

Inferring Traffic Shaping and Policy Parameters using End Host Measurements

Udi Weinsberg
School of Electrical Engineering
Tel-Aviv University, Israel

Augustin Soule
Technicolor
Paris Research Lab, France

Laurent Massoulie
Technicolor
Paris Research Lab, France

Abstract—The increasing adoption of high speed Internet connectivity in homes has led to the development of bandwidth hungry applications. This, in turn, induces ISPs to protect their core networks by deploying traffic shaping devices. End users, ISPs and regulators need to better understand the shaping policies that are enforced by the network.

The paper presents a method for inferring flow discrimination and shaping parameters in the presence of cross traffic using active probing. The key concept is a stochastic comparison of the inter-arrival times of packets and measured bandwidth of a base-line flow and the measured flow. We present *Packsen*, a framework designed to provide high detection accuracy by sending interleaved flows at a very precise bandwidth, and used it for measurements on a local testbed and on PlanetLab. Evaluation shows the accuracy and robustness of the proposed method for detecting traffic shaping and inferring its parameters.

I. INTRODUCTION

The increasing adoption of high speed Internet connectivity in homes in recent years [1] has led to a wide range of bandwidth hungry applications [2], such as video streaming, voice and video chats over IP, and peer-to-peer (p2p) file sharing. These applications cause extreme stress on networks of residential Internet Service Providers (ISPs), inducing them to protect their network infrastructure by restraining consumed bandwidth. This is commonly achieved by deploying traffic shaping devices, enabling ISPs to have a fine-grained and differentiated control over consumed bandwidth. This form of traffic shaping is performed either with or without advertising the exact policy to users, the latter being the main cause for the recent discussions about *network neutrality* [3] violations.

As traffic shaping is expected to be increasingly common [4], there exists a need for better understanding the properties of traffic differentiation by users, ISP operators and regulators. Although numerous tools were recently proposed [5], they either detect general network problems experienced at end users [6], focus on simply detecting the existence of flow discrimination [7], [4], [8] or require long measurement sessions [9] and relatively intense calculations [10] to obtain accurate results.

This work presents statistical methods, which are based on stochastic comparison of flows, for detecting relative discrimination using active probing, aiming to detect traffic shaping and infer its parameters. Three methods are presented. The first employs short flows and low computational overhead for detecting the existence of traffic shaping. The second infers

shaper parameters using a bandwidth analysis, however it does not perform well when facing significant cross traffic. The third method is more computationally intensive and manages to provide more accurate shaping parameters while being robust to cross traffic. The methods perform well using short and low-bandwidth probes, enabling fast inference and distributed continuous large-scale measurements. These, in turn, can be leveraged using tomography techniques [11] to provide the ability to detect the exact links being shaped.

Evaluation of the proposed methods is performed using a highly flexible framework called *Packsen*, which we developed and deployed on a local testbed and on a large set of PlanetLab [12] servers. These two deployments verify the proposed techniques, and provide an empirical quantification of the inference error when facing different levels of cross traffic.

II. METHODOLOGY

A. Model

Our model includes a traffic shaper in a network residing in the route between producers of various traffic flows to their consumers. The traffic shaper is comprised of one or more packet queues. A packet is first *classified* into a certain class, which determines the its queue. A *scheduler* determines the way packets are removed from the queues and sent towards their destination.

Classification of packets can use many parameters, including protocol, ports, peak hours, etc. Similar to previous work [10], we assume that classification is performed at the granularity of a flow, hence all packets belonging to the same flow are classified the same.

This paper focuses on detecting several popular scheduling policies, and identifying their parameters. When no discrimination of flows is performed, a First Come First Served (FCFS) scheduling policy is applied. Strict Priority (SP) scheduler, on the other hand, gives a specific flow preference over others. Alternatively, it is possible to specify an exact upper limit on the possible bandwidth consumed by a given flow, usually using a Leaky Bucket [13] scheduling policy. Other schedulers set an average limit on consumed bandwidth while permitting short term bursts, usually implemented using Token Buckets. As such, Weighted Fair Queuing (WFQ) is used for setting a weighted division of the overall capacity, possibly enabling flows to “borrow” bandwidth from other flows when possible.

B. Relative Discrimination

In order to test for traffic shaping along a route, we need to compare whether flows that are sent in an identical manner, are received differently. For this end, instead of comparing all combination of various applications, we use the common *relative discrimination* technique [10], [14], [9], [15], where we test each application (the measured flow) against a flow which is assumed to be non-shaped (the baseline flow).

If both flows are affected in a relatively similar manner, then it is plausible to conclude that there is no shaping performed (or more precisely, no relative discrimination) on the measured flow. If, however, the measured flow seems to be significantly altered relative to the baseline flow, then we suspect discrimination, and proceed in its measurement.

C. Cross Traffic

Cross traffic is a common problem when running measurements from end-users, as users can run other application during the measurement causing cross traffic. This traffic can have several effects on the analysis results. First, if no shaping occurs during the experiment, then cross traffic impacts other flows in a relatively equal manner, e.g., reduce the bandwidth of all flows. However, cross traffic can either be classified to a unique flow, or can have the same classification of one of the measurement flows. The first case also affects measurement flows in the same manner. However, the second one can cause bias in the analysis as the cross traffic changes the ratio of consumed bandwidth of the flows.

In order to overcome the biased introduced due to high cross traffic, our model accounts for the variability of cross traffic. Cross traffic changes the way probe traffic reaches the destination, hence it is possible to measure the variability introduced to the probe flow as a result of this cross traffic, and compensate for it. We further elaborate on this issue in the following sections.

D. Detection and Inference Methods

We present a layered approach for detecting whether ISPs employ traffic shaping and inferring the policies and the parameters used for shaping. First, we use a statistical test for measuring whether traffic shaping actually exists by stochastically comparing the distribution of the inter-arrival times of packets in the two flows. When traffic shaping is detected, a second method is used to infer the specific shaping parameters. If excessive cross traffic is detected, a different statistical model is employed, enabling parameter inference even in the presence of significant cross traffic, by using short repeated experiments. The methods are layered such that each requires longer flows and is more computationally intensive, allowing one to balance between the detection details and the effort and resources invested.

1) *Detecting Shapers*: Detecting the existence of a shaper is achieved by sending two short interleaved flows, and measuring the arrival time of the packets in the receiver end. Each flow is represented using a vector of packet inter-arrival times, i.e., the time difference between consecutive packets

of each flow. Each of the two vectors represents a set of samples of some distribution which is determined by the way packets traverse the route between the sender and receiver. In case the flows are handled similarly by devices along the route, then we expect the two sample vectors to have a similar distribution, otherwise one flow is expected to be stochastically significantly different than the other.

In order to test this stochastic difference, we use the Mann–Whitney U-test (MWU) [16] on the two vectors. MWU is a non-parametric method that measures whether two distributions are equally large based on the ranks (location in a merged ordered vector) of the samples. By comparing the means of the rank distributions to the null hypothesis that states that the two vectors are equally large, it is possible to infer if the two distributions are equally large. Otherwise, the null hypothesis is rejected and the distributions are not the same, which indicates probable existence of traffic shaping of the flow with stochastic larger values.

The MWU has several key strengths [17] making it particularly useful for detecting traffic shaping. First, it is efficient and accurate when the number of samples is small, since U is calculated precisely using the empirical values measured. Second, unlike the previously suggested [10] Kullback-Leibler (KL) divergence method, which uses the actual measured values, MWU considers ranks, making it is robust to outliers. These can be the result of various traffic events, like packet losses, accidental reordering and short term congestion. Finally, using the relaxed measures for stochastic similarity makes this method resilient against any inaccuracies introduced by the measurement process itself.

2) *Inferring Shaper Type and Parameters*: Once traffic differentiation is detected, we wish to identify the type of shaping performed and estimate the parameters used by the specific shaper, specifically the weights that are assigned to each flow. Therefore, we compare the sent bandwidth with the received one, expecting that the flows, which are sent slightly below the narrowest link capacity are received with a bandwidth determined by the shaper. In real-world deployments, the narrowest link capacity can be estimated using a standard packet dispersion technique [15].

Since the two received flows are distinguishable, it is possible to infer each received flow's bandwidth. If flow shaping is deployed, it will be visible in the ratio between the bandwidths of the received flows. Furthermore, it is possible to determine whether strict priority is employed (one flow dominates the other), or whether WFQ is employed, and obtain the weights.

The downside of this technique is that it is not robust to cross traffic, mainly due to the need to assess the received bandwidth, a task which, as we later show, is not trivial. However, as Sec. IV shows, this method is able to accurately detect shapers when low to moderate cross traffic exists, while having a low false-positive rate. In order to obtain accurate weights when facing a significant amount of cross traffic, we present a more computationally intensive method.

3) *Inferring Weights with Cross Traffic*: We develop another statistical model that can assess the amount of cross

traffic, and compensate for it when a large enough number of probe packets exists. The model assumes a WFQ scheduler with two classes of weights (c_1, c_2) (which are measured as packets per second), having $c_1 \geq c_2$. The total capacity of the bottleneck link C is assigned to these two class, i.e., $C = c_1 + c_2$.

In order to measure the effect of cross traffic, the experiment must be repeated several times, until the variance in the results is sufficiently low. For each experiment j we consider only the packets that are received after packets from both flows are observed, until the last received packet of one of the flows (depicted by the dotted line in Fig. 1). We refer to these packets as the *considered packets*, and denote by $N_{i,j}$ the number of packets belonging to flow i that appear in the considered packets. N_i is the minimum of $N_{i,j}$ over all experiments, i.e., $N_i = \min_j N_{i,j}$.

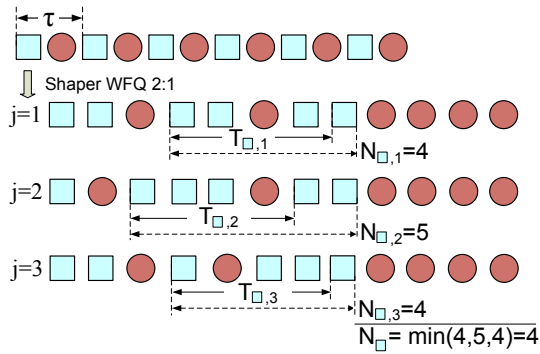


Fig. 1. Illustration of packet train before and after a WFQ shaper, enforcing 2:1 between two flows (squares and round packets). Three experiments return three different observation. The dashed line represents the considered packets $N_{i,j}$ and the solid line marks the time $T_{i,j}$ to receive N_i packets.

Assuming the existence of packet loss between the sender and the receiver, we denote by $n_{i,j}^*$ the total number of packets that were actually sent between the first and last packets in the considered packets. If no packets are lost, then $n_{i,j}^* = N_i$ otherwise $n_{i,j}^* < N_i$. The mean value of $n_{i,j}^*$ over the multiple runs is denoted by n_i^* .

Consider τ to be the time between consecutive packets of the same flow at the sender side. Assuming that flows are not altered until reaching the shaper, this time is equal to the arrival time of packets at the shaper. Therefore, N_i packets are received by the shaper during a time frame of $(n_{i,j}^* - 1)\tau$ seconds. Assuming that cross traffic in flow i arrives to the shaper in a Poisson process with mean rate λ_i , the time T_i to receive N_i packets at the receiver is:

$$T_i = \frac{1}{c_i} [N_i + \mathcal{P}(\lambda_i(n_{i,j}^* - 1)\tau)]$$

Calculating the expectation and variance of the $T_{i,j}$ over multiple experiments, yields the following equations:

$$\mathbb{E}[T_i] = \frac{1}{c_i} [N_i + \lambda_i(n_i^* - 1)\tau] \quad (1)$$

$$\text{Var}[T_i] = \frac{1}{c_i^2} [\lambda_i(n_i^* - 1)\tau] \quad (2)$$

The values of N_i , $\mathbb{E}[T_i]$ and $\text{Var}[T_i]$ are measured by repeating the experiment several times. The bandwidth used to send the flows yields the value of τ , and n_i^* is inferred by tagging the packets (e.g., using sequence number or IP ID). The only two remaining unknowns are c_i and λ_i , which can be computed by solving these two equations.

III. MEASUREMENT FRAMEWORK

In order to evaluate and deploy the proposed methodology, we developed *Packsen*, a framework that is capable of sending flows with high accuracy of bandwidth to end-hosts. *Packsen* is designed with two key concepts. First, the framework must be able to send and measure flows with high bandwidth precision, in order not to introduce bias into the measurement. Second, since packet classification can take many forms, the framework provides a wide range of methods for creating flows.

Packsen includes some key features that enables it to be highly versatile and capable of running in end-hosts. NAT and Firewall traversal is achieved by starting all measurements from the end host. This both ‘‘punches’’ a hole in the Firewall and provides the servers with a routable IP address and ports that can be used for generating upstream flows. Correctly inferring shaping parameters mandates the ability to send and receive flows with bandwidth accuracy, which is achieved by preprocessing the flow packets into memory buffers, and then sending them using raw sockets API. We evaluated this accuracy using 1Gbps link and a standard dual-core Linux machine, and were able to send back-to-back packets almost at line speed and with low jitter. Finally, *Packsen* enables source IP spoofing for testing shaping against a particular source address (e.g., shaping all streams from youtube.com).

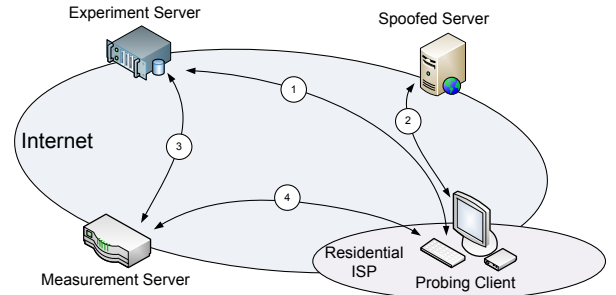


Fig. 2. *Packsen* framework setup

The *Packsen* measurement framework, depicted in Fig. 2, is comprised of a probing client, an experiment server and a set of measurement servers. The experiment starts when a probing client connects to the experiment server and requests an experiment (1). The experiment server selects an experiment from its repository and sends the experiment script to the client. In case the experiment includes a spoofed server, the client creates a connection with the server and extract the needed parameters (2). The client then sends the experiment server the parameters collected from the spoofed server, and starts sniffing traffic for the duration specified in the experiment.

The experiment server selects an available measurement server (not loaded) and sends it the script with the parameters (3). The measurement server receives the experiment script, executes it (4) and notifies the experiment server. In order to bypass NAT and firewalls, the measurement server wait until the client connects before starting to send the flows. The experiment server notifies the client that it should connect to the measurement server. The client then connects to the measurement server and sniffs the packets sent to it for a given duration. It then uploads the results to the experiment server, which stores it in a local database. A web-interface enables users to remotely examine the analyzed results of experiments.

IV. EVALUATION

We evaluate the proposed methodology using a controlled testbed, which allows us to emulate a variety of shaping policies and parameters, as well as cross traffic scenarios. In addition, we use PlanetLab as a way to further evaluate the methods, since the routes between hosts in PlanetLab are longer and the cross traffic is real.

A. Deployment Setup

The testbed is comprised of a traffic shaper connected between the measurement server and a test client, using 1Gbps link towards the server and 100Mbps link towards the test client. The machines are quad-core Xeon servers with 4GB of ram, installed with Debian Linux distribution.

Shaping policies are set using the Linux Traffic Controller [18] to emulate a DSL link. The overall capacity of this link is set to 20Mbps with an added delay of 20ms and a buffer of 100 packets.

Flows are sent using *Packsen* from the measurement server to the client, and are recorded using *tcpdump*. Cross traffic is generated with *iperf*, which simulates regular TCP traffic. We note that this type of cross traffic complicates the analysis of the experiments, since the bandwidth it consumes changes due to the congestion control mechanisms of TCP, however, it enables us to model real-world cross traffic.

The framework was also installed on slightly more than 1000 Planetlab nodes from a large set of different ASes that are spread around the globe. This set was randomly partitioned into source-destination pairs, so that half of the nodes behave like measurement servers and the second half as end-hosts. We note two issues concerning PlanetLab nodes that affect the results – first, PlanetLab nodes are mostly located in academic networks, hence behave differently in respect to traffic shaping than end-users that use commercial ISPs for Internet connection. Second, the *Packsen* framework is installed on a virtualized “slice”, alongside with other applications running on the same servers, which may also result in non-standard behavior. However, the goal of the evaluation using PlanetLab is to validate the methodology when facing real cross traffic, hence this setup is a worst-case scenario.

B. Shaper Detection

In order to evaluate the ability of the MWU stochastic test to detect shapers, we first use the local testbed to emulate flows

with a range of cross traffic. We sent two interleaved TCP flows, a base-line HTTP and a short emulation of BitTorrent, each of 20 packets. Cross traffic is emulated by creating standard TCP flows between the sender and receiver, so that they are classified as the measured BitTorrent flow. We repeated this experiment 100 times, and for each calculated the resulting MWU test value (p-Value). A p-Value below 0.05 (outside the 95% confidence interval) indicates the existence of a shaper and above it indicates that no shaper was observed.

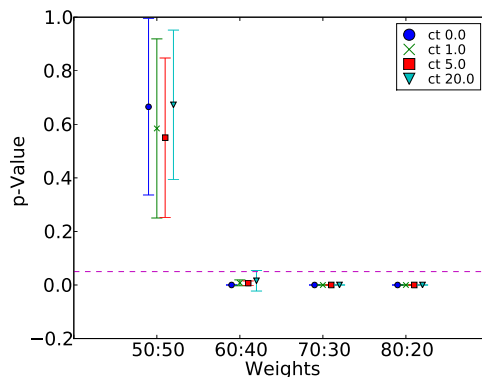


Fig. 3. MWU test results on the local testbed showing mean p-Value using different weights. Error bars mark the variance in the results, showing that the u-test correctly detects shaping (p-Value is below the 0.05 line) using various weights and cross traffic (ct)

Fig. 3 plots the mean value and variance of the p-Value using the different weights and number of cross traffic flows. The figure shows that MWU-test accurately detects the existence (or non-existence in the case of 50:50) of the shaper in any of the scenarios. In the non-shaped case (50:50) the mean p-Value ranges from 0.58 to 0.67, i.e., highly above the 0.05 rejection value. Furthermore, as the discrimination between flows increases, the MWU-test have even more accurate results, even in face of significant cross traffic. The single false-negative case, i.e., a shaper exists but the MWU-test missed it, appeared when the weights were 60:40 (i.e., close to each other) and significant amount of cross traffic exists (20 flows).

Running the experiments on PlanetLab, this time facing real cross traffic and unknown shaping strategies resulted in 99.3% of the paths do not witness a traffic shaper. The remaining 0.7% (4 out of 518 paths) are either true traffic shapers, or more likely represent a false-positive detection, i.e., a path where a shaper was detected when there is actually none.

C. Parameter Estimation

In order to evaluate the ability to estimate the weights assigned to each flow by the shaper, we repeat the above experiment, only this time we use 100 packets per flow. These additional packets are required in order to have a more precise estimation of the bandwidth. Cross traffic is emulated using 5 standard TCP flows, which are again classified as the measured BitTorrent flow.

Fig. 4 plots the received flow bandwidth (calculated using the technique described above) of each class given different weights, with and without cross traffic in the fastest class. Recall that the dotted line marks the link capacity and the arrows point from the sent bandwidth (20Mbps each flow) to the theoretic received bandwidth of each class given a perfect shaper with no cross traffic. Each dot marks a measured bandwidth of the two flows.

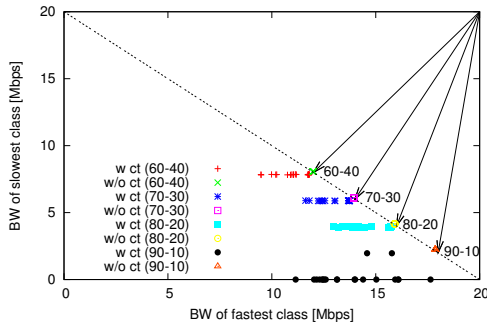


Fig. 4. Comparing received vs. send bandwidth, with cross traffic (w ct) and without it (w/o ct), showing lower accuracy in the presence of cross traffic

As the figure shows, when no cross traffic exists, the received bandwidth in each class is at the boundary of the capacity region, close to the perfect shaper and without any variation between experiments. However, in the presence of cross traffic in only one class, the variance of the measured bandwidth for this class increases, interestingly, without clear dependency in the weights assigned to each flow. When the weights assigned to the flows are significantly different, such as in 90:10, then even when no cross traffic exists, the weight inference is wrong, estimating almost zero weight to the low priority class. This is mainly the result of only a few packets of the slower flow that were received, which leads to incorrect assessment of weights. This indicates the need for much longer flows and the usage of the more accurate statistical inference method in such cases.

Following this, we study the inference error using the statistical model, which involves more computations. We define the relative error as the difference between the weight inferred by the model and the real weight of the faster class.

Fig. 5 presents the relative error using different values of cross traffic and weights. The figure shows that without cross traffic the variance between the experiments is very small resulting in all estimated weights to be within 4% of the real weight. In the presence of cross traffic, the estimated weight is mostly under-estimated, within a reasonable 10% bound. This under estimation is the result of the low variability of cross traffic. As the number of concurrent TCP streams increases, the results of the statistical model improves, mainly since the variance in received packets increases, providing higher accuracy when solving the equations.

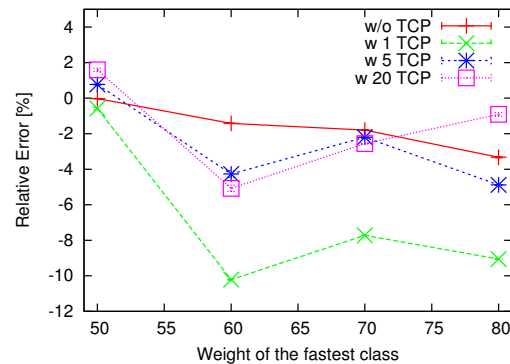


Fig. 5. Relative error between the estimated and real weight of the measured class, showing overall lower relative error with higher cross traffic

V. CONCLUSION

This paper presents three statistical methods for detecting traffic shaping at end-hosts and inferring their parameters. Evaluating the validity and applicability of these methods is performed using *Packsen*, a framework that provides easy flow generation, accurate send and receive bandwidths and a wide range of options for traversing firewalls, NATs and spoofing packets. The methods are shown to provide accurate results and are robust to the presence of cross traffic.

REFERENCES

- [1] M. Dischinger, A. Haeberlen, K. P. Gummadi, and S. Saroiu, "Characterizing Residential Broadband Networks," in *IMC*, Oct 2007.
- [2] Internet Study 2008/2009, "http://www.ipoque.com/resources/internet-studies/internet-study-2008_2009."
- [3] R. Beverly, S. Bauer, and A. Berger, "The Internet is not a big truck: Toward quantifying network neutrality," in *PAM*, 2007, pp. 135–144.
- [4] M. Dischinger, M. Marcon, and S. Guha, "Glasnost: Enabling end users to detect traffic differentiation," in *NSDI*, Oct 2010.
- [5] Measurement Lab, "<http://www.measurementlab.net/>."
- [6] The ICSI Netalyzer, "<http://netalyzer.icsi.berkeley.edu>."
- [7] M. Dischinger, A. Mislove, A. Haeberlen, and K. P. Gummadi, "Detecting BitTorrent Blocking," in *IMC*, Vouliagmeni, Greece, October 2008.
- [8] M. Tariq, M. Motiwala, and N. Feamster, "NANO: Network Access Neutrality Observatory," in *HotNets*, Calgary, Canada, October 2008.
- [9] Y. Zhang, Z. M. Mao, and M. Zhang, "Detecting Traffic Differences in Backbone ISPs with NetPolice," in *IMC*, 2009.
- [10] P. Kanuparth and C. Dovrolis, "DiffProbe: Detecting ISP service discrimination," in *Infocom*, 2010.
- [11] M. Coates, A. Hero, R. Nowak, and B. Yu, "Internet tomography," *IEEE Signal Processing Magazine*, vol. 19, pp. 47–65, 2002.
- [12] 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 CCR*, vol. 33, no. 3, July 2003.
- [13] J. Turner, "New directions in communications (or which way to the information age?)," *IEEE Communications Magazine*, vol. 24, no. 10, pp. 8–15, 1986.
- [14] M. R., M. Zhang, P. L., and P. V., "Uncovering Performance Differences in Backbone ISPs with NetDiff," in *NSDI*, 2008.
- [15] A. B. Downey, "Using pathchar to estimate Internet link characteristics," in *SIGMETRICS*. ACM, 1999, pp. 222–223.
- [16] H. B. Mann and D. R. Whitney, "On a test of whether one of two random variables is stochastically larger than the other," *The Annals of Mathematical Statistics*, vol. 18, no. 1, pp. 50–60, 1947.
- [17] M. Hollander and D. A. Wolfe, *Nonparametric Statistical Methods, 2nd Edition*. Wiley-Interscience, January 1999.
- [18] Linux Advanced Routing and Traffic Control, "<http://lartc.org/>."