Detecting and Profiling TCP Connections Experiencing Abnormal Performance

Aymen Hafsaoui[1], Guillaume Urvoy-Keller[2], Matti Siekkinen[3], and Denis Collange[4]

[1] Eurecom, France
[2] Laboratoire I3S CNRS/UNS UMR 6070, France
[3] Aalto University, Finland
[4] Orange Labs, France

Abstract. We study functionally correct TCP connections – normal set-up, data transfer and tear-down – that experience lower than normal performance in terms of delay and throughput. Several factors, including packet loss or application behavior, may lead to such abnormal performance. We present a methodology to detect TCP connections with such abnormal performance from packet traces recorded at a single vantage point. Our technique decomposes a TCP transfer into periods where: (i) TCP is recovering from losses, (ii) the client or the server are thinking or preparing data, respectively, or (iii) the data is sent but at an abnormally low rate. We apply this methodology to several traces containing traffic from FTTH, ADSL, and Cellular access networks. We discover that regardless of the access technology type, packet loss dramatically degrades performance as TCP is rarely able to rely on Fast Retransmit to recover from losses. However, we also find out that the TCP timeout mechanism has been optimized in Cellular networks as compared to ADSL/FTTH technologies. Concerning loss-free periods, our technique exposes various abnormal performance, some being benign, with no impact on user, e.g., p2p or instant messaging applications, and some that are more critical, e.g., HTTPS sessions.

1 Introduction

Several access technologies are now available to the end user for accessing the Internet, e.g., ADSL, FTTH and Cellular. Those different access technologies entail different devices, e.g., smartphones equipped with dedicated OS like android. In addition, a different access technology also implies a different usage, e.g., it is unlikely that p2p applications be used as heavily on Cellular than on wired access. Even if we consider ADSL and FTTH, which are two wired technologies, some differences have been observed in terms of traffic profile [1].

Despite this variety of combinations of usage and technology, some constant factors remain in all scenarios like the continuous usage of email or the use of TCP to carry the majority of user traffic. This predominance of TCP constitutes the starting point of our study and our focus in the present work is on the performance of TCP transfers.

In this work, we aim at detecting functionally correct TCP connections – normal set-up/tear-down and actual data transfer – that experienced bad performance. The rationale behind this study is that bad performance at the TCP layer should be the symptom of

A. Pescapè, L. Salgarelli, and X. Dimitropoulos (Eds.): TMA 2012, LNCS 7189, pp. 107–120, 2012.

bad performance at the application or user level. Note that this a different objective from the detection of traffic anomalies, where the focus is to detect threats against the network, e.g. DDoS [2] [3,4,5,6].

To tackle the problem, we adopt a divide and conquer approach, where we analyze separately connections that experience losses and connections that are unaffected by losses. Our analysis of connections unaffected by losses (the majority of connections) uses as a starting point a breakdown approach initially proposed in [7]. It enables to delineate, for each transfer, time periods due to the server or client thinking, or the time spent sending data. Once each connection is transformed into in a point in multidimensional space, we isolate anomalous TCP connections experiencing bad performance as those connections having high value(s) in one or several dimensions.

We apply our methodology to passive traces of traffic collected on ADSL, Cellular and FTTH access core networks managed by the same Access Service Provider. Our main contributions are as follows:

- Concerning losses, we extend to the case of multi-technology Internet access, what other studies have observed, namely that losses lead to a substantial, from 30 to 70% (median) increase of transfers times for all connection sizes (mice or elephants).
- We observed that the strategies observed on the Cellular technology to recover from losses seem more efficient than on ADSL and FTTH, as the time out durations are close to the Fast Retransmit durations.
- Concerning transfers unaffected by losses, we propose different definitions of what an abnormal performance means and exemplify the different approaches on our traces. A salient point is that our approach relies on an adequate normalization of those quantities in order to pinpoint abnormal performance independently of the actual size of the connection.
- While analyzing the connections flagged as abnormal, we relate the performance at the transport layer to the performance at the application layer. We show that in some cases, e.g., instant messaging applications, the low performance at the transport are unrelated to problems at the application layer. On the opposite, in some key client/server applications like HTTPS transfers, low performance at the transport layer might be perceived negatively by the end user.

2 Datasets

We collected packet level traces of end users traffic from a major French ISP involving different access technologies: ADSL, FTTH and Cellular. The latter corresponds to 2G and 3G/3G+ accesses as clients with 3G/3G+ subscriptions can be downgraded to 2G depending on the base station capability. ADSL and FTTH traces correspond to all the traffic of an ADSL and FTTH Point-of-Presence (PoP) respectively, while the Cellular trace is collected at a GGSN level, which is the interface between the mobile network and the Internet. Note that ADSL and FTTH clients might be behind 802.11 home networks, but we have no means of detecting it.

Table 1 summarizes the main characteristics of each trace. Each trace features enough connections to obtain meaningful statistical results.

Table 1. Traces Description

	Cellular	FTTH	ADSL
Date	2008-11-22	2008-09-30	2008-02-04
Starting Capture	13:08:27	18:00:01	14:45:02:03
Duration	01:39:01	00:37:46	00:59:59
NB Connections	1,772,683	574,295	594,169
Functionally correct cnxs	1,236,253	353,715	381,297
Volume UP(GB)	11.2	51.3	4.4
Volume DOWN(GB)	50.6	74.9	16.4

Our focus is on applications on top of TCP, which carries the vast majority of bytes in our traces. We restrict our attention to the connections that correspond to presumably valid and complete transfers from the TCP layer perspective, that we term functionally correct connections. Functionally correct connections must fulfill the following conditions: (i) A complete three-way handshake; (ii) At least one TCP data segment in each direction; (iii) The connection must finish either with a FIN or RST flag. Functionally correct connections carry between 20 and 125 GB of traffic in our traces (see Table 1). The remaining connections, which amount for almost one third of connections in our traces, consist for the vast majority of transfers with non complete three-way handshakes (presumably scans) and also a minority of connections, a few percents, for which we missed the beginning or the end of the transfer.

To have in idea of the applications present in our data sets, we performed a rough classification of traffic by identifying destination ports. It reveals that more than 84% of Cellular access connections targeted ports 80 and 443. This value falls to respectively 45% and 62% of bytes for our FTTH and ADSL traces, where we observed a prevalence of dynamic destination ports, which are likely to correspond to p2p applications.

3 On the Impact of Losses on TCP Performance

TCP implements reliability by detecting and retransmitting lost segments. The common belief is that the loss recovery mechanism of TCP is particularly penalizing for short transfers. However, several work have shown that even long transfers might be penalized by loss recovery, e.g. [7]. We confirm those results for the cases of all the access technology we consider. We further demonstrate that on the Cellular access technology, some counter measures have been put in place to limit the duration of TCP recovery phases.

3.1 Losses and Retransmission Periods

To assess the impact of TCP loss retransmission events in our traces, we use an algorithm to detect retransmitted data packets, which occur between the capture point and the remote server or between the capture point and the local (ADSL, FTTH, Cellular) client. This algorithm is similar to the one developed in [8]: we define the retransmission time as the time elapsed between the moment where we observe a decrease of the TCP

sequence number and the first time where it reaches a value larger than the largest sequence number observed so far. If ever the loss happens after the observation point, we observe the initial packet and its retransmission. In this case, the retransmission time is simply the duration between those two epochs. When the packet is lost before the probe, we infer the epoch at which it should have been observed, based on the sequence numbers of packets. Note that computations of all those durations are performed at the sender side, as time series are shifted according to our RTT estimate. We (heuristically) separate actual retransmissions from network out of sequence and spurious [8] retransmission events by eliminating durations smaller than the RTT of the connection. Once losses are identified, we compute for each TCP connection, its total retransmission time.

We first report, in Table 2, on two metrics: the average loss rate and the average fraction of connections affected by loss events.

Table 2. Overall loss rates

	Cellular	FTTH	ADSL
Loss rate	4%	2%	1.2%
Fraction of connections	29%	9%	5%

We observe from Table 2 that while loss rates are quite low (our traces are too short to draw general conclusions on the loss rates in each environment), the fraction of connections affected by losses are quite high, esp. for the Cellular technology. A possible reason is that losses are due to random losses on the wireless medium, which may result in small loss episodes that affect connections irrespectively of their duration or rate.

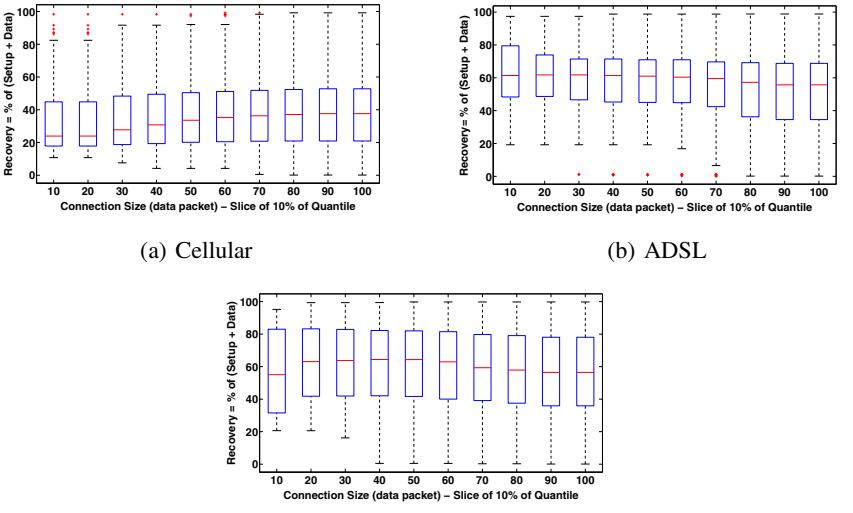
To assess the impact of the loss recovery mechanisms of TCP, we compute the fraction of transfer time that the recovery period represents. The transfer time itself is defined as the sum of set-up (three-way handshake) and data transfer time (including loss recovery periods) for each connection. We exclude the tear-down time, where only control segments are exchanged (ACK, FIN, RST,) as this duration as no impact on application performance[1] and has been observed to be extremely long in a lot of cases - see [7]. As we further want to assess the impact of the recovery process for both short and long connections, we present results for each decile of the connection size, i.e., we report results for the 10% of smaller connections, then the next 10%, etc. Results for each access technology are presented in Figure 1. We observe that for all technologies, losses lead to a significant increase of the connection duration, between 30 and 70 % when considering the medians (bars in the center of the boxplots), irrespectively of the actual size of the transfer. We also note that the lower impact is observed for the Cellular trace, while the impact on the ADSL and FTTH traces are similar for all deciles.

We further investigate this discrepancy Cellular and ADSL/FTTH in the next section.

3.2 Delving into TCP Retransmissions

To uncover the better performance of Cellular connections observed in the last paragraph, we next distinguish between losses recovered by a retransmission time-out (RTO)

[1] A server might in fact be affected by long tear down times as the resources associated to the socket are unnecessarily affected to the connection.



(a) Cellular

(b) ADSL

(c) FTTH

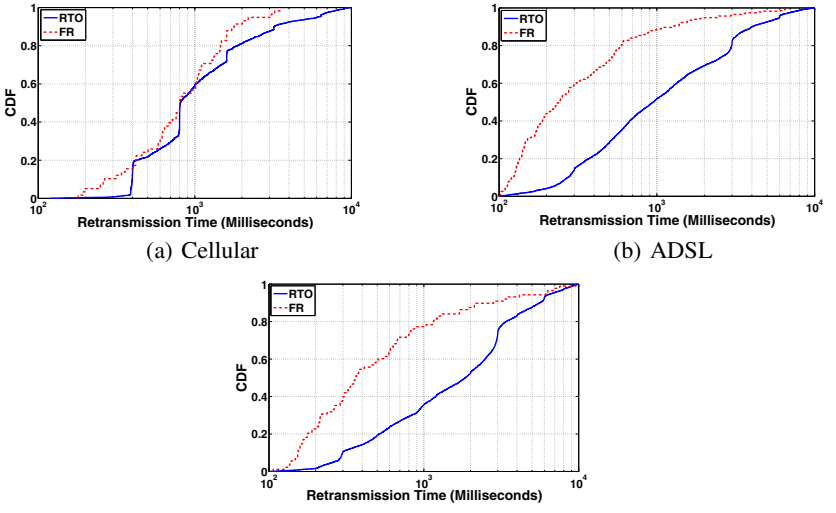
Fig. 1. Fraction of connection duration due to TCP retransmissions

and losses recovered by a fast retransmit/fast recovery (RF/R). We suppose that RTO (resp. FR/R) correspond to recovery periods with strictly less than (resp. greater or equal to) 3 duplicate acknowledgments. This definition leads to a striking result: for our traces, more than 96% of loss events are detected using RTO. This result is in line with previous studies [7].

Two factors contribute to this result. First, most of transfers are short and it is well-known that short transfers, which do not have enough in flight packets to trigger a FR/R revert to the legacy RTO mechanism. Second, long connections must often rely on RTO as the transfer, while large, consists of a series of trains (questions and answers of the application layer protocol) whose size is not large enough, in almost 50% of the cases in our traces, to trigger a FR/R.

Figure 2 plots the distribution of data retransmission time for FR/R and RTO based retransmissions. As expected, FR/R retransmission times are shorter than RTO for all access technology. However, the key result here is that under the Cellular technology, a significant attention has apparently been paid to limit the RTO duration, which results in RTO performance close to the FR/R performance. This might be due to specific mechanisms at the server side [2] or at the Access Point Name, which is the proxy that Cellular clients use to the access the Internet. Optimizing the RTO mechanism in the Cellular environment is a strategy that pays offs as the vast majority of TCP transfers rely on RTO. For instance, if we arbitrarily set a threshold in terms of abnormal performance to 1s of recovery period, we observe that with the current optimization, the fraction of abnormal recovery times is about 20% smaller in Cellular than in ADSL and FTTH scenarios.

[2] While the protocol stack of the end device might also play a role, most of the data packets flow from the server to the cellular client.



(a) Cellular (b) ADSL

(c) FTTH

Fig. 2. Retransmission times due to FR/R or RTO

4 TCP Abnormal Performance Due to Causes Other Than Losses

4.1 Methodology

We next turn our attention to connections that are not affected by retransmissions. We are left with the set-up, data transfer and tear-down times. We did not observe long set-up times, due for instance to the loss of SYN/SYN-ACK packets. We thus do not consider set-up times in our analysis. We exclude the tear-down phase from our analysis for the same reasons as in the previous section: it does not affect client perceived performance and can bias our analysis as tear-down durations can be extremely large as compared to the actual data transfer. To highlight the above assertion, we present in Figure 3 the legacy throughput (total amount of bytes divided by total duration including tear down) and what we call the Application-Layer (AL) throughput where tear-down is excluded. We already see a major difference between those two metrics. If we are to reveal the actual performance perceived by the end user, we further have to remove the durations from the epochs where the user[3] has received all data she requested from the server (which we detect as no unacknowledged data from the server to the client in flight) and the epochs where she issues her next query. We call this metric the Effective Exchange (EE) throughput. Those three metrics (throughput, AL throughput and EE throughput) are presented in Figure 3 and we can see that they present highly different views of the achieved performance.

[3] The user might be a program, e.g, a mail client sending multiple mails.

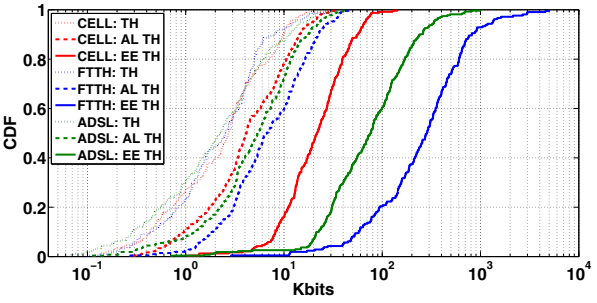


Fig. 3. Various Ways of Computing Connection Throughput

We can generalize the above approach by decomposing each transfer into 6 periods whose durations sum to the total transfer durations:

- The client[4] and server **warm-up** times, where either the client is thinking or the server is crafting data;
- The **theoretical times** computed on the client and server side, which represent the time an ideal TCP connection acting on the same path (same RTT but infinite bandwidth) would take to transfer all data from one side to the other. As a simple example, consider a TCP connection that must convey 7 data packets from a sender A to a receiver B. Assuming an infinite bandwidth, it takes $3.5 \times RTT$ to transfer the packets from A to B if we assume an initial congestion window of 1 and the use of the delayed acknowledgment mechanism.
- The difference between the transfer in one direction (say client to server) and the sum of thinking time and theoretical time is due to some phenomenon in the protocol stack, e.g. the application or the uplink/downlink capacity that slowed down the transfer. We call **pacing** this remaining duration.

Figure 4 depicts an example of our decomposition approach for the case of a browsing session.

The above methodology was presented in [7] with a different objective than detecting anomalies. We aim here at using it to isolate abnormal TCP connections. A first step is that we exclude the client side warm-ups as large thinking time at the client side should not mean anomaly. Next, we apply a normalization process on each dimension as we want to select anomalous connections irrespectively of their actual size. To do so, we apply the following normalization procedure :

- Normalized Warm-up: for each connection, we obtain the normalized warm-up as the time total warm-up divided by the number of warm-up events.
- Normalized theoretical times: for each connection, we obtain the normalized theoretical time for each direction as the total theoretical time divided by the number of packet for the corresponding direction. For long connections, the normalized theoretical times should be close to the RTT of the connection. For small connections, the speed at which the congestion window opens will constrain the normalized theoretical time.

[4] The client is for us, the initiator of the transfer.

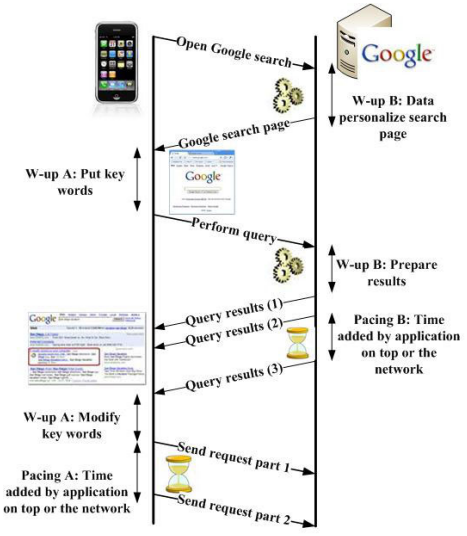


Fig. 4. Decomposition of a typical TCP transfer

– Normalized pacing: similarly to the theoretical time, we divide the total pacing time per direction by the number of packets for the corresponding direction.

We next present three different definitions of an anomaly. Note that since we excluded the Warm-up at the client side, each connection corresponds to a point in a 5 dimensional space.

– **Definition 1:** a connection is declared anomalous if its value, in any of the 5 dimension is higher than the p-th quantile for this dimension.
– **Definition 2:** a connection is declared anomalous if the sum of its values in each dimension is higher than the p-th quantile (computed over the sum).
– **Definition 3:** a connection is declared anomalous if its normalized response time, which is defined as its transfer time (set-up plus data transfer time. Again, we exclude the tear-down phase) divided by the total number of packets transferred (sum over both directions) is higher than the p-th quantile of the corresponding distribution.

Each of the above definitions has its own merits. Definition 3 is the simplest one and does not require our break-down methodology to be applied. Definitions 1 and 2 are built on our decomposition approach. Definition 1 aims at detecting outlier in at least one dimension while definition 2 aims at detecting global outliers, which might not have extremely high values in any dimension but a globally high sum.

In the present work, our objective is to understand which anomalies can be detected using our approach. We leave aside the important problem of determining which definition is the best and also, which value of p is the best. Instead, we focus on analyzing a set of connections flagged by the three definitions for an arbitrary value of p.

4.2 Selection of Anomalous Connections

We proceeded as follows to select a set of abnormal connections. Our starting point is definition 1, for which we use $p = 85$, i.e., we select a connection as an outlier if its values in any dimension is larger than the 85-th quantile in this dimension. Using this approach and this threshold value will lead to select between 15 and 75% of connections. It will be 15% if a connection that features a high value in one dimension also features a high value in all the other dimensions. Conversely, if we have disjoint sets of connections for each dimension, we obtain $5 \times 15 = 75\%$ of connections. In our case, we obtain an intermediate value of 33% of connections. Those 33% of connections correspond to 5% of the overall bytes exchanged . We next adjust the threshold in definitions 2 and 3 so as to have the same number of connections selected as in definition 1. This simply means that we set $p = 77\%$ for definitions 2 and 3. As we do not want to decide which definition is the best at this stage, we consider the intersection of the sets of connections selected using those 3 definitions. The matrix below provides the percentages of intersections using each definition:

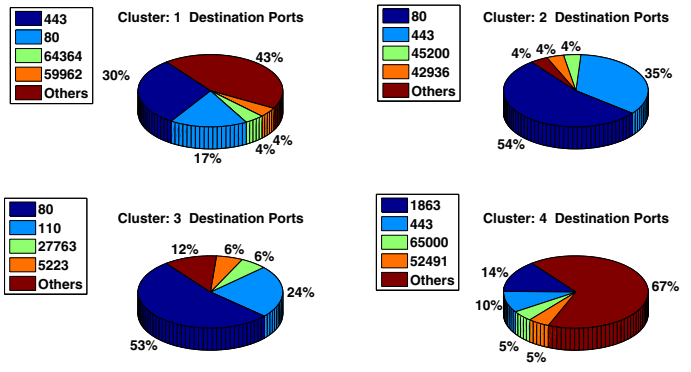
$$\begin{pmatrix} & \text{Def. 1} & \text{Def. 2} & \text{Def. 3} \\ \text{Def. 1} & 100\% & 53\% & 55\% \\ \text{Def. 2} & 53\% & 100\% & 62.1\% \\ \text{Def. 3} & 55\% & 62.1\% & 100\% \end{pmatrix}$$

4.3 Clustering Results

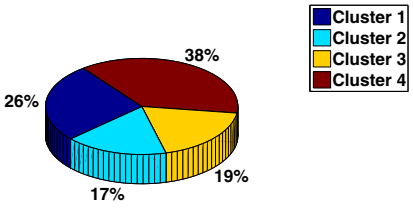
The intersection of the three sets correspond to 11% of connections and 3% of bytes. We use a clustering approach to discover similarities between anomalies. We used the popular Kmeans algorithm. Using Kmeans requires both to choose (before running the algorithm) the number of clusters and also the initial choice of the centroids of the cluster. For the first problem, we rely on a visual inspection of the data using a dimensionality reduction technique (t-SNE [9]) that projects multi-dimensional data on a 2D plane, while preserving the relative distance between points. Concerning the choice of the initial centroids, we use the classical Kmeans + approach whereby 100 initial choices of clusters are considered and the best result (in terms of intra and inter-cluster distances) is picked at the end.

For our set of bad performing connections selected in the previous section, we obtained with tSNE that 4 clusters was a reasonable choice. We present the 4 clusters obtained with Kmeans in Figure 6. We use a boxplot representation for each of the five dimensions. Note that the values reported here are non-normalized values: we normalize prior to clustering but we report initial values in the boxplot representations. We also enrich the graph of each cluster with (i) the fraction of connections for each access technology and (ii) the median size of transfers (both, on top of the graphs). We further present in Figures 5(a) and 5(b) the distributions of ports per cluster and also the volumes (in bytes) per cluster, respectively.

We can observe that the size of clusters range between 17 and 38%, which means that they are relatively homogeneous in terms of size. In contrast, the clusters are quite different in terms of the applications they correspond to. Cluster 1 corresponds to HTTP/HTTPS traffic and also a significant fraction of others, where others means



(a) Target Ports



(b) Data Volume Per Cluster

Fig. 5. Intersection

that both the source and destination ports are dynamic (a good hint of a p2p application). Cluster 2 corresponds mostly to HTTP/HTTPS traffic. Cluster 3 features mostly to HTTP and POP, while cluster 4 is dominated by dynamic ports.

When correlating the dominating ports and the fraction of connections per access technology in each cluster (on top of each plot in Figure 6), one can clearly observe that cellular access is mostly present in clusters 2 and 3 where there is little dynamic ports. In contrast, most of the ADSL and FTTH connections are in clusters 1 and 4 that contain a lot of connections corresponding to dynamic ports. Those observations are in line with intuition as dynamic ports are likely to be due to p2p applications that are more popular on ADSL/FTTH than Cellular access technology. Note that a majority of our cellular users use smartphones rather than laptop equipped with 3G dongles as we observed by mining the HTTP header (user-agent information) of their Web requests.

Let us now focus on the interpretation of the 4 clusters in Figure 6. One can adopt a quantitative or a qualitative standpoint. From a qualitative standpoint, we can observe that clusters 1 and 2 report on problems located at the server side, with extremely high warm-up or pacing times. In contrast, for clusters 3 and 4, one observes large values at both the client and the server side. If one adopts a quantitative viewpoint, the situation

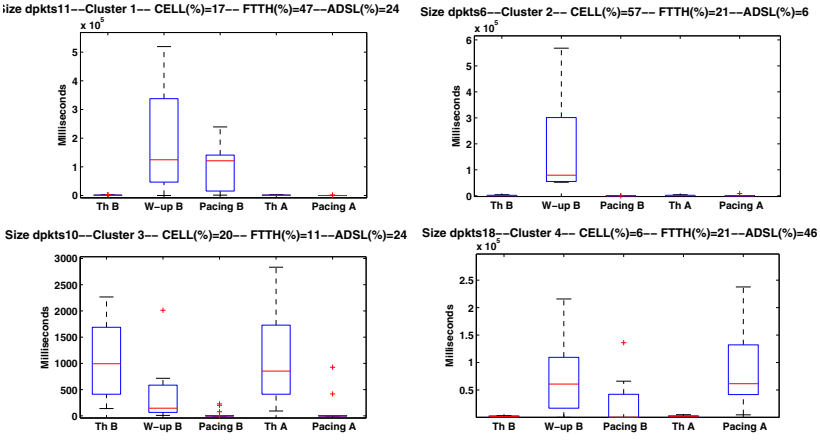


Fig. 6. Intersection: K-means Results

is quite different. Indeed, clusters 1, 2 and 4 are characterized by values (for the dimensions that characterize the anomaly) that are one to two order of magnitudes larger – in the order of tens or hundreds of milliseconds for those clusters as compared to about one second time scale for cluster 3.

Our starting point, in this work, was the hypothesis that bad performance at the TCP layer should be the symptom of a problem at the application layer or from the user point of view. A closer analysis of the clustering results reveals that while our approach indeed reveals TCP connections that perform badly, not all these connections result in bad performance from the user perspective. Let us consider the case of cluster 4. In this cluster, one observes 14% of connections using port 1863, which is the Microsoft messenger port. It is clear that here, the bad performance at the transport layer is due to the fact that the client and the server are two humans that (normally) think before typing and type relatively slowly. Bad performance at the transport layer is thus unrelated to bad performance at the application layers. Also, still for cluster 4, one observes a large fraction of dynamic ports, which is likely to be due to p2p applications. It is known that p2p applications tend to throttle the bandwidth offered to other peers. This is why we observe limitations at both the client and server side, which are the two ends of the application. It is difficult to categorize the bad performance of p2p applications as an anomaly from the user perspective since users are in general patient when it comes to download content since they treat this traffic as a background traffic (that should not interfere with their current interactive traffic, typically browsing ; hence the rate limiters in p2p applications).

The situation is different for clusters 1 and 2, for which anomalies are clearly located at the server side and a significant share of the traffic is due to HTTP and HTTPS. We observed typically large values for HTTPS connections that presumably (given the IP address) correspond to electronic payment transactions. We leave for future work a more in depth analysis of the servers flagged as anomalous (e.g., which fraction of their traffic is indeed anomalous, do they serve only a specific type of clients, e.g. cellular

clients) and we present in the next section, some typical examples of anomalies that we observed, related to warm-up or pacing problems, in order to make our case more concrete.

4.4 Examples of Anomalies

Large Warm-up B. We report in Figure 7 an example of large warm-up time at the server side,observed by a client behind an ADSL access. We notice that the acknowledgment received from the server indicates that the query (GET request) has been correctly received by the server, but it takes about 4.5 seconds before the client receives the requested object (a png image in this case).

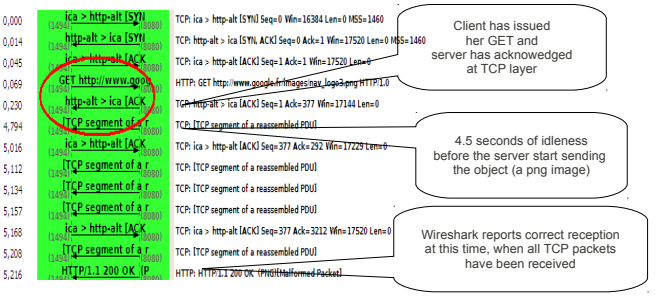


Fig. 7. Abnormal Long Response Time at The Server Side (Warm-up B value)

Large Pacing B. We report in Figure 8 an example of large pacing time for a Gmail server, observed by a client behind an FTTH access. We notice that the acknowledgment received from the server indicates that the **GET request** has been correctly received by the server. The server sends data until the last TCP segment which is delayed by 27.6 seconds, before the client receives the object.

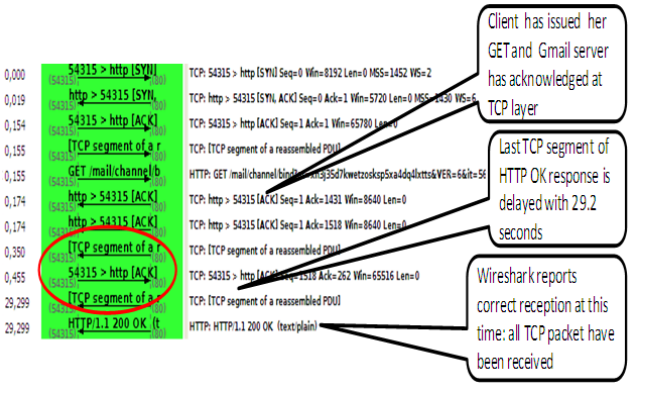


Fig. 8. Webmail: Large Pacing B in Gmail Server

Large Pacing A. We report in Figure 9 an example of large Pacing A in the case of an HTTPS connection for a Cellular client. After a successful three way handshake, the user authenticates and exchanges data with the server. If we focus on the client last data packet, we observe that it is delayed by more than 200 seconds compared to the previous data packet. This introduces a large idle time in the transfer. We see through this example that the application can play an important role in data scheduling at the network layer, which can have a detrimental impact in terms of perceived performance.

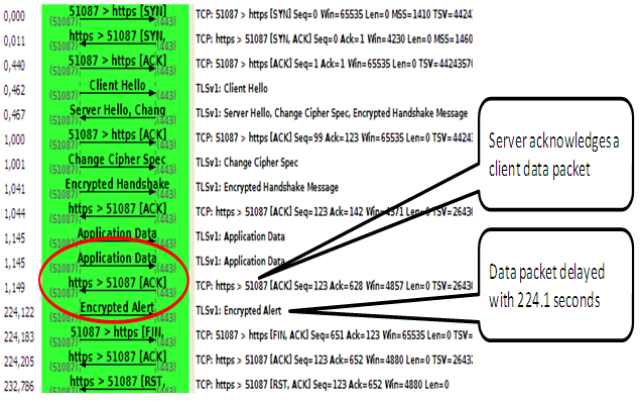


Fig. 9. Large Pacing A

5 Conclusion

In this paper, we have shed light on the problem of detecting and understanding TCP connections that experience low performance. We jointly analyzed network traces collected on a variety of networks that reflect the way people are accessing the Internet nowadays.

Losses in the network, while rare, significantly deteriorate performance. This result is not new but the added value of our work is to show that any transfer that experiences losses suffers, irrespectively of the access technology or its exact size. Furthermore, our approach of jointly profiling different access technologies enabled us to highlight that more attention is paid to limiting the impact of losses in cellular than on ADSL/FTTH networks apparently. We relate this discrepancy to the shorter durations of time out durations on the cellular network. While those results need to be confirmed over longer traces, the extent to the difference (a factor of 2 in the fraction of time required to recover losses on Cellular trace as compared to our ADSL/FTTH traces) suggests it can not be a mere coincidence. As future work, we intend to investigate whether optimizations of the recovery mechanism of TCP could be proposed based on the observations we made.

For the majority of connections that do not experience losses, we propose several approaches to detect outliers. Each of them accounts for the size of the connections so that not only long connections be flagged as experiencing abnormal performance. We

exemplify the different approaches by analyzing their intersection set, i.e., the set of connections flagged as abnormal, whatever the definition is. We use a clustering approach to form groups of similar abnormal connections. We enrich those groups with additional information like the distribution of ports per group, to understand whether low performance at the transport layer is a symptom of bad performance at the application layer. It turns out that the relation between the transport and the application layer is complex. There are cases, e.g., instant messaging, where the two are fully unrelated. This is also partly the case with p2p transfers as users are resilient to low performance at the transport layer as long as they eventually obtain the content they want. On the other hand, we observed low performance for a significant number of cases, e.g. HTTP and HTTPS transfers where the user might consider that the application is misbehaving. As future work, we intend to profile more precisely the latter set of connections (where bad performance at the transport and application layer are apparently related) to better understand the extent of those anomalies.

References

1. Vu-Brugier, G.: Analysis of the impact of early fiber access deployment on residential internet traffic. In: 21st International Teletraffic Congress, ITC 21 2009, pp. 1–8 (September 2009)
2. Barford, P., Kline, J., Plonka, D., Ron, A.: A signal analysis of network traffic anomalies. In: Proceedings of the 2nd ACM SIGCOMM Workshop on Internet Measurment, IMW 2002, pp. 71–82. ACM, New York (2002)
3. Lakhina, A., Crovella, M., Diot, C.: Diagnosing network-wide traffic anomalies. In: ACM SIGCOMM, pp. 219–230 (2004)
4. Soule, A., Salamatian, K., Taft, N.: Combining filtering and statistical methods for anomaly detection. In: Proceedings of the 5th ACM SIGCOMM Conference on Internet Measurement, IMC 2005, p. 31. USENIX Association, Berkeley (2005)
5. Tellenbach, B., Burkhart, M., Sornette, D., Maillart, T.: Beyond Shannon: Characterizing Internet Traffic with Generalized Entropy Metrics. In: Moon, S.B., Teixeira, R., Uhlig, S. (eds.) PAM 2009. LNCS, vol. 5448, pp. 239–248. Springer, Heidelberg (2009)
6. Silveira, F., Diot, C., Taft, N., Govindan, R.: Astute: detecting a different class of traffic anomalies. SIGCOMM Comput. Commun. Rev. 40, 267–278 (2010)
7. Hafsaoui, A., Collange, D., Urvoy-Keller, G.: Revisiting the Performance of Short TCP Transfers. In: Fratta, L., Schulzrinne, H., Takahashi, Y., Spaniol, O. (eds.) NETWORKING 2009. LNCS, vol. 5550, pp. 260–273. Springer, Heidelberg (2009)
8. Jaiswal, S., Iannaccone, G., Diot, C., Kurose, J., Towsley, D.: Measurement and classification of out-of-sequence packets in a tier-1 ip backbone. IEEE/ACM Trans. 15(1), 54–66 (2007)
9. van der Maaten, L.J.P.: t-distributed stochastic neighbor embedding, `http://homepage.tudelft.nl/19j49/t-SNE.html`