A First Look at 1 Gbps Fiber-To-The-Home Traffic

Matthew Sargent*, Brian Stack*, Tom Dooner*,
and Mark Allman†

TR-12-009

August 2012

Abstract

Fiber-To-The-Home (FTTH) networks are on the cusp of bringing significantly higher capacities to residential users compared to today's commercial broadband options. While fiber is available to some consumers now, the capacities offered represent an incremental bump over more traditional DSL and cable options. However, a number of high capacity experimental FTTH networks have recently become operational. One of these is the Case Connection Zone (CCZ) which connects a neighborhood adjacent to Case Western Reserve University with bi-directional 1 Gbps paths to each house. In this paper we present myriad observations from monitoring CCZ traffic for the past 14 months. We aim to better understand how users employ these high capacity links, how much of the capacity they utilize and conduct an initial exploration of why performance is limited to less than the raw capacity of the fiber.

* Case Western Reserve University, 10900 Euclid Avenue, Cleveland, Ohio, 44106
† International Computer Science Institute, 1947 Center Street, Ste. 600, Berkeley, California, 94704

This work was partially funded by the National Science Foundation (NSF) through grant number CNS: 0831535 ("Comprehensive Applications Analysis & Control). Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation.

1. INTRODUCTION

Fiber-To-The-Home (FTTH) networks are on the cusp of bringing significantly higher capacities to residential users compared to traditional commercial broadband offerings. Commercial ISPs have been offering on the order of ten megabits/second fiber service for several years (e.g., Verizon's FiOS, AT&T's Uverse). However, several research projects have started to connect residential users with significantly more capacity: Google will connect 850 homes near Stanford university [1] and 50K–500K more homes in Kansas City [11], Chatanooga's power utility has connected 100K homes via fiber and started to offer network services [3] and Case Western Reserve University has an operational testbed of roughly 90 homes near campus connected via fiber [2]. Finally, the United States' "National Broadband Plan" calls for 100 million residential connections to have at least 100 Mbps downlink and 50 Mbps uplink networks—or 1–2 orders of magnitude more than current commodity networks—by 2020 [19].

As we have thought about these networks of the (near) future, we find ourselves coming back to two basic questions:

- *What will users do with significantly higher capacity?* Traditionally, residential network capacity has lagged behind content. In other words content providers have been producing ever increasing quality of content (e.g., HD and 3D video) in advance of commodity residential networks ability to transmit the content. The envisioned FTTH networks effectively leap-frog our current content and hence a natural question is whether and how users will leverage the increased capacity.
- *Are our protocols up to the task of utilizing significantly higher bandwidth in edge networks?* In other words, do the processes we impose in our general protocols (e.g., TCP, HTTP) impose a bound on performance that makes it difficult to use significantly higher capacity?

In this paper we tackle both of these big picture questions in the context of monitoring Case Western Reserve University's Case Connection Zone (CCZ). This unique university experiment connects each of roughly 90 residences adjacent to campus via a bi-directional 1 Gbps fiber link. After discussing the CCZ network, our data collection and the calibration of our data in the next section we then address these questions from a number of angles in the remaining sections in the paper.

We stress that this paper is an initial study. Without fail, each analysis we present begs additional and deeper questions that are not addressed in this paper. Our goal in this paper is to gain an initial empirical understanding of the workings of a high capacity FTTH network in the wild. This study is (as far as we know) the first word on the topic and by no means anywhere near the last word.

2. DATA

We use a packet-level monitor to record the traffic at the border between hosts on the CCZ and the broader Internet. The CCZ user population is roughly 60% students and 40% full-time residents of the neighborhood [10]. Each of the CCZ homes is connected via a 1 Gbps fiber. These fibers ultimately come together in a switch which is served by a 1 Gbps link to the broader network. Traffic is mirrored to a switch port where our monitor can observe and record the traffic. This switch can represent a choke point given the 100x mismatch between the aggregate capacity to the residences and the capacity to the Internet. However, we never observe aggregate usage of anywhere near 1 Gbps. As detailed in § 5 we generally find daily average utilization to be around 10 Mbps. Our data collection period for this paper ranges from January 25, 2011 through March 31, 2012. We collect two general kinds of data as detailed in the next two subsections.

We stress that we are not claiming our data is in some way "*representative*" or "*typical*". We present our analysis as an initial look at a single FTTH edge network. While no single vantage point will give a conclusive picture we believe this study provides an initial look at these emerging networks. Indeed we encourage the community to replicate our analyses in additional FTTH environments such that we can gain a better overall sense of the area. Additionally, we note that this study also provides additional data about residential networks, as others have studied, e.g., [13].

2.1 Protocol Behavior Logs

First, we use Bro 1.5.2 [17] to continuously monitor the CCZ network and record various traffic logs using its collection of protocol analyzers. We denote this dataset $\mathcal{L}_x$ where x identifies the particular protocol log. In particular, we record the following logs that we use in our study:

Connection Logs: $\mathcal{L}_c$ includes information about each transport-layer connection[1]. For each connection these logs include the start time, involved IP addresses and port numbers, number of bytes transmitted in each direction, duration of the connection and ancillary information (e.g., whether TCP's three-way handshake was successfully completed).

HTTP Logs: $\mathcal{L}_h$ includes records for each HTTP [9] transaction. These records include the IP addresses and port numbers of the underlying TCP connection, the type of request (GET, POST, etc.), the URL being

[1]Including UDP transactions, even though these are not strictly speaking "connections".

requested, the response code from the server and the number of bytes of each request and response.

BitTorrent Logs: $\mathcal{L}_b$ includes records for BitTorrent [8] transactions. These records include IP addresses and port numbers for all observed peers as well as myriad information about the protocol, such as *have* messages, *choke* and *unchoke* messages and handshaking information.

The log data—especially $\mathcal{L}_c$—requires calibration before use such that measurement errors and ambiguities[2] do not skew our analysis. This turned into an arduous multi-step process as outlined below.

Bro will report in $\mathcal{L}_c$ information for all attempted connections. For 14 days during our collection period we hit a bug in Bro whereby each arriving packet was reported as its own "connection". While often this manifests after some particular time within the day we remove all such logs from further analysis as obviously Bro was not working properly. Since we cannot readily re-create the situation that triggers this bug we were not able to determine its cause. It may be caused by some network effect and hence bias our analysis by preventing us from observing that phenomenon. However, given the relatively small number of days removed we do not believe this has a large biasing effect. Further, even if we understood the cause, the data is left in such a state such that accurately analyzing it would be impossible.

For the remaining days in the $\mathcal{L}_c$ dataset we find a large number of "connections" that are essentially bogus for the purposes of further analysis. For instance, scanning traffic that sends a SYN to a non-active service port will show up in the connection logs as a "connection", but accomplishes no useful work. We filtered connections from our analysis for five basic reasons:

Filter F.1: As noted above, $\mathcal{L}_c$ includes both successful and unsuccessful connections (e.g., port scanning, or contacting an unresponsive host). This latter type of connection typically manifests as either (i) having a duration that is either zero or unknown or (ii) reporting unknown values for the number of bytes sent in one or both directions. Since these entries are incomplete, we remove them from further analysis unless otherwise specified.

Filter F.2: While monitoring the network, Bro logs unexpected events in its "notice" and "weird" logs. We can then use these unexpected events to further remove connections from $\mathcal{L}_c$ that may skew our analysis. An example unexpected event that we have found particularly useful is an entry that indicates Bro has observed a TCP ACK for sequence number x, but has only observed data sequence numbers through y (for $x > y$). In constructing the $\mathcal{L}_c$ logs Bro will assume all bytes through x have been transmitted. However, given the missing data this may be a wrong assumption. We assume that in the cases where the gap—i.e., $x - y$—is large the difference is caused by an errant ACK and not by measurement loss (since we have verified that measurement-based loss is rare, see § 2.2). By using the size of the reported gap—i.e., $x-y$—we remove connections from further consideration for which we do not observe large amounts of the data. We verified our assumption by spot checking potentially problematic entries in $\mathcal{L}_c$ with packet-level traces (see § 2.2) and we indeed find connections with relatively few packets and a large errant ACK, which corrupts the byte counts.

Filter F.3: Given that one errant packet can lead to a gross mis-calculation of a connection's size we started an augmented logging scheme on January 11, 2012 using Bro's "conn-stats" policy. Using this the $\mathcal{L}_c$ logs include packet counts (in each direction). This makes it straightforward to filter out cases where we observe a disproportionate amount of data volume for the number of packets observed.

Filter F.4: Additionally, the logs in $\mathcal{L}_c$ contain Bro's version of each connection's "history". The history records (for each direction) whether Bro has observed: SYNs (and SYN+ACKs), data-carrying packets, ACKs, FINs and resets. We can then use this to ensure connections progress as expected. We adopt the following criteria (and order) for removing connections from further analysis: (1) Connections with no SYN. (2) Connections with no SYN+ACK. (3) Connections reporting multiple SYN+ACKs with intervening packets (i.e., not simply retransmitted SYN+ACKs). (4) Connections reporting non-zero data bytes sent for a given direction, but for which no data-carrying packets are observed. (5) Connections that have no history of acknowledging transmitted data.

Filter F.5: The above checks winnow $\mathcal{L}_c$ to what looks to be a reasonable set of connections in the logs taken after January 11 2012—i.e., when we started recording packet counts and hence could leverage those in our filtering process. However, we are left with a small number of connections that if accurate would reflect an extremely fast sending rate (e.g., just under 1 Gbps). We suspect these connections are not accurate since after January 11, 2012 we no longer find such high transmission rates. Therefore, we adopted the following strategy. We observe a maximum aggregate rate for one CCZ IP address of roughly 160 Mbps when using the filtered post-January 11, 2012 logs. Based on this, we set a per-connection threshold of 200 Mbps—25% higher than the maximum observed aggregate—and remove from further analysis any connection that occurs before January 11, 2012 and exceeds the threshold. This is a

[2]We are not the first to notice such in the Bro connection logs. E.g., see the discussion in [6] about the ambiguities of determining a connection's final "state"—e.g., "properly terminated"—from these logs.

Filter Type	Pre-Jan/11	Post-Jan/11
All Conns.	849M	166M
F.1	365M	44M
F.2	11.6K	2.1K
F.3	—	6.2K
F.4	30M	6.8M
F.5	3.8K	—
Remaining	454M	114M
	53.5%	68.7%

Table 1: Connection filtering breakdown.

fairly crude filter. However, given the lack of such high transmission rates after January 11, 2012 we are confident that Bro erred in some way when logging these connections.

In table 1 we present a high-level summary of the connections filtered from further analysis by our calibration strategy. Since we used a slightly different strategy for the logs before and after January 11, 2012 we list the results separately. We find that a significant fraction of the connections are removed for various reasons. This is not surprising given previous analysis of similar connection logs (e.g., [6] shows more than 90% of logged connections are caused by scanners and hence are junk). We find the bulk of the filtered connections are removed by filter F.1 due to incomplete information or by filter F.4 for showing an unexpected history. Since the logs before January 11, 2012 do not include packet counts filter F.3 is not employed. Likewise, filter F.5 was added as a heuristic in the absence of packet counts and hence is not used after January 11, 2012. In total we find nearly 570M legitimate connections over the course of our 14 month data collection.

Finally, recall that we find single hosts utilizing roughly 160 Mbps across connections and hence set a filtering threshold of 200 Mbps for a single connection in the pre-January 11, 2012 logs. Our final dataset contains 69 connections that show performance between 160–200 Mbps. We consider these connections to be somewhat suspect. However, since the number is small they do not overly skew our results in general.

2.2 Packet Traces

The second source of data is packet-level traces. The torrent of traffic precludes the capture of all traffic for the entire measurement period. We therefore collect packet traces from the 11^{th} through the 18^{th} of each month, as follows. We divided each day in the collection period into six hour blocks and then collect a one-hour trace starting at a random time within each block. While we captured full packet payloads, saving all such traces quickly became logistically burdensome. We therefore randomly chose one trace to retain in full for each day and stripped the payload from the remaining three traces. This leaves us with 7 hours of full payload traces and an additional 21 hours of header-only traces for every month in our collection.

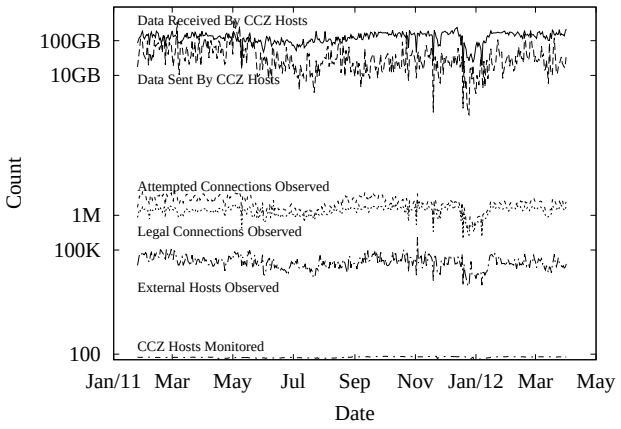

Figure 1: Overview of CCZ traffic.

Failing to collect all packets that cross our monitoring point can lead to biased or wrong conclusions. Therefore, before using our packet traces we must assess measurement-based packet loss. While *tcpdump* reports a number of losses this is often not telling as it is difficult to understand (and hence count) what the application did not observe. Rather, we analyze the traces themselves for signs of missing packets. In particular, we analyze TCP traffic for cases where we observe an acknowledgment for data we never observe. These "gaps" represent cases where the ultimate recipient clearly received the data, but the data was not recorded in our traces. This analysis provides only an indication of the measurement-based loss rate since it does not include all traffic (e.g., excluding non-TCP traffic).

In 330 of the the 392 traces in our dataset we find no measurement-based loss. In the remaining traces we find loss rates of (*i*) less than 0.001% in 43 traces, (*ii*) between 0.001% and 0.01% in 18 traces and (*iii*) 0.013% in 1 trace. Therefore, while individual measurement-based loss events may impact individual analyses we undertake, we conclude that in a general and statistical sense the measurement-based loss rate is low enough to not impact the insights we derive from our dataset.

2.3 A First Look

Figure 1 gives a sense of the overall CCZ network and traffic. The remainder of the paper will further analyze these aspects of the dataset, but here we aim to provide an overview to build the reader's intuition. First, we observe 90 local IP addresses active on each day of our dataset. This is as expected based on the scope of the CCZ project and because the project provides a standard router to each home that does NATing for all internal devices. In § 3.1 we further analyze the data to understand how many devices are active behind these NATs on each day. The figure also shows that local

hosts communicate with tens of thousands of remote peers each day. On median we observe traffic to 44K remote peers with the 5^{th} percentile at 20K and the 95^{th} percentile at 88K. We next turn our attention to the observed traffic. As shown in the figure, we observe a median of 2.3 million connections in $\mathcal{L}_c$ each day. Of these, a median of 1.3 million—or 57%—are valid (per the discussion in § 2.1). Finally, the top two lines on the figure show the number of bytes sent and received from/to CCZ hosts on a daily basis. The amount of data sent by CCZ hosts is roughly 30 GB per day on median, with a maximum of 411 GB. Meanwhile the amount of data received by CCZ hosts is 129 GB per day on median, with a maximum of 240 GB.

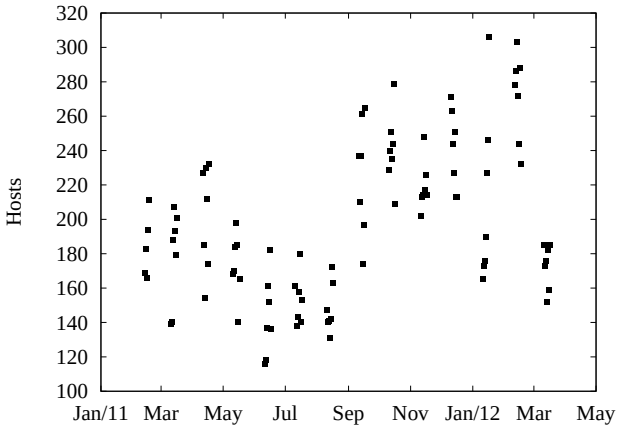

Figure 2: Number of unique devices observed.

3. ORIGINS AND DESTINATIONS

Our first set of analyses aims to understand traffic sources and destinations CCZ users employ.

3.1 Devices

We know that there are roughly 90 homes connected to the CCZ network, but we do not know the number of devices connected within these homes. The CCZ project provides each residence with a router which connects in-home devices to the fiber link via Wifi or wired Ethernet. This router acts as a NAT such that each house uses only a single public IP address. We use a technique similar to that described in [14] to count hosts behind these NATs. For each of our 392 hour long packet traces we run *p0f* [21] to obtain SYN-based operating system determinations for each connection. Further, if payload is present—as it is for 98 of our traces—*p0f* reports the HTTP user agent string associated with HTTP requests.

To determine the number of devices behind a particular NAT we start with the number of operating systems produced by the SYN-based signatures, denoted S_i for each CCZ IP address i. This is a lower bound on the number of devices. For instance, if we find a particular CCZ IP address x has traffic from both Windows 7 and OSX we can determine that at least two devices are present behind the NAT, but we cannot determine how many of each device is present. Therefore, S_x will be set to two.

For the connections on which we also have HTTP user agents we look for opportunities to augment our count by noticing that one OS has traffic from multiple browsers and assuming each device would have only one primary browser this indicates multiple devices. Continuing the example from above if we note OSX traffic from Chrome, Firefox and Safari then we would change OSX's device count contribution in S_x from 1 to 3 devices. Therefore, in total for this example we have $S_x = 4$. This use of the user agent string can in one sense be viewed as a lower bound. Similar to not being able to tease apart two Windows 7 machines based only on SYN signatures we cannot tease apart two Windows 7 machines both running Chrome, even with the extra user agent information. On the other hand, if our assumption that each device runs only a single browser in general is wrong we are overestimating the number of devices by attributing different browser traffic to different devices. We stress, however, that we aim to only roughly determine the number of devices behind each NAT, hence small errors are not problematic.

Finally, we note that our methodology is inherently conservative because our traces cover only 4 hours each day in which we can used SYN-based fingerprints and we have only a single random hour per day of user agent information. Therefore, we are no doubt undercounting the number of devices behind each NAT. This is likely especially true for mobile devices that may not be around when our traces are being taken.

Figure 2 shows the number of devices we find to be active on each day of our packet-level dataset. We find that like many of the other plots in this paper the graph is divided into three portions by the academic summer break. The general dip that we observe from May–August during various analyses corresponds to the decreased student user population during summer break. Before summer we generally observe 160–220 devices per day. In the summer we see the user population drop off as we only find 140–180 devices per day. After the summer we find 200–300 devices per day, a healthy increase over the previous academic year. We find the January and March 2012 device counts to be low. This is explained by the January data being collected before spring semester classes started and the March data being collected over spring break.

In terms of operating systems, table 2 shows the breakdown of the minimum, maximum and average percentage of various operating systems per day in our dataset. We find variability across time—which we largely as-

OS	Min	Mean	Max
Windows 7/8/XP	32	40	58
OSX	15	21	28
iOS	9	15	21
Linux	3	9	15
Other	8	15	20

Table 2: Minimum, mean and maximum percentage of OS observed per day.

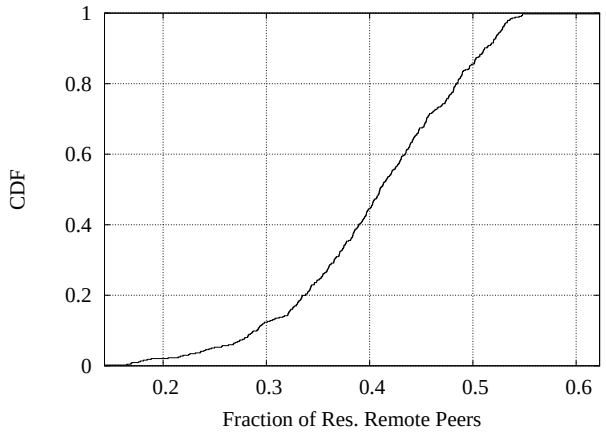

Figure 3: Distribution of residential remote peers.

cribe to our data collection methodology. As discussed above, with a more comprehensive view we believe the variability would be reduced. In general each OS's maximum population is twice its minimum population and the mean is the middle of the two ends of the spectrum. Also, we note that the median, while not shown, is ±1% of the mean in all cases.

3.2 Peer Location

We next assess the location of the remote peers in our $\mathcal{L}_c$ dataset. In particular, we assess the prevalence of remote peers that correspond to end-user devices and not servers. We use the SpamHaus PBL [20] to determine if a remote IP address is an end-user device. While the PBL is maintained with help from the owners of some address blocks, the list still represents an approximation. However, we note that the list is generally viewed as "good enough" for operational tasks such as email filtering as evidenced by the widespread use of the PBL. Colloquially we will refer to end-user devices as "residential"—even though strictly speaking they are not all in homes—and all other hosts as "non-residential".

CCZ hosts make valid connections to a median of 44K remote peers per day over the course of our $\mathcal{L}_c$ dataset. Figure 3 shows the distribution of the fraction of peers that are residential for each day of the dataset. We find that as little as 10% and as much as 55% of the peers are residential across the dataset. The median of the distribution indicates that approximately 41% of the remote peers are residential. In § 4.4 we revisit peer location with an eye towards correlating it with traffic characteristics.

3.3 Fanin / Fanout

We next aim to get a sense of the number of hosts each CCZ household interacts with on a daily basis. We do this by analyzing $\mathcal{L}_c$ for the number of hosts that each local CCZ household contacts ("fanout") and the number of remote hosts that initiate contact with local CCZ hosts ("fanin"). The top two plots in figure 4 show the median and maximum daily fanout across CCZ hosts, while the bottom plots show the fanin. The plots on the left side include all connections, while the plots on the right include only valid connections.

In terms of fanout we make several observations. First, we find that the median household contacts roughly 1,000 remote peers per day. This holds when considering all connections and only valid connections. We see slight growth in the median fanout over the course of our dataset, but we also see the expected dip during the academic summer break. Additionally, we find a short but pronounced dip in fanout during December of 2011. This occurs around the holidays and hence we believe this reflects a generally smaller user pool at that time (e.g., students leave for break, residents visit family).

While more variable than the median, we find the maximum fanout to be generally around 10K—an order of magnitude higher than the median. Similar to the median, the maximum is not heavily influenced by the removal of bad connections. While there may be small dips in the maximum fanout due to the academic calendar, we do not see as much influence on the heaviest users (in term of fanout) as on users in general.

We now turn to the bottom plots in figure 4 which show the fanin. The median fanin when considering all connections (bottom left) is between 50 and 100—with an increasing trend—across the dataset. However, when only considering valid connections the median fanin is generally zero (no point shown due to the log scale of the y-axis). The exception is from September to December we find the median to be a handful of valid connections initiated by remote peers. This matches our general model of remotely initiated traffic to a residence, which suggests that aside from peer-to-peer traffic there are unlikely to be many externally available services running in homes. Given that the results show an uptick of fanin when considering all connections and also the fanin does not show the dips during academic breaks we conclude that there are a moderate number of scanners probing CCZ hosts every day. The maximum fanin is not only larger but also more variable than the median. This again suggests that there is an effect from scanners

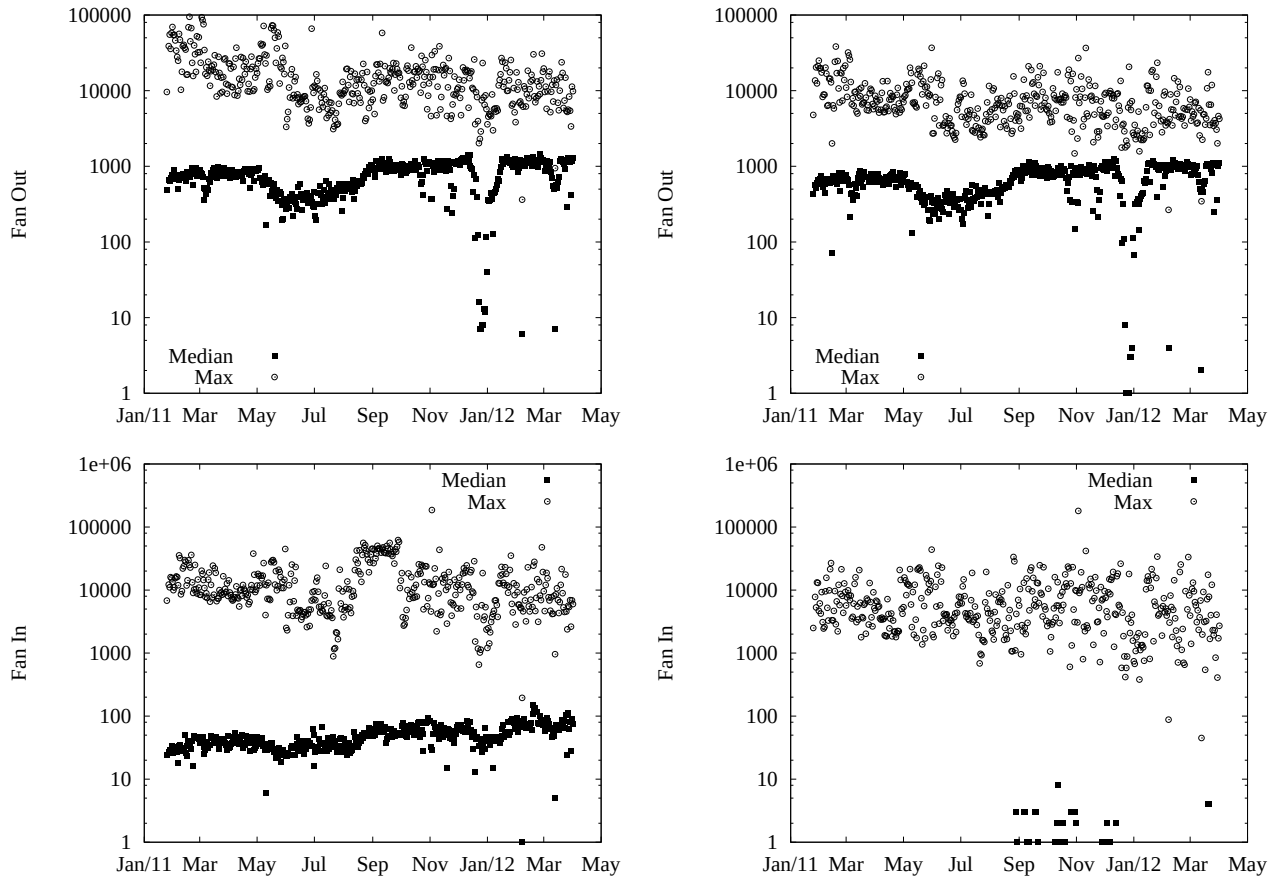

Figure 4: Fanout (top plots) and fanin (bottom plots) for both all connections (left plots) and valid connections (right plots).

at play.

Finally, we note that the average fanin is 514 and 202 for all connections and good connections, respectively, across the dataset. Further, we find the percentage of hosts that exceed this average is 13% and 10% for all and good connections, respectively. This shows that the distribution is skewed and a few hosts are generally responsible for much of the fanin. For fanout we find an average of 1,364 and 938 for all connections and good connections, respectively, across the dataset. We also determine that the percentage of hosts exceeding the average is 28% and 36% for all and good connections, respectively. This again shows skew towards a smaller set of hosts driving the aggregate fanout discussed above, even though the skew is not as pronounced for fanout as for fanin.

4. TRAFFIC MIX

We next turn our attention to providing an overview of the traffic patterns we observe.

4.1 Transport Protocols

We first briefly assess the transport protocols we find on the CCZ network. Our $\mathcal{L}_c$ dataset includes rich summaries of all TCP and UDP traffic. To understand what is not captured in the $\mathcal{L}_c$ logs, we analyze the packet traces. Over our 392 hours of packet traces we find that more than 99.8% of the packets are either TCP (69.3%) or UDP (30.5%). In the remaining traffic we find that ICMP makes up roughly 0.1% and a handful of other protocols each send much less than 0.1% of the packets across our dataset.

We use $\mathcal{L}_c$ for the remainder of the analysis in this section. In terms of connections, recall that figure 1 shows the overall number of connections per day in our dataset. We find that TCP generally constitutes roughly two-thirds of the connections per day—with a minimum contribution of 53% and a maximum contribution of 84%—while UDP makes up the balance.

Finally, in terms of the data volume we find that TCP carries over 99% of the data on each day in $\mathcal{L}_c$. This stands in stark contrast to the packet- and connection-based results above, which indicate that UDP's contribution is roughly one-third. This shows that UDP transactions, while frequent in number are small in size.

4.2 Heavy Hitters

We next turn to assessing each CCZ host's contribution to the aggregate traffic observed. Figure 5 shows the fraction of bytes sent to each CCZ host (left) and from each CCZ host (right). We sorted the hosts by their relative contribution (the sort is independent for each plot). The horizontal line shows the average per-host contribution. In terms of arriving data (left plot) we find that 31 hosts receive at least an average share of the traffic. Further, we find that the top host receives roughly four times as much as the average—receiving roughly 4% of the bytes across our dataset. Meanwhile, the smallest share is 10,000x less than the average.

The right-hand plot in the figure shows the relative data volume transmitted by CCZ hosts and shows a different result than when considering data received. We find that the top CCZ sender transmits nearly 20% of the data volume across our dataset! Meanwhile in relative terms we find some hosts contribute nearly nothing to the overall data volume sent. Finally, this direction is more skewed compared to the data received as only 17 hosts contribute at least the average share of data transmission volume.

Finally we note that there is some overlap between the top senders and top receivers. Considering the top ten hosts in each direction we find four hosts that appear on both lists. The host sending nearly 20% of the data departing from the CCZ network is not within the top ten in terms of data receivers. However, the top host in terms of data reception ranks fifth in terms of data transmission.

4.3 Applications

We now analyze our dataset to determine the top applications. We identify applications first by Bro's "service" determination in the $\mathcal{L}_c$ logs. This leaves a significant fraction of the traffic unclassified. In particular, while Bro logs fine-grained details about BitTorrent traffic in $\mathcal{L}_b$ it classifies such traffic as "other" or "http" in the $\mathcal{L}_c$ logs. We therefore analyze the $\mathcal{L}_b$ logs to determine which "other" and "http" connections to classify as BitTorrent and change their designation. Additionally, Bro does not recognize some traffic using TCP port 51314 as BitTorrent. We have found this to be the default port used by the Transmission BitTorrent client and the traffic we find on this port is consistent with BitTorrent activities and therefore we roll this traffic into the BitTorrent count. Finally, we use the service port number—i.e., the destination port of the SYN that starts a connection—to identify the remaining applications.

Using this process we find thousands of applications. To focus our attention on the most prevalent applications we winnow our analysis to applications that constitute one percent of either (*i*) the aggregate number of connections, (*ii*) the total traffic volume sent by CCZ hosts or (*iii*) the total traffic volume received by CCZ hosts. Using this approach identifies the nine most popular application protocols.

Service	Hosts	Conns.	Sent	Rcvd.
HTTP	90	242 M	789 GB	41 TB
Flash	89	343 K	4.4 GB	2.9 TB
BitTorrent	70	23.6 M	7.8 TB	2.3 TB
HTTPS	90	33 M	437 GB	1.1 TB
Steam	58	36 K	142 MB	584 GB
DNS	90	187 M	7.8 GB	43 GB
Other-1111	25	1.4 M	724 GB	37.4 GB
Other-8332	20	6.5 M	7.1 GB	8.2 GB
Minecraft	22	6.2 M	329 GB	7.2 GB
Unclassified	88	68 M	7.0 TB	3.7 TB
	98%	12%	41%	7%

Table 3: Aggregate traffic volume for popular application protocols across the entire dataset.

Table 3 shows the traffic breakdown for the top applications. There are two service port numbers that are ambiguous in our data. Port 1111 traffic seems to correspond to either Flash or Daodan malware. Port 8332 traffic seems to correspond to both a wireless security camera being tested as part of the CCZ experiment, as well as some BitCoin [7] traffic. Rather than settling on specific designations we leave these nebulous in the table.

The second column of the table shows the number of CCZ IP addresses we observed using the particular application over the course of our 14 month dataset. As expected we find DNS, HTTP, HTTPS and Flash are used by (nearly) all users. We find BitTorrent is used by over three-quarters of the users. The Steam gaming application is used by nearly two-thirds of the users. The remaining top applications are used by roughly one-quarter of the user population each. The last two columns of the table show the traffic volume in each direction. We find that even among the top applications there is a difference of four orders of magnitude in data volume across the list (for both directions). Further we find that the top application—HTTP for data reception and BitTorrent for data transmission—comprises an order of magnitude more traffic volume than the second ranked application.

Table 3 gives the amount of traffic we do not attribute to one of the top nine applications in the "Unclassified" row. As shown in the table, in terms of connections and bytes received the fraction of traffic involving other applications is modest and expected in that we know from previous studies and intuition that the top applications will not be responsible for all traffic. However, the percentage of bytes sent that are unclassified is 41%—which is striking. We know from our analysis to determine the top applications that no one port number is responsible for more than 1% of the bytes sent across any day in our dataset. This suggests that

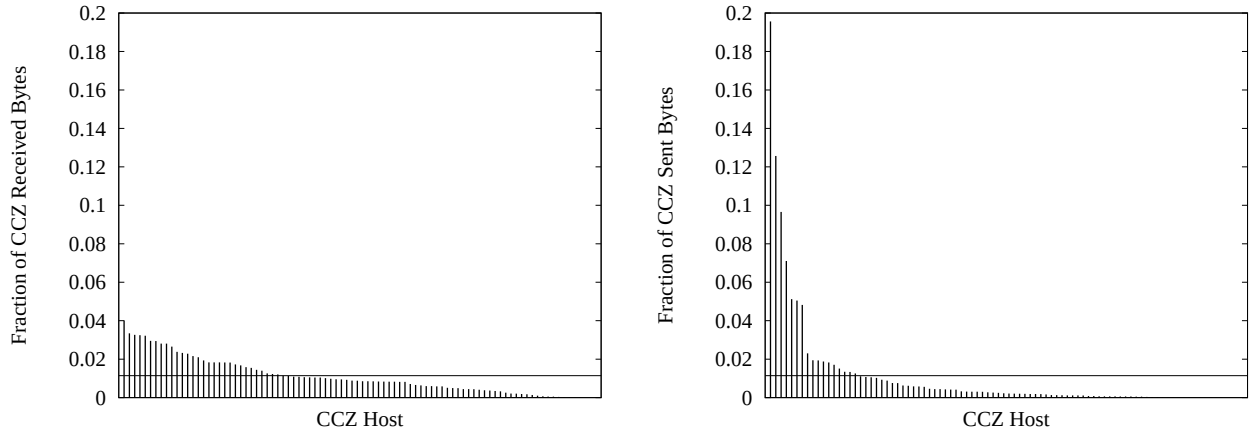

Figure 5: Relative contribution of each CCZ host for arriving bytes (left) and transmitted bytes (right).

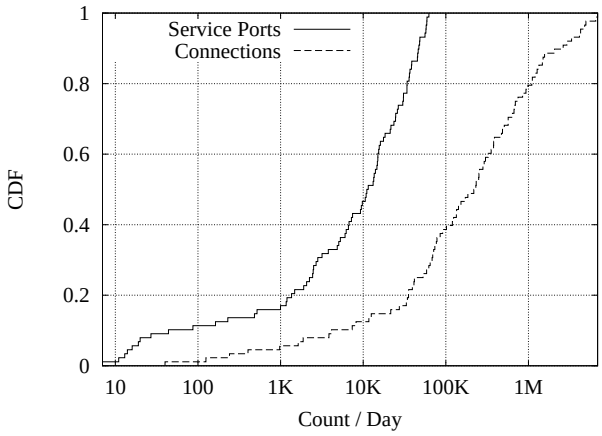

Figure 6: Distribution of the number of connections and service ports for each day.

this uncategorized traffic volume may well be using a large variety of ports as a policy evasion technique.[3]

In figure 6 we plot the distribution of both the number of unclassified connections per day and the number of distinct service port numbers found in the unclassified traffic. The difference between the number of service ports and the number of connections is roughly a factor of 20 across the distribution—with the difference growing towards the tail. We also find the number of service ports has a median of over 11K and a maximum of 64K, which is the number of possible ports. This illustrates the heterogeneity of the traffic and that it belies aggregation given the information present in the $\mathcal{L}_c$ logs. However, the port spread and the traffic volume suggests that much of the unclassified traffic is likely peer-to-peer traffic trying to avoid detection.

[3]The CCZ does not impose policy-based restrictions based on port number, but the remote peer could well be under such constraints.

Service	Host	Conn.	Sent	Rcvd.
HTTP	2.04	1.02	2.35	1.01
Flash	4.00	5.74	5.60	2.38
BitTorrent	5.01	3.81	1.13	3.19
HTTPS	2.97	3.47	3.14	3.85
Steam	5.99	6.75	6.76	5.03
DNS	1.01	2.02	4.80	5.79
Other-1111	7.60	7.64	7.52	7.65
Minecraft	7.93	6.33	5.38	7.69
Other-8332	8.45	8.22	8.33	8.40

Table 4: Average rank of each of the top application protocols per day.

While table 3 gives an overview of the entire 14 month dataset it does not give any indication of the volatility across time. To gain a sense of the relative contribution of each of the nine top applications we rank them by (*i*) number of hosts using the application, (*ii*) number of connections, (*iii*) bytes sent and (*iv*) bytes received for each day in our dataset. Table 4 gives the average rank of each of the top nine applications in each category across the entire dataset. In terms of the number of hosts using an application and the amount of traffic received the average daily rank is nearly identical to the aggregate rank (with only the last two applications swapping positions in both cases). This suggests that these rankings are fairly stable across our dataset. However, in terms of the data volume sent by CCZ hosts and the number of connections we find that only the top three or four, respectively, applications are identical across the aggregate and average daily ranking. After that the ranking diverges and hence shows that non-trivial daily variations in application usage do in fact occur.

4.4 Volume vs. Peer Classification

In § 3.2 we consider the prevalence of residential hosts

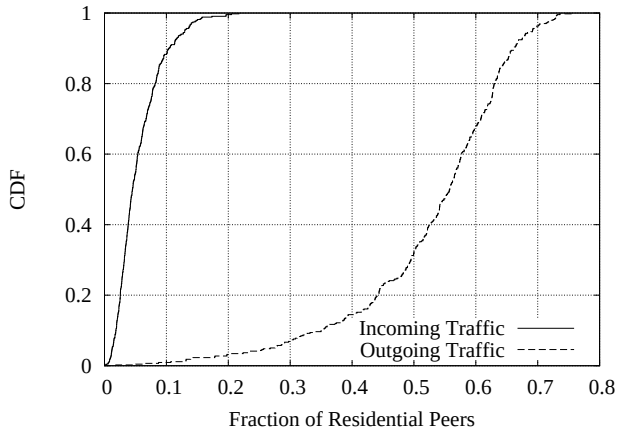

Figure 7: Distribution of the fraction of traffic exchanged with residential hosts.

in the set of remote peers with which CCZ hosts communicate. We now turn our attention to understanding the contribution of these remote peers to the overall traffic volume. Figure 7 shows distributions of the fraction of incoming and outgoing bytes exchanged with residential peers for each day in our dataset. The plot shows a dramatic difference between incoming traffic and outgoing traffic. We find that residential peers contribute around 5% of the incoming traffic on median and at most 20% of the incoming volume. However, we find that in terms of outgoing traffic the median is around 55% and the maximum is approximately 75%. This is consistent with the volumes found for various applications in § 4.3. The top applications in terms of received traffic volume are traditional client/server applications (e.g., HTTP) where we would expect to be receiving data from a centralized server rather than a end-user host. Meanwhile, in terms of outgoing data volume we find that the top application is BitTorrent, which is by its very nature a distributed system that relies on end hosts and not on infrastructure level servers. Therefore, the disparity in figure 7 is not surprising, even if striking.

4.5 Web Servers

Table 3 above shows that HTTP is by far the largest source of traffic flowing to CCZ hosts and is the second ranking application in terms of traffic sent by CCZ hosts. Given HTTP's prominence in the traffic mix we briefly assess the popular servers being accessed as an indication of how users are employing their large capacity networks. For this analysis we consider the "Host" header in HTTP traffic, which is recorded in out $\mathcal{L}_h$ logs. This is then correlated without our $\mathcal{L}_c$ dataset to produce traffic volumes for each site. We aggregate sites based on the second level domain.

Table 5 shows the top 15 web sites in terms of traffic downloaded to CCZ hosts. The table gives the site, the percentage of traffic volume the site contributes, the number of days the site is ranked in the top 15 sites across the dataset and then the minimum, median and maximum daily ranking of the given site. The table largely shows popular sites (e.g., as listed by Alexa [4]) and CDNs are the most common. Further, we find volatility in the top 15 sites. For instance, only three sites are within the top sites nearly every day in our dataset. Further, over half the top sites are ranked first at least once. We also find that some sites are bursty. An example is "espn.com" which ranks as high as two, is ranked in the top 15 only 62 times (roughly one-seventh of the days) and more than half the time ranks above 4,087. In this particular case, the likely cause is popular sporting events driving the access patterns (e.g., in particular the one-in-seven pattern suggests perhaps NFL football games have some influence). We also find that roughly one-third of the aggregate data volume is from video streaming sites. Another roughly 25% of the volume is from content distribution networks.

Site	Vol (%)	Days	Min,Med,Max		
youtube.com	19	420	1	2	5
llnwd.net	15	402	1	3	247
nflximg.com	14	416	1	3	60
edgesuite.net	8.4	335	1	4	145
apple.com	2.8	332	1	7	67
xvideos.com	1.5	268	4	9	60
hulu.com	1.4	101	1	19	119
fbcdn.net	1.2	245	5	10	26
megaupload.com	1.0	133	1	20	6574
filesonic.com	1.0	78	1	62	5012
espn.com	0.9	62	2	4087	6554
tumblr.com	0.8	97	5	16	223
akamaihd.net	0.8	91	2	24	5250
pandora.com	0.7	45	5	15	1291
google.com	0.6	9	6	17	44

Table 5: Top 15 sites in incoming traffic volume.

Site	Vol (%)	Days	Min,Med,Max		
google.com	20	420	1	3	8
facebook.com	6.8	420	1	2	10
youtube.com	5.8	419	1	4	13
datei.to	4.5	3	1	2	5321
4shared.com	4.3	26	1	1142	6721
yieldmanager.com	4.0	420	1	3	9
fbcdn.net	2.5	413	3	5	12
doubleclick.net	2.2	419	3	6	11
revsci.net	1.1	188	5	11	34
yahoo.com	0.9	146	1	13	38
google-analytics.com	0.9	151	7	11	20
adnxs.com	0.8	116	4	15	67
citizengroove.com	0.8	1	1	1642	3253
twitter.com	0.6	46	2	18	38
peeje.com	0.6	3	2	568	1835

Table 6: Top 15 sites in terms of outgoing traffic volume.

Finally, table 6 shows the top 15 web sites in terms of data transmitted by CCZ users to web sites. In this case we see more consistency in the sites across the dataset than for incoming traffic, as six of the sites are in the top 15 list nearly every day. However, we again find

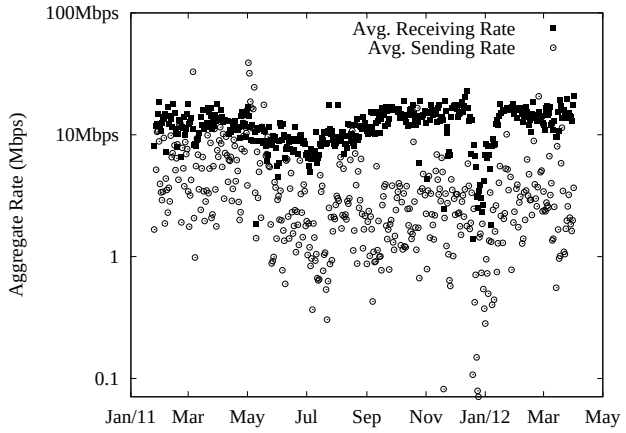

Figure 8: Daily aggregate sending and receiving rates.

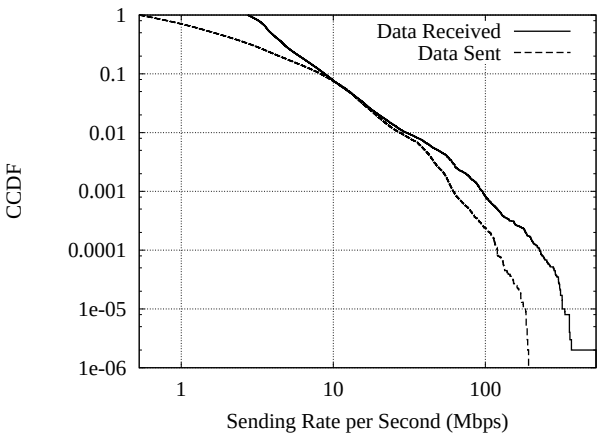

Figure 9: Throughput for top 1% bins.

volatility, as well. For instance, more than half the top 15 sites again reach the highest rank on at least one day. In addition, we again find bursty behavior. E.g., "citizengroove.com" is overall in the top 15 sites in terms of volume, but appears in the daily top 15 list only once—at the top position—while its median rank is 1,642. Finally, we observe some low-rate, but consistent sites such as Twitter, which only spends roughly 10% of the days in the top site list, but is ranked between 2–38 for each day in the dataset and ultimately accrues enough volume to be listed in the top 15.

5. OBSERVED TRANSMISSION SPEED

Thus far we have considered who CCZ users communicate with and what protocols and applications drive the communication. We now turn our attention to the salient feature of FTTH networks: speed. Figure 8 shows the aggregate sending and receiving rates across all hosts in the CCZ for each day in our dataset. The plot shows that the daily aggregate incoming traffic rate is roughly 10 Mbps. As in some of the results from previous sections we find dips in the plot during summer and winter break. In terms of local hosts transmitting data we find that in general the aggregate rate falls between 1–10 Mbps with similar modest dips during academic breaks.

5.1 Per-Host Speed

We will now focus on individual CCZ hosts.[4] For each connection in the $\mathcal{L}_c$ logs we evenly distribute the number of bytes transmitted over the duration of the connection into 1 second bins (86,400 bins per day). We track each direction independently. This even spreading of data across a connection does not reflect reality for two basic reasons: (i) applications do not send and receive data uniformly across the duration of a connection and (ii) TCP's congestion control algorithms [5] constantly adjust the sending rate based on the perception of the network conditions. However, both of these dynamics happen outside our view and therefore for this initial analysis using the average rate suffices. However, future work will include a more fine-grained packet-level analysis.

As sketched above, we break our data into 7.2 billion one-second bins—i.e., 86,400 bins for each direction, day and user in our dataset. To concentrate on periods when hosts are transmitting relatively rapidly we winnow our dataset to the top 1% (36M) bins in each direction. Figure 9 shows rate distributions for the top bins in each direction. The first point on each line shows the 99^{th} percentile of our entire dataset, which is a sending rate of approximately 0.5 Mbps and a receiving rate of roughly 2.7 Mbps.

The figure shows that more than 90% of the top 1% receiving bins represent a rate under 10 Mbps. In other words, 99.9% of the overall bins do not exceed a rate available from common commodity residential networks.[5] Or, on average each user spends approximately 14.5 minutes per day employing higher-than-commodity network capacity. We also find that 0.0008% of the overall bins—or roughly 69 seconds per day per user—show an aggregate receiving rate of more than 100 Mbps.

Due to commercial networks often being asymmetric, the CCZ network provides a larger relative improvement in uplink rates than in downlink rates. While users only exceed a nominal commodity receiving rate (10 Mbps) 0.1% of the time, they exceed a nominal commodity uplink of 0.5 Mbps 1% of the time.[6] Further, we note

[4]While we attempt to count the number of actual hosts behind each IP address in § 3.1, we have no way to do that for the $\mathcal{L}_c$ dataset that forms the basis of the analysis in this section. Therefore, "host" in this section is an IP address.

[5]We are aware of faster community networks, but 10 Mbps is the right order.

[6]Again, our aim is not to quibble about commodity rates,

Service	Recvd (%)	Buckets (%)
HTTP	91	98
Steam	3.3	1.3
BitTorrent	0.9	17
HTTPS	0.9	79
Unclassified	3.9	55

Table 7: Breakdown for top receiving applications.

Service	Sent (%)	Buckets (%)
BitTorrent	36	80
HTTPS	8.1	57
HTTP	5.2	75
Minecraft	2.7	15
Other-1111	2.0	2
Unclassified	46	90

Table 8: Breakdown for top sending applications.

that CCZ user transmission rates exceed 10 Mbps approximately 0.1% of the time and 100 Mbps roughly 0.0002% of the time. The data suggests that residential users' current usage patterns and applications are generally well-served by commodity downlinks, but not by commodity uplinks.

Finally, we note that we find 2.6 million (7.3%) times whereby a bin for the same host is included in both the top 1% sending and receiving lists. This illustrates that in a non-trivial number of cases a particular host is engaged in high-speed data transfers in both directions, e.g., as part of a peer-to-peer network.

5.2 High-Rate Applications

Having considered the aggregate and per-host capacity utilization observed in the CCZ network, we now analyze which applications are active during these periods of high capacity use. The following analysis takes into account only the top 1% of the bins as discussed in § 5.1. Table 7 shows the percentage of the incoming data volume for each application that receives at least approximately 1% of the incoming data volume. Additionally, the table shows the percentage of the top 1% bins in which the service was active. The table shows that HTTP is responsible for 91% of the data volume when the capacity is being highly utilized. Further, 98% of the top 1% bins contain HTTP traffic. The results in table 7 are similar to the results in § 4.3, even if some of the more minor or less bulky (i.e., DNS) applications have fallen off the list developed in § 4.3.

Table 8 shows our observations for top applications in terms of data transmitted by CCZ users during the top 1% utilization periods. Much as we found in § 4.3 we find BitTorrent to be the largest contributor—both in terms of volume and active bins. The unclassified category is significant during high-rate sending periods, but to illustrate the difference between the uplink use and downlink use by CCZ users.

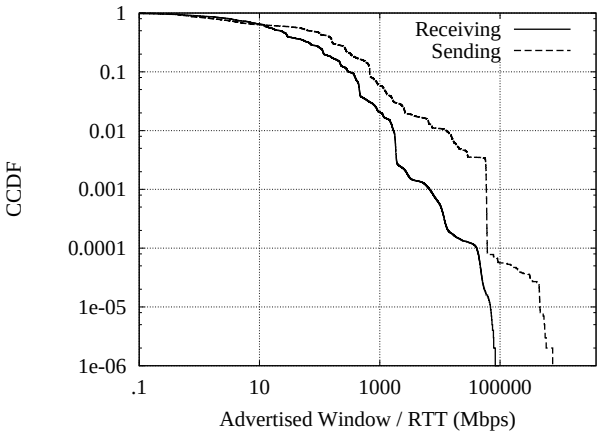

Figure 10: Distribution of advwin/RTT for all connections in our packet traces.

just as it was for traffic sending in general. This result is expected given the general prevalence of unclassified traffic found earlier and the analysis in § 4.3 that suggests this traffic is peer-to-peer file trading, which we expect would aim for high capacity utilization.

6. TRANSMISSION SPEED CAUSES

In § 5 we study observed transmission rates on a host-level basis. We find that these rates do not often use anywhere close to the full capacity. A natural question is: *why?* In this section we analyze our packet traces to try to get an initial understanding on what is limiting performance. TCP's performance is dictated by a set of congestion control algorithms [5] and has a number of dependencies, including (i) the TCP receiver's advertised window, (ii) the size of the TCP sender's retransmission buffer, (iii) the RTT of the network path, (iv) the loss rate along the network path and (v) the application's sending pattern. Of these, (ii) and (v) are not readily visible in packet traces, while the others are either exposed directly by the protocol or can be readily estimated from packet traces. In this section we use these pieces of information to study connection-level transmission speed.

6.1 Potential Speed

TCP's performance is ultimately constrained by the RTT of the network path and the receiver's advertised window. In particular, the upper-bound on throughput is $\frac{advwin}{RTT}$. This upper-bound requires the sender's retransmission buffer to be at least as big as the advertised window, the application feeds a constant stream of data to TCP and no loss occurs such that TCP's congestion window dynamically adjusts. For the purposes of assessing how fast hosts *can* send and receive data we assume these requirements hold.

We analyze our 392 packet trace files for RTTs and

advertised windows found in each TCP connection's three-way handshake.[7] We filtered connections per the process in § 2, which left us with 13.3 million connections. Figure 10 shows the distributions of potential sending and receiving rates with respect to the CCZ hosts. This shows that roughly one-third of the connections for both directions cannot attain 10 Mbps as the advertised window is too small for the given RTT. Also, 26% of the connections can receive in excess of 100 Mbps, while 46% can transmit at least 100 Mbps. Finally, we find that only 2% of the connections can receive at least 1 Gbps, while 6% of the connections can transmit at 1 Gbps. These results show that in roughly two-thirds of the cases the CCZ users can potentially use more capacity than a 10 Mbps commodity broadband connection provides via a single TCP connection. While two-thirds of connections have the potential to exceed a commodity rate, recall that we only ever see users exceeding this rate when aggregating across connections about 0.1% of the time. Therefore, there must be some other factor(s) that are responsible for limiting the sending and receiving rates we observe on the network which we attempt to identify in the following subsections.

6.2 Connections Without Loss

We now move on from the potential best-case rates that TCP can possibly attain to examining the rates TCP does attain in our dataset and the reasons for those rates. As an initial investigation we examine the largest 10 connections in terms of data transmitted from a CCZ host in each of our 392 one-hour trace files. Connections that send a large amount of bulk traffic are likely to be trying to do so as fast as possible while shorter connections may have different goals and/or application dynamics that make studying performance more difficult. Our corpus of 3,920 connections involves 79 CCZ hosts and 3,079 remote hosts. We analyze the data flowing from the CCZ to the remote host as our vantage point is then close to the sender and hence makes estimation of various sender properties straightforward (e.g., RTT, congestion window size, loss rate). A vantage point close to the receiver makes these sender properties difficult to estimate.

We first only consider connections that do not experience loss—or 633 of the 3,920 connections in our corpus. (We will consider the balance in the next subsection.) Using Bro, we determine the advertised window and the maximum flight size for each connection. The flight size is the amount of data transmitted but not ACKed at any given time and approximates TCP's congestion window. We then compare the maximum flight size with the maximum advertised window to assess whether the TCP sender is limited by the receiver's advertised window. We find that in 12% of the connections the sending TCP is in fact constrained by the advertised window. In the remainder of the connections there is some other phenomenon that is constraining the sending rate.

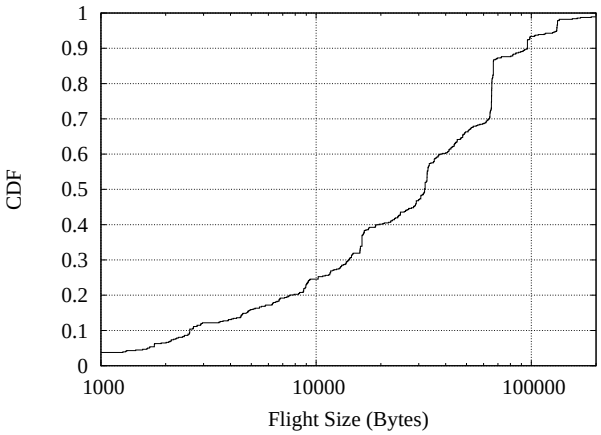

Figure 11: Distribution of flight size in connections without loss.

In figure 11 we plot the distribution of maximum flight size over all connections with no loss and that we did not find to be advertised window limited above. We find modes of varying size in this plot at 16KB, 32KB, 64KB, 96KB and 128KB. These are suggestive of some sender-side buffering issue that is limiting the flight size. The natural candidate would be the sender's TCP retransmission buffer—which limits the amount of data that can be transmitted before receiving an ACK in case the data is lost and needs resent. The limit could also come from an application—e.g., in an attempt to limit the overall sending rate. While the 1 Gbps fiber link is unlikely to become overloaded it is possible that some applications are trying to protect infrastructure within a house (e.g., a wireless network). When summing the various modes we find they account for roughly 23% of the 633 connections with no loss.

While 12% of the connections without loss are constrained by the advertised window and the data suggests that another 23% are hampered by some sender-side buffer, that leaves nearly two-thirds of the connections unexplained. Without additional insight from the end hosts themselves it is hard to reason about the performance of these connections. While we cannot pinpoint the cause, we can say that the network path and the receiver's buffer do not appear to be the constraining issue.

[7]Note, this analysis assumes the advertised window in the handshake represents the largest amount of buffering available for the connection. Systems that auto-tune their buffers [18] will therefore likely utilize a larger advertised window than given in the SYN handshake. However, the results in this section are at least a lower bound on the potential performance.

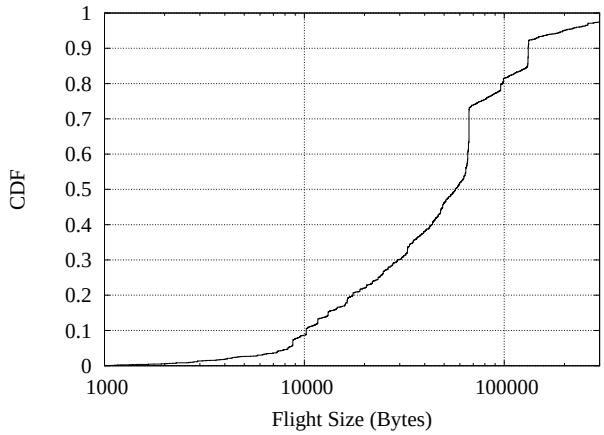

Figure 12: Distribution of flight size in connections with loss.

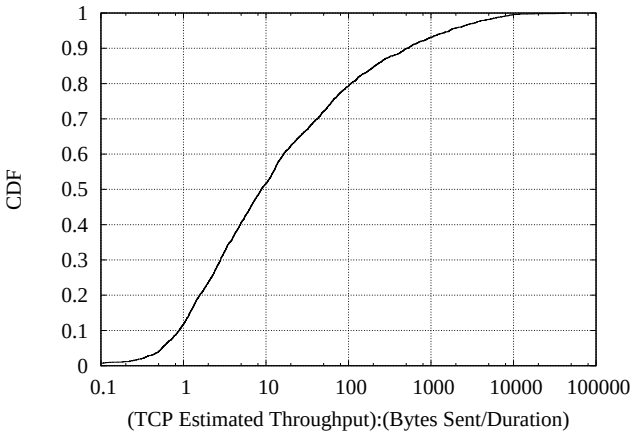

Figure 13: Ratio of theoretical throughput to actual throughput for each connection.

6.3 Connections With Loss

Finally, we turn to the 3,287 connections in our large connection corpus that experience loss. In these cases theory suggests the loss rate and RTT contribute to dictate performance [15, 16]. However, the advertised window, retransmission buffers and application behavior can still limit performance. As such, we repeat the analysis we did for the no loss case above. First, we find that the maximum flight size reaches the maximum advertised window size in 20% of the connections with loss, which is a higher proportion than in the no loss connections.

As above we next plot the maximum flight size distribution in figure 12. As with the no loss case we find several modes that suggest a sender-side buffer limit that ultimately constrains performance. In this case we find this happens in roughly 22% of the connections that experience loss. Therefore, overall we find 20% of the connections with loss constrained by the advertised window and another 22% likely constrained by a sender side buffer (such as TCP's retransmission buffer). That leaves 58% of the connections to be constrained by some phenomenon outside our purview.

As a final test we assess how much slower the connections with loss progress than theory suggests should be possible given the network path characteristics. As there are 58% of connections with loss and two-thirds of connections without loss being constrained by an unknown phenomenon, we attempt to see if TCP itself is a limiting factor given network characteristics. We used the TCP model that was developed in [15, 16] and applied in [12] to calculate the rate each connection could potentially use given the connection's RTT and loss rate.[8] Figure 13 shows the distribution of the rate suggested by the TCP model versus the rate we observed for each connection that experienced loss in our dataset. We find that in less than 10% of the cases the observed throughput actually outperforms the model. In the remaining roughly 90% of the cases the observed throughput is less—often by orders of magnitude—than the performance predicted by the model. This plot reenforces our finding that the network path and TCP's congestion control algorithms would allow connections to transmit more rapidly. However, host limits and application behavior are holding back performance.

[8]We used the median RTT for the connection and $b = 1$ for the model given in [12].

7. SUMMARY

This paper aims to present an initial broad characterization of traffic from an operational 1 Gbps FTTH network. We make several contributions, as follows.

- We are the first (to our knowledge) to characterize myriad aspects—from structural aspects to traffic patterns to capacity utilization issues—of an operational FTTH network over a 14 month time period.
- Our study provides another data point on the use of residential networks (e.g., a reappraisal of some aspects of [13]).
- We find that even when given virtually unlimited bandwidth the majority of the time users do not retrieve information from the Internet in excess of commercially available data rates. However, in terms of transmitting data we find the FTTH users in our study use more capacity than available via commodity broadband.
- Similar to the last point we find 1 Gbps links to a single household are not well utilized, which presents an opportunity for new applications and services to capitalize on such resources.

- We find that TCP connections do not attain anywhere near 1 Gbps in performance even though plenty of unused capacity exists and TCP theory suggests the network paths are amenable to (much) higher rates than realized.
- We find that the likely reasons for TCP's low performance are end host buffering issues in many cases. In some cases this manifests in TCP's advertised window, but in others we find evidence of a sender-side buffer limitation.

Finally, we note that our goal in this paper was for broad characterization. Every analysis we present begs many additional questions. Our future work will involve digging more deeply into many of these questions.

Acknowledgments

Lev Gonick, Mike Chalkwater, Lou Changeri, Tu Ouyang and Michael Rabinovich facilitated our monitoring of the CCZ network. Discussions with Michael Rabinovich aided this work. The work was funded in part by NSF grant CNS-0831535. Our thanks to all!

8. REFERENCES

[1] Bringing ultra high-speed broadband to Stanford homes. `http://googleblog.blogspot.com/2010/10/bringing-ultra-high-speed-broadband-to.html`.

[2] Case Connection Zone. `http://caseconnectionzone.org/`.

[3] Chattanooga, Tenn Announces only 1 Gigabit Broadband Service in U.S. for Both Residential and Business Customer. `http://chattanoogagig.com/pdf/Chattanooga_GPON_EPB.pdf`, 2010.

[4] Alexa: The Web Information Company. `http://www.alexa.com`.

[5] M. Allman, V. Paxson, and E. Blanton. TCP Congestion Control, Sept. 2009. RFC 5681.

[6] M. Allman, V. Paxson, and J. Terrell. A Brief History of Scanning. In *ACM SIGCOMM/USENIX Internet Measurement Conference*, Oct. 2007.

[7] Bitcoin: P2P Digital Currency. `http://bitcoin.org`.

[8] Cohen, B. The BitTorrent Protocol Specification. `http://bittorrent.org/beps/bep_0003.html`, 2008.

[9] R. Fielding, J. Gettys, J. C. Mogul, H. Frystyk, L. Masinter, P. Leach, , and T. Berners-Lee. Hypertext Transfer Protocol – HTTP/1.1, June 1999. RFC 2616.

[10] L. Gonick. Personal Communication, Apr. 2021.

[11] Ultra high-speed broadband is coming to Kansas City, Kansas. `http://googleblog.blogspot.com/2011/03/ultra-high-speed-broadband-is-coming-to.html`.

[12] M. Handley, S. Floyd, J. Padhye, and J. Widmer. TCP Friendly Rate Control (TFRC): Protocol Specification, Jan. 2003. RFC 3448.

[13] G. Maier, A. Feldmann, V. Paxson, and M. Allman. On Dominant Characteristics of Residential Broadband Internet Traffic. In *ACM Internet Measurement Conference*, Nov. 2009.

[14] Maier, G. and Schneider, F. and Feldmann, A. NAT usage in residential broadband networks. In *Passive and Active Measurement*, pages 32–41. Springer, 2011.

[15] M. Mathis, J. Semke, J. Mahdavi, and T. Ott. The Macroscopic Behavior of the TCP Congestion Avoidance Algorithm. *Computer Communication Review*, 27(3), July 1997.

[16] J. Padhye, V. Firoiu, D. Towsley, and J. Kurose. Modeling TCP Throughput: A Simple Model and its Empirical Validation. In *ACM SIGCOMM*, Sept. 1998.

[17] V. Paxson. Bro: A System for Detecting Network Intruders in Real-Time. *Computer Networks*, Dec. 1999.

[18] Semke, J. and Mahdavi, J. and Mathis, M. Automatic TCP buffer tuning. In *ACM SIGCOMM Computer Communication Review*, volume 28, pages 315–323. ACM, 1998.

[19] The national broadband plan. `http://www.broadband.gov/plan/`.

[20] The Spamhaus Project - PBL. `http://www.spamhaus.org/pbl/`.

[21] Zalewski, M. p0f: Passive OS Fingerprinting tool. `http://lcamtuf.coredump.cx/p0f.shtml`, 2006.