Virtual Speed Test: an AP Tool for Passive Analysis of Wireless LANs

Peshal Nayak

Samsung Research America, USA

Edward W. Knightly

Rice University, USA

Abstract

Internet speed tests assess end-to-end network performance by measuring throughput for 10's of MB of TCP uploads and downloads. While such tests provide valuable insights into network health, they are of little use to network administrators since (1) the results are only available on the client that performs the test and (2) the tests can saturate the network, increasing load and worsening performance for other clients. In this paper, we present virtual speed test, a measurement based framework that enables an AP to estimate speed test results for any of its associated clients without any special-purpose probing, with zero end-user co-operation and purely based on passively observable parameters at the AP. We implemented virtual speed test using commodity hardware, deployed it in office and residential environments, and conducted measurements spanning multiple days having different network loads and channel conditions. Overall, virtual speed test has mean estimation error less than 6% compared to ground truth speed tests, yet with zero overhead, and outcomes available at the AP.

1. Introduction

TCP speed tests are an end-to-end test of network health and are available via a plethora of online apps [1, 2, 3]. As a part of the measurement process, the client performs an active TCP download and an active TCP upload to a server to measure the download and upload TCP throughput respectively. Since more than 80% of the modern day internet traffic is transmitted over TCP [4], the performance of numerous online applications is crucially dependent on the maximum TCP throughput achievable over an underlying network path.

Email address: peshalnayak@gmail.com (Peshal Nayak)

If the speed test happens from a nearby server (*i.e.*, a server with minimum possible latency to the AP), the WLAN becomes a key part of the end-to-end path. Consequently, the results become valuable to the network manager to assess WLAN performance and make decisions on network infrastructure alterations to improve the quality of service experienced by the end user.[1] However, these results can only be seen by the end user and are unavailable to the administrator without seeking end user co-operation. Moreover, regularly performing these speed tests impose additional traffic load on the network and hence doing so can potentially disrupt user traffic and drain the battery of mobile devices.

In this paper, we make the following contributions:

First, we present a framework that enables an AP to estimate the outcome of a speed test, *i.e.*, the upload and download TCP throughputs that any of its associated STAs should obtain from a nearby server, yet, without any special-purpose probing, with zero cooperation of endpoints (*i.e.*, the server and the client), and solely based on measurements that are passively observable at the AP. We call our measurement based framework *virtual speed test*. Virtual speed test employs a novel L2 edge TCP model to perform throughput estimation. The key challenge for the AP to estimate these inherently bi-directional, end-to-end and layer-4 throughputs, is that the AP only has a limited view of the network. Since the AP is unaware of the presence of hidden terminals, interference from neighboring BSS to the STAs, etc. (which affect the STA's queuing delays, NAV timers and packet retransmissions), the AP cannot estimate how long it takes a STA to successfully transmit after it starts to attempt. Our design is motivated by the fact that since the WLAN is the final hop for any TCP segment directed towards a STA, this duration can also be estimated by measuring the delay incurred between the transmission of a TCP segment on the downlink to the reception of the corresponding TCP ACK on the uplink from the STA. This TCP segment, therefore, can belong to any TCP flow (e.g., a Netflix video stream) and need not be a part of a flow from a nearby server. To carry out these measurements, the AP must identify TCP flows. To this end, we leverage TCP's inherent bi-directionality and packet size signatures to spot TCP flows. Specifically the fact that TCP flows involve TCP segment traversing on the forward path and small sized TCP ACKs on the reverse path enables the AP to identify these flows and perform its measurements.

Second, we experimentally validate our framework via an implementation on commodity hardware followed by extensive field trails. We deploy a virtual speed test enabled AP (VST AP) in two environments: an office located inside a university building and an apartment in a residential complex. The VST AP is deployed in the office for a period of 2 days and in the apartment for a period of 7 days. Both deployment settings are characterized by interference from non-BSS devices co-existing in the same frequency band, human mobility and link diversity with respect to signal propagation (*i.e.*, LoS vs non-LoS paths) and supported PHY rates. The office and the residential scenario cover a total of 36 and 49 topologies respectively with a varying number of STAs. Overall, the VST AP observes a total of 113,047 TCP flows across both deployments. These TCP flows result from multiple applications running on end devices such as video streaming, music streaming, pdf downloads and email activities. Virtual speed test demonstrates

[1]These modifications could be either software upgrades to a device to fix a temporary network problem, a hardware upgrade to provision additional capacity or network optimization in the form of channelization strategies to alter interference, prioritization strategies, etc.

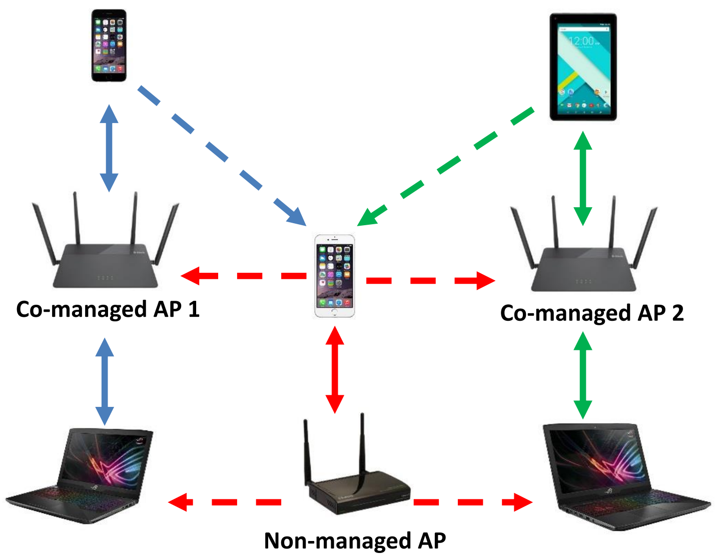


Figure 1: Enterprise WLAN scenario: bold lines indicate connectivity while dotted lines indicate interference

a high level of estimation accuracy with an average estimation error under 6% for both upload and download speed estimation.

Finally, we implement virtual speed test into ns-3's source code and perform extensive simulations to investigate operating conditions beyond those encountered in our field trials. Our simulations also concur with our field trial conclusions demonstrating estimation errors below 5%.

To the best of our knowledge, our framework is the first to estimate both upload and download TCP throughputs of STAs in the network by using passive measurement metrics at only the access point, *i.e.*, without any active probing, additional hardware infrastructure or user participation.

The remainder of this paper is organized as follows. In Sec. 2, we describe our target network scenario along with the necessary background on speed tests and a high level problem formulation. In Sec. 3 we present the L2 edge TCP model and describe its parameter details while techniques to estimate these parameters are presented in Sec. 4. Our commodity hardware implementation, experimental setup and field test details are presented in Sec. 5 and the corresponding results are reported in Sec. 6. We present related work in Sec. 7 and conclude in Sec. 8.

2. Virtual speed test: scenario description and problem formulation

2.1. Enterprise WLAN setup

We consider an enterprise WLAN environment such as illustrated in Fig. 1. As depicted, the network comprises of multiple APs. While the network might use channelization, for ease of exposition we will consider only APs with at least partially overlapping channels such that they can potentially interfere with each other. Moreover, we consider that in addition to the managed infrastructure, there may be one or more non-managed

WLANs that may be interfering. Such WLANs could correspond to an LTE hot spot or a neighboring WLAN under different administrative control.

Ideally, all such networks would have sufficient physical separation to enable full spatial reuse for each AP (*i.e.*, simultaneous transmission for each network). However, as depicted, the unwanted interconnectivity creates interference and contention among nodes. Moreover, inter-node connectivity can form a complex relationship: while all STAs are necessarily connected to the APs that they associate with, a particular STA may or may not be in range of other APs. Likewise, STAs might be "hidden" from each other or mutually in range. It is further possible that a STA is in range of other APs which are not in range of the AP that is serving it. The interference and contention possibilities are further compounded by the need to consider both downlink transmissions (AP to STA), uplink transmissions (STA to AP), and mixes.

We do not make any assumptions about the PHY layer capabilities of the AP or the STAs. For instance, the AP may have advanced physical layer capabilities such as multi-user MIMO. Likewise, the AP can have any channelization strategy, *e.g.*, dynamically bonding channels to 80 MHz as available.

The STAs can have traffic going either uplink and/or downlink which can either be TCP or UDP. Furthermore, the traffic may have varying levels of burstiness which can introduce idle times at the AP and/or the STA.

2.2. Background on TCP upload and download speed test

Speed tests measure the upload and download TCP throughput that a client would get from a server on the internet. If the speed test happens from a nearby server, the WLAN becomes a key part of this end-to-end path and the network manager can use these results to assess WLAN performance. For the remainder of the paper, we will only focus on speed tests that happen from a nearby server. A speed test is user initiated and the results are visible to the user at the end of the measurement. Speed tests primarily consist of two phases: a setup phase during which the speed test parameters are configured and a measurement phase which involves an active TCP upload and download.

Setup phase. The setup phase begins with a server selection process which can either be manual or app driven. If this is app driven, a server is selected by probing a pool of available servers and a download or an upload session is established with it. Typically the server is selected such that the backbone delay between the server and the AP is as minimum as possible to ensure a maximum TCP throughput [5]. An ideal case would be one in which the selected server is in the same LAN as the AP since this would completely eliminate the effect of backbone delay on the measurements. Since the goal is to measure the maximum TCP throughput, while running a speed test, a STA is recommended to turn off other applications. Next, the client and server side TCP parameters are configured. The exact mechanism used for performing this configuration differs from one speed test application to another. A commonly used mechanism is to conduct a test download and a test upload from the STA. For instance, in the case of Ookla speed test, the STA initially downloads or uploads a small sized file to estimate an initial throughput. Following this initial phase, the STA adjusts the file size, buffer size and number of parallel TCP flows (limited to maximum of 8) to maximize the network connection usage while preventing congestion during the measurement phase [6].

Measurement phase. As shown in Fig. 2, the measurement phase consists of two sessions: an upload session and a download session. A vast majority of the speed test

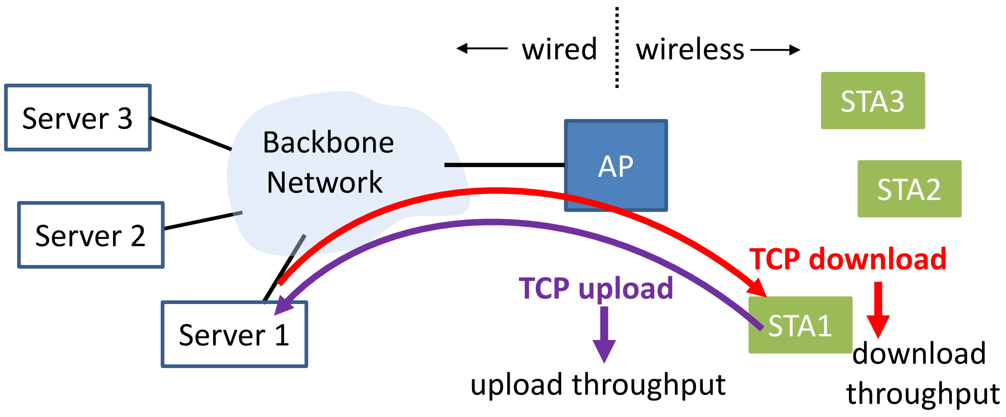


Figure 2: Illustration of an upload and download speed test: here STA 1 performs a speed test. Server 1 is selected from a pool of servers consisting of server 1, 2 and 3. Here server 1 has minimum backbone delay to the AP.

apps available online follow a flooding based mechanism in the upload and download sessions [7]. A flooding mechanism involves establishment of several parallel TCP flows between the server and a STA with a calculation of aggregate throughput across all the flows. This ensures that the results obtained are robust to factors such as a small TCP window size [8] (due to, for instance, loss of a TCP segment) or any bounds on the maximum window size [9] (for instance, due to a small receive window size advertised by the receiver) which could potentially make the total number of circulating TCP segments the prime bottleneck. The number of parallel flows to be established is determined in the setup phase. During the upload phase, a STA performs an active upload to the selected server and measures the TCP throughput by averaging the total data transmitted end-to-end over the total time taken. During the download phase, the STA performs an active download and measures throughput in a similar fashion.

2.3. Virtual Speed Test: High level problem definition

Analogous to online speed tests, our goal is to realize a virtual speed test and enable an AP to estimate the TCP download and upload throughput that a STA can achieve from a nearby server. As described in our network scenario, an AP can have an arbitrary number of STAs associated with it and the AP should be able to estimate the throughput for any of the associated STAs. Note that the STA does not perform the actual speed test.

The AP is required to make the prediction using only passively collected information available on the AP side whereas no reports are available from STAs and out-of-network APs. Further, we consider that no additional commands can be required of STAs, *e.g.*, STAs cannot be requested to send packets for testing purposes. Moreover, STAs cannot be re- quested to download special purpose software or report STA- side measurements. Instead, we consider that by leveraging AP side observables, the AP can estimate the following metrics [10].

Aggregate AP metrics. We consider that the AP can measure the airtime usage due to transmission and reception, defer time, contention time, idle time (no backlogged downlink traffic) as well as byte counts for downlink and uplink frames.

Per-STA metrics. Likewise, while the STAs do not report STA-side statistics, the AP can estimate per-STA metrics observable at the AP such as the following: downlink and uplink MCS and PHY parameters including use of advanced PHY features such as channel bonding, uplink RSSI and SNR, spatial multiplexing, and multi-user transmission and downlink retransmission statistics.

Non-associated device metrics. Lastly, the AP may be in range of a number of non-associated 802.11 devices that are transmitting on a different BSS. When the AP is forced to defer to a non-BSS device, it can record interferer air time consumption.

While the above might appear to be an exhaustive set of information for performance characterization, there are a number of STA-side statistics that remain unknown to the AP. For instance, the AP does not know the STA's idle times or the STA's defer times due to NAV especially when the STA is deferring to a non-BSS device. Since our network scenario considers a complex inter-node connectivity leading to a rise of hidden terminals, inter-cell interference, etc., these parameters cannot be directly calculated based on aggregate AP or STA metrics mentioned above. However, the throughputs that we want the AP to estimate are inherently bi-directional, end-to-end and layer-4 and can indeed be degraded by the above factors. To this end, we infer the impact of these unknowns using the above AP side observables with the help of techniques which we describe in Section 4.

3. L2 edge TCP model

3.1. Assumptions for mathematical analysis

Our objective is to empower an AP to estimate the upload and download throughputs that a STA would obtain if it performs a speed test. Virtual speed test is powered by a L2 edge TCP model that enables estimation of download and upload TCP throughput when supplemented with AP side observables. Here we state the key assumptions that we make to capture important aspects of the aforementioned speed test setup and measurement phases in our model.

As mentioned previously, in the measurement phase, multiple TCP flows are initiated between the server and the client. Recall that the reasoning for doing so is two fold: first, to ensure that the measurement phase is not bottlenecked by the number of circulating TCP segments, and second to prune the impact of TCP's slow start phase on the obtained results. Instead, we capture this in our analysis by representing the packet flow dynamics by a single long lived TCP flow with a maximum congestion window size of W_{m}. We assume that W_{m} is large enough so that there are a sufficient number of TCP segments circulating in the network. We hereby refer to this flow as the speed test flow and the STA under consideration as the target STA.

The server selection process in the setup phase selects a server with minimum latency to the AP to reduce the impact of backbone elements on the measured results. To account for this in our analysis, we consider congestion and delays on the backbone for the speed test flow segments as second order effects and ignore them. We further assume that no speed test segments are permanently lost in the network. This is not to say that collisions or packet errors do not occur on the wireless channel. Rather, packets lost on the wireless channel are locally retransmitted by the MAC layer and we do account for these collisions and retransmissions in our analysis. Also recall that the parameters of the TCP flow used during the speed test are adjusted by the STA based on an initial measurement performed to ensure that TCP does not drive the network into congestion.

We further assume that devices in the WLAN are work conserving, *i.e.* whenever a packet is available in the queue, the device will begin to contend for channel access. While limited to first order effects, these assumptions enable us to derive simple and instructive throughput expressions that will nonetheless lead to accurate results (as will be shown in later sections).

3.2. Virtual end point representation

The discussions in this sub-section are mainly in the context of a download speed test. However, the arguments and explanation are applicable to upload speed tests as well and will be generalized later.

To estimate TCP throughput from nearby server it is important to understand the packet flows that exist in the WLAN. In our network scenario, we do not impose any restrictions on the traffic flows of non-target STAs and they may be UDP and/or TCP traffic going on the downlink and/or the uplink. Note that the number of these flows per non-target STA can also be variable and differ from STA to STA. Since we make no assumptions about the type or number of flows for other STAs in the network, it is not possible to state precisely the inputs for a queuing model. We remark that a majority of TCP models for WiFi are hindered by a requirement for AP side knowledge of network topology. Not being restricted by this requirement is instead vital to the realization of virtual speed test.

For our analysis, we divide all flows into two categories: speed test flow and non-speed test flow. A speed test flow packet can either be a TCP segment or a TCP ACK. First, we analyze the speed test flow by considering the journey of a speed test flow segment from the server to the target STA. On the forward path, a TCP segment experiences delays on the queues of devices on the backbone.[2] When the packet enters the queue at the AP, it will encounter another delay before reaching the head of the queue, part of which arises from the AP serving non-speed test flow packets. We denote the average amount of time the AP spends on non-speed test flow packets prior to serving a speed test flow packet by Q. Upon reaching the head of the queue, the AP begins to contend to access the channel. It is possible that as the AP counts down, the target STA or another STA or another AP wins the channel, causing the AP to defer. It is also possible that a transmission from the AP fails either due to collision or poor channel quality, forcing the AP to double its contention window size and re-contend and transmit (with the same or adapted data rates[3]). We denote the mean time the AP takes to win the channel prior to a successful transmission by d_{access} as shown in Fig. 3. Observe that the value of this parameter can vary depending on the STA being considered as the target STA. The average amount of time to transmit the TCP segment is represented by d_{tx}. This includes any MAC and physical layer overhead, MAC frame transmission time, all interframe spacings and the MAC layer acknowledgement. Just like the TCP segment, the TCP ACK will also face a similar journey back to the server. The terms u_{access} and u_{tx} are defined in a similar manner.

[2]A example cause of these delays is that due to cross traffic sharing a common queue on the backbone with the TCP segment

[3]Generally, the exact rate adaptation policy will depend on a particular vendor implementation

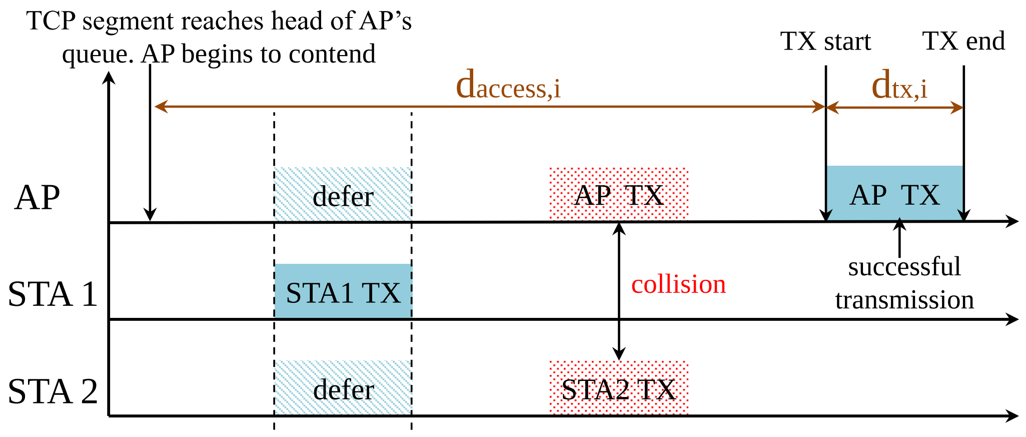


Figure 3: Example timeline of a downlink transmission to depict d_{access} and d_{tx}. $d_{\text{access,i}}$ and $d_{\text{tx,i}}$ denote the values for the i^{th} downlink transmission. d_{access} and d_{tx} denote the mean values.

The transmission time of all non-speed test flow packets going on the uplink is captured in the 'access' term of either speed test or non-speed test flow packets going on the downlink or the 'access' term of the speed test flow packets going on the uplink.

For our analysis, we represent the WLAN (AP and STAs) as a virtual end-point consisting of two queues: a forward queue and a reverse queue. For now, let us assume that the non-speed test flows are non-existent and that only the speed test flow packets exist in the WLAN (we subsume the impact of non-speed test flow packets into the model parameters later). With this consideration, we can treat the virtual end point as a black box replacing the WLAN that runs a speed test. The socket level TCP client (not to be confused with the physical STA) runs on the virtual end point itself as shown in Fig. 4. We can think of TCP segments and TCP ACKs as jobs circulating in the network. The service time of each job will be a sum of its 'access' term and its 'tx' term. E.g., for jobs in the forward queue, the service time will be a sum of d_{access} and d_{tx}. Since we account for the 'tx' term in the service time itself, the jobs themselves become indistinguishable. As we have not yet subsumed the effect of non-speed test flows, the throughput of the virtual end point is not the same as that of the target STA in our WLAN. In our second step, we account for the impact of non-speed test flow packets on the throughput of this system by inflating the service times of each queue to account for the non-speed test flow packets. In essence, this inflation makes the effective speed of each server as seen by the speed test flow packet in the virtual end point system the same as that in the original system where some server time would have been consumed by non-speed test flow packets as well. Consequently, on the forward queue, the service time will be inflated by Q. However, since the target STA has no other uplink traffic while performing a speed test, the reverse queue service time requires no inflation. Similarly, we can subsume the impact of cross traffic on the backbone queues into their respective service times.

3.3. Throughput analysis

To analyze the throughput of the network shown in Fig. 4, we consider two cases. First we consider a case wherein TCP performs no ACK thinning. Consequently, in this case, each TCP segment received by the STA results in the generation of a TCP ACK.

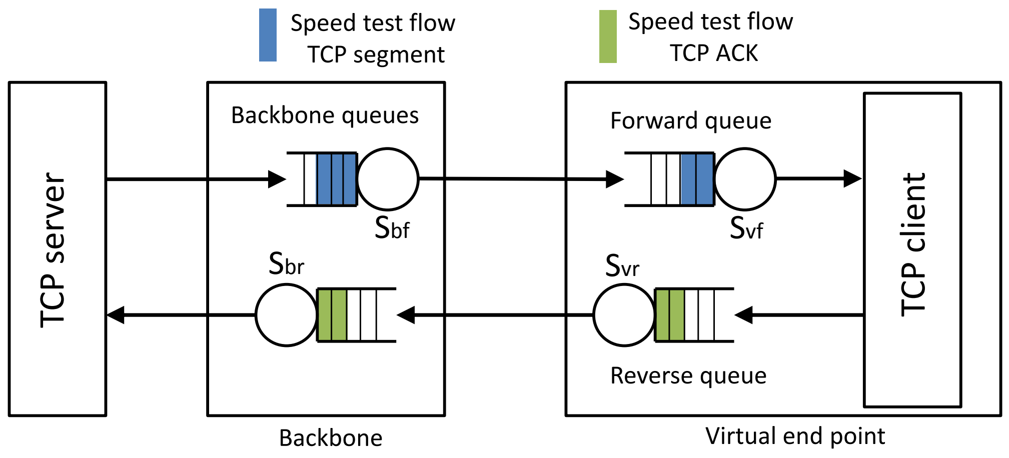


Figure 4: WLAN representation as a virtual end point consisting of a forward and reverse queue. The TCP client here refers to the socket level client and is not to be confused with the physical STA itself.

Next, we generalize this to account for the case of ACK thinning with an ACK thinning ratio of n. In this case, the client generates a TCP ACK following the receipt of every n^{th} TCP segment.

3.3.1. No TCP ACK thinning

Ignoring the initial transient stage during which TCP's window size grows, the speed test flow will reach a steady state wherein TCP operates at W_{m}. Consequently, the number of packets that are contained in the speed test flow, which can either be TCP segments or TCP ACKs, remain constant and the system behaves as closed queuing network with tandem servers and a constant number of jobs circulating inside it.

Based on the aforementioned notations, the mean service times for each individual queue in the virtual end point will be given by:

$$S_{vf} = d_{\text{access}} + d_{\text{tx}} + Q \tag{1}$$

$$S_{vr} = u_{\text{access}} + u_{\text{tx}} \tag{2}$$

Let $S = S_{bf} + S_{br} + S_{vf} + S_{vr}$, $S_{max} = max(S_{bf}, S_{br}, S_{vf}, S_{vr})$ and θ denote the throughput in terms of jobs per second. It can be shown [11] that

$$\theta \leq min\left(\frac{W_{\text{m}}}{S}, \frac{1}{S_{max}}\right) \tag{3}$$

where $\frac{W_{\text{m}}}{S}$ is an asymptotic bound for small values of W_{m} and $\frac{1}{S_{max}}$ acts as an asymptotic bound for large values of W_{m}. The cases of small and large here are relative to a critical value W_{m}^{*} which is the point at which the asymptotes cross each other. Consequently,

$$W_{\text{m}}^{*} = \frac{S}{S_{max}} \tag{4}$$

To understand the physical relevance of the two components of Eq. (3), let us consider two extreme case scenarios. Let us assume that $W_{\text{m}} = 1$ which makes the number of jobs circulating in Fig. 4 the botteneck. The throughput, therefore, is given by $\frac{W_{\mathrm{m}}}{S}$. On the other extreme, if W_{m} is sufficiently large (again large as compared to W^*_{m}) to not bottleneck the system, then the slowest queue acts as a bottleneck. In this case the slowest queue always remains busy and in accordance with the utilization law, $\theta = \frac{1}{S_{max}}$.

Recall that due to the server selection process, S_{br} and S_{bf} are not the bottleneck in the system. To understand the typical values that W^*_{m} can take, let us consider the critical point wherein $S_{br} = S_{bf} \sim max(S_{vf}, S_{vr})$. Substituting in Eq. (4), we will get $W^*_{\mathrm{m}} = \frac{2*(S_{vf}+S_{vr})}{max(S_{vf},S_{vr})}$. As shown in Appendix A, maximum value of W^*_{m} occurs when $S_{vf} = S_{vr}$ and thus $\max(W^*_{\mathrm{m}}) = 4$. In practice, $W_{\mathrm{m}} \gg 4$ and consequently, we can see that $\theta \leq \frac{1}{S_{max}}$ will act as a asymptotic bound on the values of θ. In fact, we find in our experimental evaluation that for a typical speed test, the values of W_{m} is extremely large as compared to 4 and θ will tend to the bound yielding

$$\theta \sim \frac{1}{S_{max}}. \tag{5}$$

The throughput in Eq. (5) is expressed in terms of jobs/sec. To obtain the TCP throughput in terms of bits/sec, we need only multiply the right hand side either by $\mathbb{E}[\text{TCP segment size}] * F_{\mathrm{AP}}$ (in the case of the download speed test) where F_{AP} denotes the average number of frames transmitted by the AP in a single downlink transmission to the target STA. In this case, the values of S_{vf} and S_{vr} have to account for aggregated frame transmissions. These multiple frames may be transmitted using frame aggregation in single stream transmissions (e.g., SISO) or multi-stream transmissions (e.g., MIMO) or a combination of both frame aggregation and multi-stream transmissions. For the case of upload speed test, we use F_{STA} instead.

3.3.2. TCP ACK thinning

Now, we consider the more general case of TCP ACK thinning: for an ACK thinning ratio of n, we can view a maximum of only $\frac{W_{\mathrm{m}}}{n}$ TCP segments circulating in the system and the remaining segments can again be accounted for by further inflating the service times of each of the devices (just as for non-speed test flows). Consequently, in the absence of frame aggregation, the service times of both the forward and reverse queue in the virtual end point will stretch by an amount equal to $(n-1)\times(d_{\mathrm{access}} + d_{\mathrm{tx}} + Q)$ for the case of the download speed test. Here we inflate the service time of the reverse queue to account for the fact that the TCP ACK is not generated until the n^{th} TCP segment is received. The numerator will also inflate by a factor of n to account for the shrinking of the total number of TCP segments. For the upload speed test, the service times will stretch by $(n-1)\times(u_{\mathrm{access}} + u_{\mathrm{tx}})$. However, in the presence of frame aggregation, such an inflation is not necessary since the STA receives all the TCP segments in a single aggregated frame and there is no additional delay in generation of a TCP ACK. We emphasize that this is possible since typical ACK thinning ratios of TCP are much smaller than the frame aggregation levels allowed under 802.11 [12, 13]. In summary,

$$\theta_{\mathrm{dl}} = \frac{\mathbb{E}[\text{TCP segment size}] \times F_{\mathrm{AP}}}{max(S_{vf}, S_{vr})} \tag{6}$$

$$\theta_{\mathrm{ul}} = \frac{\mathbb{E}[\text{TCP segment size}] \times F_{\mathrm{STA}}}{max(S_{vf}, S_{vr})} \tag{7}$$

where we denote θ_{dl} and θ_{ul} as the download and upload TCP throughputs respectively.

Note that while calculating S_{vf} and S_{vr} for Eq. (6), d_{tx} will be the average time to transmit F_{AP} number of TCP segments at the AP's datarate and u_{tx} will be the average time to transmit F_{STA} number of TCP ACKs at the target STA's datarate. In Eq. (7), this will be reversed since the target STA is now the one transmitting TCP segments and the AP is the one transmitting the TCP ACK. S_{vf} and S_{vr} further vary depending on which STA is chosen as the target STA. Consequently, the AP has to estimate these two parameters with respect to the particular STA that is chosen as the target STA.

We remark that while the L2 edge TCP model needs to be supplemented with measured values from the AP, it is not restricted by a requirement for an AP side knowledge of inter-node connectivity or an assumption on network traffic characteristics.

4. Obtaining AP-side measurements

In this section, we show how the AP can measure all of the parameters required for the above model, thereby enabling a dynamic AP-side speed test estimate for each STA.

4.1. AP-side estimation problem

We observe that Eq. (6) and (7) are independent of S_{br} and S_{bf}. To estimate θ_{dl} and θ_{ul} at the AP, the key challenge is computation of S_{vr}, as the remaining parameters are based on common AP side observables described in Sec 2.3. Recall from Eq. (2) that S_{vr} is composed of u_{tx} and u_{access}. While the average uplink transmission time u_{tx} is known to the AP via per-STA metrics, the uplink access time u_{access} is known only at the STA side. Let $t^{U,i}_{\text{hq}}$ denote the time at which the i^{th} *uplink* packet reaches the head of the STA's queue, $t^{U,i}_{\text{ts}}$ denote the start time corresponding to the successful transmission of this packet and $t^{U,i}_{\text{te}}$ denote the end time of this packet transmission. By definition, $u_{\text{access}} = \mathbb{E}[(t^{U,i}_{\text{ts}} - t^{U,i}_{\text{hq}})]$. While the AP can observe $t^{U,i}_{\text{ts}}$ for any uplink transmission, $t^{U,i}_{\text{hq}}$ remains unknown. If the STA is assumed to be fully backlogged, the end time of the previous transmission can be approximated to be the time when the next packet reached the head of the queue. However, STA backlog is user activity dependent and is not known to the AP. As a result, the AP cannot estimate u_{access} by a simple observation of packets received on the uplink.

4.2. Snooped handshakes for estimation of uplink access time

Suppose that the client is performing a TCP download from a server (e.g., streaming a Netflix video). This can be any server on the internet with any backbone delay to the AP. The client will attempt to return a TCP ACK as fast as possible after reception of the corresponding TCP segment. This TCP ACK is "data" at layer 2. For now, consider a case where there are no other flows on the uplink from the target STA and no ACK thinning. Since the WLAN is the final hop for the TCP segment, upon reception of a TCP segment, *i.e.*, at the end of the AP's successful downlink transmission (denoted by $t^{D,i}_{\text{te}}$), the STA has the corresponding TCP ACK and begins to contend. Consequently, in this case, $t^{U,i}_{\text{hq}} = t^{D,i}_{\text{te}}$ and thus the AP will have inferred a parameter that is not directly observable. In essence, the delay incurred between the transmission of the segment to the reception of the TCP ACK enables the AP measure how long it takes the STA to successfully transmit after it starts to attempt. Thus, our general approach is to selectively sample TCP data-ACK handshakes from any TCP download performed by the target STA and use them to drive a measurement based prediction of θ_{dl} and θ_{ul}. We refer to such TCP flows as *snooped flows*.

This can be generalized under a flow hypothesis (*i.e.*, knowing that a given flow on the downlink is a TCP flow) by the following two cases.

ACK queuing. This case occurs when the target STA has other uplink flows whose packets get queued prior to the TCP ACK. Consequently, in such scenarios, $t_{\text{hq}}^{U,i} = t_{\text{te}}^{U,i-1}$. In such cases, we abuse the term $t_{\text{te}}^{U,i-1}$ to refer to the end time of transmission of the immediately preceding uplink packet.

ACK immediate. However, if the target STA has no other uplink flow, it begins to contend as soon as the TCP ACK is queued. Consequently, $t_{\text{hq}}^{U,i} = t_{\text{te}}^{D,i*n}$ where the superscript 'D' refers to a downlink transmission.

4.3. TCP flow inference

Because the layer four handshake is needed to estimate u_{access}, it is crucial to identify this handshake at the AP, which does not have layer four visibility. To this end, we employ IP addresses and size signatures as follows.

IP address signature. Due to the inherent bi-directionality of TCP, the source and destination addresses for TCP segments traversing on the forward path are swapped for the corresponding TCP ACKs on the reverse path. This key factor enables us to distinguish individual TCP flows and separate them from the remainder of the downlink and uplink traffic.

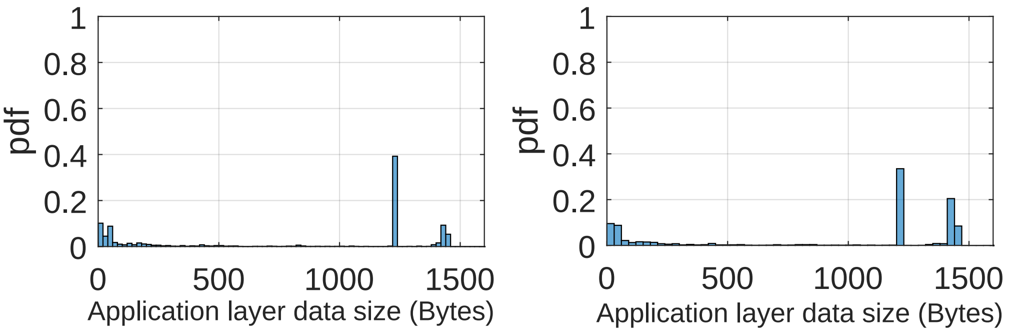
Packet size signature. Although the above signature enables identification of a bidirectional flow, it does not aid in spotting the forward and reverse paths distinctly. While the size of TCP segments on the forward path may fluctuate during the course of a download, the reverse path is characterized by small TCP ACKs whose size remains fixed during the entire duration of the flow. Typically a TCP ACK is 20 bytes long [14]. Having distinctly identified the forward and reverse paths, the AP can employ the u_{access} estimation process described in the previous sub-section.

5. Experimental Methodology

5.1. Testbed characterization

Platform specifications. Our AP runs on a Linux operating system and is factory installed with 32 GB DDR4 SO-DIMM RAM, 2.4 GHz dual core CPU with slots for USB, HDMI and a Gigabit LAN port. The AP is equipped with a Ralink RT3070 off-the-shelf WiFi chipset. The radio card supports IEEE 802.11b/g/n utilizing up to 40 MHz bandwidth and a peak PHY rate of 300 Mbps. The STAs are a mix of portable laptops running on either Windows or Linux OS whose network interface card supports 802.11b/g/n as well. This AP is hereby referred to as the VST (virtual speed test) AP.

Per-packet statistics. We build APIs that enable the acquisition of a number of per packet statistics at the access point as well as the STAs. While a rich raw characterization of each packet is available at all the devices, we only feed the AP side packet timestamps, source and destination IP addresses, frame sizes, and PHY rates into the L2 edge TCP model for throughput estimation. Nonetheless, the remaining statistics



(a) Office (b) Residential

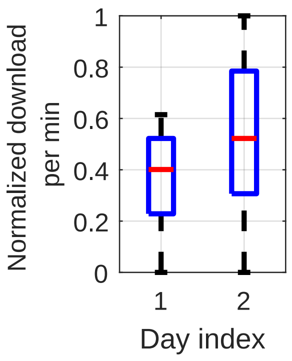
Figure 5: pdf of application layer data sizes encountered in the office and residential deployment.

enable us to characterize the operating environment as will be described later. As described earlier, the parameter estimation methodologies employ packet timestamps as a part of the computation process. While the absolute value of these timestamps can be infected with system dependent offsets, their post-subtraction residue becomes negligible as they have a low second moment. The timestamps on the VST AP are available on a nanosecond granularity. However, timestamp resolution has hardware dependency and can vary based on processor architecture, system clock, operating system time stamping policies, etc. Since the packet timeline operates on a scale greater than microsecond, even a microsecond resolution - a capability available on many hardware platforms [15], is sufficient to capture the time domain information between packets necessary to estimate u_{access}.

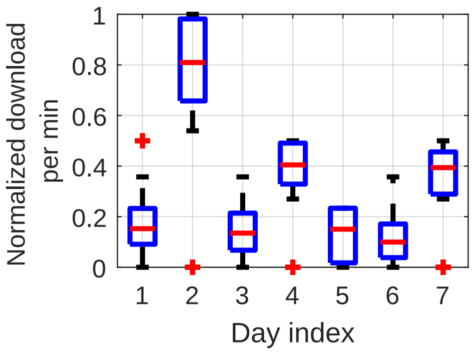
5.2. Field trails

To understand the estimation accuracy of virtual speed test, we deploy the VST AP and STAs in two environments.

Deployment description. The first deployment is in a 5m x 3m student office located in a 3 storied building on a University campus. In this deployment, the VST AP and the STAs co-exist with a University administered enterprise network and 2 APs deployed in nearby student offices. Here the VST AP has 6 STAs associated with it. We deploy a second network in a 3 storied residential building primarily consisting of apartments with 1 or 2 bedrooms per unit. This residential network consists of 7 STAs. These devices co-exist with 4 APs from neighboring apartments. The VST AP deployed in the office performs measurements for a period of two days whereas the residential scenario measurements are carried out for a period of one week. During the entire duration of deployment, the VST AP observed a total of 113,047 snooped flows. These flows are the result of multiple applications running on end devices such as video streaming (via YouTube), music streaming (via Pandora), pdf downloads (via IEEE Xplore) and sending and receiving emails (via Gmail). The STAs use Mozilla Firefox web browser for performing these online activities. The traffic flows consist of single application traffic where each STA runs only one web application as well as a mix of web applications running concurrently. Aside from the applications mentioned above, some of the STAs have Dropbox installed on them which occasionally adds to the uplink traffic from these devices in addition to that generated by email activities. We do not suppress any control packet transmissions at any of the layers.



(a) Office



(b) Residential

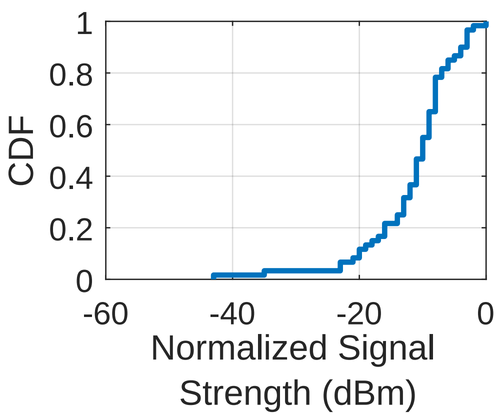
Figure 6: Variation in the amount of data downloaded per minute in the office and residential deployment. The values are normalized with respect to the overall maximum encountered in that particular deployment.

The ground truth values of the L2 edge TCP model parameters are affected by operating conditions determined by the traffic characteristics, active STA count and the MAC and PHY statistics. It is important that virtual speed test is able to estimate both the upload and download throughputs accurately despite a variation in these factors.

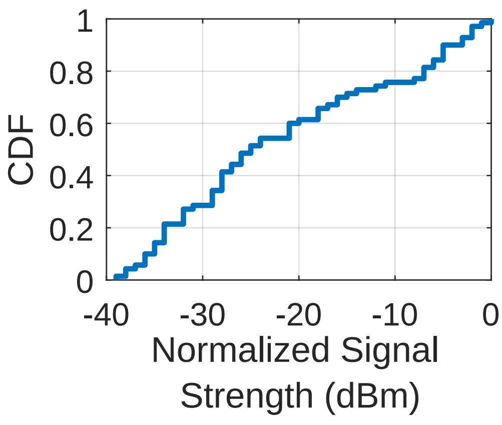
Traffic statistics. Packet size variation causes a microscopic fluctuation in the air time utilization for non-target STAs and transmit time for the target STAs. Fig. 5 shows the application layer data size distribution for both the scenarios. The minimum size encountered in the trace is 20 bytes whereas the maximum size is 1.4K bytes. In addition to this, a fluctuation in the per minute download statistics as shown in Fig. 6 causes a variation in the network load affecting queuing delays and idle times for both the target and non-target STA.

Active STA count diversity. A variation in the number of active STAs in the network affects ground truth value of Q and the 'access' parameters. In both deployments, the number of active STAs is varied to cover a total of 36 and 49 combination of STA sets in the office and residential scenario respectively. In these combinations, active STA sets of all possible sizes are covered and the VST AP makes predictions for each device in the set.

MAC and PHY statistics. Link diversity in terms of signal propagation characteristics (due to LoS and non-LoS links to the VST AP), overheard transmissions from neighboring BSSs, active STA count variation and traffic statistics mentioned above result in a variation in MAC and PHY statistics across devices. Fig. 7 shows the normalized signal strength distribution of the VST AP across all STA locations in the office and residential scenario. The signal strength distribution is normalized with respect to the maximum VST AP signal strength encountered in that environment across all STA positions. Due to the small size of the office, the received signal strength of the VST AP across all STA locations is very close to the encountered maximum. In comparison, the residential scenario exhibits a wide distribution of the VST AP's signal strength across the apartment resulting in a variation in supported PHY rates across different STA positions which affects the 'tx' parameters.



(a) Office



(b) Residential

Figure 7: Distribution of signal strength of the VST AP across all the STA positions in the office and residential deployments. The values are normalized with respect to the maximum signal strength encountered in that particular deployment.

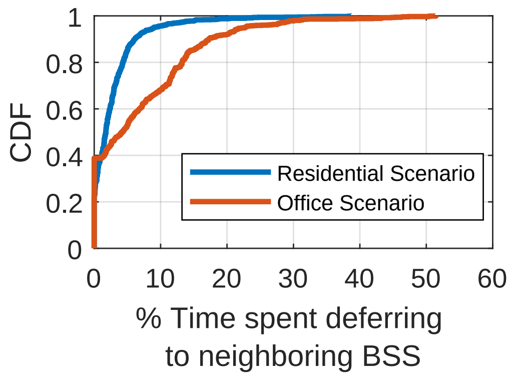


Figure 8: Distribution of fraction of a TCP flow's total duration that a device spends deferring to transmissions from neighboring BSS

Moreover the multi-AP nature of these experiments results in devices in the network deferring to transmissions from co-existing BSS thereby affecting the 'access' parameters. Fig. 8 depicts a distribution of the fraction of time during a download that a device spends deferring to transmission from neighboring BSS in both the office and residential scenario. The activity in the neighboring BSS varied in both the deployments. For instance, during approximately 30% of the flows in both the office and the residential scenario, there was no activity on the neighboring BSS. On the other hand, the maximum portion of a TCP flow's duration that a STA spent deferring to neighboring BSS is 38.8% for the residential scenario and 51.4% in the office scenario.

All the above environment characteristics result in a variation in all the model parameters which the VST AP intends to estimate for the purpose of throughput prediction.

5.3. Ground truth procurement

Ideally, we would compare the outcome of virtual speed test with that obtained from one of the online speed test applications [1, 2, 3]. Recall that during the setup phase, the server selection process of these speed test applications tries to minimize the backbone delay between the server and the AP with the ideal case being a server in the same LAN as the AP. In practice, an ideal server may not be available in the server pool probed by these applications. Consequently, the results obtained from a sub-optimal server may be affected by factors such as variable backbone load, delay factors, server load etc. Instead, we create the ideal case by running an iperf between the AP and the STA. To demonstrate the difference, we keep one 802.11n laptop associated with the AP and run these speed test applications 10 times with a gap of 10 mins between each consecutive run. Fig. 9 shows the values obtained for upload and download throughput from some exemplary online speed test applications as well as iperf. Compared to iperf, the speed test applications demonstrate deterioration and fluctuations due to sub-optimal server selection. On the other hand, the values obtained from iperf remain consistent and all the flow parameters are controlled by us. Therefore, to obtain consistent ground truth values in our experiments, we use the iperf tool.

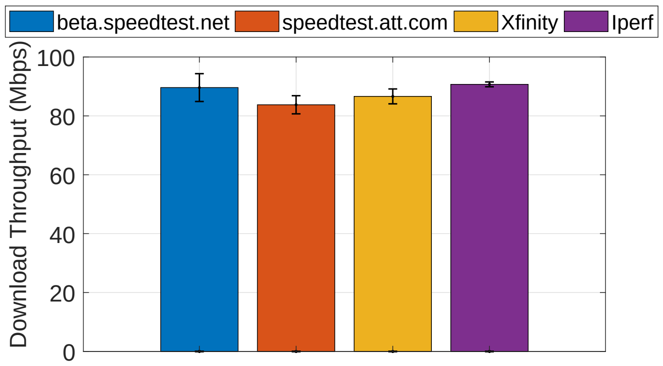
6. Experimental Evaluation

In this section, we investigate the performance of virtual speed test in the two deployment scenarios. The key idea behind virtual speed test is to employ TCP data - ACK handshakes from existing network traffic to estimate AP side unknowns. Specifically, the AP uses these samples to estimate u_{access} which is then used for throughput estimation.

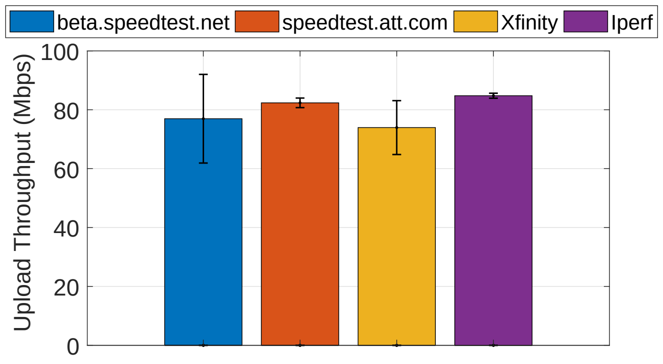
In our deployment, ground truth estimates are acquired every 10 mins by employing iperf tool. This is then compared against the throughput estimates obtained by supplementing the L2 edge TCP model with the measurements acquired by VST AP. The network traffic in our experiments is composed of both download as well as upload traffic. Virtual speed test only leverages the download flows for u_{access} estimation.

6.1. Precision of u_{access} estimation

Recall that the AP requires a knowledge of both S_{vf} and S_{vr} to estimate θ_{dl} and θ_{ul}. As a result, for throughput computation, an accurate estimate of u_{access} at the AP is

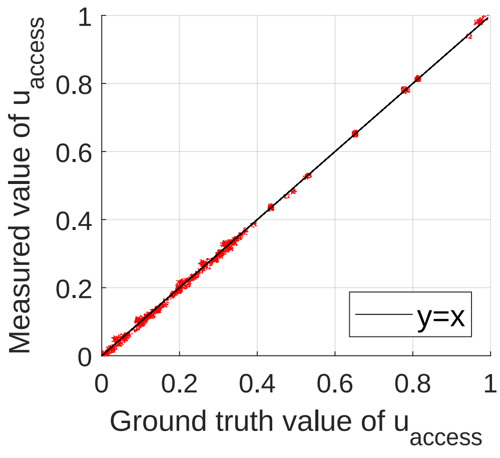


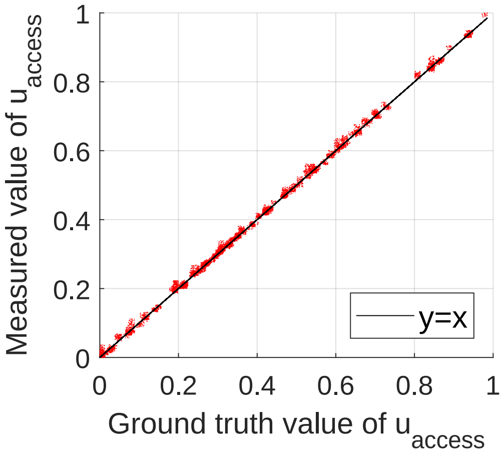
(a) Download speed test



(b) Upload speed test

Figure 9: Comparison of exemplary online speed test applications with iperf for procuring ground truth





(a) Office Scenario

(b) Residential Scenario

Figure 10: Scatter plot of u_{access} measured and ground truth value in the office and residential deployments. In each sub-figure, both the axis are normalized with respect to the maximum ground truth value in that particular deployment.

Table 1: Best, worst and example average case for download throughput predictions in office deployment.

	Download		
	Ground truth (Mbps)	**Estimate (Mbps)**	**%Error**
Best case	85.7	86.57	1.02
Example average case	75.4	72.13	4.33
Worst case	14.3	15.92	11.34

Table 2: Best, worst and example average case for upload throughput predictions in office deployment.

	Upload		
	Ground truth (Mbps)	**Estimate (Mbps)**	**%Error**
Best case	63.7	63.063	1.00
Example average case	36.9	38.38	4.01
Worst case	36.9	40.92	10.89

essential. Consequently, it is necessary to establish the efficacy of the previously discussed u_{access} estimation method.

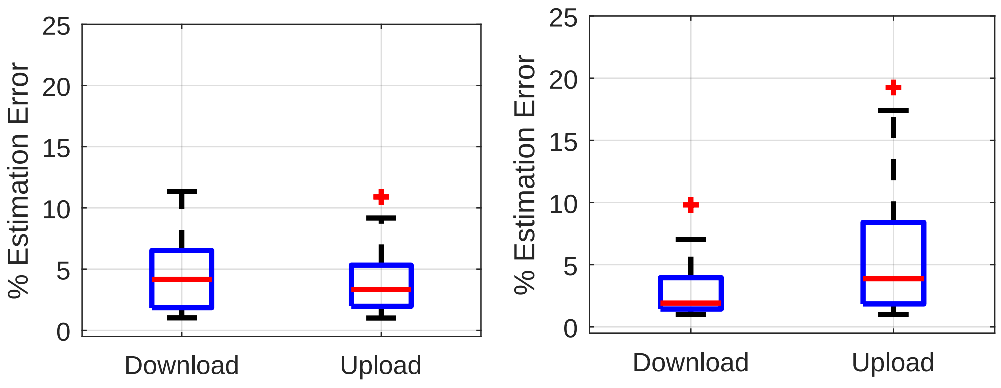
To this end, we demonstrate an existence of homoscedasticity between the ground truth value of u_{access} and the value measured by the estimation technique. In essence, this shows that as the ground truth value of u_{access} varies due to the numerous factors mentioned previously or factors not observed in our deployment, the variation in estimated value of u_{access} does not depend on the ground truth value of u_{access}.

Fig. 10 shows a scatter plot of measured value of u_{access} vs the ground truth value. In our deployment, the AP observes cases involving only single application traffic as well as those with mixed application traffic. In case of mixed application traffic, u_{access} is obtained by averaging over all the concurrent traffic flows of the target STA. The ground truth values are obtained by utilizing STA side statistics. For each deployment scenario, both of these values are normalized with respect to the maximum ground truth value in that particular scenario.

In both the deployment scenarios, as the measured values of u_{access} varies, the measured values continue to be closely located to the identity line. Moreover, the variance in the measured value is independent of the ground truth value of u_{access}. This confirms the effectiveness of the u_{access} estimation methodology and its general applicability.

6.2. Throughput estimation accuracy

The ultimate goal of virtual speed test is to estimate θ_{dl} and θ_{ul}. To demonstrate the throughput estimation accuracy, we calculate the percent estimation error between the ground truth values obtained from iperf and the estimates of virtual speed test as $\%error = \frac{|(\theta_{\text{gt}} - \theta_{\text{est}})| * 100}{\theta_{\text{gt}}}$ where θ_{gt} denotes the ground truth value of throughput and θ_{est} denotes the estimated value from virtual speed test. We remark that since the $\%error$ is calculated after weighing by θ_{gt}, it is sensitive to the value of θ_{gt} *i.e.* the same absolute error at a smaller ground truth value is bound to yield a higher estimation error.



(a) Office Scenario (b) Residential Scenario

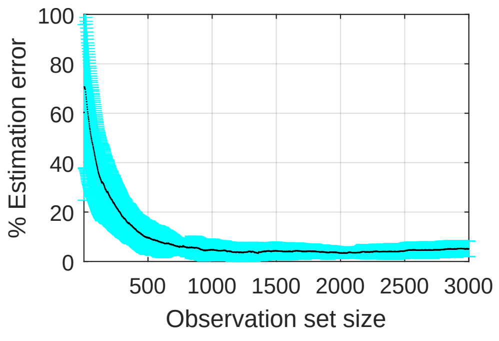
Figure 11: Percent throughput estimation error statistics in the office and residential deployments.

Table 3: Best, worst and example average case for download throughput predictions in residential deployment.

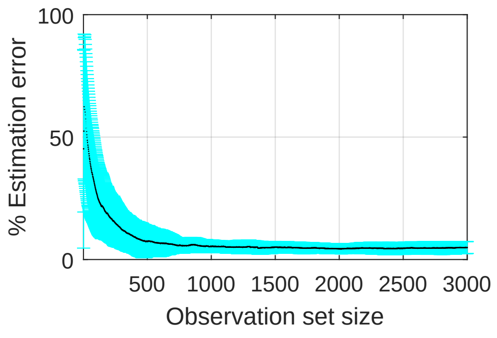
	Download		
	Ground truth (Mbps)	**Estimate (Mbps)**	**%Error**
Best case	82	81.18	1.00
Example average case	82	86.33	5.28
Worst case	47.5	52.15	9.81

Table 4: Best, worst and example average case for upload throughput predictions in residential deployment.

	Upload		
	Ground truth (Mbps)	**Estimate (Mbps)**	**%Error**
Best case	58.6	57.98	1.00
Example average case	51.6	50.39	2.34
Worst case	4.44	5.29	19.25



(a) Download



(b) Upload

Figure 12: Mean throughput estimation error as a function of the observation set size available at the AP for office scenario.

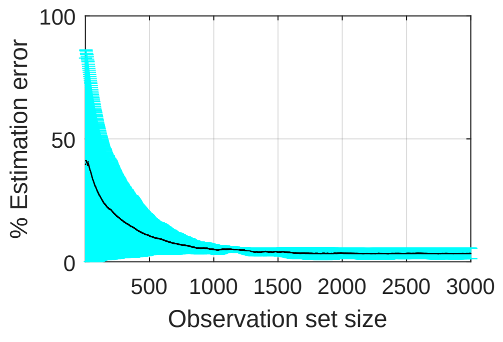
Fig. 11 summarizes the percent throughput estimation error statistics in both the office and residential deployments. Overall virtual speed test shows a good match against ground truth values with a mean percentage error of 4.09% and 4.3% for upload and download speeds respectively in the office scenario. In the residential scenario, these values are 2.9% and 5.51% respectively. The ground truth and estimated values corresponding to the worst, the best and an example case close to the mean estimation error along with their corresponding percent estimation errors are shown for the office and residential scenario in Table 1, 2, 3 and 4 respectively.

6.3. Observation set size characterization

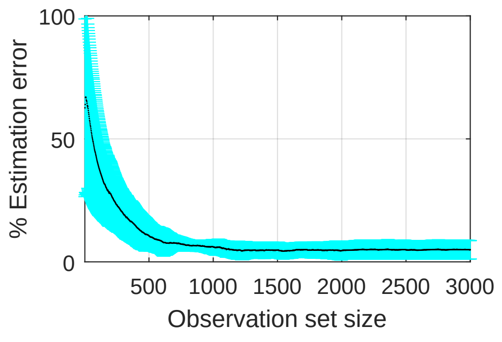
Virtual speed test estimates the model parameters from measurements collected via TCP traffic in the network. The model parameter estimation accuracy is a function of the number of measurement samples available at the AP which in turn is dependent on the activity of user corresponding to the target STA. On one extreme, a large number of observations obtained from either a single large file download or a series of consecutive small file downloads results in a refined parameter estimation. It is important to quantize the observation set size that is required to avoid a deterioration in the estimation accuracy.

We experimentally investigate the observation set size required as follows. Since the backbone delay between the snooped flow's server and the AP is inconsequential, we initiate a single TCP flow from the AP to the target STA and treat it as a snooped flow. We control the observation set size available at the AP by changing the download file size and computing the percent throughput estimation error as before. In each deployment, we vary the number of active STA size and combinations as before and repeat these experiments for half of a day.

Fig. 12 and Fig. 13 depicts the mean estimation error as a function of the size of the observation set available at the VST AP. The mean estimation error demonstrates an exponential decrease with an increasing set size. When the sample set is size in on the order of a few 10s of samples, the estimation error is as high as 70%. The variance in this case is also very high. This is due to the fact that the number of measurement samples are insufficient to accurately characterize the model parameters. With an increasing sample set size, the precision in the model estimation increases as expected. Fig. 12 and 13

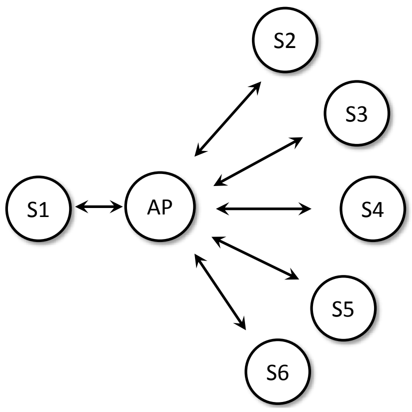


(a) Download

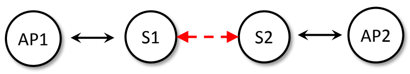


(b) Upload

Figure 13: Mean throughput estimation error as a function of the observation set size available at the AP for residential scenario.



(a) Hidden terminals



(b) Cell edge STA

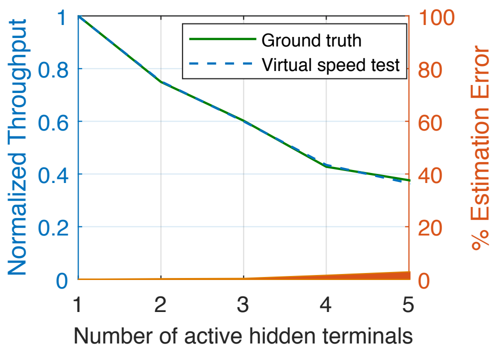
Figure 14: Network topology for (a) Hidden terminal scenario (b) Cell edge STA scenario. In the hidden terminal scenario, the STAs S2 - S6 are beyond the carrier sense range of STA S1 and vice versa.

reveal that even when the observation set contains a few 1000s of samples (which could easily be obtained from observing the download of a research paper from IEEE Xplore), the mean estimation error of virtual speed test is under 5% in both the deployments.

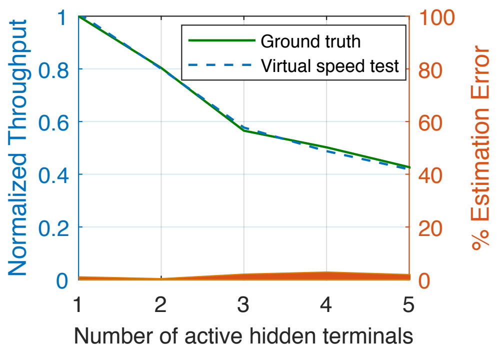
6.4. Hidden terminals and cell edge scenarios

To analyze the performance of virtual speed test under operating conditions beyond those observed in our deployment, we implement virtual speed test in the source library of ns-3 [16]. In particular, we investigate network topologies involving hidden terminals and cell edge STAs as shown in Fig. 14. Both of these cases are characterized by a high number of collisions and retransmissions which are captured by the 'access' parameters of the L2 edge TCP model. In the hidden terminal scenario, the STAs S2 - S6 are beyond the carrier sense range of STA S1 and vice versa. In this scenario, the number of hidden terminals is varied from 1 through 5.

For obtaining ground truth, we emulate a speed test in as follows. The STA performs an active download and an active upload to a server with a backbone delay of 1 ms to

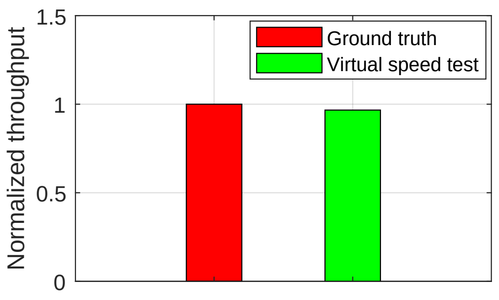


(a) Download

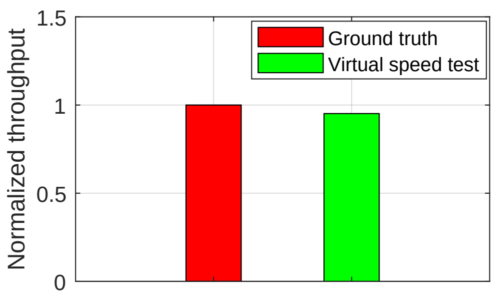


(b) Upload

Figure 15: Throughput and estimation errors for the hidden terminal scenario for (a) Download throughput estimation (b) Upload throughput estimation. In each case, the throughput is normalized with respect to the maximum ground truth value.



(a) Download



(b) Upload

Figure 16: Throughput and estimation errors for the case of cell edge scenario for (a) Download throughput estimation (b) Upload throughput estimation. The estimation error for download throughput is 3.32% and for upload throughput is 4.8%.

the AP. The number of parallel TCP flows initiated is 10. The ground truth values are measured by averaging the total data transmitted from all the parallel TCP flows over the total time. The background traffic is fully backlogged uplink UDP flows with segment size fixed at 1KB. The snooped flow here is a single long lived TCP flow with $W_{\mathrm{m}} =$ 50 segments. The snooped flow server has a backbone delay of 10 ms to the AP and each segment in the snooped flow is 1KB long. Fig. 15 and Fig. 16 show the normalized throughput and estimations in the case of the hidden terminal and cell edge scenarios respectively. Our simulation findings concur with our experimental conclusions on the effectiveness of virtual speed test. The overall accuracy of virtual speed test is below 5%.

7. Related Work

Analytical Models. Powerful analytical models are available in literature which enable throughput prediction. [17], [18], [19] and [20] incorporate Wi-Fi's random count-down process along with changing contention windows due to binary exponential backoff. [21] accounts for a number of realistic factors that are critical for model predictions to match results obtained in measurement studies (*e.g.*, variable channel rates, mixes of hidden and exposed terminals, variable rate traffic, and capture effects) while [22] account for bottlenecks in the presence of TCP flows. While it may seem promising to extend these models to enable throughput prediction, they are hindered by a requirement for AP side knowledge of topology, interfering nodes including those from neighboring BSS, their traffic patterns, PHY capabilities, data rates etc. Obtaining this information requires a co-operation and regular reporting from the STAs. On the other hand, virtual speed test enables throughput estimation with zero STA side co-operating and no reporting.

Active probing. Active probing techniques involve usage of probing packets to estimate bandwidth. These have also been extended to wireless networks by [23], [24] and [25]. These techniques impose additional traffic load on the network and hence their periodic usage can disrupt user traffic or drain the battery of mobile devices. On the other hand, virtual speed test performs passive measurement based estimation and hence does not impose any additional traffic load on the network.

Passive sniffer. Careful deployment of sniffers can be used to make passive observations to collect traffic traces of various users to estimate throughput [26],[27]. However, such methods require installation and maintenance of additional hardware for data collection. In contrast, virtual speed test requires no additional infrastructure, no client side software and no cooperation from clients.

Training via ground truth measurements. In this approach, followed by [10, 28, 29, 30] network clients are instrumented with special software that stores empirical throughput of all TCP sessions and reports them to the AP to build a database of throughputs of TCP sessions of its clients. Along side the TCP throughput data, the AP also records the wireless conditions during the session, e.g. , the sessions MCS, busy air time, and collision rate. The AP can subsequently perform prediction of TCP throughput without performing an actual test for any client by correlating the current wireless conditions with historical averages that correspond to similar conditions. However, this approach requires client side co-operation to obtain ground truth as network and traffic conditions change, a requirement that is not allowed in our problem formulation.

TCP flow analysis. Analysis of TCP flow statistics has been leveraged for both network performance evaluation and security threat detection. Tools like Tstat [31] perform analysis of headers in TCP flows to provide several IP and TCP statistics such as segment reordering, duplication, etc. which aids network measurement research. On the other hand [32] utilizes TCP's spectral characteristics to separate TCP flows from those corresponding to malicious attacks and [33] demonstrates P2P Botnet detection based on successful connection rate of TCP flows. In contrast, we utilize TCP flow dynamics to measure L2 parameters to facilitate upload and download throughput estimation for WLANs.

Packet sampling techniques. A number of packet sampling techniques have been proposed in literature to minimize the amount of computation needed. The IETF has also studied information models and protocols for packet sampling[34]. In [35], packet sampling has been used to improve the accuracy of network traffic identification. The authors of [36] explore the impact of packet sampling on malware traffic detection. In [37], the authors present a reinforcement learning based packet sampling based approach. Such approaches can be leveraged to further improve the computational efficiency of our framework.

8. Conclusion

In this paper, we present virtual speed test - a measurement based framework that enables an AP to estimate the TCP speed test results for any of its associated STAs without any end-user co-operation, with no additional traffic load on the network and solely based on passively obtained AP-side observables. We deploy a virtual speed test (VST) enabled AP in a university office for a period of 2 days and in a residential apartment for a period of 7 days. These deployments cover a total of 36 and 49 different topologies for the office and residential scenario respectively with varying number of STAs. Further, these scenarios are characterized by a variety of operating conditions involving presence of multiple BSS co-existing with our network, link diversity in terms of signal propagation and supported PHY rates and variation in traffic characteristics. Overall, virtual speed test exhibits a high level of estimation accuracy with mean estimation errors below 6% and best case estimation errors as low as 1%.

Appendix A.

We break the proof into two parts. (a) In the first part, we show that W^*_{m} is upper bounded by 4. (b) In the second part we show this maximum value occurs when $S_{vf} = S_{vr}$.
(a) Suppose that $S_{vf} = max(S_{vf}, S_{vr})$. Consequently,

$$W^*_{\mathrm{m}} = \frac{2 * (S_{vf} + S_{vr})}{S_{vf}} \leq 4 \tag{A.1}$$

The same can be shown when $S_{vr} = max(S_{vf}, S_{vr})$.
(b) It can be further stated that

$$W^*_{\mathrm{m}} = \frac{4 * (S_{vf} + S_{vr})}{\frac{(S_{vf}+S_{vr}+|S_{vf}-S_{vr}|)}{2}} \tag{A.2}$$

Suppose that $S_{vf} \geq S_{vr}$. When W^*_{m} reaches its upper bound, it follows that

$$\frac{2 * (S_{vf} + S_{vr})}{S_{vf}} = 4 \tag{A.3}$$

It follows that $S_{vf} = S_{vr}$. Again, the same can be shown when $S_{vf} \leq S_{vr}$. In the absence of the equality condition, Eq. A.3 will not be solvable.

References

[1] Ookla Speedtest, `http://www.speedtest.net/`, accessed: 2018-05-27.
[2] AT & T Internet Speed Test, `http://speedtest.att.com/speedtest/`, accessed: 2018-05-27.
[3] Xfinity Speed Test, `http://speedtest.xfinity.com/`, accessed: 2018-05-27.
[4] D. Murray, T. Koziniec, S. Zander, M. Dixon, P. Koutsakis, An analysis of changing enterprise network traffic characteristics, in: Communications (APCC), 2017 23rd Asia-Pacific Conference on, IEEE, 2017, pp. 1–6.
[5] O. SpeedTest, How does the Begin Test button select a server? (2012).

[6] O. SpeedTest, How does the test itself work? How is the result calculated?” (2012).

[7] O. Goga, R. Teixeira, Speed measurements of residential internet access, in: International Conference on Passive and Active Network Measurement, Springer, 2012, pp. 168–178.

[8] E. Altman, D. Barman, B. Tuffin, M. Vojnovic, Parallel TCP Sockets: Simple Model, Throughput and Validation., in: INFOCOM, Vol. 2006, 2006, pp. 1–12.

[9] S. Bauer, D. Clark, W. Lehr, Understanding broadband speed measurements (2010).

[10] A. Patro, S. Govindan, S. Banerjee, Observing Home Wireless Experience Through WiFi APs, in: Proceedings of the 19th Annual International Conference on Mobile Computing &#38; Networking, MobiCom ’13, ACM, New York, NY, USA, 2013, pp. 339–350. doi:10.1145/2500423.2500448.
URL `http://doi.acm.org/10.1145/2500423.2500448`

[11] M. Harchol-Balter, Performance modeling and design of computer systems: queueing theory in action, Cambridge University Press, 2013.

[12] D. Murray, T. Koziniec, The state of enterprise network traffic in 2012, in: 2012 18th Asia-Pacific Conference on Communications (APCC), 2012, pp. 179–184. doi:10.1109/APCC.2012.6388126.

[13] R. Braden, RFC-1122: Requirements for internet hosts, Request for Comments (1989) 356–363.

[14] P. Jon, Transmission control protocol–darpa internet program protocol specification, Tech. rep., RFC-793, DARPA (1981).

[15] P. Guide, Intel® 64 and IA-32 Architectures Software Developer’s Manual, Volume 3B: System programming Guide, Part 2 (2011).

[16] ns3 manual, `https://www.nsnam.org/docs/manual/html/random-variables.html`, accessed: 2018-07-15.

[17] G. Bianchi, Performance analysis of the IEEE 802.11 distributed coordination function, IEEE Journal on Selected Areas in Communications 18 (3) (2000) 535–547. doi:10.1109/49.840210.

[18] T. Sakurai, S. Hanly, Modelling TCP flows over an 802.11 wireless LAN, in: 11th European Wireless Conference 2005 - Next Generation wireless and Mobile Communications and Services, 2005, pp. 1–7.

[19] D. Miorandi, A. A. Kherani, E. Altman, A queueing model for HTTP traffic over IEEE 802.11 WLANs, Computer Networks 50 (1) (2006) 63 – 79. doi:https://doi.org/10.1016/j.comnet.2005.04.004.
URL `http://www.sciencedirect.com/science/article/pii/S138912860500112X`

[20] J. Yu, S. Choi, D. Qiao, TCP dynamics over IEEE 802.11E WLANs: Modeling and throughput enhancement, in: Broadband Communications, Networks and Systems, 2007. BROADNETS 2007. Fourth International Conference on, 2007, pp. 66–75. doi:10.1109/BROADNETS.2007.4550408.

[21] J. Camp, E. Aryafar, E. Knightly, Coupled 802.11 Flows in Urban Channels: Model and Experimental Evaluation, IEEE/ACM Trans. Netw. 20 (5) (2012) 1452–1465. doi:10.1109/TNET.2011.2181863.
URL `http://dx.doi.org/10.1109/TNET.2011.2181863`

[22] P. Nayak, M. Garetto, E. W. Knightly, Multi-user Downlink with Single-User Uplink can Starve TCP, in: IEEE INFOCOM 2017 - IEEE Conference on Computer Communications, 2017, pp. 1–9. doi:10.1109/INFOCOM.2017.8057048.

[23] M. Li, M. Claypool, R. Kinicki, WBest: A bandwidth estimation tool for IEEE 802.11 wireless networks, in: Local Computer Networks, 2008. LCN 2008. 33rd IEEE Conference on, IEEE, 2008, pp. 374–381.

[24] K. Lakshminarayanan, V. N. Padmanabhan, J. Padhye, Bandwidth estimation in broadband access networks, in: Proceedings of the 4th ACM SIGCOMM conference on Internet measurement, ACM, 2004, pp. 314–321.

[25] R. Kapoor, L.-J. Chen, L. Lao, M. Gerla, M. Y. Sanadidi, Capprobe: A simple and accurate capacity estimation technique, in: ACM SIGCOMM Computer Communication Review, Vol. 34, ACM, 2004, pp. 67–78.

[26] L. DiCioccio, R. Teixeira, C. Rosenberg, Impact of Home Networks on End-to-end Performance: Controlled Experiments, in: Proceedings of the 2010 ACM SIGCOMM Workshop on Home Networks, HomeNets ’10, ACM, New York, NY, USA, 2010, pp. 7–12. doi:10.1145/1851307.1851310.
URL `http://doi.acm.org/10.1145/1851307.1851310`

[27] Y.-C. Cheng, J. Bellardo, P. Benkö, A. C. Snoeren, G. M. Voelker, S. Savage, Jigsaw: Solving the Puzzle of Enterprise 802.11 Analysis, in: Proceedings of the 2006 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications, SIGCOMM ’06, ACM, New York, NY, USA, 2006, pp. 39–50. doi:10.1145/1159913.1159920.
URL `http://doi.acm.org/10.1145/1159913.1159920`

[28] C. Rattaro, P. Belzarena, Throughput prediction in wireless networks using statistical learning, in:

LAWDN-Latin-American Workshop on Dynamic Networks, 2010, pp. 4–p.

[29] W. Zhou, Z. Wang, W. Zhu, Mining Urban WiFi QoS Factors: A Data Driven Approach, in: Multimedia Big Data (BigMM), 2017 IEEE Third International Conference on, IEEE, 2017, pp. 9–16.

[30] M. Mirza, K. Springborn, S. Banerjee, P. Barford, M. Blodgett, X. Zhu, On the accuracy of TCP throughput prediction for opportunistic wireless networks, in: Sensor, Mesh and Ad Hoc Communications and Networks, 2009. SECON'09. 6th Annual IEEE Communications Society Conference on, IEEE, 2009, pp. 1–9.

[31] M. Mellia, A. Carpani, R. L. Cigno, Tstat: TCP statistic and analysis tool, in: International Workshop on Quality of Service in Multiservice IP Networks, Springer, 2003, pp. 145–157.

[32] Y. Chen, K. Hwang, Tcp flow analysis for defense against shrew ddos attacks, in: IEEE International Conf. on Communications, 2007, pp. 1203–1210.

[33] L. Zhou, Z. Li, B. Liu, P2P traffic identification by TCP flow analysis, in: Networking, Architecture, and Storages, 2006. IWNAS'06. International Workshop on, IEEE, 2006, pp. 2–pp.

[34] T. Dietz, B. Claise, P. Aitken, F. Dressler, G. Carle, Information model for packet sampling exports, Tech. rep. (2009).

[35] S. Dong, Y. Xia, Network traffic identification in packet sampling environment, Digital Communications and Networks (2022).

[36] C. Novo, J. M. C. Silva, R. Morla, An outlook on using packet sampling in flow-based c2 tls malware traffic detection, in: 2021 12th International Conference on Network of the Future (NoF), IEEE, 2021, pp. 1–5.

[37] M. Bachl, F. Meghdouri, J. Fabini, T. Zseby, Sparseids: Learning packet sampling with reinforcement learning, in: 2020 IEEE Conference on Communications and Network Security (CNS), IEEE, 2020, pp. 1–9.