying the structure and topology of the Internet. DIMES' strength lays within the large volunteer community that installs its agents on their machines. The agents perform round trip measurements using traceroute and ping. The results are sent to a central database at Tel Aviv University.

The IDM extension augments DIMES with the ability to emit highly programmable packet trains into the Internet. Thus, it enables configurable distributed experiments on the Internet, aiming to measure the impact of the network on packets in terms of bandwidth, capacity, reordering and queueing delays. Each packet in the train can be independently configured: IP header parameters, layer 4 type and parameters, packet length and control of transmission timing. Due to DIMES large scale distribution, researchers can emit IDM packet trains from hundreds of different locations towards available destinations. Each experiment can be complemented by traceroute and ping to measure the round trip time and middle routing hops.

As the set of receivers, we selected the ETOMIC [7] servers due to their ability to measure packet arrival time at sub μSec accuracy, which is achieved by DAG hardware. ETOMIC have a fair distribution in roughly 18 locations throughout

Europe (and Israel), with plans to double this number in the near future. The IDM was developed under the assumption that to get a wider distribution of receivers, general purpose machine will have to be used. Therefore it will be easy to port to different operating systems and hardware. For example PlanetLab boxes [3] could be used, but the ability to accurately stamp incoming packets must be studied first.

Performing combined one-way and round-trip measurement within large scale experiments requires enhanced system level coordination. The system design must be able to manage concurrent measurement from many sources to many destinations, control the result collection from both ends of path, and finally, efficiently aggregate the results into a central database for researcher analysis.

The high programmability of the IDM packet trains enables researchers to examine and analyze routing differences between packets with different combination of packet sizes, protocols, port numbers etc. Thus, identify splitters, shapers, and possibly queueing disciplines along the route. This paper focuses on the system, therefore does not dwell on these issues.

We suggest here to use software based emitters to transmit accurately spaced packets. Specifically, enhancing the DIMES software based agents to be traffic emitters for path characteristics measurements.

Although the majority of agents are based on MS Windows, we show that, contrary to the prevailing perception, accurately spaced packet trains can be emitted from general-purpose Windows machines at the line speed. We discuss the combination of various MS Windows mechanisms, which were built to provide the agent emitting ability. A parameter space is defined which describe system conditions to accomplish successful operation of the DIMES agents.

The rest of this paper is organized as follows. In the next section, we describe our system design and implementation. Next we present experiments that prove that packets are indeed emitted equally spaced in a train and analyze the conditions that enables this. Finally, we show a short example of an experiment conducted with the system.

II. IDM DESIGN AND IMPLEMENTATION

In designing a highly distributed one-way measurement system one would like to create a system, which is as flexible as possible, allowing easy generation of experiments that meet a variety of the goals. The IDM system is designed to deliver programmable packet train emissions, since it is needed for most active measurement techniques. Thus, one would need to have control over the following aspects of the experiment:

- Selecting the set of emitters and sinks. The selection process should enable choices based on geographic locations, such as country or city, AS (autonomous system), IP prefix, etc.
- Setting the daytime period and the repetition interval.
- Setting the number of packets per the train.
- Controlling the inter-packet and inter-train transmission timing.
- Setting characteristics independently **per packet**:
  ⇒ Packet size in bytes.
  ⇒ IP layer: destination address, various IP fields such as TOS & TTL
  ⇒ Layer 4: TCP, UDP, or ICMP; destination and source port numbers.

The ability to transmit packets from distributed emitters to the same source while taking into account packet loss, duplication, and reordering requires the packet identification within the payload. For this purpose the emitters stamp each packet with unique identification for the agent-experiment-train. The packets also contain the time-stamp taken before transmission, which is compulsory for accuracy verification, since on the emitting machine packets from other applications may be transmitted at the same time, causing corruption of timing (see Sec. III-B). Another important capability is to avoid synchronizing packets from multiple emitters to pass through some link, or to arrive to a destination node at the same time. This DDoS like feature requires accurate absolute timing (in contrast with the relative timing between packets in the train).

A. Implementation infrastructure

As mentioned, the DIMES infrastructure performs the system management, coordination, and the probe emitting functionality. The current DIMES measurement portfolio is based on several variants of traceroute and ping measurements, which are all round trip measurements. Namely, an agent emits packets and waits for a response for each packet emitted. Thus, each agent works independently of other DIMES agents. IDM requires coordination between emitters and sinks and thus complicates the system operation. Section II-B describe the flow which synchronizes the entire system. A web based planner can be used to configure the experiments.

The ETOMIC servers perform the sink part of the system. Each ETOMIC server contains a special hardware, a DAG card, which captures the packets with time resolution of sub μsec. A GPS receiver synchronizes the ETOMIC clock. A modified DIMES agent was written to utilize the ETOMIC special hardware. Throughout this paper DIMES agent over ETOMIC is simply called ETOMIC.

B. IDM experiment flow

Fig. 1 presents the flow of preparing, executing, and retrieving the results for the IDM experiment. Since the ETOMIC infrastructure cannot perform two experiments concurrently, the first step in the experiment is reservation of time slots in ETOMIC through its web interface at www.etomic.org (step 1 in the figure). Next the researcher submits a request for an experiment in the DIMES planner at the web site www.netdimes.org (step 2). The DIMES request contains the experiment parameters, such as the profile of the agents, the commands to issue: ping, traceroute, and IDM with parameters as discussed in the beginning of Section II. Currently, each such request is manually approved to avoid malicious use.

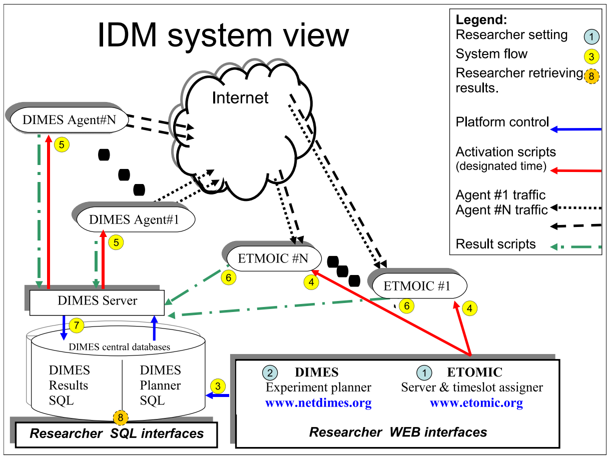


Fig. 1. IDM system view

If approved, the experiment is stored in the DIMES central database (step 3). At the chosen time, the ETOMIC servers will start running the modified agent (step 4). The server will transfer execution scripts to the designated DIMES agents (Transfers are initiated by the agents when they ask for new scripts). The asynchrony in the scripts assignment, which has its limitation, prevents traffic synchronization, which may interfere with the results. This lack of synchronicity also mitigates a potential opening for malicious acts. The agents execute the script (step 5). The results of the round trip measurements (ping/traceroute) are sent to the central database with the experiment tag. The probing packets are emitted through the Internet towards the ETOMIC. The ETOMIC agent analyzes each packet according to the sender Agent ID, train ID and intra train index, and stores the relevant information. At the end of the execution (step 6) the relevant results are transferred to the DIMES central database (steps 7, 8) with the same tag as the round trip results, where they can be retrieved for evaluation by the researcher.

C. Setting Gaps Between Packets

In order to transmit packets with precise gaps we insert between each packets a gap packet with the TTL field set to 1. The gap packet size is calculated such that its length, including the Ethernet overhead (see Sec. III for a calculation example of the packet length), will match the desired rate (see Fig. 2). While this method is limited to gaps larger than the size of the minimum size packet, we have found it valuable for many purposes.

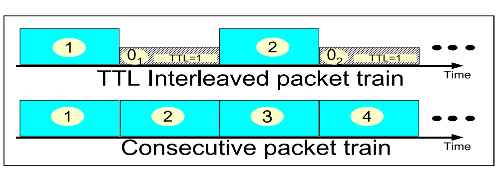


Fig. 2. Packet train interleaved vs. consecutive

III. Validation Test

The ability of agents, running over a commonplace operating system (like MS Windows or Linux), to send packets in an accurate rate is essential to the system correctness. Therefore the following section describes the set of validation tests.

Most of the presented validation results were performed from the same PC; however, similar behavior was observed in the rest of the agents that were examined. The validation DIMES agent was installed on a PC running MS Windows (HP Compaq dc7100 Pentium(R) CPU 3GHz, 512MB RAM). It was placed as close as possible to an ETOMIC server to reduce network interferences. They were both connected using Fast Ethernet (100Mbps) LANs located at the Hebrew University in Jerusalem Israel (HUJI) with only the university firewall separating between them. The network path distance between them was only one IP-TTL hop.

The testing trains were constructed from 100 packets. The tests were repeated with both UDP & ICMP protocols and various packet sizes. There was no difference in the evaluation setup between UDP & ICMP (in long haul experiments there were great differences)

We used two types of graphical views to quantify and evaluate our results. The first shows the submission timestamp vs. the received timestamp for each packet. All times are relative to the submission or reception of the first packet. The timestamps of the receiver are sorted according to the sender index, namely in case of packet reorder, the graph is not monotone. The plot is displayed in "stem" mode for better visualization of transmission rate. The density of the vertical lines represents the inter-packet time, and thus corresponds to the rate.

Slope in this graph represents the rate ratio between sender and receiver. A 45 degree slope means that the sender packets were received at the same rate as they were sent. Lower slopes means that packets were delayed by the network.

The second type of graph shows the inter-packet gaps at both ends. The DIMES transmission timestamp are marked by blue circles (o), and the ETOMIC reception timestamp marked by red pluses (+). The packets at the reception are ordered according to the transmission order, and thus we can have negative inter-arrival at the ETOMIC. The plots include a solid line that represents the minimal possible inter-packet gap for the interface in use, Fast Ethernet in our case. The calculation includes the headers and layer 1 overhead of 100Mbps MAC Ethernet link. For example, packet trains of IP packets of size 1400 bytes have a minimal inter packet gap of 115.04μsec (including 14 bytes MAC header, 4 bytes FCS, and 20 bytes of MAC IFG & SFD).

A. Trains of large-Sized packets

The typical result for packet trains of 1400 byte packets is presented in Figs. 3 & 4. Fig. 4 shows that the 1400 byte packets arrive at the ETOMIC box back to back, as the interarrival (the plus marks) are on the solid line (with deviation of less than the timestamp resolution - 1μSec).

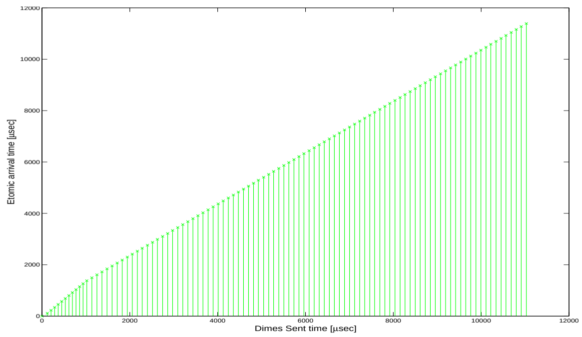


Fig. 3. TimeStamp view: a consecutive transmission of a train comprised of large packets at HUJI.

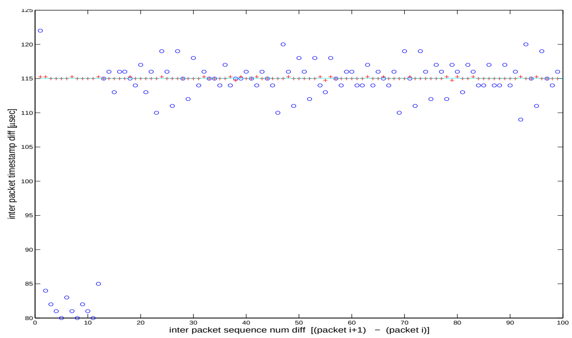


Fig. 4. InterPacket differential view: a consecutive transmission of a train comprised of large packets at HUJI.

Most of the DIMES inter-sending times are spread above and below the solid line up to 6μSec in each direction (this 5% inaccuracy is discussed below). But clearly the packets leave the emitting machine back to back as demonstrated by the accurate receiver time stamping. The first 12 packets' inter sending time is 85μSec implying a rate higher than 130Mbps, which is physically impossible. It can also be noted by the slope of the first packets in Fig. 3.

To explain this phenomenon the packet handling mechanism of the DIMES agent need to be described. The DIMES agent uses the Network Device Interface Specification (NDIS) [1] for emitting the IDM packets. This interface resides between the data-link-control and the media-access-control layers in the OSI model [10]. The device driver creates the packet descriptor in pre-allocated buffers in shared memory and performs a memory mapped I/O access (also known as a doorbell-write) to the NIC hardware (see [2]). The NIC wakes-up upon the doorbell-write, reads the packet descriptor, and performs a DMA transfer of the packet data into the NIC hardware queue. Typical buffer memory sizes are 32KB-128KB but less than half is devoted for the outgoing packets. [1]

[1]Various vendors mentioned the internal memory in their vendors' product brief. For example http://www.broadcom.com/collateral/pb/5703-PB03-R.pdf http://www.marvell.com/products/pcconn/yukon/Yukon_88E8001_10_073103_final.pdf, and http://www.intel.com/support/network/adapter/pro100/sb/cs-010531.htm.

The IDM software prepares all probing packets of a train in shared memory buffers. The transmission is done in a simple `for` loop. The only actions performed between two consecutive transmissions are sampling and insertion of the timestamp into the packet and transferring the packet descriptor to NDIS, which forwards it to the network interface card (NIC). The transfer is a blocking action; its rate is determined by the communication rate with the NIC. As discussed earlier, part of the train packets in Figs. 3 and 4 had a higher time-stamping rate than the Ethernet rate. This can now be explained by the ability of the PC to pass packets to the NIC at a higher rate than the line rate until the NIC's internal buffer fills up. When the memory is full, a new packet can be admitted only when one is evacuated, and as a result packets are transferred to the NIC at approximately the line rate. This also explains the $\pm 6\mu$sec inaccuracy in timestamping the packets since it has to do with the scheduling of the NIC interrupt.

The amount of internal memory dedicated to store packets can be calculated as follows. It takes the system 1100μSec until it becomes full (the knee in Fig. 3) in this period 12 of 1400bytes packets where handed to the NIC. The NIC could empty at this time $\lfloor 0.00011 \times 100,000,000/((1400+38)*8) \rfloor = 8$packets, namely, the NIC become full with 4 packets inside, or 44,000 bits. Calculating for other experiments done from two different machines we obtained values in the range between 30Kb to 50Kb. This may hints that the vendors devote much less than half of their buffer space to outgoing packets. Generalizing this method to continuous memory level calculation enables the researcher to detect whether other applications on the emitter machine transmitted packets which interleaved with the probing trains.

B. Identifying gaps in trains

As DIMES is a volunteer based infrastructure, the DIMES agent priority is intentionally set low to protect the volunteer's machine from suffering performance degradation due to DIMES activity. This can lead to two undesired phenomena: First is the possibility that another applications will use the CPU in the middle of train transmission and thus disrupts train continuity; Second, other applications may transmit packets that may interleave with the train packets causing the probing packet to loss their back to back characteristics.

Simply looking at the DIMES timestamps on the packets is not sufficient to identify whether a train was disrupted. For example, if the packet insertion to the NIC ceased for a short period the NIC's internal buffer may allow us to maintain a steady flow of back to back packets. Thus, we seek a method to identify when the packet train is disrupted. To this end, we use the DIMES time stamps in the arriving packets to calculate the amount of memory which the emitted packets use. This is simply the amount of packets handed to the NIC minus the amount of packets that could be transmitted since the insertion of the first packet. Analysis of the memory consumption indicates whether the packet train was disrupted and why, as we demonstrate below.

To test the risk of interference, IDM was tested under extreme conditions. The first test examined IDM working in a high CPU utilization scenario, which was achieved by running anti virus scanning on the host machine. The second test examined IDM working with concurrent networking application, which was accomplished by uploading a file from the host machine using Skype. Both scenarios were tested for tens of trains, and the memory consumption was analyzed to detect gaps in the trains.

Fig. 5(a) depicts an example of the perturbations in the packet train time stamping in a CPU utilization scenario. It presents the sending times and arrival times of a 1400 byte packet train from a DIMES agent in Tel Aviv towards an ETOMIC server in Chania, Crete, Greece. As mentioned above the density of the vertical line depicts the inter-packet time. It can be seen that the density is not uniform 4 and 10 mSec after the train start (corresponds to packets 35 and 88 in the train) and there is a large gap half a mSec later (after packet 94). Fig. 5(b) shows the memory utilization analysis of this train. The first two events are short, and the packets indeed arrive at the destination back to back (this is hardly a proof since the network could cause this). Analysis of the memory utilization shows that the memory consumption is kept strictly positive meaning that the NIC has enough packets in the buffer to transmit continuously the train's packet. The third and long event cause the NIC buffer to empty and the train to be disrupted. Note that after the first event we see a similar fast packet insertion at a rate equal to the rate at beginning of the train.

Fig. 6(a) shows a similar train to the previous, which was transmitted in a network utilization scenario. Interestingly, about a third of the trains in this scenario showed no indication of interference. The figure shows a 'foreign' packet insertion around packet 36, this can be seen by the larger gap between packet stamps at the origin and also at the receiver (again, the network could be responsible for this). Fig. 6(b) shows that the memory estimation initially decreased as before, but returned to a stable level, lower than before. The difference between the stable levels (36Kbits and 24.8Kbits) is the size of an uploading packet, 1400 bytes, which fits the packets sizes of the uploading application.

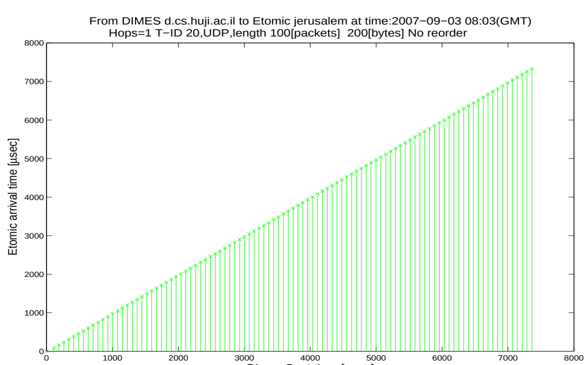


Fig. 7. A 200 byte packet train at HUJI.

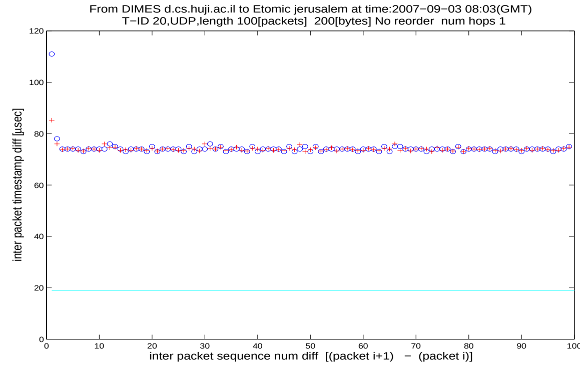


Fig. 8. A 200 byte packet train at HUJI.

C. Trains of small packet sizes

Transmitting trains with uniform sizes of 100, 200 (see Figs. 7, 8), and 700 bytes in the IDM validation test, resulted in inter-sending times around the constant value of about 75μsec for all trains (more than 97% of the packet had less than 5μsec deviation), while 100Mbps Fast Ethernet back to back packets sending time are 11μsec, 19μsec, and 59μsec for 100, 200, and 700 bytes packets respectively. Further study confirmed that each PC had a limitation on the minimal sending rate.

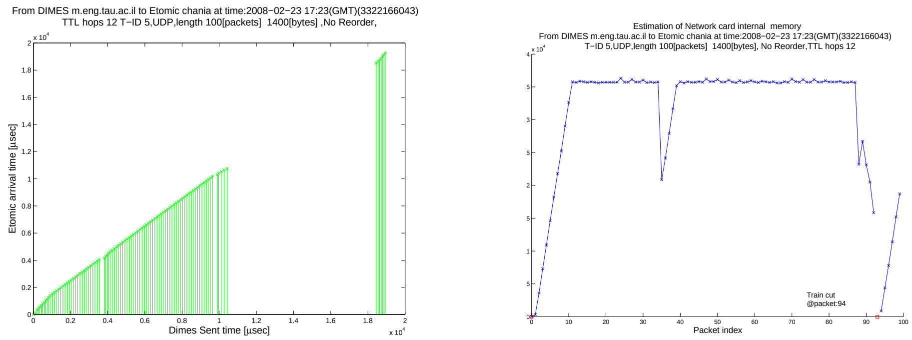
Thus, IDM cannot transmit small to medium packets back to back, however the small packets can be transmitted at constant rate. The ETOMIC receiver timestamp intervals follow tightly the emitter's; more than 95% had less than $\pm 5\mu$sec deviation, whereas the rest appears to be caused by cross traffic (omitting the first packet which is discussed in III-D). Thus, in this case, too, the DIMES transmission time-stamping accurately represents the sending time.

D. The first packet of the train

The first inter-packet gap in the DIMES time stamping is larger than the rest, up to 100μsec more (about 40μsec in Figs. 4 and 8). This can be easily explained by the need of the first packet to wait for the NIC to wake up. At the other end, the receiving inter-packet-interval is just slightly larger in the above figures, which shows that once the packets are placed in the NIC they are sent back to back. However, occasionally larger gaps are visible for the first packet pair, which we failed to explain. They are two ways to overcome this: The easiest, is to ignore the first packet pair in the train. The other way is to send a first packet with TTL $=1$. Thus the first network device drops the problematic packet.

E. Validation summary

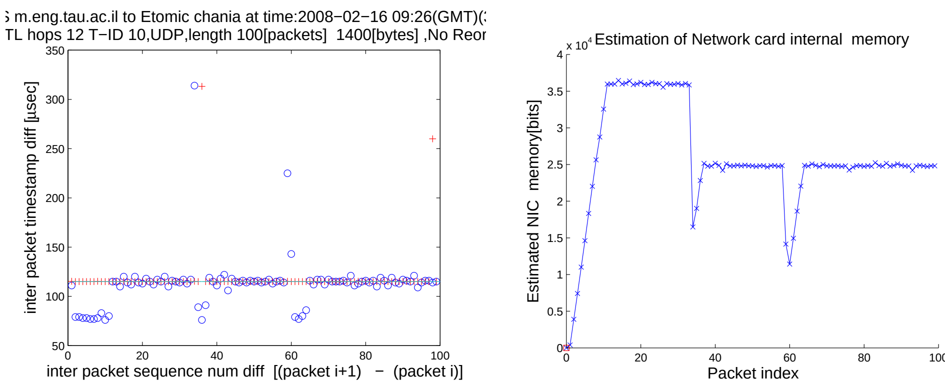
The validation tests show that, using DIMES agents with IDM, it is possible to transmit large packets in Fast Ethernet line rate back to back. For smaller packet, the PC cannot utilize the entire line rate, but sends packets at constant intervals. The rate depends on the host machine, and can be easily calculated from the transmission time stamps. Furthermore, the IDM can diagnose the events of OS task switch, or a different application transmitted packet interleave.



(a) TimeStamp view

(b) Memory in NIC estimation

Fig. 5. IDM Train of 1400bytes packets under heavy CPU load



(a) TimeStamp view

(b) Memory in NIC estimation

Fig. 6. IDM Train of 1400 bytes packets under network load

In the validation test, where both DIMES and ETOMIC were placed at HUJI, there was no difference between UDP and ICMP trains. That is contrary to the Internet end to end measurements which are discussed in [5].

IV. Concluding Remarks

We showed that large scale distribution of sufficiently accurate packet trains is possible. We have confidence that the DIMES with the IDM will prove to be an important tool in large scale studies of the Internet QoS characteristics, such as path bandwidth, capacity, loss, jitter, etc. We also believe that this tool can also be used to detect rate limiters, shapers, active queue management schemes, and load balancers. The experiments described in this paper were made on an alpha version of a DIMES agent. A few months ago, a new version of the DIMES agent which includes IDM (ver. 0.5.0) was released. Since then, over 700 installation were reported, over 600 of them are sending regular keep-alive messages to the system. This opens the door to large scale experiments with this new tool.

References

[1] http://www.microsoft.com/whdc/archive/ndis5.mspx.

[2] Introduction to intel I/O acceleration technology and the microsoft windows server 2003 scalable networking pack. Intel/Microsoft white paper. http://www.intel.com/technology/ioacceleration/317106.pdf.

[3] B. Chun, D. Culler, T. Roscoe, A. Bavier, L. Peterson, M. Wawrzoniak, and M. Bowman. PlanetLab: an overlay testbed for broad-coverage services. *ACM SIGCOMM Computer Communication Review*, 33(3), July 2003.

[4] C. Dovrolis, P. Ramanathan, and D. Moore. Packet dispersion techniques and capacity estimation methodology. *IEEE/ACM Transactions on Networking*, 2004.

[5] E. Kaplan. On the Feasibility of a Large Scale Distributed Testbed for the Characterization of Path Quality on the Internet. Master's thesis, School of Electrical Engineering, Tel Aviv University, Israel, Sept. 2008.

[6] S. Lee, P. Sharma, S. Banerjee, S. Basu, and R. Fonseca. Measuring Bandwidth Between PlanetLab Nodes. In *6th International Workshop on Passive And Active Network Measurement (PAM)*, Boston, MA, USA, Mar./Apr. 2005.

[7] D. Morato, E. Magana, M. Izal, J. Aracil, F. Naranjo, F. Astiz, U. Alonso, I. Csabai, P. Haga, G. Simon, J. Steger, and G. Vattay. The european traffic observatory measurement infrastructure (ETOMIC): A testbed for universal active and passive measurements. In *TRIDENTCOM 2005*, Trento, Italy, Feb. 2005.

[8] R. Prasad, M. Murray, C. Dovrolis, and K. Claffy. Bandwidth estimation: Metrics, measurement techniques, and tools. *IEEE Network*, 2003.

[9] Y. Shavitt and E. Shir. DIMES: let the internet measure itself. *ACM SIGCOMM Compututer Communication Review*, 35(5):71–74, 2005.

[10] A. Silberschatz, P. B. Galvin, and G. Gagne. *Operating System Concepts*. John Wiley & Sons, seventh edition, 2005.

[11] B. Ye, A. Jayasumana, and N. Piratla. On end-to-end monitoring of packet reordering over the internet. In *The International Conference on Networking and Services (ICNS)*, July 2006.