WebRTC Based Network Performance Measurements

by

Miranda McClellan

Submitted to the Department of Electrical Engineering and Computer Science
in partial fulfillment of the requirements for the degree of

Masters of Engineering in Computer Science and Engineering

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

June 2019

© Massachusetts Institute of Technology 2019. All rights reserved.

Author .....
Department of Electrical Engineering and Computer Science
May 24, 2019

Certified by .....
Steven Bauer
Research Scientist
Thesis Supervisor

Accepted by .....
Katrina LaCurts
Chair, Master of Engineering Thesis Committee

WebRTC Based Network Performance Measurements

by

Miranda McClellan

Submitted to the Department of Electrical Engineering and Computer Science
on May 24, 2019, in partial fulfillment of the
requirements for the degree of
Masters of Engineering in Computer Science and Engineering

Abstract

As internet connections achieve gigabit speeds, local area networks (LANs) become the main bottleneck for users connection. Currently, network performance tests focus on end-to-end performance over wide-area networks and provide no platform-independent way to tests LANs in isolation. To fill this gap, I developed a suite of network performance tests that run in a web application. The network tests support LAN performance measurement using WebRTC peer-to-peer technology and statistically evaluate performance according to the Model-Based Metrics framework. Our network testing application is browser based for easy adoption across platforms and can empower users to understand their in-home networks. Our tests hope to give a more accurate view of LAN performance that can influence regulatory policy of internet providers and consumer decisions.

Thesis Supervisor: Steven Bauer
Title: Research Scientist

Acknowledgments

I am grateful to the Internet Policy Research Initiative for providing support and context for my technical research within a larger ecosystem of technology policy and advancements.

Finally, I would like to thank my academic advisor, Leslie Kaelbling for keeping me grounded and focused on my goals in addition to my friends and family who inspire me to continue setting and achieving new goals for myself. In particular, I'd like to thank Maryam Archie and Karia Dibert for selflessly sacrificing their personal computers for testing throughout my software development process.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Everyone has experienced moments where the connection is slow or severed abruptly while browsing the internet. We understand the frustration behind not being able to solve internet connectivity issues, and even worse not even understanding what has gone wrong in the complex network system that delivers online content to our homes and workplaces. Network performance measurements aide in diagnosing these connectivity issues and otherwise understanding the health of an internet connection.

1.1 Network Performance Measurements

Network performance measurements are used to measure and understand the viability of the connection between a users local machine and another point in the Internet. Network measurement is important to multiple stakeholders for technical, regulatory, and political reasons. The ability to measure network connectivity and capacity has growing importance to ensuring the fulfilment of service expectations between consumers and internet service providers (ISPs), especially as internet connectivity becomes a global commodity. However, determining the viability and strength of an internet connection is complex, and numerous existing tests attempt to evaluate performance based on a medley of measurements including reliability, speed, availability, or fairness among other factors.

The designer of a network measurement tests selects factors based on the assump-

tions about the network state, topology, and resource demands. Tests are typically performed between a users machine, a personal computer, and the test providers server. To understand the connection, many speed tests measure performance metrics such as the download and upload speeds, latency, loss, and round trip time (RTT), which is the time to deliver a message and receive a reply. Speed is the most common metric for measuring network performance [2]. There are many tests already developed for measuring wide-area network (WAN) performance, but differences in assumptions and methodologies, such as prioritizing download speed over minimizing off-net congestion or throughput, cause large discrepancies in the results [2, 3]. In addition, the complexity of WAN tests introduces variables in measured metrics that are outside of the end users control and locality, such as such as network connection dependencies.

Our WebRTC-based network performance measurements and supporting web application developed are an important step in understanding the performance of high-speed networks. Existing tests focused on wide area network (WAN) performance produce results with high discrepancy, up to an order of magnitude [4], in addition to lacking a nuanced view of LAN performance. The discrepancy is caused by a wide range of assumptions and methodologies about network configuration in current tests. Lack of LAN-specific testing and large result discrepancies complicated our understanding of users experienced connection and the true capabilities of networks.

1.2 Technical Background

In this thesis, we focus on speed measurements within the field of network performance measurements. Speed measurements which are often used to assess a networks quality of service. Existing, alternative testing methodologies are discussed in detail in Section 2. We choose to focus on bulk transfer capacity (BTC), defined as the largest number of bits that a network can be expected to deliver from point A to point B over some period [5]. The connections performance is evaluated by collecting measurements and using a network model to determine if the path can meet given

target performance input standards.

We construct and evaluate two network performance tests over WebRTC data channels, one between peer machines on the same LAN and the second from an end users personal machine to a server using a data stream with periodic bursts. Both tests measure bandwidth (speed), round trip time, and loss rate.

To evaluate network speed, we utilize the Mathis Equation below, which calculates the bandwidth for a network with periodic loss and a receiver that acknowledges every packet, like one operating with TCP congestion control:

$$BW = \frac{MSS * C}{RTT * p}$$

The Mathis Equation determines the relationship between bandwidth (BW) also known as throughput, maximum segment size (MSS) which is the largest expected packet size, round trip time (RTT) and a constant nonzero random packet loss rate (p) [6]. The speed of wireless LANs that we measure is affected by higher loss along wireless links and the small RTT of nearby peer machines.

We consider our network performance measurements within the framework of Model Based Metrics (MBM) for Bulk Transport Capacity introduced by Mathis and Morton in an IETF RFC [1]. The MBM test framework allows one to input target performance values (what characteristics we require a network to exhibit) and network models (our understanding of a networks behavior, that may affect our measurement of it) to run suites of performance tests. In our WebRTC Based Network Performance Measurements, we set target round trip times and loss rates and the Mathis Equation serves as our network performance model. The tests developed, described in detail in Section 3, evaluate the path between two peers by aiming to collecting measurements and using the network models to determine if the network is performant enough to meet the target performance standards.

1.3 Motivating Scenarios

Why are isolated LAN performance measurements and tests necessary? To answer this question, we look to two motivating scenarios.

First, consider the typical in-home wireless connection, which can be characterized by high loss rates in comparison to physical links and low round-trip time (RTT) due to the physical proximity of the peers. When measured with a common measurement tool, like IPerf3, When measured with a common measurement tool, like IPerf3, the capability of the path will be overstated for any flows with longer RTTs. By utilizing the Mathis equation in our calculation of throughput, we capture both the effects of both the short RTT and the loss rate in our approach and gain a more nuanced and accurate view of bandwidth for LANs.

Second, consider the movement of ISPs to offer high speed connections. As capacity increases towards Gigabit speeds, performance bottlenecks experienced by consumers are more likely to occur in LANs because of issues like high loss rates in wireless networks [3, 7]. Traditional speed tests obfuscate this detail in their end-to-end measurements. The ability to test LAN performance in isolation will allow end users to first understand if their in-home wireless network is the source of a performance bottleneck before inquiring about the performance of the networks of the ISPs, target platforms like Google or Netflix, or the interconnection points between these large corporate networks.

1.4 Contributions

This experimental thesis project contributes five main additions to the network performance measurement field:

1) determines whether WebRTC peer-to-peer data channels are performant enough to be used for high-demand network measurements,

2) develops a system to perform local area network (LAN) performance test in isolation using WebRTC peer-to-peer data channels,

3) provides a platform-independent web application for performing the tests,

4) provides the LAN component essential to the development of future efforts in the composability of network tests as outlined in the Model Based Metrics RPC, and

5) evaluated our systems performance against the outcomes of existing tools on MITs wireless networks.

Chapter 2

Background

Network performance tests are used to understand the viability of the connection between a users local machine and another point in the Internet, typically between a users machine and the test providers server. Traditional network performance tests focus on end-to-end performance, which introduces variables outside of the end users control or locality such as such as network connection dependencies between Internet service providers (ISPs) or the topology and demand of the test providers network. To understand the connection, network performance tests measure various characteristics of the data transmission such as the ping time, round-trip time (RTT), speed, and loss rates [4, 7]. Understanding the performance and being able to model these metrics over time for a network is valuable to many stakeholders including network administrators, internet service providers (ISPs), consumers making purchasing decisions, and policy makers regulating ISPs. We will explain the current network testing ecosystem, the goals and methodologies of a representative sample of existing network performance tests, and their limitations.

2.1 Purpose and Methods of Network Measurement Tests

Speed is the most common metric for measuring network performance [2]. Modern networks are defined by their speed (also known as throughput), a measurement of the networks physical capacity to move data along links. Speed can be considered a measurement of the average speed at which a user can send traffic within the available capacity on the internet link. The speed that is considered sufficient depends on the task of the network and the stakeholder evaluating the results. For example, an ISP might be interested in capturing end-to-end network measurements that can be used to understand how users experience their internet service, a concept called quality of service, and how closely the ISP is meeting their quality of service promises.

Understanding network performance is a complex task. Modern networks support many concurrent users and tasks without allowing performance to suffer because of necessary resource sharing, congestion along network links during heavy usage periods or during outages in other segments of the network, and the increasingly heavy delivery demands as user adopt more data-intensive uses like video streaming [8]. In addition, growing connection points between ISPs and corporate networks for internet services like Google or Netflix can experience bottlenecks that are difficult to locate and can require multi-corporation agreements to mitigate. Understanding performance bottlenecks in large networks is further complicated by variability in results of performance measurements depending on factors like locality of users and time of day [9].

Network measurement methodologies can be categorized into two main groups: passive measurement and active measurement. Passive measurement methods do not transmit test-specific data on the network but rather monitor existing traffic on the network, while active measurement methods generate test-specific traffic and transmit them across the network link in question. Active measurement methods send generated test traffic from a source host (sender) to a destination host (receiver), select a set of properties for the test traffic such as packet size and sending rate, and

calculate end-to-end performance metrics by analyzing the traffic delivery [10]. The measurement tools discussed in the following section and the framework implemented by this thesis all follow active measurement methodologies.

Bulk transport capacity is the most common method to measure a networks speed for active measurement frameworks. The bulk transport capacity of a network link is the maximum amount of data that can be transmitted over the link in a congestion-aware transport protocol (for example, TCP) over a determined time period [5]. Bulk transport capacity is a useful measurement for the modern internet because it helps understand a networks behavior in data-intensive use cases like sending large files (user file or software downloads) or high-rate video streaming. Bulk transport capacity can be measured by transmitting a large file over a network and measuring the performance using the active measurements of the generated file packets. However, even though many existing testing tools seek to measure speed through bulk transport capacity, test results vary due to differences in assumptions of network topology, the services the network supports (static pages, email, media, or video), and protocols used (TCP, UDP, etc.) [10].

2.2 Existing Network Tests and Frameworks

There are many tests already developed for measuring wide-area network (WAN) performance, but differences in assumptions and methodologies, such as prioritizing download speed over minimizing off-net congestion or throughput, cause large discrepancies in the results [2, 3]. According to Bauer, Clark, and Lehr, many deployed network performance measurement tools use methodology that is inaccurate for assessing broadband network quality. Understanding the methodological differences between tools is vital to making valid inferences from the measurement data. Below, we explore the purpose and limitations of existing testing tools developed in industry.

Most available tests are designed for WAN measurement, are browser-based for easy consumer adoption, and are motivated by the business interests of the test provider. Examples of browser-based tests with different use cases and results are

Speedtest.net by Ookla [11] and Fast.com by Netflix [12].

Netflix is a content provider in part of the growing video streaming trend for which, at peak times, over one-third of internet traffic can be produced. Fast.com tests only download speed from Netflix servers because the bulk of Netflix traffic is large video data flowing in one direction to users [12].

In contrast, Speedtest.net by Ookla allows users to measure the performance from their machine to any public Oookla test server around the globe, measures upload and download speeds in addition to latency, and uses multiple TCP connections for later filtering, and chooses the destination server geographically closer to the client by default [3]. These performance tests measure WAN performance between a users machine and a distant server and depend on the interactions between internet service providers networks, routing metrics, and geographical and political separation [3]. Servers are not equidistant or of equal path and link quality from all users, so results can be influenced by a users location.

2.3 Challenges in Testing Gigabit Speed Networks

For high speed internet connections, those over 20Mbps, the in-home wireless networks are more likely to be the dominant source of bottlenecks [4]. Wireless networks can cause local bottlenecks in network performance. Routers utilized in LAN data transmission have maximum speeds of 54 Mbps (for new models supporting 802.11 g/n) or as low as 11 Mbps (for older models supporting 802.11b). Network congestion signals cause the senders to self-limit data rates to even lower levels [7]. The low maximum speeds can cause noticeable differences in quality of service and can be insufficient to stream videos or other data-intensive internet applications. In addition, wireless networks have higher loss rates and larger variance in latency that are often unacknowledged in traditional active measurement methods [13].

Despite the effects of in home wireless networks on performance, locating the source of bottlenecks is still a challenging task. Measuring within the locality of the home router can help distinguish between performance issues caused within the home

network or those caused by external access connections outside the users locality or control [2, 3]. There are few tools for testing LAN performance despite the need for understanding wireless network performance in isolation.

With the introduction of their commercial home Wi-Fi services, Google offers network performance tests between Wi-Fi routers and any connected device within the local area network (LAN) [14] to fill the gap in LAN testing for their customers. Googles in-home network tests online function for devices connected wireless, not including Ethernet connections, and are only available through the Google Wifi mobile app. Googles Wi-Fi tests report network speeds and categorizes the measured speed into three groups in comparison to ideal performance. However, there are several shortcomings in Googles LAN testing: First, the Google Wifi app can provide a set of network tests for LAN, a largely ignored testing arena. Second, it is only applicable to the small number of consumers who subscribe to Google Wi-Fi. Third, Google maintains opaque records of the proprietary methodology used. And, fourth, the tests lacks platform-independence require proprietary application downloads.

2.4 TCP Behavior

The Transmission Control Protocol (TCP) is the most widely-used transport layer internet protocol. Designed to improve network reliability, TCP adapts its transmission rates by identifying congestion in the network and resending dropped packets that trigger congestion signals. TCPs built-in congestion control algorithms aims to transmit as many packets as possible within the networks resource limits, which are affected by resource sharing across applications and users, changes in available bandwidth, and buffer capacity along the connection. Dropped packets, those not delivered successfully to the receiver, are the main triggers for congestion signals that indicate transmission failures in a network.

TCP utilizes congestion signals to identify network failure states and adjusts the sending rate to relieve stress on the networks limited resources. Congestion signals are commonly triggered by dropped packets, which can be identified using a system

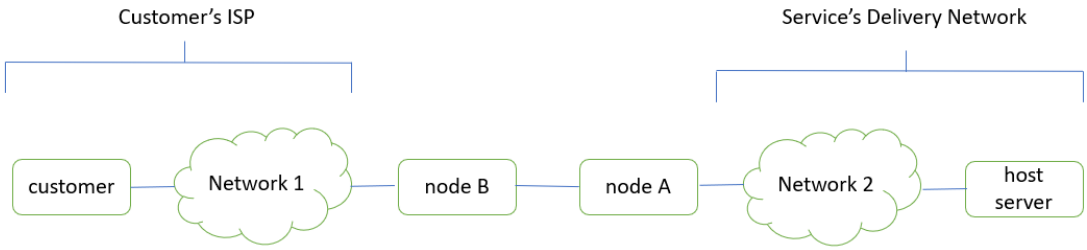

Figure 2-1: Overview model of commonly tested network links.

of acknowledgement packets (ACKs) that are sent from the receiver to the sender to confirm receipt of the packet. Each ACK received initiates an increase in speed in the absence of packet loss. If the sender does not receive ACKs for recent packets, then the congestion control algorithm can infer that packets were dropped along the connection, potentially due to connectivity issues or over-filled buffers. Packet drops then trigger a congestion signal that can cause re-transmission of un-acknowledged packets and a decrease in the sending rate.

We revisit the motivating example from Section 1.3 to better understand the congestion control system can be complicated by differences in distances between sender and receiver. Consider a scenario in which there are two interconnection points, nodes A and B, and you want to measure the bulk transport capacity on the link between them as in Figure 2-1. Nave methods would measure the performance by transmitting data along the interconnection link only by sending data between A and B. Proximity between sender and receiver allows the sender to more rapidly adjust its sending rate to the changes in network resource capacity than between peers that are far apart and must wait longer RTT before determining if a packet has been dropped. If the sender and receiver are very close in terms of round-trip time (RTT), then the sender will wait shorter time periods on average to receive ACKs and be able to respond more quickly to congestion signals. However, in a typical end-to-end use case, the round-trip time will be longer because users are relatively further from the the server that hosts their desired internet content. In the typical case, the loss rate along the interconnection link between A and B would need to be

much smaller in order to support the required throughput given the longer expected round-trip time along the end-to-end paths. Traditional testing methodology does not account for the different maximum loss rates obtained from changing the point of measurement.

This intuition into TCPs response to network congestion signals led to the proposal of the "Mathis Equation", a novel formula for calculating throughput (BW), the amount of data sent via a link during a time period:

$$BW = \frac{MSS * C}{RTT * p} \tag{2.1}$$

where

a) BW is the amount of data transmitted over a link in a given transmission window

b) MSS is maximum segment size, (1460 bytes by default)

c) C is a constant here (1 by default)

d) RTT is round trip time in seconds, and

e) p is the loss percentage, or number of congestion signals per acknowledged packet. [6]

2.5 Model-Based Metrics Framework

In this thesis, we implement the network performance framework outlined in Mathiss Model Based Metrics (MBM). The MBM framework is designed to assess whether a network can support a set of target performance criteria by providing a model for expected network behavior, a manner to create and send test data traffic across links of interest, and a statistical method for evaluating the performance based on a set of observed measurements [1]. Combined with the improved model of TCP performance described in the previous section, MBM provides a framework for more meaningful and accurate network measurements, especially when applied to LANs.

The MBM framework notes several limitations in current methods for capturing

throughput and provides mathematically backed solutions to produce more meaningful results of network measurements. First, it discourages the triggering of TCP-style congestion response. The test traffic used to probe the network can influence congestion on the link of interest and then measure the very congestion the test caused while robbing the link of any measurable ground truth condition. Next, MBM employs a strategy such that the networks delivery of the test traffic and any dropped packets do not also influence the traffic sending patterns. Finally, MBM provides a framework for internet performance diagnosis that statistically evaluates link performance in a vantage-point independent manner.

Chapter 3

System Design

In general, a test framework consists of a suite of network performance test, each of which is designed to measure a different type of network based on the networks topology and behavior such as delivery and congestion patterns. Tests within the suite can also be customized to evaluate networks based on their typical use cases. For example, existing network speed tests make different assumptions about network performance existing tests including Netflixs Fast.com, which focuses on download speed to model the heavy video streaming demands their network must manage. The WebRTC-based network performance measurements implemented in this thesis follows the MBM framework for component structure as shown in Figure 3-1 (from top to bottom of diagram):

1. A set of target performance goals,

2. Network models appropriate for the typical application,

3. A method for generating test traffic by which the networks performance is measured, and

4. An evaluation of the collected measurements to determine the tests outcome.

3.1 Design Goals

The WebRTC-based network performance measurements are designed to serve as both a proof of concept of WebRTCs performance for low-latency applications

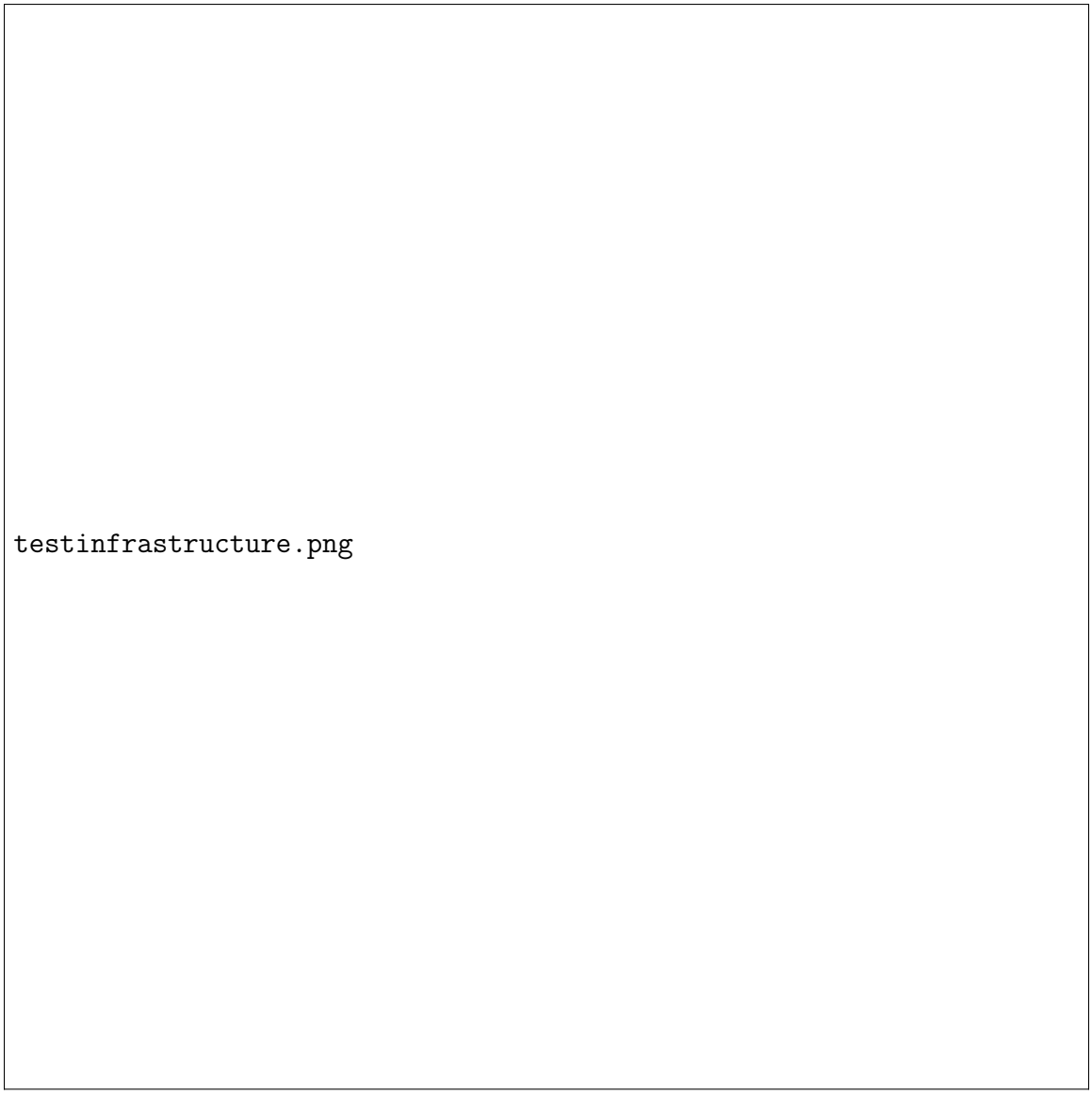

Figure 3-1: MBM test framework design detailing logical flow for active measurement of network performance image [1]

and a production speed test available to the public. This thesis provides a test framework that measures, records, and evaluates the network connection between two peer machines within the same LAN or between a hosted server and an end users machine. The test framework also evaluates whether the connection has sufficient capacity to meet target performance metrics. While designed with the measurement of high-speed networks in mind, the test framework is extensible to measure and evaluate LAN performance against new network performance profiles as future users find necessary.

Our goal in this project is to provide network performance tests through a portable web application that allows users to gain an accurate view of LAN performance based on throughput and loss that is independent of WAN performance. To accomplish this, we use the WebRTC application-level framework to allow peer machine discovery and direct peer-to-peer communication which enables measurements isolated to the LAN. The test framework operates completely within the browser, for this thesis limited to Google Chrome, completely at the application layer. We selected this design paradigm despite potential additional latency from operating at a high stack layer for these benefits:

1) system interoperability,

2) accessibility to users within familiar browser,

3) lack of overhead or required downloads.

Making a network speed test that operates within the browser is a trade-off between usability and performance. Browser-based applications are more familiar to users and more likely to be adapted than solutions requiring downloaded software specific to certain operating systems. In addition, common browsers like Google Chrome, for which our tool is implemented, have already gained the trust of users as secure programs that are safe to run.

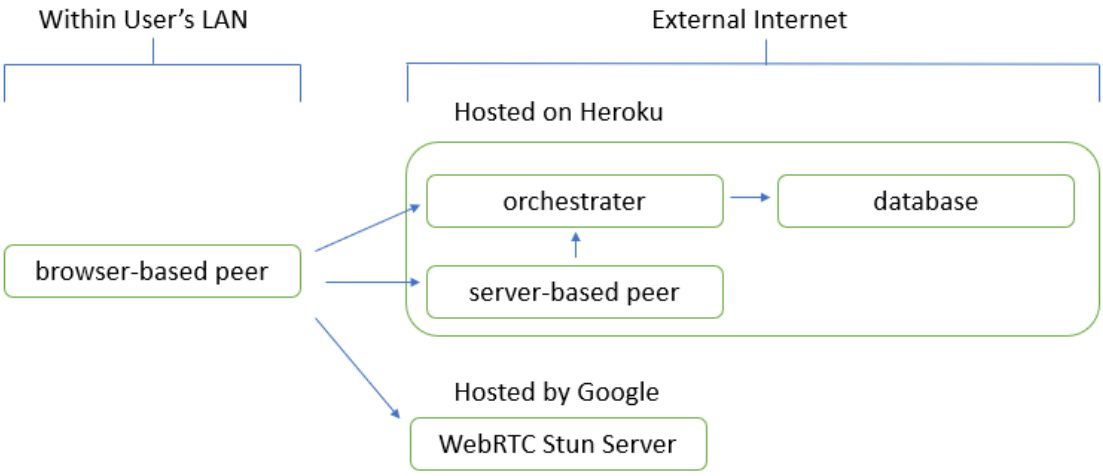

Figure 3-2: Diagram of the main components of the WebRTC-based test infrastructure. The WebRTC system allows separation between peer machines and components external to the LAN (orchestrating server, database, STUN server) during the network performance measurements.

3.2 Infrastructure Design and Overview

At the top level, the test framework that supports our WebRTC-based network performance measurements has four parts: a user interface, the WebRTC STUN and TURN signaling servers, an orchestrating server, and a database for storing test results. The user interface implements the mechanisms for users to view and select from available peers, view results of previous measurements, and initiate new measurements after the signaling servers and orchestrating server identify peers within the same LAN. The orchestrating server is also equipped to serve as a WebRTC peer to measure the connection outside of a users LAN. After each initiated measurement, the results are stored to the database for future analysis. Figures 3-2 depicts these key modular components and how they interact.

Combined, these steps are designed to allow peer-to-peer network performance measurements without server connectivity or involvement. The components external to the LAN are only utilized for data gathering (IP and peer discovery) and data storage (database). The connections shown in Figure 3-3 can be easily extended to the scenario where the chosen peer is server-based.

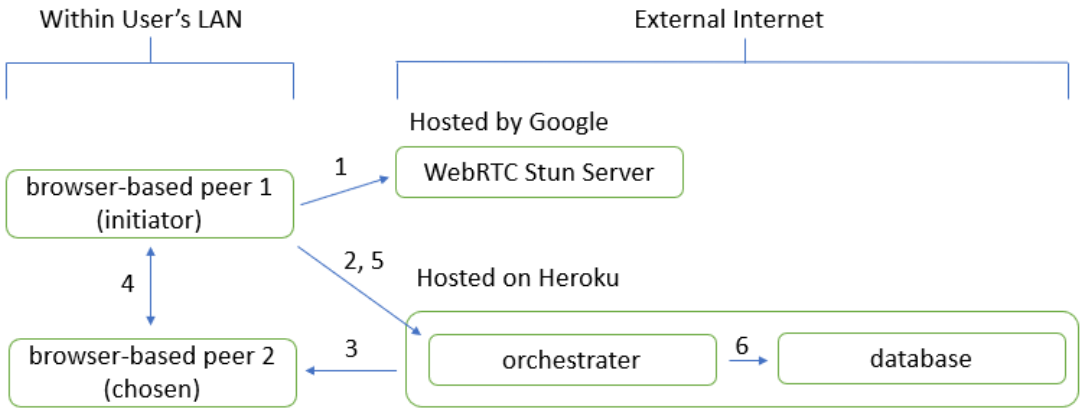

Figure 3-3: Diagram of WebRTC test infrastructure including the ordered steps for communication from information and IP discover (1) to storage of test results (6)

3.3 Key Components

In this section, we provide a detailed explanation of each of the five main components of the test infrastructure described in Section 3.2.

3.3.1 Peer Machines

All peers connect to the orchestrator, a NodeJS server hosted on the internet. Peers communicate to each other and perform network performance measurements via WebRTC data channels. There are two types of peer machines: browser-based and server-based.

Browser-based peers are created within the browser of an end-users device, a computer that supports Google Chrome. One physical machine can host multiple browser-based peers, each in a separate tab, though hosting multiple peers on same machine can decrease performance at the non-active browser tab. From the browser, users select other peers within the same LAN to which they want to perform the network performance measurements. The measurements must always be initiated from a browser-based peer.

Server-based peers are created within the orchestrator and function similarly to

a browser-based peer but lack a user interface (UI) with which users can interact or invoke commands. Server-based peers are always the receiving peer and cannot initiate test measurements to other peers. A server-based peer can maintain multiple peer connections simultaneously to browser-based peers.

The initiating peer selects a testing profile (described in Section 3.5) that determines the benchmark performance requirements against which the performance measurements will be compared. The initiating peer also records measurements for loss, latency, upload rate. The receiving echoes received packets back to the initiating peer during the measurement test and records the download speed for the measurements. Because JavaScript within the browser is single-threaded, more complex computation is performed in Web Workers spun by the peer instance to allow larger data processing with minimal effect on the runtime of the main test code.

3.3.2 Signaling Servers

The Session Traversal of UDP Through NATs (STUN) servers used are operated by Google. Potential peers within the same LAN that connect to the web application initially communicate with the STUN server to determine their public IP address. The public IP address is then shared with the NodeJS server and used to determine available peers within the LAN between which a speed test could be run. In addition, when server-based peers rely on signaling messages relayed through a Traversal Using Relays around NAT (TURN) server because they are located outside of the LAN of a user's browser-based peer.

3.3.3 Orchestrator

The orchestrator is a NodeJS server, hosted on Heroku, a free web hosting service, serves mainly to accept connections from new peers on WebSocket connections, and assign them to a group or room associated with the machines LAN based on the public IP address of the newly connected peer. When a new peer is assigned a room or a peer disconnects via closing or reloading the browser, the orchestrator sends a

broadcast to all machines in the room notifying them of the up-to-date list of available peers in their LAN. The orchestrator also broadcasts the initiating peers request to connect to available peers by echoing the WebRTC offer and answer messages used to set up WebRTC data channels within the room.

3.3.4 Data Channel

WebRTC data channels support peer-to-peer communication between any two peers. The WebRTC data channel used in the tests run the configurable SCTP transport-layer protocol. Traditional TCP-based communication channels hide natural packet loss by resending for reliable delivery, require in-order delivery, and congestion control metrics necessarily cause congestion in order to detect the available capacity of the network. We use SCTP configured to be give packet delivery that is unreliable, unordered, and without retransmit such that the metrics captured represent the performance of the unaltered network connection.

After connected through a WebRTC data channel, the orchestrator is no longer needed to facilitate communication between peers. The connection provided by the WebRTC data channel is used to capture the test metrics for the network performance measurements. During the measurement test process, the peers communicate with a series of messages numbered for quick parsing and response from the receiver. The messages and their purposes are outlined in Table 3.1.

3.3.5 Database

The Postgres database hosted on Heroku receives the results of the peer-to-peer network performance measurements and records them for later analysis. The database stores the IP addresses, operating system, and browser version for both initiating and receiving peers, the benchmark performance profile, the measured average bandwidth, average loss, average upload rate, average download rate, if the initiating peer is behind a NAT, and time and date at which the measurements were completed. A separate table also stores the latency observed for each connection.

Type	Sent From	Purpose	Response
0	Initiator	Alert receiver to incoming test packets and how many bytes to expect over test round	Record test information and send Type 7 message to initiator
1	Initiator	Send packet with time sent and random string of data as payload. The network performance while delivering Type 1 messages are reported as test results	Send Type 3 message as ACK
2	Initiator	Signal end data packets used for current round of performance measurements	Send Type 4 message to initiator
3	Receiver	ACK received Type 1 messages.	Count message transmission as successful and calculate RTT for message
4	Receiver	Share the download speed experienced at the receiving peer	Record download speed, calculate averages for all metrics, store results on database
5	Initiator	Share averages for all metrics with peer for display in web app	Display test results
6	Either	Notify the other peer that the connection has been terminated by the user or with failed network connection	Close peer connection and data channel
7	Receiver	Share receiving peer's IP address, operating system, and browser with initiating peer	Record the information for storage later with test metrics
8	Initiator	Send test message to receiver with sending time attached	Measure latency from initiator to receiver. Send Type 9 message
9	Receiver	Share latency with initiator	Calculate average latency measured and display results

Table 3.1: Provides a detailed description of the messages used between peers during all stages of the network performance measurement process after the WebRTC data channel has been established. Messages 0 through 9 relate the organization of the network performance measurements. Messages 8 and 9 are required for the latency metrics captured upon creation of the WebRTC data channel to understand the connections baseline performance.

3.4 Traffic Generation and Measurement Design

Using this list of components, we created tests composed of staged sending intervals to measure LAN performance at high demand. Data test packets (Message type 1) are sent over the WebRTC data channels according to the target bandwidth of pre-defined network profile described in Section 3.4 or according to user-input requirements.

3.4.1 Network Performance Profiles

Network Profiles consist of a target sending rate and a maximum round trip time (RTT) metrics. The pre-designed network performance profile represents the requirements slightly greater than that of high definition video streaming, a common high-intensity internet application that will be encompassed by Gigabit network connections. In the pre-defined profile, the target sending rate is 100 Mbps and the threshold RTT is 100ms. Users can also create custom network performance profiles by inputting new pairs of RTT and target sending rate within the UI.

We transmit Type 1 messages during staged sending intervals to measure LAN performance at the level of demand specified by the network performance profile. Data is sent over the WebRTC data channel at the specified sending rate (100 Mbps by default) for 10 seconds, long enough to overcome the ramp up time for the connection. Within the 10 second interval for the rate, bursts of data messages are sent in 50 milliseconds windows to provide stress on the network that averages to the target sending rates.

3.4.2 Captured Metrics

First, we capture a latency measurement. Upon connection through the WebRTC data channel and before the network performance measurements begin, we automatically trigger a latency measurement between the connected peers. The latency measurement consists of five ping-like messages sent from the initiating peer to the receiver. The latency is calculated as the difference between the time stamp

in the message and the time at which the receiver obtains the message. The latency displayed is an average of the calculated latency metrics.

The test traffic transmitted as Type 1 messages is used to observe the loss percentage (p) and the experienced round-trip time RTT. Combined with the preset values for MSS and the constant $p = 1$, the observed metrics are used to calculate the throughput (BW) using the Mathis Equation

$$BW = \frac{MSS * C}{RTT * p} \tag{3.1}$$

at the end of each 50ms round. Then, we compute the average of the throughput calculations over all rounds and report this and other averaged metrics to the user and store in the database.

The data messages payload combines a time stamp and random data. RTT is measured as the time difference between the time stamp in the data message and the time at which the initiating peer receives the ACK message from the receiver. We calculate a loss event when the observation of a gap between the sent times of messages in relation to the RTT and more information is available in Section 4. In this thesis, the bandwidth measures the data rate as seen by an application above the network transport layer in bytes per second, which is useful for measuring link capacity in bulk transport conditions. This measurement does not include transport layer headers or re-transmissions that are not part of the data delivered to the application layer [6].

Additional metrics captured include upload rate, which is the number of Type 1 messages transmitted through the data channel by the initiating peer within the sending time windows, and the download speed, which is the number of Type 1 messages processed by the receiving peer within the sending time windows.

3.5 User Interface

The user interface utilized by internet users who desire to test their internet connectivity. Several iterations of interface designs were used before deciding on the final form, which is influenced by a study of layouts of existing browser-based network

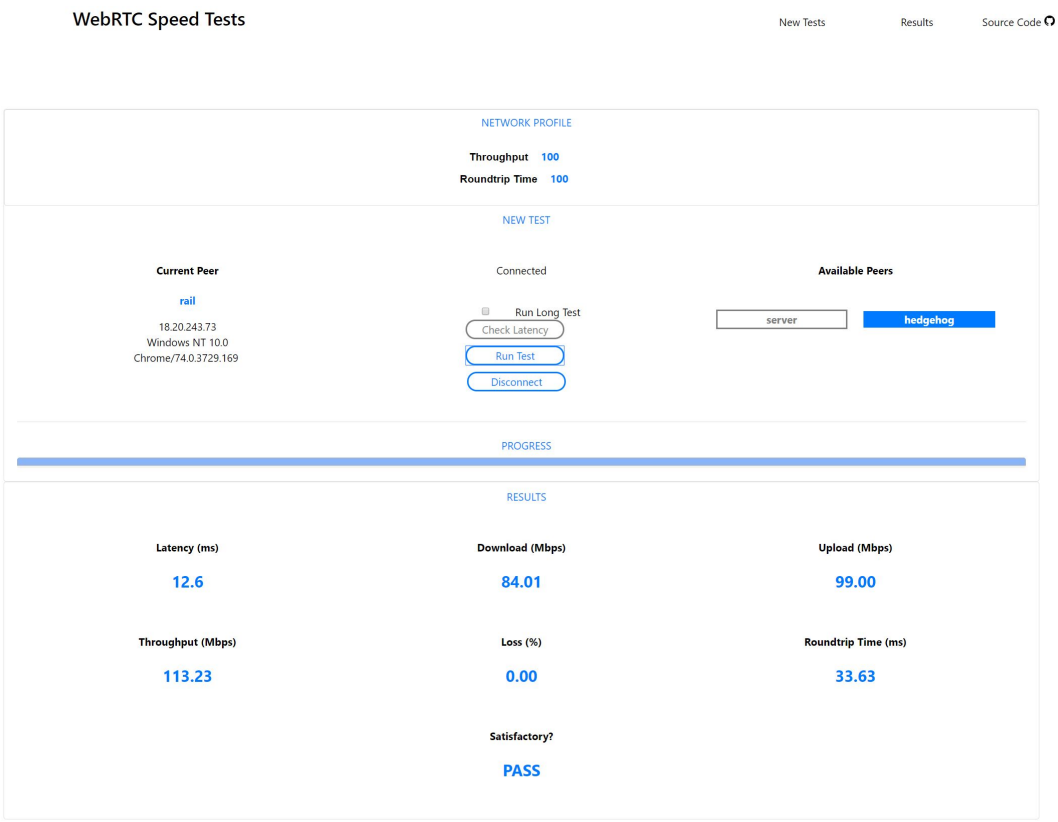

Figure 3-4: The main user interface in the browser through which new browser-based peer instances are created and network performance measurements are initiated. This image shows the results of a successful test between the browser-based peer and a server-based peer.

speed tests available for free, including those provider by ISPs like ATT, Verizon and Xfinity and independent test providers such as Ookla, Google, and Netflix. The user interface was designed for simplicity and ease of use for the user despite the additional information required to be displayed for peer-to-peer testing as seen in Image 1. The user interface is available at webrtc-tests.herokuapp.com and functions on the Google Chrome browser up to Version 74.0.3729.157.

In addition to the test UI where users can initiate new measurements, the website also enables users to view historic data of how their internet connectivity measurements taken using our test framework. The Results page displays information only from the users LAN, based on IP address, as seen in Image 2.

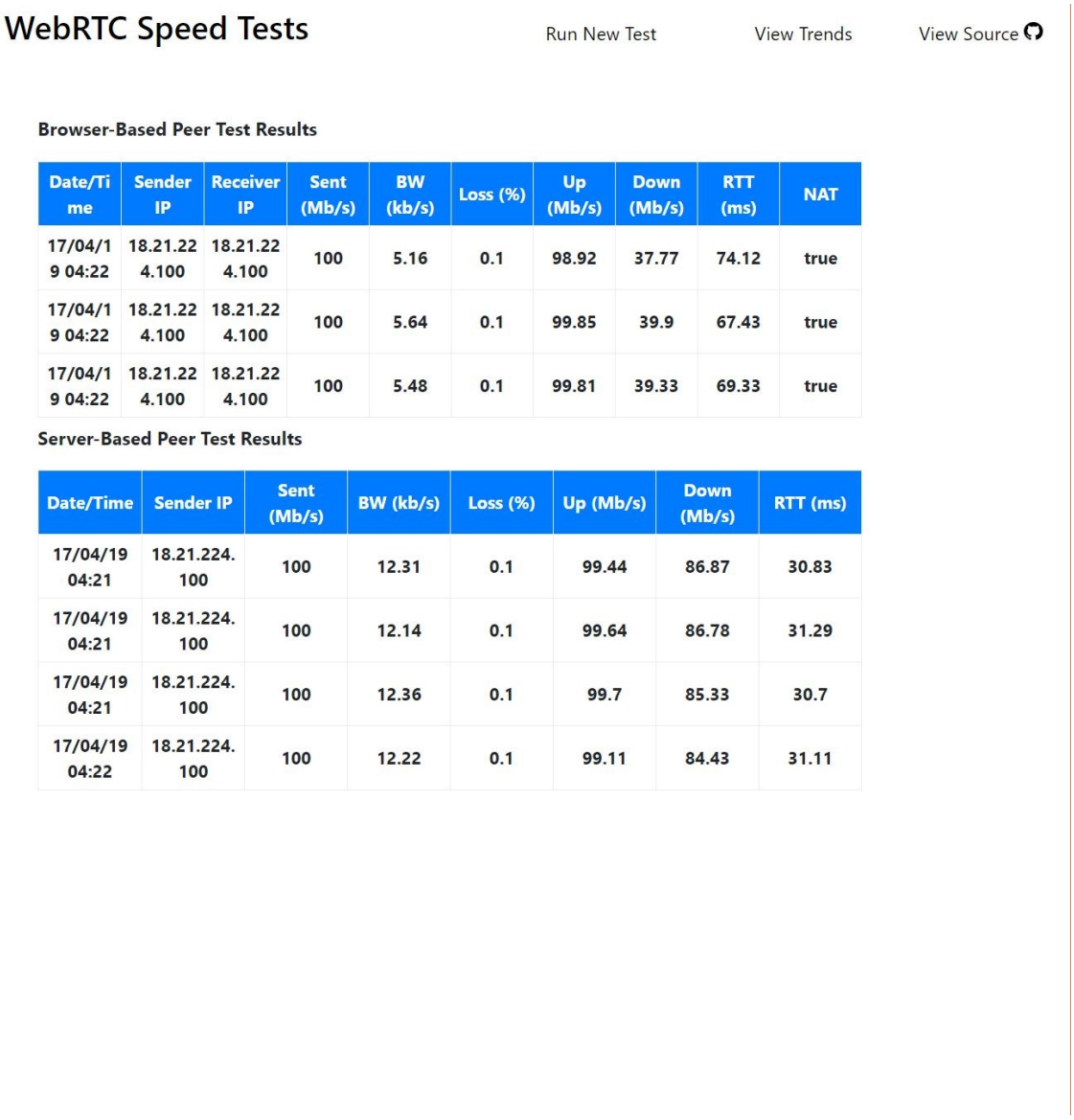

Figure 3-5: The page in the user interface where users can view data from past test completed within their current LAN.

Chapter 4

Model-Based Statistical Evaluation

The final step in the test framework outlined previously is network performance evaluation. In this thesis, a series of statistical tests are used to evaluate whether the internet connection between the peers is performant enough to obtain the benchmark performance required by the chosen network profile. We use the Model-Based Metrics (MBM) network performance measurement test framework designed by Mathis and Morton in Model-Based Metrics for Bulk Transport Capacity [1]. MBM statistically evaluates the observed round-trip time (RTT), maximum segment size (MSS), and throughput against the target values defined by the chosen network performance profile. The MBM system can be used for a variety of network performance profiles, measurement specifications, and network models.

4.1 Benefits of Model-Based Metrics Evaluation

The MBM system helps achieve the goal of vantage-point independent network performance measurements. To achieve vantage-point independence, we send test traffic (Type 1 messages as described in Table 3.1) across a given link (WebRTC peer-to-peer data channels) while observing the loss rate and end-to-end RTTs taken to complete the communication between peers. After finishing the transmission of test traffic, we compare the observed metrics to the performance in the chosen network profile that the link should be able to support and determine if the local-area network

is able to achieve the target throughput. We use MBM to evaluate the performance of links in local-area networks.

4.2 Evaluation Steps

This section details the implementation of the network performance evaluation which provides a pass, fail, or inconclusive result after analyzing the captured metrics. The output result is determined by the links ability to carry data above the target throughput by a sufficient statistical margin, to have an observed packet loss rate below that of a suitable TCP performance model, and that the link or longer connection between the two peer machines has sufficient buffering capability to overcome bursts during data transfer. The metrics and mathematical steps required to compute the performance evaluation are described in the sections below.

4.2.1 Required Measurements

As described in Section 3.4.2, the test suite we developed measures latency, round-trip time (RTT), loss ratio, upload speed, download speed, and maximum segment size (MSS). From a subset of these raw measurements, we calculate the observed throughput along the connection between two peer machines. The evaluation stage of the test framework begins by determining if the performance meets test preconditions then continues to evaluate whether the loss rate is low enough to support the target throughput.

4.2.2 Test Preconditions

The test traffic sent over a link must satisfy some traffic requirements before the statistical evaluation of loss rate can be utilized. The test preconditions include meeting the target roundt trip time RTT and the target throughput (BW) for the chosen network performance profile. A test will return a failure result if both preconditions are not met (the observed values are equal to or higher than the preconditions) be-

cause lower observed performance signifies that the link was not capable of meeting the required performance conditions. If the performance requirements are met, then we further statistically evaluate the performance as described in the next few sections.

4.2.3 Calculating Maximum Acceptable Loss Rate

During the evaluation stage, we compare the observed loss rate along the peer-to-peer connection to a theoretical cutoff loss rate. The cutoff loss rate is the maximum loss rate with which the connection can still support the test traffic under the target conditions specified. We calculate the theoretical cutoff loss rate (p_0) using the target throughput BW_{target} and the target round trip time RTT_{target} (as specified in the test profile at the beginning of the measurements) in addition to the MSS_{target}, which is preset to 1460 in our tests traffic simulations. Following the MBM RFC, we first begin by calculating the target window size W, the average window size in messages needed to meet the *target-rate*, based on the target performance conditions:

$$W = \frac{BW_{target} * RTT_{target}}{MSS_{target}} \tag{4.1}$$

Using the target window size W, we next calculate the target run length R, the minimum number of messages or packets that must be transmitted between loss events in order to meet target performance conditions. The target run length can be calculated:

$$R = (\frac{3}{2}W)(2W) = 3W^2 \tag{4.2}$$

Finally, the calculate the maximum acceptable loss rate p_0 using the target run length:

$$p_0 = \frac{1}{R} \tag{4.3}$$

The observed loss rate p then must be less than the cutoff loss rate p_0 in order to meet test requirements.

4.2.4 Statistical Evaluation of Observed Loss Rate

According to Mathis, observed loss events are not defects but signals, indicating that the transport protocol should slow down in a rate-controlled sending mechanism like TCP. In this these, each loss event also demonstrates that the connection cannot maintain the current transmission rate without failure. As test traffic is transferred between the peers, the evaluation module calculates the loss rate by the number of incomplete and severely delayed message end-to-end transmissions along the WebRTC data channel.

We perform the Sequential Probability Ratio Test (SPRT) [15] to determine which hypothesis is best supported by the observed loss ratio p. The measured p of the link is statistically determined to be low enough to support the specified performance criteria if p $< p_0$ was hypothesis H_0. The alternative hypothesis, H_1 is the situation in which observed loss p is determined to be too high to support target performance when if p $> p_1$ where the MBM IETF Internet-Draft defines $p_1 = 4^* p_0$. When p is larger than p_1, the observed network performance indicates that the connection between the peers would not be able to sustain the target performance criteria under TCP traffic flow because high number of loss events and frequent congestion signals would prevent the senders window size from remaining large enough to transmit traffic at the given rate.

Given these definitions, we can specify the pair of hypotheses for performing SPRT as given below:

$$H_0 = p \leq p_0 = \frac{1}{R} \tag{4.4}$$

$$H_1 = p \geq p_1 = 4 * p_0 = \frac{4}{R} \tag{4.5}$$

In these calculations and in the MBM paper, each message transmission is assumed to follow a Bernoulli distribution with probability of successful end-to-end transmission defined as $1 - p$ and probability of the message being lost in transmission defined as p. We compute our evaluation based on the assumption that packet

Result	Reasoning
PASS	Test preconditions are met, and the statistical test accepts H_0 with a 95% confidence level. The observed loss rate affirms that connection can support the traffic for a given target performance.
FAIL	Test preconditions are met, and the statistical test accepts H_1 with a 95% confidence level. The observed loss rate DOES NOT affirm that the connection can support the traffic for a given target performance.
INCONCLUSIVE	Test preconditions are met, but neither the H_0 nor H_1 can be accepted because the observed loss rate lies between the boundary lines. More data would be required to obtain a definitive result.
ERROR	Test preconditions are not met.

Table 4.1: The four possible outcomes of statistical evaluation of the network metrics captured. "PASS" means that the performance criteria in the network profile was met. Otherwise, the link could not support the criteria specified.

loss probabilities are independent and identically distributed.

4.2.5 Test Evaluation Output

We establish four potential outcomes from test evaluation by comparing the observed loss rate to the decision boundaries informed by the two hypotheses and by determining whether the test preconditions were met. The outcomes are explained within Table 4.1.

Two decision boundaries represent accepting one of the hypotheses. A loss ratio p above the X_{accept} lines signifies accepting the H_0 that the loss rate is low enough to support the target conditions. A loss ratio p below X_{reject} signifies that the connection has failed to sustain the target conditions, in which case we H_0 in favor of H_1. Observed loss ratios in between the two boundary lines are labeled inconclusive. The boundary lines are based on the specified Type I and II error; we use Type I error = $\alpha = 0.05$, and Type II error = $\beta = 0.05$. The boundaries for hypothesis testing are defined as below:

$$AcceptanceLine : X_a = -h_1 + s * n \tag{4.6}$$

$$RejectionLine : X_r = h_2 + s * n \tag{4.7}$$

where and n is the number of total messages and increases linearly during the transmission periods and

$$h_1 = \frac{\log \frac{1-\alpha}{\beta}}{k} \tag{4.8}$$

$$h_2 = \frac{\log \frac{1-\beta}{\alpha}}{k} \tag{4.9}$$

$$k = \log \frac{p_1 * (1 - p_0)}{p_0 * (1 - p_1)} \tag{4.10}$$

$$s = \frac{\log(\frac{1-p_0}{1-p_1})}{k} \tag{4.11}$$

.

Chapter 5

System Evaluation

In this chapter, we evaluate the efficacy of the tests. First, we evaluate the WebRTC channel and determine if it is appropriate for low-latency applications like testing. Then, we explore the ways in which the WebRTC Based Network Performance Measurements developed in this thesis contribute positively to the network testing ecosystem by providing a more accurate depiction of predicted network performance in likely internet applications.

5.1 Evaluation Methodology

We evaluate the test framework developed in this thesis in two ways using the web application that supports the WebRTC Based Network Performance Measurements. First, we observe the latency achieved through successive runs of the test through the web application. Second, we observe other metrics such as throughput, upload speed, download speed, loss rates, round trip time, and the output of the statistical evaluation methods.

We evaluate our network test over live WiFi network links on the MIT free network ubiquitous on campus with over 4,000 wireless access points. LAN testing between local peers evaluates the connection between the sending peer, nearest router, and receiving peer. Testing over the WAN from a local peer to a hosted server transmits data over the WiFi links mentioned before and through any additional network in-

terconnects until the server is reached, similarly to traditional network tests. The difference in network links used is demonstrated in Figure 5.1 and 5.2.

We maintain the network profile parameters throughout all of the tests to evaluate the networks ability to maintain high speed data into the era of Gigabit internet. We set the target throughput to 100Mbps and the *target-rtt* to 100ms. These parameters were chosen based on real-world scenarios in which a user would be streaming video (high throughput requirements for user experience) from a far away or potentially international server (high RTT due to distance and interconnects).

5.2 Peer-to-Peer Local Area Network Testing

5.2.1 Test Configuration

Next, we evaluate the WebRTC based network performance measurements by measuring and reporting the connection between two peer machines on the same LAN. Again, we measure connectivity on MITs open wireless network. We conduct a series of tests between multiple peers. First, between two Microsoft Surface Book laptops running Windows 10. Second, between a Surface Book laptop and an Android Phone. For each pair, we test them at different distances (5ft from access point and 30ft from access point) to cause changes in RTT even within the LAN. This follows the network test model displayed in Figure 5.1.

5.2.2 Results

Performing our WebRTC-based network performance measurements between the two laptops and between the laptop and the smart phone gives fairly consistent results within the pairs. The throughput calculated over 4hr periods are displayed in Figure 5-2 and 5-3 for laptop and smart phone peer-to-peer evaluations, respectively.

The majority of the variation in throughput between the two positions is determined by the difference in RTT. The difference in RTT is more pronounced among the laptop to phone pairing than the laptop to laptop pairing. A comparison of the

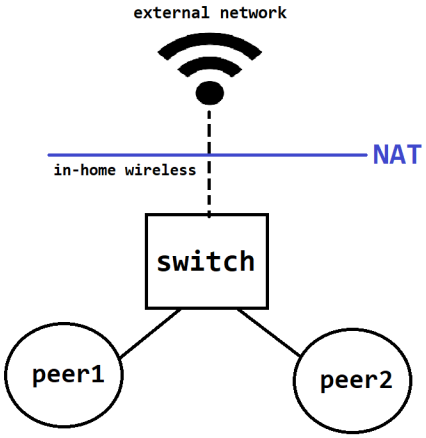

Figure 5-1: A LAN test between two local peers in the same WiFi network

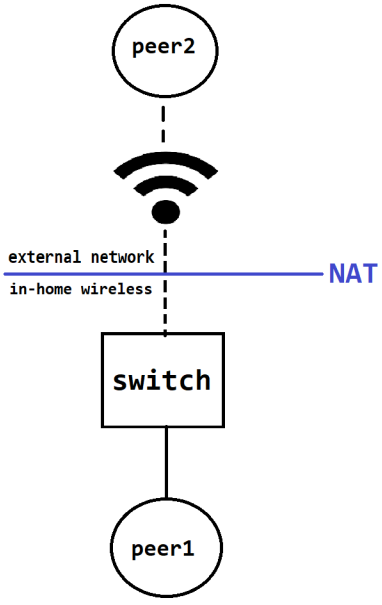

Figure 5-2: WAN test between local peer1 and hosted server acting as peer2

RTTs for the laptop to laptop measurements are shown in Figure 5-5 and for laptop to phone measurements are shown in Figure 5-6. In general, the laptop to laptop measurements have smaller variation in RTT even given the larger distance covered and have RTT around half as long as those measured by the laptop to phone pairings. For long distance measurements, the laptop to phone pairing has wide spread (over 100ms).

The high RTTs measured between the phone and laptop peers negatively affect throughput experienced. In Figures 5-3 and 5-4, the experienced throughputs (solid lines) are compared to the throughput required to meet the performance requirements specified in the network profile (dotted line) and obtain a PASS outcome according to the statistical analysis conducted after each run. While the laptop to laptop connection link was able to meet or surpass the target performance metrics for almost every run, the connection to the cell phone failed to meet the target throughput about half the time for short distance measurements (5ft from access point) and failed to meet the target throughput at all for the long distance measurements (30ft from access point).

5.2.3 Discussion

By comparing the throughput achieved in each of these pairs, we hope to demonstrate that the test outcomes still provide worthwhile insight because of the use of Mathiss throughput calculations based on varying RTT. Because the loss rates were consistently low or zero across the pairings and distances measured, the main variability in throughput measured was caused by changes in the RTT for messages to be transmitted from sender to receiver and back again. This observation is consistent with Mathiss concept of network behavior and is appropriate to be captured in network performance tests.

While differences in system specifications and capabilities can affect test performance, our WebRTC based performance measurements provide an understanding of networks given the system capabilities of the peer machines used. The RTT can be affected by multiple factors. For example, the weaker signal receivers on mobile

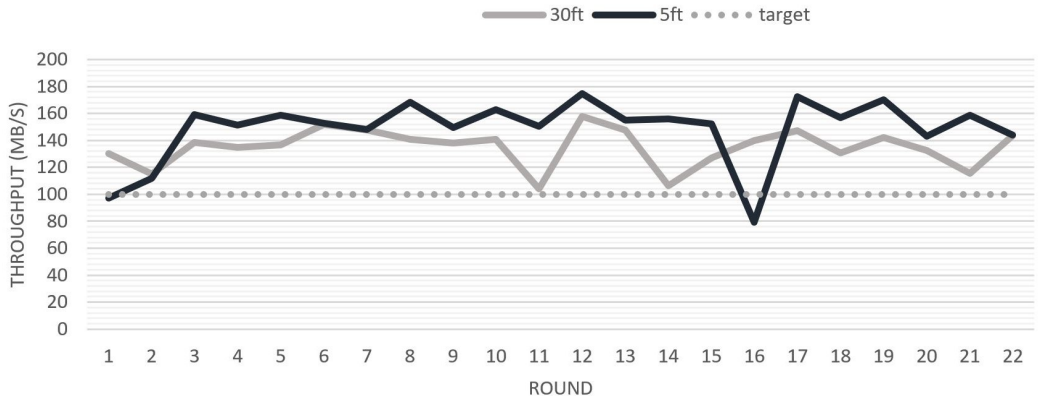

Figure 5-3: Laptop to laptop throughput measurements for peer-to-peer connection evaluation

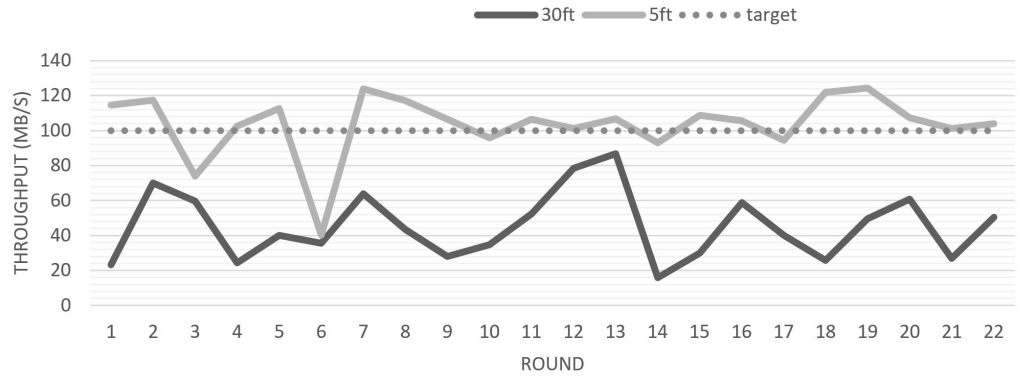

Figure 5-4: Laptop to smart phone throughput measurements for peer-to-peer connection evaluation

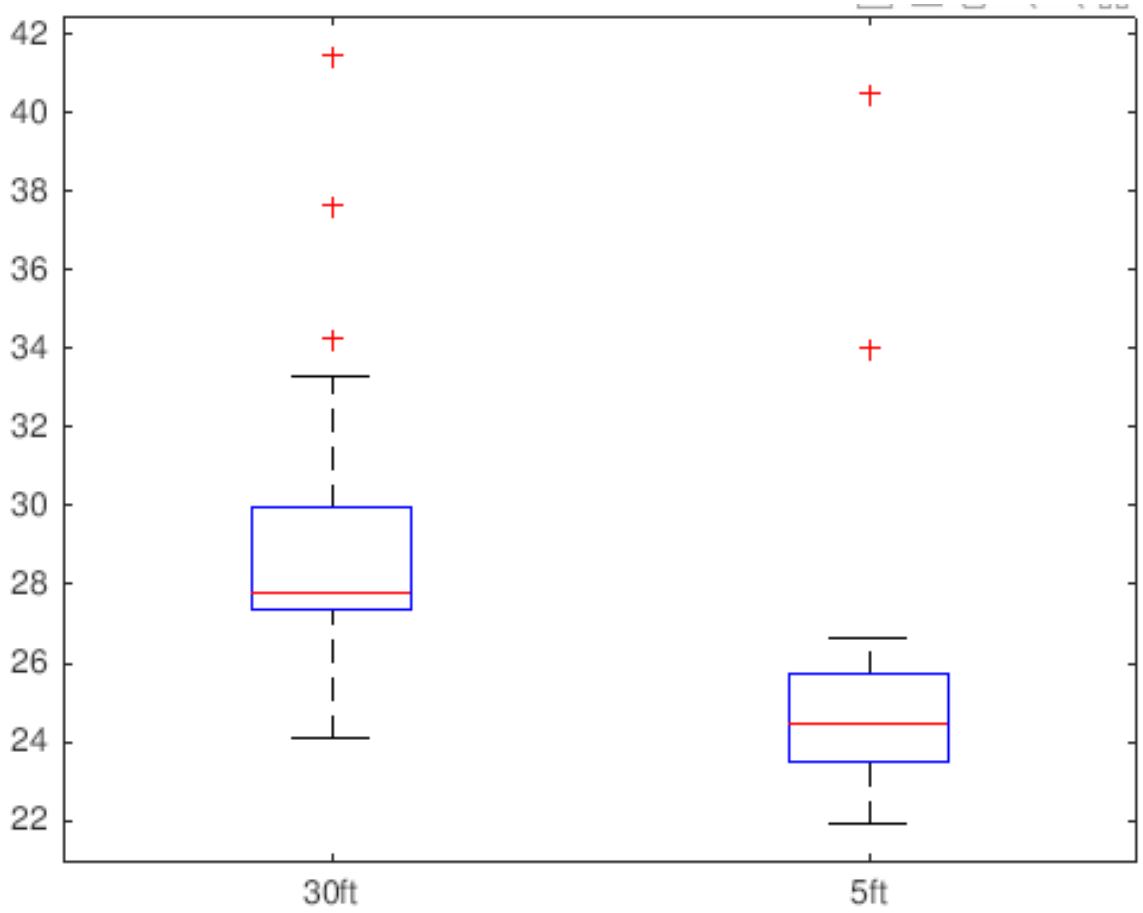

Figure 5-5: Box plot of distribution of RTT measurements (in ms) for laptop to laptop tests

phones may have impacted the large RTTs measured at the longer 30ft distance. Knowing the limitations of the machines used to measure can help users understand which test suites or frameworks are more appropriate for their system.

5.3 Wide Area Network Testing to Servers

5.3.1 Test Configuration

In the final evaluation step, we compare measurements captured with our WebRTC based network performance measurements to those captured by Speedtest.net

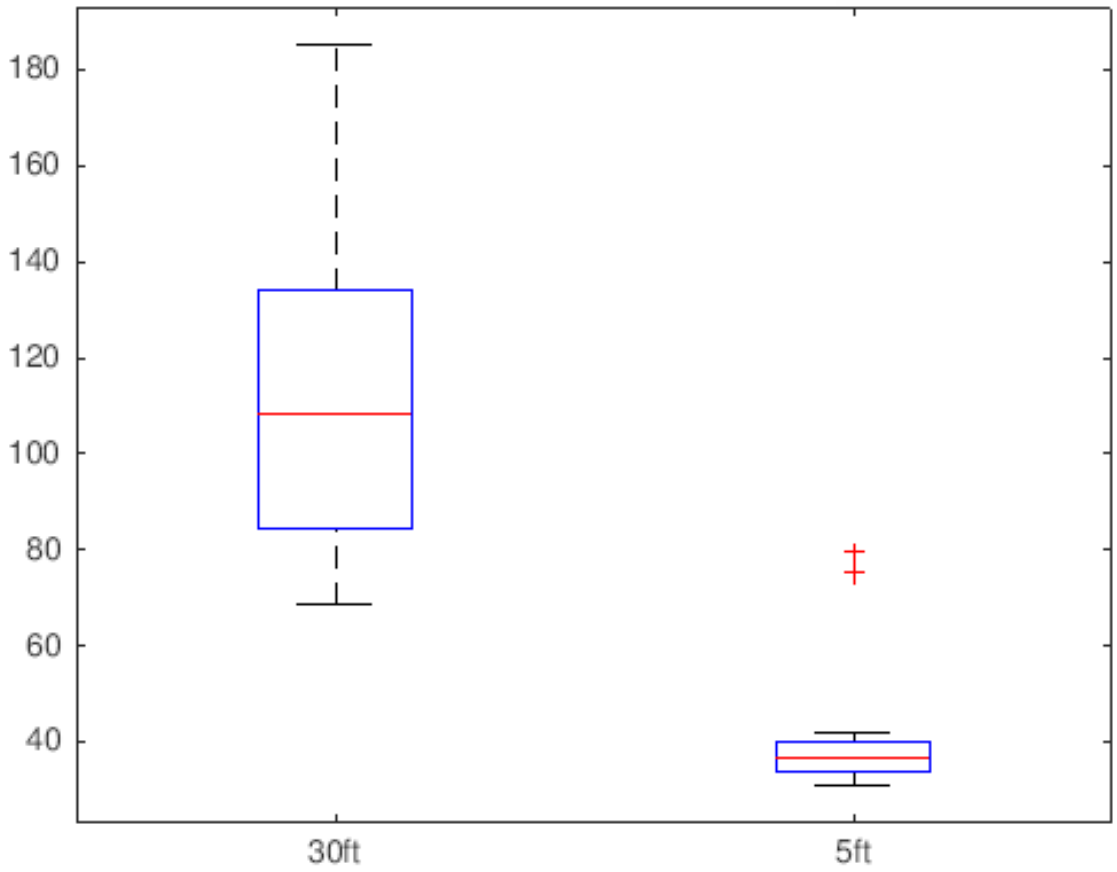

Figure 5-6: Box plot of distribution of RTT measurements (in ms) for laptop to mobile phone tests

by Ookla. Over an hour, we measure the connectivity between a peer machine and the WebRTC server hosted on Heroku, the Boston-based server hosted by Comcast (the default server chosen by Ookla to be closest to the user), and the Speedtest.net server hosted in Chicago. We then compare the impact that distance between the home network and the test servers has on the test results. This follows the network test model displayed in Figure 5-2.

5.3.2 Results

We observed from the test measurements that the performance experienced over a network link decreases as the distance between the sender and the destination increases. First, latency and RTT increase because packets must travel further and through more additional network interconnection points for more distant servers. Transmission time is also affected because of natural limits on the speed of data transmission (the speed of light in ideal circumstances), and performance is additionally constrained queuing in case of congestion and varying link capacities.

The upload and download speed also decrease with increasing distance to the destination, or test server in this case. According to Mathiss model for network throughput, the throughput experienced along a link is inversely related to the RTT. Likewise, upload and download speed are related to throughput. In traditional TCP connections, upload speed is regulated by congestion controls and rate limitations based on the current network throughput. Download speed is also negatively connected to decreasing throughput because downloads cannot occur faster than new packets arrive. Existing network speed tests like Speedtest.net assign the nearest server by default, and the reported results portray a false picture of network performance based on the inaccurate assumption that most internet content users will like is nearby.

5.3.3 Discussion

Our WebRTC based network performance measurements and certain runs of Speedtest.net give conflicting results and ideas about the internet connectivity on campus. First, we notice that the results from Speedtest.net vary greatly depending on the test server chosen and the servers proximity to the us as the user. Although Speedtest.net was determined to be the most accurate of the currently available internet speed tests [7], the results depend a lot on the distance (measured in RTT) between the user and the test server.

This dependency can cause the data to be misinterpreted by users who do not understand the relationship. In addition, the dependency causes inaccurate results. Many users will run a speed test if their connection to a website or online service seems slow. Most online content is not as close to the user as the default Speedtest.net server, and the results of the test does not account for the additional delay caused by retrieving data from far away data centers.

We compare the upload speed, download speed, and RTT between the Speedtest.net measurements and those captured by our tool. Because Speedtest.net does not report speed, we compare latency and upload and download measurements. We make two main observations on how our tool improves the existing testing ecosystem. First, our WebRTC based tests give an accurate view of network performance because of the application-centered design methodology that considers typical use cases and allowing users to determine the performance requirements against which to measure their network connection based on their own online behavior.

Second, our tests consider the fact that users may be further from servers distributing online content and a medium-distance server more accurately captures the RTT, and subsequently, the throughput, they would experience on average to an online service. Heroku offers web hosting in data centers spread across eight global regions. Website locality is ensured by hosting the application in one of the two North American data centers closest to the location where the site content is uploaded [16]. In our case, because the application is developed in the Boston area, we

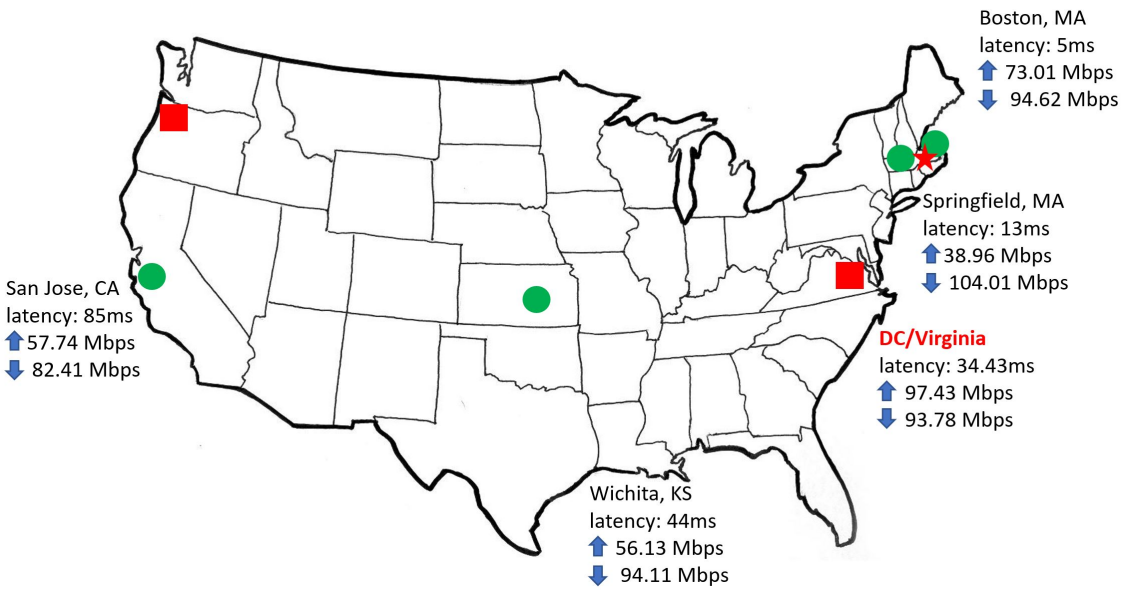

Figure 5-7: Demonstrates the increasing latency that accompanies access to servers that are further from the user. Here, the red star designates the Boston area from where our tests originate. Speedtest.net servers are identified with green circles and Heroku data centers where web applications are hosted are designated with a red squares.

are guaranteed hosting space in the Herokus data center in Virginia.

Chapter 6

Conclusion

This thesis demonstrates a methodology for testing local area networks and implements the methodology in a practical web application. The WebRTC based network performance measurements explained in this thesis provide several contributions to the network measurement space including isolated LAN network speed tests, a platform-independent web application and user interface, and an implementation of Model Based Metrics to provide a more accurate model of network throughput.

6.1 Continued Discussion

Chapter 5 demonstrates that our network speed tests capture the affect of RTT on observed throughput as described in Mathis Model in Equation 2.1. We also show that testing to servers that are outside of the vicinity of the user provides a more accurate model of the RTT to typical internet services, and, as an extension, results in a more accurate understanding of expected network throughput and quality of service. Below, we discuss the MIT networks and the system limitations of the peer machines used to understand their potential impact on the results reported in the previous chapter.

We conduct each set of network tests over wireless networks provided by MIT. MIT serves as its own ISP and operates several open wireless networks on campus, the "MIT", "MIT SECURE", and "MIT GUEST" WLANs, are now all running on Cisco

8540 wireless controllers and were updated in March 2019. Email correspondence with MITs Information Systems and Technology office gave important insight into the networks capabilities: Cisco Aironet 3702E APs serve as indoor wireless access points (AP) on campus and can support up to 802.11AC. Each wireless controller is connected to a router via a 4 by 10Gbps (40Gbps) LACP link, and the routers are then connected to the MIT backbone routers at 40Gbps. The backbone routers and the MITnet border routers are connected with 100Gbps links with various other network providers including multiple commodity Internet service providers. These external peerings can range from 5Gbps up to 100Gbps depending on the network [17]. Note that the target performance metrics from the evaluation section are within the advertised capabilities of the MIT network.

Next, we describe the system specifications of the peer machines utilized in the evaluation of our online tool. The laptops used are both first generation Surface Book computers and the Samsung S8 smart phone contain network interface cards compatible with IEEE standards for 802.11 ac that support data rates up to approximately 867 Mbps and 300 Mbps, respectively [18]. Both systems have been independently shown to support speeds such as those used in this thesis.

6.2 Limitations of the Browser

Providing an implementation through a web-based application in the browser has several tradeoffs. While consumers are more familiar with browser-based applications, some precision and performance benefits are lost at the application level. In addition, while the WebRTC peer-to-peer data channels are performant enough to support low-latency applications like network measurement, there are performance tradeoffs in using cutting edge technology.

One drawback to working in the browser is the lack of consistently synced system clocks across machines. There are two main functions for determining time. First, the Date.now Javascript function gives the milliseconds since January 1, 1970 according to the machines internal clock and only ensures accuracy to the second. Next,

the performance.now function represents the milliseconds since the browser session was begun on that machine. The difference in the timing methodologies produces noticeable differences across the two timing measurements taken as part of our network tests. Because the RTT is calculated as the time required to transmit from the send to the receiver and back, using the performance.now function returns consistent timing metrics because the same browser session is used. However, there is more variance in the latency measurements recorded using the Date.now function because the system clocks are rarely synced between two peer machines. The system clocks may be set to similar enough times to be useful for typical daily tasks for people, the discrepancy can cause illogical results in end-to-end latency measurements. For example, in one of the rounds between the laptop and the smart phone, the measured RTT is 19.03ms while the latency is -1632.15ms. The lack of clock synchronization produces non-negligible effects.

An additional tradeoff of designing our WebRTC based network performance measurements in the browser is that JavaScript is single threaded. In our implementation, we use Web Workers to run data analysis as a background process to not interfere directly with the performance of the main window process in which test traffic for the measurements are transmitted and received. However, WebRTC peer-to-peer data channels do not currently function in the Web Worker background processes and additional processing time taken in the main thread increases RTTs.

6.3 Future Work

There are interesting improvements that can be made to the existing application and evaluation methods:

First, the performance limitations of using WebRTC in the browser could be mitigated by conducting an in-depth analysis of the message analysis, sending, and receiving functionalities that may cause processing bottlenecks. Once the primary bottlenecks are isolated, the functionalities can be simplified as necessary to reduce processing time with the hopes of reducing RTT. As the WebRTC infrastructure

continues to mature, support for operating the data channels in background workers has already been spoken about and may be added.

Second, the web application should be released to the public. Evaluating our WebRTC based network performance measurements and the accompanying web application on a real-world network has already provided insight into the practical implementation of Model Based Metrics and understanding maximum and experienced LAN throughput. Expanding the system evaluation to the general public would allow us to test on a larger pool of networks and understand the situations in which our methodology and network models are appropriate. In addition, a public release of our web will allow users to measure the performance of their in-home networks while contributing to a growing data set of LAN throughput and loss measurements that could inform public policy.

Bibliography

[1] Matt Mathis and Al Morton. Model based metrics for bulk transport capacity. *IP Performance Working Group: RFC 8337*, February 2017.

[2] Shirley Hung S[teven] Bauer, William Lehr. Gigabit broadband, interconnection, propositions and the challenge of managing expectations. *TPRC 43: The 43rd Research Conference on Communication, Information and Internet Policy*, 2015.

[3] Renata Teixeira Srikanth Sundaresan, Nick Feamster. Home network or access link? locating last-mile downstream throughput bottlenecks. *PAM 2016 - Passive and Active Measurement Conference*, pages 111–123, March 2016.

[4] Merry Mou Steven Bauer, William Lehr. Improving the measurement and analysis of gigabit broadband networks. March 2016.

[5] Matt Mathis et al. A framework for defining empirical bulk transfer capacity metrics. *IETF Network Working Group RFC 3148*, July 2001.

[6] Matt Mathis et al. The macroscopic behavior of the tcp congestion avoidance algorithm. *Computer Communication Review, 27(3)*, July 1997.

[7] William Lehr Steven Bauer, David D. Clark. Understanding broadband speed measurements. *TPRC*, August 2010.

[8] William Lehr Steven Bauer, David D. Clark. The evolution of internet congestion. *TPRC*, August 2009.

[9] Xiaohong Deng Yun Feng Srikanth Sundaresan, Danny Lee and Amogh Dhamdhere. Challenges in inferring internet congestion using throughput measurements. *IMC*, November 2017.

[10] Francis Lepage Fabien Michaut. Application-oriented network metrology: Metrics and active measurement tools. *IEEE Second Quarter, Volume 7 No.2*, 2005.

[11] Ookla. Speedtest.net. http://www.speedtest.net/. Accessed: 2017-10-05.

[12] Netflix. Fast. https://fast.com/. Accessed: 2017-10-05.

[13] Mary Baker Kevin Lai. Measuring bandwidth. *IEEE*, 1999.

[14] Google. Test wi-fi speed to devices on your network. https://support.google.com/wifi/answer/7578264?hl=en. Accessed: 2019-04-27.

[15] Abraham Wald. Sequential analysis. *John Wiley and Sons, Inc.*, 1947.

[16] Heroku. Regions. https://devcenter.heroku.com/articles/regionsdata-residency. Accessed: 2019-05-12.

[17] Brian Stephens. "inc0289851 resolved — questions about the specifics of mitnet for master's thesis". Email. Message to Miranda McClellan. 14 May 2019.

[18] Marvell. Documentation 88w8897a featuring 802.11ac, nfc and bluetooth 4.2. https://www.marvell.com/wireless/88w8897/. Accessed: 2019-05-11.