Network Analysis using SoNIC

Junyu Chen, Yicheng Liang, Zhihong Liu
{jc2838,yl2533,zl456}@cornell.edu
Cornell University

***Abstract*—The physical and data link layer of the network stack contain valuable information. SoNIC (Software-defined Network Interface Card) provides software access to these layers by implementing them in software. It provides complete control over network stack in real-time. With SoNIC, we are able to perform precise network measurements. Also, GENI (Global Environment for Network Innovations) provides a virtual laboratory for networking and distributed system research and education. Therefore, we can gather valuable information, in particular, available bandwidth estimation, on GENI using SoNIC and perform corresponding analysis.**

***Index Terms*—SoNIC, Available bandwidth estimation, Pathload, Probe**

I. Introduction

For different uses of network, people have strict requirement on available bandwidth estimation, like designing high performance network system, improving network protocol and building distributed system. However, bandwidth estimation is a problematic measurement where accuracy is difficult to achieve, particularly in high-speed network.

Current way for active available bandwidth estimation fall into one main idea, which is sending a train of probe packets through a network path to destination point, then analyzing the data receiving end from probe packets timestamps of gap. To its credit, it makes full use of the flexibility of user space control and compatibility of current bandwidth estimation. However, there are several problems in this estimation approaches. Firstly, creating explicit probe packet that consume extra bandwidth and CPU resource. Besides, the probe packet which is created in user space, due to the design of operating system, are often perturbed in level activities. Lack of precise timestamp, the accuracy will be affected. Lastly, bursty cross network traffic will also make a passive effect on the estimation approach.

This paper presents an approach to address the above problems. While keeping the advantage of approaches above, we can reduce the negative effect on explicit probing packets through using the existing application packets running in the network path, and avoiding the perturbed activities in operating system. Finally, operating in the user space provides users more flexible to control the whole measurement and make an immediate improvement when finding problems during the estimation.

As we know, the physical and data link layers of the network stack contain valuable information. Through controlling over the physical and data link layers, we can control over the entire network stack in real-time. SoNIC (Software-defined Network Interface Card) provide us an possibility to access to these two layers in software by implementing them in software. With SoNIC, we are able to perform precise network measurements, accurately characterizing network component such as routers, switch and network interface cards. SoNIC is able to change Idle characters, accurately measure inter-packet delays.

In this paper, we implement the approach we raised and test it in several environments. we run experiments on syslab machines that are equipped with SoNIC boards. Under different topologies in local, we get efficient and useful experiment data to complete and automate data analysis process. Also, we use this approach to estimate the available bandwidth on GENI. GENI provides a virtual laboratory for networking and distributed system research and education. we load script and re-run the above experiments.

All tests are run successfully on the different environment, including crossing traffic, and we got plenty useful and detailed data from it. Through analyzing the data and improving parameter in the experiment, we can make whole and accurate understanding of the network performance. Especially, this approach allow program in user space to accurately measure the available bandwidth of high speed, 10Gbps, network.

II. Background

Bandwidth estimation techniques, specifically available bandwidth estimation algorithms, measure network throughput, which is closely related to congestion. Like the experiment on cross traffic, we need to make an accurate analysis on that. To deal with the problem mentioned above, we make a full use of the feature of SoNIC to characterize the pattern in packet chains, while SoNIC provide us a way to access the packet without affecting the accuracy. GENI allow us make full test and improvement on the estimation way on different environment.

In this section, we introduce and discuss some key ideas and tools underlying existing, which play important role in this paper, including GENI, SoNIC and some methods on bandwidth estimation. And through those analysis, we can make a whole picture on the approach design following.

A. GENI

GENI is a facility concept being explored by the United States computing community and its goal is to enhance experiment research in computer network and distributed system. Based on its design, it is useful in experiment to obtain compute resource from location around United States and control how network switches in their experiment handle

traffic flows. It is a large-scale experiment infrastructure, which provide us with more resources than is typically found in one lab. In our experiment, one of important part is using the GENI to make bandwidth estimation, in which we made connections between eastern coast and west coast, UC Davis to NC Chapel Hill, which can make a full evaluation on the current network situation.

B. SoNIC

SoNIC, Software-defined Network Interface Card, which provides access to the physical and data link layers in software by implement them in software. So SoNIC provides complete control over the entire network stack in real time. As we known, the physical and data link layers are usually implemented in hardware and not easily accessible to system programmers, while handling PHY enables highly accurate traffic analysis and replay capabilities. Indeed, SoNIC allows us to measure on network by generate packets at full data rate with minimal inter packet delay. Importantly, SoNIC accurately captures incoming packets at any data rate by handling idle characters from PHY. This brings us a useful method to use the application packets in the network rather than creating new packets, which reduces efficiently negative impact on the conflict of the OS perturbed activities. Moreover SoNIC be capable of improving the accuracy of packet timestamp, which can change the rate of the forwarding packet as the need in the experiment through changing the number and position of idle in the physical layers. At the receiver side, this feature brings us a way to profile network components and create timing channels that are undetectable from software application, which is accurate and stable analyzing way. Thus, SoNIC allows cross-network-layer research explorations by systems programmers.

C. Current Methods

An end-to-end approach is the main idea in many current available bandwidth estimation methods. As mentioned above, trains of probe packets are sent to destination side. Through observing and analyzing the change in some packets characteristics at the destination side, we can get the bandwidth information because the gaps between first packet and the last packet will be change, which is queuing delay, if the probing rate exceeds the available bandwidth. Some estimation tools such as Spruce, which samples the arrival rate at the arrival rate at the bottleneck by sending pairs of packets spaced so that the second probe packet arrives at a bottleneck queue then calculate the number of bytes between two probes at the receiver.

III. Related Work

For the bandwidth estimation, there are some active and famous works, like IGI, Spruce, which are differ in the size and temporal structure of probe stream, while using the main idea by analyzing the receiving packets. Spruce uses dozens of packet pairs having a certain input rate defined to be roughly around to the capacity of the path and the packets are located with exponential intervals in order to emulate a poissonian sampling process. Comparing with Spruce, IGI makes a use of sequence of 60 spaced packets as probe packets. The gap between two consecutive packets is increased until the average output and initial gaps match. Through the analysis of the gaps pattern, the bandwidth can be learnt. And the Pathload use constant rate streams and change the sending rate every round, which varies the probing rate using a binary search structure.

Most of those approaches compared the performance against other approaches in their solution so that they work better than other in some how, for example, Pathload focus on a controlled environment and Spruce raises its performance over hundreds of real Internet paths. The above mentioned experiments have also been performed considering different environment and test configuration, so the various results are hard to comparable.

IV. Design

Our measurement achieved high-fidelity by direct access to the physical layer in realtime using SoNIC. First, we describe the characteristic of 10 Gigabit Ethernet(GbE)and how SoNIC help the measurement take advantage of access to the physical layer.

A. 10 GbE Physical Layer

According to IEEE 802.3 standard, when 10Gb Ethernet are passed to physical layer, the data will be reformatted. The physical layer encode every 64bits into a 66 bits block with two bit synchronization header (sync-header). So actually the 10GbE link are working at 10.3125 Gb (10Gb x 66/64). The 10GbE always sends 10.3125Gb per second(Gbps), encoding the raw data (10Gb)with the header and scrambles each block, then , pass the packet to the stack to be transmitted. The receiver? physical layer remove the two-bits header and de-scrambles each block.

There are some special symbols, Idle symbols(/I/), fill the gaps between any two packets. Since the 10GbE always sends the 10.3125Gbps, when there are no packets need to be transmitted, the physical layer continuously adds idle symbols to fill the 10.3125Gbps. To be specific, the standard require at least 12 /I/s after every packets. An idle symbol can be 7 or 8 bits depending on Ethernet frame and physical layer alignment. More importantly, SoNIC can access to physical layer and thus, control the Idle symbols, which enables high-fidelity control of the network traffic.

B. High Fidelity Measurement

That our measurement can achieves high-fidelity is because its direct access to the physical layer, controlling the /I/ characters by the SoNIC. In specific, our measurement can measure(count) and generate(insert and remove) exact number of /I/ characters between each packets to get exact interpacket gap, which leads to the exact rate of traffic. Further more, if two subsequent probe packets are separated by packets from

different flow(cross traffic), the gap between probe packets will include /I/ characters and also the data characters of the cross traffic, but the measurement can still be exact. However, before the SoNIC platform was created, this kind of precise measurement could never be created because /I/ characters were typically inaccessible from higher layer, since they are discarded by hardware and definitely were not accessible in software. In traditionally computer, an end host may timestamp packets in user-space, kernel or network interface hardware which will come up with significant noise. None of the traditional method provides enough precision for high-fidelity network measurement. As a result, many existing bandwidth measurement tools are not accurate.

Our measurement leverages the facility of SoNIC, which control the physical layer traffic by insert or remove /I/ characters from the original traffic. In particular, two variables of the traffic are of interest: The gap between packets and the overall rate of packets trains. The SoNIC provide API to user-space program that can specify the number /I/ characters between two packets, allowing user space programs to perform high-fidelity pacing.

C. Generalized Probe Model

Fig. 1 shows a generalized probe model that able to emulate a number of existing bandwidth estimation algorithms and used for the rest of the paper. The horizontal dimension is time. the Vertical dimension is the probe rate. Each pulse(we call it train) contains multiple probe packets sent at particular rate, as depicted by height. In specific, the parameter N represents the number of packets in a probe train. R stands for the probe rate of the train, we control the probe rate R by place exact number of inter-packet gap (/I/ characters) between N packets. Packet sizes are also considered in computing the probe rate. The gap between probe train are specified by parameter G(inter-train gap). Finally, the measurement can also be consisted of several probe samples(a set of probe trains with increasing probe rate) and the distance between the two measurement sample are calculated by D(inter sample distance). With these four parameter we can emulate most of the probe traffic of existing algorithm. For example, to emulate Spruce probes, we can set the parameters (N,R,G,D) as (2, 500Mbps, 1.2ns, 48us) Similarly, we can reproduce IGI experiment probe with (N,R,G,D) equals to (60,[0.1:0.1:9.6]Gbps, 30s, 30s). Most types of existing probe trains can be emulated with our generalized probe model.

In the following sections, we will use a parameterization which is similar to Pathload with parameters (N,R,G,D) as (20, [0.1:0.1:9.6], G, D). G and D are variable can be decided by user. In Pathload algorithm, we estimate available bandwidth based on the increasing one-way delay(OWD) in probe trains. The basic idea is if the available bandwidth of the bottleneck link is larger than the probe trains, the gap between probe packets suppose to be remain the same as what is sent. On the contract, the probe train will induce congestion in the network

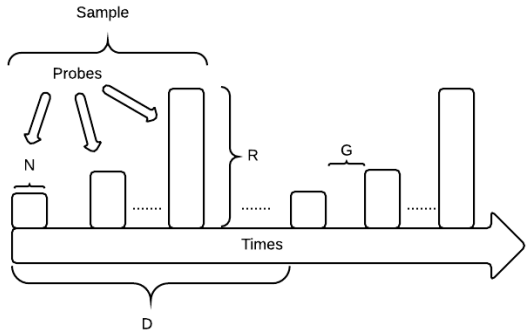


Fig. 1: Probe Train Model

so that the last packet of the probe train will experience longer queuing delays compared to the first packet of the same probe train. The difference between the OWD of the last packet and the first packet can be used to compute the increasing OWD in the probe train. For our measurement, we generate the increasing rate of probe train and where the OWD increases could be used to estimate the available bandwidth.

A bandwidth estimation algorithm based on Pathload needs to compute the increased OWD between the first packet and the last packet in a probe train. Since we can measure the gaps between subsequent packets, it is easy to proove that the increase in the OWD of a probe train $Q_n - Q_1 = \sum G_i - \sum H_i$, H is the sending gaps between the packets in the probe train, G is the receiving gaps between the packets.

V. Evaluation

In this section, we discuss our environment setup, algorithm that we are using to estimate available bandwidth, and show the results with the corresponding algorithm.

A. Environment Setup

To get familiar with SoNIC and a better understanding of current available bandwidth measurement process, we have first set up a topology using a Fractus node. The topology is configured into a simple loopback, which is demonstrated in Fig. 2. Two of the Fractus ports are connected to a switch, with port 0 generating packets (pkt_gen mode) and port 1 capturing packets (pkt_cap mode). With this topology, the Fractus machine is able to generate and transfer packets out of port 0, but port 1 is not able to receive any such packets. We tested the topology also with UDP packets, which are not being forwarded by the switch to any of the other machines connected. Therefore, we have concluded that the switch is not forwarding packets correctly, and could not figure out the cause of such problem. As a result, we have decided to migrate our experiments onto a set of SoNIC machines in system lab.

We obtained access to three SoNIC machines, sonic1, sonic2, and sonic3 in system lab. We had the topology set

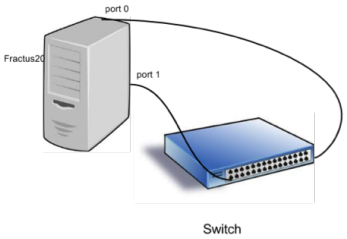


Fig. 2: Fractus loopback topology

up as shown in Fig. 3. In between the machines are two virtual switches. Port 1 of sonic1 is connected to port 20 of vlan2. Port 0 of sonic2 is connected to port 18 of vlan2. Port 0 of sonic 3 is connected to port 23 of vlan3. The two virtual switches vlan2 and vlan3 are connected via port 24 and port 45 respectively. Machine sonic2 was used as the sender that passes the generated probe packets to sonic3 through port 0 using pkt_rpt mode. The switch can also be accessed with sonic3 to view and update the switch configuration, such as mac address table. Because of the ease of access and management and the problems with GENI described below, most of our experiments have been run on the system lab machines.

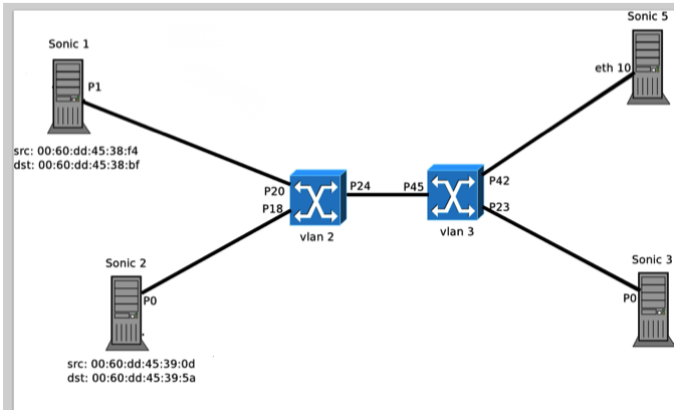
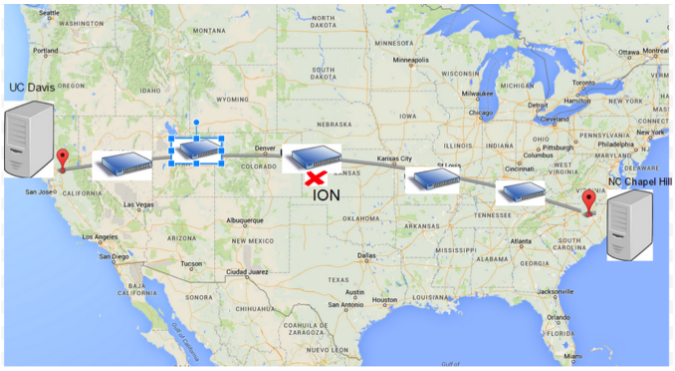
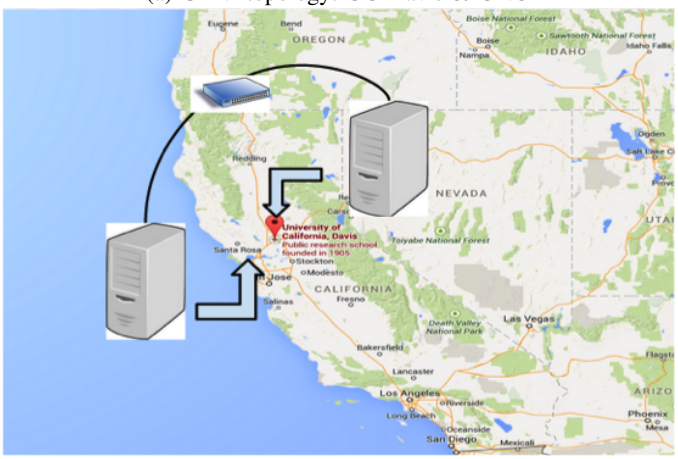


Fig. 3: syslab machines topology

For our topology on GENI, we have decided to use two nodes, one in University of California - Davis, and one in University of North Carolina - Chapel Hill. Since GENI is a virtual laboratory for networking and distributed system research, it is easy to set up different topologies for research purpose for the selected nodes. However, in our case, whenever we want to connect the two selected nodes, the link has to pass through a switch called ION. The request has always failed on this switch. We think that such problem is caused because ION switch is managed by Internet2, and a request for such is needed. As a result, we have decided to instead use two nodes that are both on the west coast (two nodes in University of California - Davis). In this case, the topology did not include the above mentioned switches, and was successfully set up as illustrated in Fig. 4 (b). The two selected nodes are connected through a switch. We were able to successfully run experiments with this topology. However, the experiments are limited because the GENI is often under maintenance, and we have a little less control over the network as with the SoNIC machines in system lab.



(a) GENI topology: UC Davis & UNC



(b) GENI topology: two nodes in UC Davis

B. Algorithm

We generated a probe pattern with the parameterization that is similar to Pathload, which estimates available bandwidth based on the increasing one-way delay (OWD). The probe train will induce congestion in the network if the rate of this probe train is larger than the available bandwidth. Therefore, we are interested in the difference in the one-way delay between the first and last packet of the same probe train. If the probe train is greater than the available bandwidth, the last packet will have longer queuing delays compared to the first packet. The difference in the OWD can be used to compute the increasing OWD in the probe train. The lowest probe train rate where such queuing delay increases is approximately the

available bandwidth of the bottleneck link.

In our algorithm, we chose the parameters of (N, R, G, D) to be (20, [0.1:0.1:9.6]Gbps, 120,000Bytes, variable). To be specific, there are 60 packets in each probe train. The rate of the probe train starts from 0.1Gbps, and increases to 9.6Gbps with 0.1Gbps increment each time. Also, the gaps between successive probe trains are defined to be 120,000Bytes, and use some selected value for the distance in time between each measurement sample.

To verify our algorithm and evaluate the estimations, we also want to limit the available bandwidth to some values of our choice to see if our algorithm successfully estimate each available bandwidth. To achieve the goal of limiting the available bandwidth, we have written a script to generate cross traffic with corresponding rate. In our script, we compute the number of idle characters of corresponding rates that need to be inserted into the traffic and the number of such packets in order to span the entire time of our experiment.

After generating the cross traffic, we use sonic1 to pass the cross traffic to the switch with the pkt_rpt mode, to sonic5 in Fig. 3. Therefore, the cross traffic is not received by the receiver, which is sonic3 in our case.

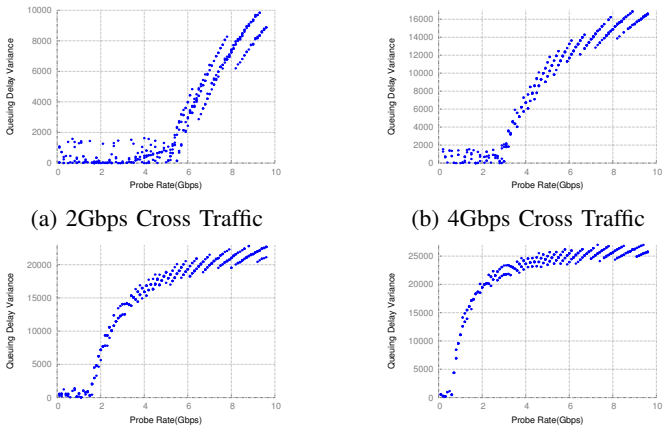
After received the packets, we also implemented a user friendly scripts to summary and estimate the available bandwidth by the received packets. In particular, we process the packets and compute the idle and translate idles to OWD.The script also draw an image, which display the whole OWD changes of this experiment. While calculate the estimated point of where OWD significantly increased, we use both Least mean squares (LMS) algorithms and the Random sample consensus (RANSAC).

C. Results

We have run the experiments for cross traffic rate ranging from 1Gbps to 10Gbps with 1Gbps increments. Fig. 5 shows the results of four of them for our experiments. It states the cross traffic rate, and thus the actual available bandwidth is the difference between 10Gbps and the stated cross traffic rate. However, from our experiments, we actually saw that the available bandwidth should be around 7.8Gbps because the sum of the estimated available bandwidth (reading roughly from the graph or use our script to calculate the lowest probe train rate) and the corresponding cross traffic rate adds up to this value. We have yet found a reasonable explanation for this problem. In future work, we will try to use other method to estimate the actual available bandwidth, think about the possible influences the cross traffic could bring to the network, and discuss with Han about this situation.

VI. Discussion

In this section, we discuss the challenges and issues we encountered in this project and the possible future work related to our current project. In addition, we explain the portion of the project that are dedicated for our M.Eng project.



(a) 2Gbps Cross Traffic (b) 4Gbps Cross Traffic

(c) 6Gbps Cross Traffic (d) 8Gbps Cross Traffic

Fig. 5: Results With Cross Traffic on syslab Machines

A. Challenges

During the process of our project, we have found that SoNIC is a little difficult to deploy. First, in order to load SoNIC, certain kernel version requirements must be met. Previously, the Fractus machine that we were using had a newer kernel version, and the SoNIC is not compatible with such version and thus cannot be compiled or loaded. It only got to work when Han downgraded the kernel version of the machine. Second, switch configuration can affect the use of SoNIC. As described in Environment Setup section, with the Fractus node, the switch was not able to forward packets generated by SoNIC.

Moreover, SoNIC is a little unstable. Our same experiments work some time, and do not some other time. Since the hardware can be unstable, it is also a little hard to debug this unstable situation.

We have also spotted a SoNIC script issue in the SoNIC code provided. For pkt_rpt command, the user space configuration only works for port 0. The configuration cannot be changed to port 1 unless in kernel space. This problem thwarted our cross traffic experiments for a long time as we could not understand why the packets are not being passed through.

Estimation measurement application is built upon the lower layers. Since the lower layers already has too much problem to deal with, and are somewhat unstable, we think it is hard to build such an estimation measurement application without the lower layers working properly.

B. M.Eng Portion

We are also using this project as our M.Eng project. In addition to the course project, we will also run more experiments with additional algorithms such as Pathchirp and IGI, and use their corresponding analysis methods to estimate available bandwidth. Moreover, we will try to improve our estimation algorithm, to provide a more accurate estimation.

C. Future Work

For future works, since our focus is mainly on the network analysis on GENI, more experiments need to be run on GENI. These experiments could be adjusting cross traffic rate and characteristics, using different algorithms, and adjusting the topologies to run similar experiments.

Currently, we are still injecting probe packets into the network to estimate available bandwidth. To address the intrusive issue that we discussed in introduction, we want to use application packets for such measurement, and have the middle box filter the packets to determine which ones to use for the bandwidth estimation.

Lastly, it is always good to have a user space measurement application. We would still want to explore the possibility and feasibility of building such an application in user space.

VII. Conclusion

After the struggle we had with the environment set up, we have successfully set up a working environment on system lab machines. We have written a script to generate cross traffic packets, and completed the automation for the process of data analysis. On the machines, we have also successfully conducted the experiments with cross traffic to estimate the available bandwidth. Moreover, we also have a working GENI topology that is able to perform similar experiment and where we intend to run more experiments. There are still a few tasks to complete over the winter break, which includes running more experiment over GENI, and exploring more algorithms. However, overall, we have come a long way, and have overcome many of the problems.

VIII. References

1

Steven J. Murdoch and Stephen Lewis. 2005. Embedding covert channels into TCP/IP. In Proceedings of the 7th international conference on Information Hiding (IH'05), Mauro Barni, Jordi Herrera-Joancomart, Stefan Katzenbeisser, and Fernando Prez-Gonzlez (Eds.). Springer-Verlag, Berlin, Heidelberg

2

Emanuele Goldoni and Marco Schivi. 2010. End-to-end available bandwidth estimation tools, an experimental comparison. In Proceedings of the Second international conference on Traffic Monitoring and Analysis (TMA'10), Fabio Ricciato, Marco Mellia, and Ernst Biersack (Eds.). Springer-Verlag, Berlin, Heidelberg

3

Han Wang, Ki Suh Lee, Erluo Li, Chiun Lin Lim, Ao Tang, and Hakim Weatherspoon. 2014. Timing is Everything: Accurate, Minimum Overhead, Available Bandwidth Estimation in High-speed Wired Networks. In Proceedings of the 2014 Conference on Internet Measurement Conference (IMC '14). ACM, New York, NY, USA

4

Hao Jiang and Constantinos Dovrolis. 2005. Why is the internet traffic bursty in short time scales?. In Proceedings of the 2005 ACM SIGMETRICS international conference on Measurement and modeling of computer systems (SIGMETRICS '05). ACM, New York, NY, USA

5

M. Jain, "Pathload: A measurement tool for end-to-end available bandwidth," in Proc. Passive and Active Measurements (PAM) Workshop, Mar. 2002, pp. 14-25

6

Jacob Strauss, Dina Katabi, and Frans Kaashoek. 2003. A measurement study of available bandwidth estimation tools. In Proceedings of the 3rd ACM SIGCOMM conference on Internet measurement (IMC '03). ACM, New York, NY, USA