Passive Analysis of TCP Anomalies [1]

Marco Mellia [a,*] Michela Meo [a] Luca Muscariello [b] Dario Rossi [c]

[a] *Dipartimento di Elettronica, Politecnico di Torino, Corso Duca degli Abruzzi 24, 10129 Torino, Italy*

[b] *France Télécom R&D - Orange Labs, 38-40 rue du général Leclerc 92794 Issy-les-Moulineaux, France*

[c] *ENST ParisTech, 46 rue Barrault, 75634 Paris, France*

Abstract

In this paper we focus on passive measurements of TCP traffic. We propose a heuristic technique to classify TCP anomalies, i.e, segments that have a sequence number different from the expected one, such as out-of-sequence and duplicate segments. Since TCP is a closed-loop protocol that infers network conditions from packet losses and reacts accordingly, the possibility of carefully distinguishing the causes of anomalies in TCP traffic is very appealing and may be instrumental to understand TCP behavior in real environments. We apply the proposed heuristic to traffic traces collected at both network edges and backbone links. By comparing results obtained from traces collected over several years, we observe some phenomena such as the impact of the introduction of TCP SACK which reduces the unnecessary retransmissions, the large percentage of network reordering, etc. By further studying the statistical properties of TCP anomalies, we find that, while their aggregate exhibits Long Range Dependence, anomalies suffered by individual long-lived flows are on the contrary uncorrelated. Interestingly, no dependence on the actual link load is observed.

Key words: Traffic Measurements, TCP.

* Corresponding author.
Email addresses: `mellia@tlc.polito.it` (Marco Mellia), `meo@tlc.polito.it` (Michela Meo), `luca.muscariello@orange-ftgroup.com` (Luca Muscariello), `dario.rossi@enst.fr` (Dario Rossi).

[1] This work was funded by the European Community FP6 Network of Excellence "EuroFGI", and by the Italian Ministry of University, Education, and Research (MIUR) under the PRIN MIMOSA. We would like to thank the GARR for allowing us to monitor some of their backbone links, the Politecnico di Torino network facilities CESIT for their patience in supporting the monitoring at our campus LAN, and finally the people from NLANR for allowing people to access their invaluable traces.

1 Introduction

In the last fifteen years, the interest in data collection, measurement and analysis of Internet traffic has increased steadily. Indeed, by acknowledging the failure of traditional modeling paradigms, the research community focused on the analysis of the traffic characteristics with the twofold objective of i) understanding the dynamics of traffic and its impact on the network elements and ii) finding simple models for the design and planning of packet-switched data networks.

By focusing on passive traffic characterization, we face the task of measuring Internet traffic, which is particularly daunting for a number of reasons. First, traffic analysis is made very difficult by the strong correlation in both space and time, due to the closed-loop behavior of the client-server communication paradigm. Second, the complexity of the involved protocols, and of TCP in particular, is such that a number of phenomena can be studied only if a deep knowledge of the protocol details is exploited. Finally, some of the traffic dynamics can be understood only if the forward and backward packets are jointly analyzed – which is especially true for the detection of erratic flows behavior.

TCP is the dominant transport protocol currently deployed in the Internet, and supports a wide range of applications such as web and email applications, and newly emerging peer-to-peer applications. Given this reliance on TCP, there is currently great interest in understanding TCP performance and its limiting factors (such as network congestion and sender/receiver buffer limits). TCP relies on a sliding window protocol to implement both error recovery and congestion control, numbered segments and acknowledgments are used to detect and react to network losses and out-of-sequence delivery. Moreover, TCP identifies a segment loss as a congestion notification event, upon which the sender bitrate must be reduced. Defining, measuring and quantifying the impact of those events can be instrumental in both understanding and improving TCP performance.

Therefore, identify and classify all these *anomalies* is of primary importance. The term anomaly is used for any segment that has a sequence number which is not the expected one, such as out-of-sequence and duplicate segments. In this context, the objective of this paper is to propose a heuristic classification technique of anomalies that may occur during the lifetime of a TCP connection. After describing and validating the heuristic, we apply it to a set of real traffic traces collected at different measurement points in the Internet. Compared to all previous work, for the first to the best of our knowledge, we provide the reader an extended set of measurements collected in different periods, and form different networks. Results show that the proposed classification allows us to inspect a plethora of interesting phenomena: e.g., the impact of TCP SACK on the occurrence of unnecessary retransmissions, the almost negligible impact of the daily load variation on the occurrence of anomalous events, the surprisingly large amount of network reordering, the correlation of anomalies, just to name a few.

The identification and characterization of anomalies carried out in this paper can help in verifying simplistic assumptions that researchers adopt when facing both modeling and design problems. For example, one commonly accepted assumption is that packet loss in the Internet can be modeled as a Bernoulli process. While this assumption is used to simplify the modeling process, to the best of our knowledge, it has never been verified. Our study is therefore instrumental to the definition of crucial processes, such as loss and out-of-sequence processes, that drive network performance.

Finally, we believe that the results presented here have a wider scope, since they come from an extended measurement campaign and analysis of the possible events a packet may suffer in the current Internet. While these observations are very important for issues related to TCP, they are also essential when developing some applications such as real-time applications. For example, the fact that network reordering occurs only to back-to-back packets (i.e., back-to-back packets can arrive at the receiver in inverted sequence, but separated by a negligible amount of time) can be considered and exploited when designing a VoIP application, in which packets are rarely sent back-to-back due to the low source bitrate. The picture changes when considering a high-quality IPTV service, in which the out-of-sequence delivery of packets must be carefully managed.

The remainder of the paper is organized as follows: after presenting the related work in Sec. 2, we introduce the methodology developed to classify TCP anomalies in Sec. 3. Results obtained from test-bed, Campus LAN and Backbone traces are presented in Sec. 4, Sec. 5 and Sec. 6 respectively. Finally, Sec. 7 concludes the paper.

2 Related work

Internet measurement is the subject of many studies, that can be broadly classified as using either active or passive methodologies: in the former case, injecting traffic into the network is instrumental to perform the measurement, whereas passive studies analyze existing traffic obtained from packet traces. The advantage of the latter measurement technique, on which we restrict our attention in the following, is that it allows to leverage large amounts of traffic (possibly millions of samples) without the active probe overhead.

Several studies exploit passive analysis of traffic to focus on packet loss. Many of them pertain to the area of Network Tomography [1–3], which concern the inference of the internal network characteristics based on end-to-end observations. Specifically, such studies focus on identifying the "root cause" links, i.e., the links that are responsible for the majority of packet losses and reordering. [1] requires the senders to cooperate with the receiver by time-stamping the packets in order to detect shared congested links using end-to-end passive measurements. By correlating the delay experienced by packets belonging to the same flows, authors detect shared bottleneck links. Other techniques to identify lossy links are presented in [2], in which authors compare Bayesian inference, random sampling and linear programming technique to correlate link loss rate and to identify bottleneck links. Similarly, [3] identifies TCP loss and reordering events based on the IP identifier field, thus not requiring the observation of both directions of a TCP connection, with the aim of identifying links that are responsible for poor end-to-end TCP performance.

Rather than focusing on the description of the network links properties, we focus on investigating how the link and path loss process affects end-to-end TCP performance. This topic has already been explored in [4–7]. Specifically, in [4] authors estimate the loss ratio in the network by observing TCP sequence number of only *data* packets . In order to devise a methodology that applies also to the asymmetric routing case, acknowledgment packets are disregarded. Rather than coping with all the possible network dynamics, a subset of all possible segment arrival patterns are selected to be representative for the whole figure. All unknown sequences are disregarded. Adopting a different approach, [5] employs inference techniques to detect packet losses from reordering events: on arrival of a duplicated acknowledgment, the delay since the transmission of the first unacknowledged packet is tested using a Bayesian framework based on the distribution of round trip times of the immediately preceding packets. In case of packet drop, a high delay is suffered from the highest sequence numbered packets, since an almost-full queue must be traversed. On the contrary, that penalty is not suffered in case of packet reordering. The authors show that the proposed methodology is very effective to predict packet reordering from packet drops on long-lived flows, but fails with short-lived ones. Our aim in this paper is to provide a reliable classification of *all* possible events, and then to study their statistical properties.

The closest work to ours is presented in [6,8,7]. In [6], authors monitor the connection at the sender side only and, aiming at being as accurate as possible, they leverage information from both the data and acknowledgment streams. However, the classification considers retransmission and duplicated acknowledgment events only, therefore being very coarse. In [8] authors describe a tool for passive analysis of TCP segment traces which explicitly accounts for implementation-specific details in prominent TCP stacks. However, the tool offers great performance only if the source operating system (O.S.) is known, which is hardly verified when performing passive trace analysis. Moreover, the tool must be updated to include details about new O.S. versions. On the contrary, the algorithm we propose stems from IETF TCP standards [9] and follows a conservative approach, so that ambiguous cases are not misclassified.

Finally, in [7], authors propose a simple but detailed and efficient classification algorithm of out-of-sequence TCP segments, of which the classification proposed in this paper is a modification and an extension. Specifically, our work (of which preliminary versions appeared in [10,11]) aims at identifying and analyzing a larger subset of phenomena, thus obtaining a finer-grained classification of anomalous events. In particular, we introduce distinctions based on the cause of possible retransmissions, i.e., retransmissions due to timer expiration or Fast Retransmit, and include novel classification classes, such as false retransmissions caused by TCP flow control mechanisms.

3 Methodology

The methodology adopted in this paper extend the approach followed in [7]; the original algorithm discriminates among out-of-sequence segments due to i) packet retransmissions by the TCP sender, ii) network duplicates, and iii) network reordering. We adopt a passive measurement technique and, building over the same idea, we correct and complete the classification rules to include other event types. Besides, we distinguish among other possible *kinds* of out-of-sequence or duplicate packets, but we also focus on the *cause* that actually triggered the segment retransmission by the sender. Since different implementations of the TCP/IP stack may use different algorithms and parameter setups, instead of trying to identify the plethora of TCP dialects, the algorithm we propose conforms to TCP standards, as defined in [9] and following RFCs. We prefer to possibly have a larger number of unknown events, rather than suffering from a larger misclassification probability.

TCP Packets flowing in both directions are recorded so that both data segments and ACKs are analyzed; to track the TCP sender status, the IP and TCP layers are exposed to the analyzer. Figure 1 sketches a typical TCP flow evolution: connection setup, data transfer, and tear-down phases are highlighted. The measurement point (or sniffer) can be located anywhere in the path between the client and the server; as shown in the picture, the sniffer can be close to clients (and servers) when measurements are taken at the network edge (as it happens, e.g., on campus LAN) . Since we rely on the observation of both data and ACK segments, problems may arise if asymmetric routing forces data and ACK segments to follow different paths. In such case, flows are ignored.

A flow starts when the first SYN is observed, and ends when either the tear-down sequence (FIN/ACK or RST messages) is observed, or when no segment is monitored for an amount of time larger than a given threshold [2] . By tracking both flow

[2] Since for some flows the tear-down sequence is never observed, to avoid memory starvation we use a 15-minute timer, which is large enough to avoid discarding on-going flows [12].

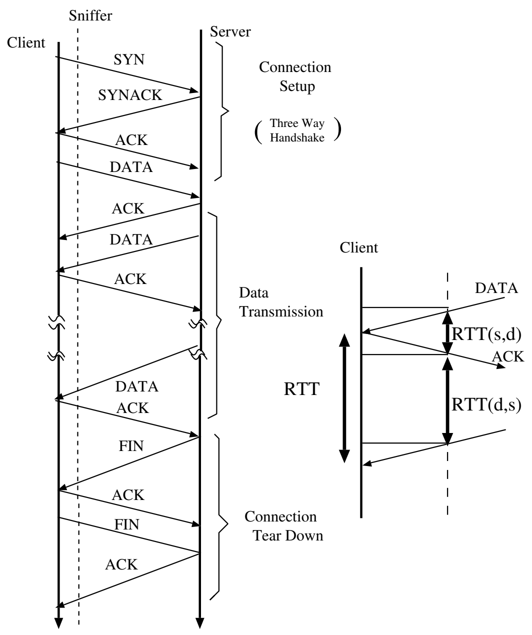

Fig. 1. TCP flow evolution over time and *RTT* Estimation Process.

directions, the sniffer correlates the sequence number of data segments to the ACK number of backward segments. Three possible cases are in order:

- *In-sequence*: if the sender sequence number of the current segment corresponds to the expected one;
- *Duplicate*: if the data carried by the segment has already been observed before;
- *Out-of-sequence*: if the sender sequence number is not the expected one, and the data carried by the segment has never been observed before.

The last two cases are indication that some *anomalies* occured during the data transfer: to further discriminate such anomalous events, we devise a fine-grained classification and implement it in Tstat [13], which we then use to analyze the collected data. The decision process used to classify such anomalous events makes use of the following variables:

- $RTT_{\min}$: the Minimum Round Trip Time (*RTT*) estimated since the flow started;
- *RT*: the Recovery Time, i.e., the time interval between the observation of the anomalous segment and of the previously observed segment with the *largest* sequence number;
- ΔT: the inverted-packet gap, i.e., the difference between the time the anomalous segment and of the last previously received data segment have been observed;
- *RTO*: the sender Retransmission Timer value, computed according to [14] from the running estimate of the average and standard deviation of the RTT;
- $DupAck$: the number of duplicate ACKs referring to the current segment sequence number.

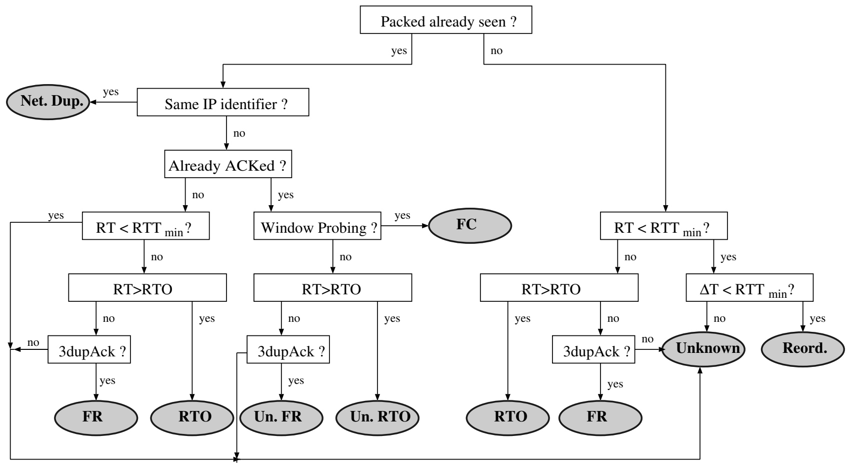

Fig. 2. Classification heuristic Flow-Chart.

3.1 Heuristic Classification of Anomalies

Following the flow diagram of Figure 2, we describe the classification heuristic. Given an anomalous segment, the process initially checks if the segment has already been observed by comparing the current TCP sequence number with those carried by segments observed so far. The anomalous segment can be classified as either duplicate or out-of-sequence. In the first case (left branch of the decision process), the IP identifier field is compared with the original packet: if they are the same, the anomalous packet is classified as **Network Duplicate** (Net. Dup.). Network duplicates may stem from malfunctioning apparatuses, mis-configured networks (e.g., Ethernet LANs with diameter larger than the collision domain size), or, finally, from unnecessary retransmissions at the link layer (e.g., when a MAC layer ACK is lost in a Wireless LAN). Unlike in [7], we classify as network duplicate all packets with the same IP identifier *regardless* of ΔT: indeed, there is no reason to exclude that a network duplicate may be observed at any time, and there is no relation between the *RTT* and the time a network can generate some duplicate packets.

When the IP identifiers are different, the TCP sender may have performed a retransmission. If all the bytes carried by the segment have already been acknowledged, then the segment has successfully reached the receiver, and therefore this is an unneeded retransmission. Either a Retransmission Timer (*RTO*), or the Fast Retransmit mechanism has fired: i) if the recovery time is larger than the retransmission timer ($RT > RTO$), the segment is classified as an **Unneeded Retransmission by *RTO*** (Un. *RTO*); ii) if 3 duplicate ACKs have been observed, the segment is classified as an **Unneeded Retransmission by Fast Retransmit** (Un. FR).

Unneeded retransmissions are also due to the *window probing* algorithm implemented by TCP flow control mechanisms. When receiver flow control kicks in, the TCP sender keeps retransmitting old segments to force the immediate transmission of an ACK so as to probe if the receiver window *RWND* is larger than zero. In particular, a retransmission due to **Flow Control** (FC) is identified if i) the sequence number is equal to the expected sequence number decreased by one, ii) the segment payload size is of zero length, and iii) the last announced *RWND* in the reverse ACK flow was equal to zero. This is a new possible case which was previously neglected in [7]. Finally, if none of the previous conditions holds, we label the anomaly as **Unknown** (Unk.). Unneeded retransmissions may be due to failures in performing selective retransmission of missing segments, a misbehaving source, a wrong estimation of the *RTO* at the sender side, or, finally, an ACK loss on the reverse path; distinguishing among these causes is impossible by means of passive measurements.

Let us now consider the case of segments that have been already observed but have not been acknowledged yet: this is possibly the case of a retransmission following a packet loss. Indeed, given the recovery mechanism adopted by TCP, a retransmission can occur only after at least an *RTT*, since duplicate ACKs have to traverse the reverse path to trigger the Fast Retransmit mechanism. When the recovery time is smaller than $RTT_{\min}$, then the anomalous segment can only be classified as **Unknown** [3] ; otherwise, it is possible to distinguish between **Retransmission by Fast Retransmit** (FR) and **Retransmission by *RTO*** (*RTO*) adopting the same criteria previously detailed. Retransmissions of already observed segments may be due to data segments lost on the path from the measurement point to the receiver, and to ACKs delayed or lost before the measurement point.

Finally, let us consider the right branch of the decision process, which refers to out-of-sequence anomalous segments. In this case, the classification criterion is simpler: indeed, out-of-sequence can be caused either by the retransmission of lost segments or network reordering. Since retransmissions can only occur if the recovery time *RT* is larger than $RTT_{\min}$, by double checking the number of observed duplicate ACKs and by comparing the recovery time with the estimated *RTO*, we distinguish **retransmissions triggered by *RTO* or FR**. On the contrary, if *RT* is smaller than $RTT_{\min}$, then a **Network Reordering** (Reord.) is identified if the inverted-packet gap ΔT is smaller than $RTT_{\min}$. Network reordering can be due to load balancing on parallel paths, route changes, or parallel switching architectures which do not ensure in-sequence delivery of packets [15].

3.2 Dealing with wrong estimates

The proposed classification algorithm relies on estimates whose values must be derived from the packet trace itself: especially at the flow setup, these estimates

[3] In [7], authors use the *average RTT*; however, many *RTT* being smaller the average *RTT*, we believe that the use of the average *RTT* forces a uselessly larger amount of unknown.

may not be very accurate, or even valid. In the following, we explain how to cope with wrong estimates and limit their impact.

3.2.1 The RTT estimation

All measurements related to the RTT are critical, since they are part of RTT_{min} and RTO estimation. RTT estimation is updated during the flow evolution according to the moving average estimator standardized in [14]. Given a new RTT measurement m, the average RTT estimate is updated according to a low pass filter $E[RTT] = (1-\alpha)E[RTT] + \alpha\, m$ where α is equal to $1/8$.

Since the measurement point is neither co-located at the transmitter, nor at the receiver, the direct RTT measurement is not available. As shown in Figure 1, let $RTT(s,d)$ denote the *half path RTT sample*, which represents the delay at the measurement point between an observed data segment flowing from the source to the destination, and the corresponding ACK on the reverse path. Let $RTT(d,s)$ denote the delay between the ACK and the following segment. An estimate of the total round trip time RTT is given by

$$RTT = RTT(s,d) + RTT(d,s).$$

The estimation of the average RTT is not biased, given the linearity of the expectation operators. Therefore, it is possible to estimate the *average RTT* by

$$E[RTT] = E[RTT(s,d)] + E[RTT(d,s)].$$

The standard deviation of the connection RTT, $std(RTT)$, can be correctly estimated if $RTT(s,d)$ and $RTT(d,s)$ are independent measurements, which usually holds.

Finally, given $E[RTT]$ and $std(RTT)$, it is possible to estimate the sender retransmission timer as in [14]:

$$RTO = E[RTT] + 4std(RTT).$$

Regarding the estimation of RTT_{min}, we have

$$RTT_{\text{min}} = \min(RTT(s,d)) + \min(RTT(d,s)).$$

In general, this estimator gives a biased estimate of the real minimum RTT, since $RTT_{\text{min}} \leq \min(RTT)$. This leads to a conservative classification algorithm, which possibly increases the number of anomalies classified as unknown, rather than risking some misclassification.

3.2.2 No RTT Sample Classification

There are some cases in which the RTT measurement is not available, but an anomalous event is detected. This happens in particular at the connection start, since no valid RTT samples may be available at the very beginning. Since most of TCP flows are very short [13], these events are quite frequent and cannot be neglected. The choice of the initial values of RTO and RTT_{min} is critical, since inappropriate choice may lead to wrong classifications. We adopt the following approach:

- if no valid RTT samples have been collected, the heuristic uses $RTO = 3s$ and $RTT_{min} = 1ms$ as default values
- the RTO is forced to assume values larger or equal to $1s$, according to [14]
- the RTT_{min} is forced to be larger than $1ms$.

3.2.3 Batch Classification

Given that a TCP sender can transmit several segments per RTT, it may happen that more than one anomalous segment is detected. This occurs, for example, when the TCP sender adopts the SACK [16] extension and retransmits more than one segment per RTT, or when packets belonging to the same window travel on a path in which packets are reordered and arrive with “strange” patterns. In such cases, the measurements of RT and ΔT may be wrong and lead to incorrect classifications. We therefore implement a filter in the classification heuristic that correlates the classification of the current anomaly with the classification of the previous segment. In particular, if the current recovery time RT is smaller than $E[RTT]$ (suggesting that the segment belongs to the same window as the previous one) and the previous segment was not classified as in sequence, we classify the current anomalous segment as the previous one.

For example, consider a simultaneous SACK retransmission of two segments triggered by a Fast Retransmit. The first retransmitted segment is correctly classified given that three duplicate ACKs have been observed on the reverse path, and the RT is larger than RTT_{min}. However, the second retransmitted segment cannot be correctly classified, given that no duplicate ACK has ever been observed, and $RT < RTT$. By explicitly considering the classification of the first segment, it is possible to correctly identify the second segment as a retransmission triggered by Fast Retransmit.

4 Testbed Validation

To proof-check the correctness of the heuristic, we setup a simple testbed to perform controlled anomalies injection and verify the accuracy of the proposed heuristic.

The testbed involves two hosts, acting as transmitter and receiver, connected by a 100 Mbps Ethernet link through a router capable to artificially inject anomalies. All hosts run Linux operating system, with the `netem` [17] network emulator directly enabled in the kernel. Linux is forced to use TCP NewReno, with no SACK option. The traffic is captured at both the sender and receiver side, so that it can later be processed to inspect the TCP anomaly identified by the proposed heuristic. Furthermore, for comparison purposes, kernel-level TCP statistics at the server side are also recorded using `netstat`. `netstat` gives access to some statistics about TCP loss recovery mechanisms, and duplicate/out-of-sequence segments.

A number of experiments were performed, varying the amount of data to be transferred, and controlling the percentage of packet loss, duplication and re-ordering injected by the network emulator. In particular, link capacities were limited to 5 Mbps by means of a token bucket. The propagation delay is set to 50 ms, and the *minimum* Round Trip Time equal to 100 ms.

Not surprisingly, the analyzer correctly identifies all **network duplicates** and all **network reordering**, so that the results perfectly match those of the kernel level log.

Therefore, in the following we focus on the impact of **packet loss**. For a more detailed analysis, we consider two scenarios. In the first one (named Short), a TCP connection transferring 3 kB of data was generated, while in the second case (named Long), a TCP connection transferring 1 MB of data was involved. Each experiment was repeated 5000 and 500 times, for Short and Long file size respectively. I.i.d. packet loss were introduced, with average percentage set to 5%.

The comparison of the heuristic versus kernel-level classifications is reported in Table 1 in terms of the number of identified RTO, FR and Unk. events. In the table, we distinguish results derived considering the transmitter/receiver side packet trace.

As a general remark, the heuristic classification exhibits a very good matching with the kernel truth. To further inspect the result, let us first consider the short flow scenario. An excellent match between the heuristic classification and the kernel log is achieved. Indeed, 952 out of 971 RTO events and 31 out of 50 FR events are correctly classified at the transmitter; 39 missing events were classified as unknown. Similar results are observed at the receiver side.

Considering the long flows case, 97% of losses are recovered by FR and 3% by RTO according to the kernel measurements. The heuristic classification identifies 85% of FR, 17% of RTO and 6% of unknown events when run at the sender side. The apparent overestimation of the RTO percentage is due to the batch classification. Indeed, the Linux kernel considers as "fast retransmit" multiple losses recovered after the first RTO, while, due to the batch classification, the heuristic classify these losses as RTO events. In Table 1 the values in brackets correspond to the classifications obtained by Linux kernel naming. In this case, the match is excellent. Notice also that in the testbed scenario $std(RTT)$ is very small, since no cross traffic is queued at the router. This is therefore a worst case scenario. Nonetheless, excellent results are achieved by the proposed heuristic classification.

Table 1
Comparison of Heuristic versus Kernel Classifications.

Scenario	Measurement point	RTO	FR	Unk.
	Heur. transmitter	952	31	39
Short	Heur. receiver	970	40	0
	Kernel transmitter	971	50	
	Heur. transmitter	302(99)	1553(1756)	106
Long	Heur. receiver	300(93)	1506(1713)	86
	Kernel transmitter	58	1762	

5 Measurement Results - Campus Network

In this section, we present results obtained running the algorithm on campus LAN traces. We consider measurements that have been gathered from the sole Internet edge link of our institution. Our campus network behaves like an Internet stub, because the access router is the sole gate to the Internet. More than 7,000 hosts are present: most of them are clients, but some servers are regularly accessed from outside. We collected all packets flowing into the access link that connects the campus border router to the GARR network [18], the nation-wide ISP for research and educational centers. Measurements were performed over several years. We present only a subset of results, and in particular:

- from the 6th to the 7th of February 2001. The bandwidth of the access link was 14 Mbit/s;
- from the 29th of April to the 5th of May 2004. The bandwidth of the access link was 28 Mbit/s.

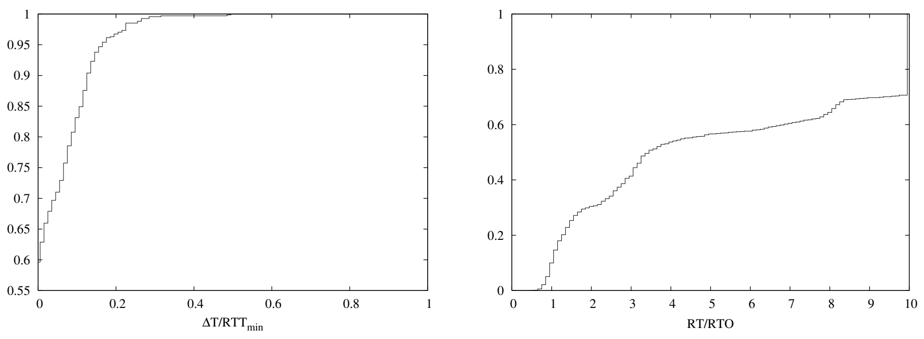

Fig. 3. CDF of the ratio between ΔT and RTT_{min} (left plot). CDF of the ratio between RT and the actual RTO estimation (right plot).

5.1 Impact of RTT_{min} and RTO

We first present a set of measurements to double-check the impact of the choices described in Sec 3.2. In particular, we are interested in the impact of the measurement of RTT_{min} and RTO. The first one is involved in the classification of the network reordering anomalies that may occur when identifying two out-of-sequence segments separated by a time gap smaller than RTT_{min}. Left plot of Figure 3 shows the Cumulative Distribution Function (CDF) of the ratio between the inverted packet gap ΔT and the value of the RTT_{min} considering only TCP anomalies classified as network reordering. Measurements referring to 2004 are reported, and similar results are obtained considering the 2001 dataset. The CDF clearly shows that ΔT is much smaller than the RTT_{min}. This is due to the very small inverted packet gap of reordered packets and suggests that the initial choice of RTT_{min} is appropriate. The fact that the inverted packet gap of reordered packets is very small is an interesting observation that can be useful when designing protocols that have to cope with network reordering. For example, multimedia applications like VoIP or IPTV services may exploit this in order to reduce receiver buffer and play-out delay.

Right plot of Figure 3 reports part of the CDF of the ratio between the actual Recovery Time RT and the corresponding estimation of the RTO when considering anomalous events classified as retransmissions by RTO. Also in this case, results referring to the 2004 dataset are reported. The CDF shows that $RT > RTO$, which is a clear indication that the estimation of the RTO is not critical [4] . Moreover, it can be noted that about 50% of the cases have a recovery time which is more than 5 times larger than the estimated RTO. This apparently counter intuitive result is due to the fact that the heuristic neglects the RTO back-off mechanism implemented in TCP, which consists in doubling the RTO value at every retransmission of the same segment. Not considering the back-off mechanism leads to a robust and conservative approach.

[4] The small percentage of values below 1 is due to the batch classification heuristic. In those cases, $RT < RTT$, and therefore $RT < RTO$ holds.

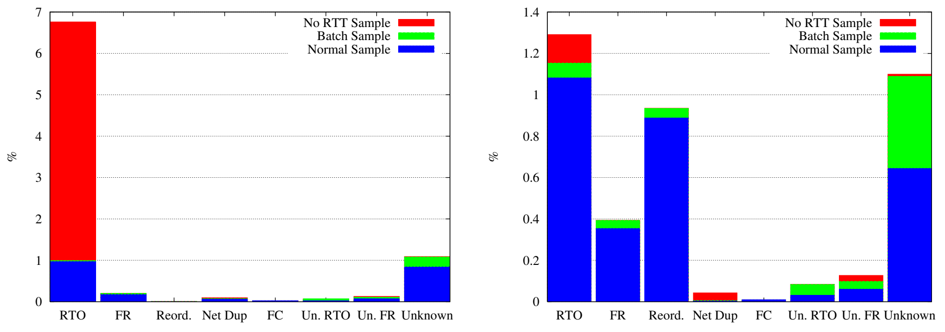

Fig. 4. Average percentage of anomalous events: outgoing traffic on the left, incoming traffic on the right.

Table 2
Traffic volume and anomaly breakdown, incoming direction for 2004 traffic.

Pkts %	Flows %	Anom %	**Flow Class**	RTO %	FR %	Reord %	NetDup %	FC %	Un.RTO %	Un.FR %	Unkn %
3.77	76.22	9.86	**Short**	5.13	1.84	20.56	35.24	0.02	39.29	0.01	3.19
5.14	16.71	6.63	**Medium**	6.32	2.87	10.38	13.24	0.37	12.21	0.06	3.76
91.09	7.07	83.51	**Long**	88.55	95.30	69.06	51.52	99.61	48.50	99.93	93.05
100.0	100.0	100.0	*Breakdown*	33.65	12.07	28.12	0.60	0.54	3.23	1.45	20.35

5.2 Aggregate Results

Figure 4 reports the percentage of identified anomalous events during the 2004 period. The right plot of Figure 4 refers to incoming traffic, i.e., traffic whose destination host is inside our campus LAN; the left plot reports measurements on outgoing traffic, i.e., traffic whose destination host is outside our campus LAN. Each bar in the plot explicitly underlines the impact of the batch classification (light pattern) and of the lack of RTT samples (dark pattern).

Considering the incoming anomalies classification (right plot of Figure 4), we observe that there is a large dominance of network reordering and retransmissions due to RTO expiration. Fast Retransmit occurs only for a very small portion of the total retransmissions. This is related to the characteristics of today's data traffic, which is mainly composed of very small file transfers that do not allow TCP to trigger Fast Retransmit [19]. The coarser classification in [7] did not allow to observe such effect. Network reordering is also quite common, while network duplicates are seldom identified. Similarly, receiver flow control is rarely used to slow down server transmission, and a small percentage of unnecessary retransmissions is also present. Finally, a rather large percentage of anomalies that could not be classified is observed. Investigating further, we noticed that, for most of them, the recovery time is smaller than the estimated RTO (therefore missing the retransmission by RTO classification), but larger than the RTT_{min} and the number of duplicate ACKs is smaller than 3. We suspect that they may be due to either i) transmitters that trigger the Fast Retransmit with just 1 or 2 duplicate ACKs, or ii) servers with aggressive an RTO estimation that triggers the retransmission earlier than that assumed by the RTO estimation at the measurement point. We prefer to classify these events as unknown rather than risking to misclassify them to follow a conservative approach.

For what concerns the impact of the batch classification and the lack of a valid RTT sample, observe that the first is evenly distributed among all classification cases, while the latter one has a large impact on the identification of retransmissions by RTO. This is due to the lack of valid RTT samples at the very beginning of the TCP connection.

When considering outgoing traffic (left plot of Figure 4), the heuristic correctly identifies the anomalies. RTOs with no RTT samples are the large majority, since client HTTP requests are so short that no RTT sample is ever collected. Neither Network Reordering nor Duplicates are identified. This is correct, given that our Campus LAN is a switched Ethernet LAN in which packets can be duplicated or reordered only in case of malfunctions. This supports the validity and robustness of the classification heuristic.

Considering the average percentage of total anomalies, about 4% and 8% of incoming and outgoing traffic, respectively, is affected by an anomaly; however, the average values are not representative due to the process non-stationaries.

Finally, in order to further verify the validity of our heuristic, we split the flows into three different classes based on their length. *Short* flows (also called mice in the literature) have payload size no longer than 5 segments. *Long* flows (the so called elephants) have payload size larger than 20 segments, and the *medium* sized flows whose payload size is larger than 5 segments, but shorter than, or equal to 20 segments. Let us focus on the classification of anomalous events for incoming traffic during the 2004 time period; results relative to outgoing flows and to the 2001 dataset are similar and they are not reported here due to lack of space.

Table 2 reports some statistics on the amount of flows, packets and anomalies observed discriminating flows based on their length and the occurrence of different anomaly types; for the sake of clarity, we omit to further mention the "batch" or "no RTT sample" classification. More specifically, the table reports the occurrence of each anomalous event, i.e., the ratio between the number of anomalies occurring in a given class over the total number of such anomalies. The bottom line reports the anomaly *breakdown*, i.e., the relative occurrence of each event type irrespectively of the flow class. Considering the percentage of packets and flows, it can be seen that a small amount (about 7%) of long flows accounts for more than 91% of the totally observed packets but only 83.51% of the anomalies. Also, short flows account for about 9.86% of the total anomalous events despite generating only 3.77% of the packets. This means that short flows are relatively more penalized by the occurrence of anomalies than long flows.

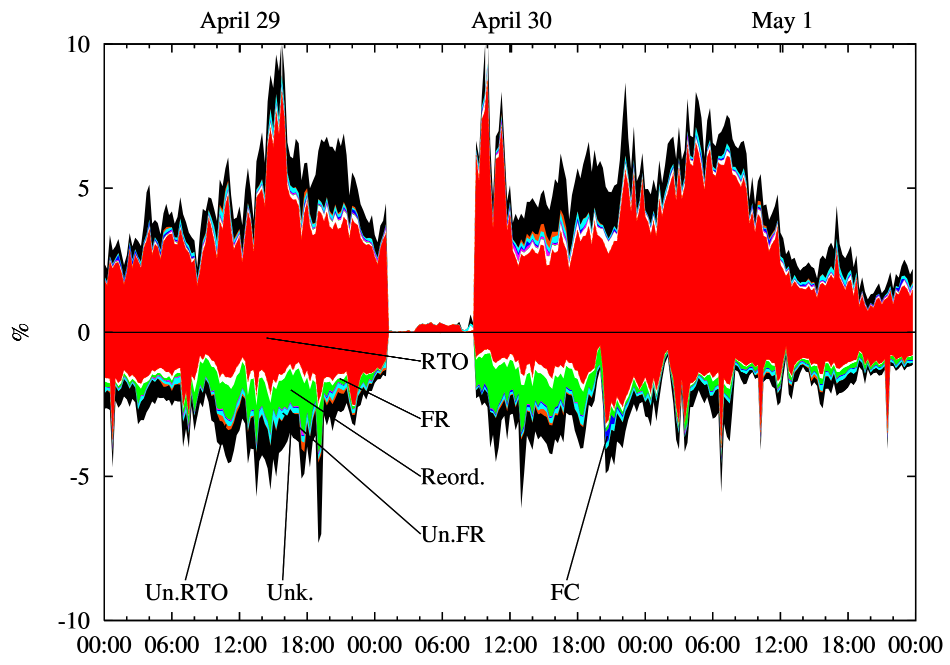

Fig. 5. Anomalous events occurrence versus time: outgoing traffic (positive values), and incoming traffic (negative values); 2004 measurements.

As expected, retransmissions due to RTO expiration are distributed (almost) proportionally to the flow length among all classes, with a little bias toward short flows, which have a larger percentage of RTOs. Retransmissions due to Fast Recovery are mainly triggered by long flows; indeed, short flows cannot trigger Fast Retransmit [19]. Interestingly, short flows have a significant number of anomalies due to packet reordering, which account for more than one fifth of the total number of anomalies: a packet belonging to a short flow has a larger chance to suffer from a reordering than a packet belonging to a long flow. Probably, in this scenario, many long flows are transported over paths in which packet reordering does not occur. Indeed, packet reordering is peculiar to specific paths: there are paths over which no reordering is ever observed, while in some others reordering is quite frequent. The same reasoning applies to network duplicates. Neglecting flow control and unnecessary retransmissions as their occurrence is marginal, we observe that the anomalies classified as unknown are mainly associated to long flows. A significant portion of those events, namely 15%, is represented by packet retransmissions triggered by a Fast Retransmit occurred after the reception of the second (rather than the third) duplicate acknowledgment, as proposed in [20]. This behavior, which is *not* compliant to the IETF standard [9], is typical of most flavors of Microsoft Windows platforms; indeed, as reported in [21–23], by default Windows retransmits a segment when the second [5] duplicate acknowledgment is received. Two additional remarks can be made on this regard. First, we point out that, although the wide

[5] This feature is controllable with the registry parameter `HKEY\_LOCAL\_MACHINE\SYSTEM\CurrentControlSet\Services\Tcpip\Parameters\TcpMaxDupAcks`

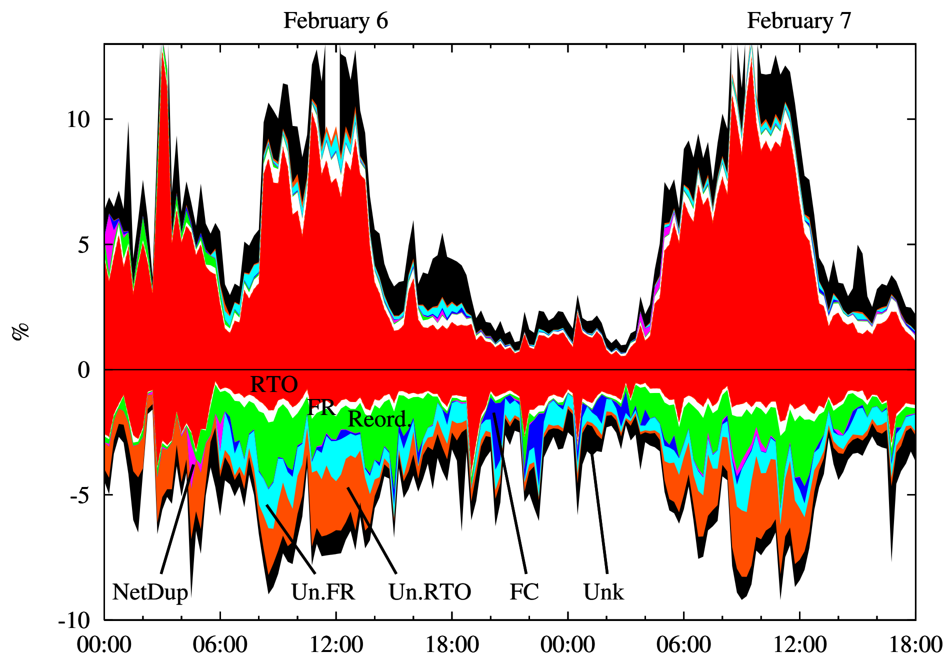

Fig. 6. Anomalous events occurrence versus time: outgoing traffic (positive values), and incoming traffic (negative values); 2001 measurements.

majority of today *clients* is represented by Windows platforms, the direction of the traffic we are observing is representative of incoming data downloaded by internal clients from Internet *servers*, where the mixture of OS is less biased and thus the phenomenon is not prominent. Second, we observe that, by classifying as Fast Retransmit the retransmission triggered by the second DUPACK, the overall percentage of Fast Recovery would raise up to 14% of all anomalous events. However, we prefer to be strictly compliant to the standard, and thus we conservatively avoid any possible misclassification.

5.3 Time characterization

We report in this section the occurrence of anomalies versus time. We omit the sub-classification due to batch arrivals or to the lack of valid RTT samples for the sake of clarity. Figures 5 and 6 depict the time evolution of the volume (in percentage, normalized over the total number of packets in each direction) of anomalous segments. Measurements aggregate anomalies over an interval of time equal to 15 minutes. The detailed classification is outlined in colored slices whose size is proportional to the percentage of that particular event. Positive values refer to outgoing traffic; negative values refer to incoming traffic. Figure 5 refers to the 2004 period, while Figure 6 refers to the 2001 period.

Apart from the network outage that is evident between midnight and 9:00 of April 30th 2004, the first result is that TCP anomalies are highly non stationary. There

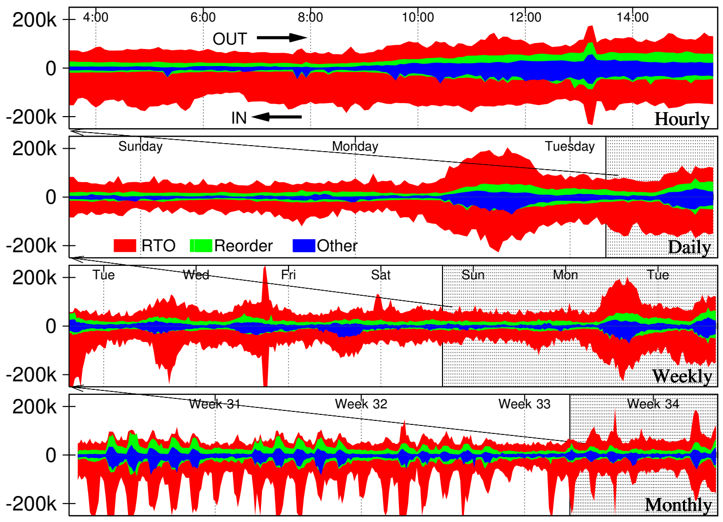

Fig. 7. Amount of incoming (negative values) and outgoing (positive values) anomalies at different timescales.

are some peaks of significant magnitude that reach 10% during 2004 and almost 15% during 2001. Considering the incoming traffic of 2004 trace, Retransmissions by RTO, Network Reordering and Unknowns are the largest parts of the anomalies. TCP Flow control seldom kicks in, and a negligible number of Unnecessary Retransmissions is identified. Surprisingly, the typical night and day effect, which is commonly present on the total traffic volume (and also in the considered link), is not anymore visible when considering TCP anomalies. Notice also that the last two measurement days are weekend-days. In particular, Labour Day was celebrated on Sunday the 1st of May in Italy: therefore the link load during that weekend was particular low. Nonetheless, the RTO fraction is almost equal to the one observed during busy hours of weekdays; only the Network Reordering seems to disappear. This indicates a weak correlation between link load and TCP anomalies: we will further investigate this considering backbone traffic in Section 6.

If we compare 2004 and 2001 incoming traffic plots, it can be noted that after three years Unnecessary Retransmissions almost disappear. Indeed, in the 2001 trace, a significant portion (about 40% or more) of the anomalous events is identified as unnecessary retransmissions. This fact is explained by the vast popularity of TCP SACK that was used by only 21% of flows during 2001. Its usage increased to 90% of flows during 2004 (this information is obtained from the three-way-handshake negotiation). At the same time, a reduction of the fraction of TCP anomalies is noticeable comparing 2001 and 2004 measurements: this may be related to access link capacity that was a major bottleneck during 2001, so that severe congestion was common.

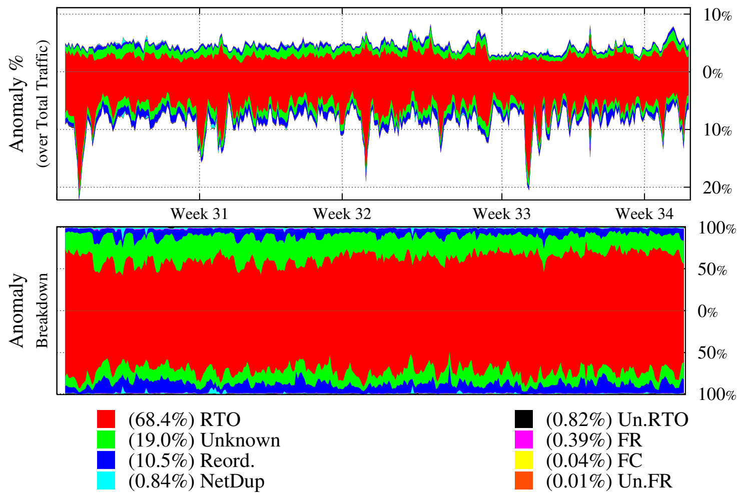

Fig. 8. Anomalies percentage normalized over the total traffic (top) and anomalies breakdown (bottom).

6 Measurement Results - Backbone Traffic

In this section we present results obtained through traffic traces collected from different backbone networks. The first one is a packet-level trace gathered from the Abilene Internet backbone, publicly available on the NLANR Web site [24]. The trace, which is 4 hours long, has been collected on June 1st, 2004 at the OC192c Packet-over-SONET link from Internet2's Indianapolis node toward Kansas City. The second measurement point is located on the GARR backbone network [18], the nation-wide ISP for research and educational centers. The monitored link is a OC48 Packet-over-SONET from Milano-1 to Milano-2 nodes, and statistics reported in the following are relative to August 2005. Traces show different *routing symmetry*: while GARR traces reflect almost 100% of traffic symmetry, only 46% of ABILENE traffic is symmetric; this is of particular relevance, since the algorithm relies on the observation of both data and ACK segments to track TCP flow evolution.

All the gathered flow-level and aggregated statistics, related to anomalous traffic as well as to many other measurements that we do not report here, are publicly available on the Tstat Web page [25]. In the following, we *arbitrarily* use "In" and "Out" tags to discriminate between the traffic directions – even though neither an inner nor an outer network region actually exists since the measurement point is located in the backbone.

Figure 7 depicts the time evolution of the volume of anomalous segments classified by the proposed heuristic applied to the GARR traffic; in order to make plots readable, only *RTO* and network reordering are explicitly reported, whereas the other kinds of anomalies are aggregated. Measurements are evaluated with different granularity over different timescales: each point corresponds to a 5 minute window in the hourly plot (top plot), a 30 minute window for both daily and weekly plots and a 2 hour interval in the monthly plot (bottom plot). Both link directions are displayed in a single plot: positive(negative) values refer to the Out(In) direction. The plot of the entire month clearly shows a night-and-day trend, which is mainly driven by the link load. The trend is less evident at finer time scales, allowing us to identify stationary periods during which advanced statistical characterization can be performed. The same monthly dataset is used in Figure 8, which reports the volume of anomalies normalized over the total link traffic (top plot), and the anomaly breakdown, i.e., the amount of anomalies of each class normalized over the total number of anomalous events (bottom plot). Labels report, for each kind of anomaly, the average occurrence over the whole period obtained aggregating both traffic directions. The *RTO*s account for the main part of anomalous events, being almost 70%: as previously noted, *RTO* events correspond to the most likely reaction of TCP to segment losses. Packet reordering is also present in a significant amount. The average total amount of anomalous segments is about 5% considering Out traffic direction, and about 8% considering In direction, with peaks ranging up to 20%.

Besides the precise partitioning of such events, it is interesting to notice that the periodical trend exhibited by the *total* amount of anomalies observed early in the monthly plot of Figure 7, almost completely vanishes when considering the *normalized* amount shown in top plot of Figure 8 (previously noted from Figures 5 and 6). Thus, while the total amount depends on the traffic load, the percentage of anomalies seems independent from the load. This is an interesting finding that clashes with the usual assumption that the loss probability increases with load. Very similar results were also obtained for the Abilene trace, confirming that the percentage of anomalies over the total amount of traffic remains steadily constant even when the link load exhibits large fluctuations. An explanation of the possible cause of such a counter-intuitive result may lay in the greedy nature of TCP, which pushes the instantaneous offered load to the bottleneck capacity causing packet losses. Indeed, even if the average offered load is much smaller during off-peak periods, congestion may arise on bottleneck links (i.e., possibly at the network access).

We now focus on the correlation structure of the time series counting the number of anomalies per time unit (i.e., the anomalies counting process). Though this could be presented in many different ways, we decided to use the Hurst parameter H as a very compact index to identify the presence of Long Range Dependence (LRD) in the series. Intuitively, $H = 0.5$ indicates uncorrelated processes, while

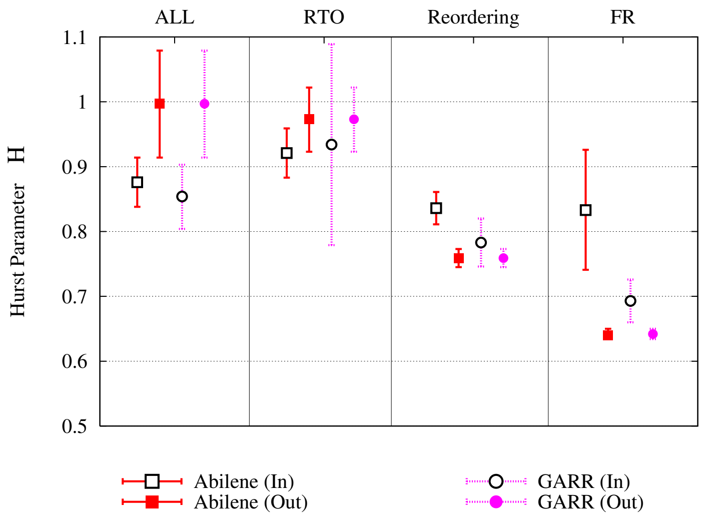

Fig. 9. Hurst parameter estimate for different anomalies and traces.

increasing values of $H \in [0.5, 1]$ are symptomatic of increasing correlation in the process, with values around $H = 0.7$ usually considered typical of LRD. The H estimation is performed using the Multi Resolution Analysis (MRA) [26,27] with the code provided in [28]. Stationary time-windows have been considered. In particular, stationarity holds for the two backbone link traces when considering few hours periods. H estimate is depicted in Figure 9 distinguishing anomalies, measurement points and traffic directions. In all cases, H is considerably higher than 0.5, meaning that these series are LRD. Reordering and FR series exhibit H smaller than for the *RTO* series, for all traces and traffic directions. By further measuring H in different subsets of the original trace and by using different time units (from 10 ms to a few seconds), we gathered sufficient information to confirm the presence of LRD in the anomalous traffic.

6.2 Analysis of a Single Elephant Connection

Other interesting observations can be drawn by focusing on a specific network path, where it can be assumed that network setup is homogeneous during the observation period: in particular, we consider the longest TCP flow that has been observed on our campus LAN. It corresponds to a large file download from a host in Canada which lasted about eleven hours (from 9:10am to 8:12pm of Wednesday the 5th of May), transferring 359 MBytes in 307,124 packets. The flow throughput varied from 60 kbit/s up to 160 kbit/s, while the measured *RTT* varied in the $[300, 550]$ ms range. Figure 10 plots time series curves where each series corresponds to one of the anomalies [6]. First of all, we point out that, while each time series is non-stationary, the aggregated series is stationary. The largest number of anomalies refer

[6] Time series curves have been smoothed using a low pass filter to allow a better understanding of the process.

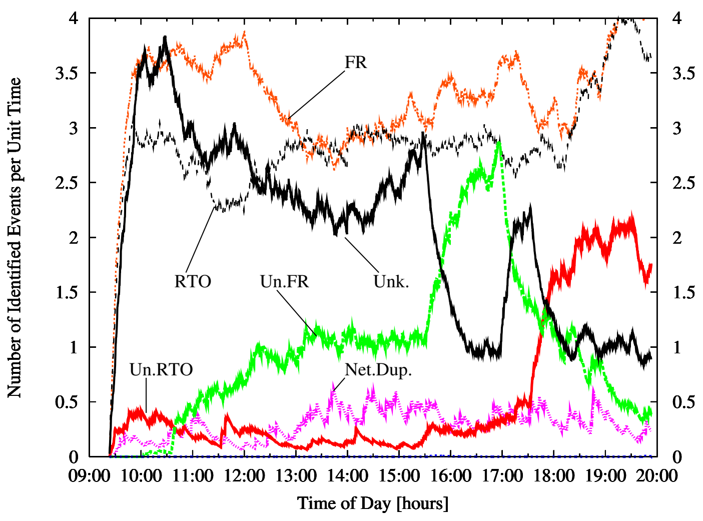

Fig. 10. Time plot of anomalous events for the selected elephant flow.

to *RTO* and FR, and this holds for the whole flow lifetime. It is worth noticing that typically, while the number of *RTO* increases, the number of FR decreases (and vice-versa): large values of *RTO* probably correspond to high congestion levels, during which the sender congestion window is reduced and the chance of receiving three dup-ACKs is therefore smaller. Similar behaviors were observed for other long flows not shown here for the sake of brevity. No network reordering has been identified during the whole flow lifetime. As previously observed, this suggests network reordering to be path dependent, and a similar reasoning applies to network duplicate events.

We now focus on the fluctuation of the rate and the statistical properties of the counting process of i) the received packets per time unit and ii) the anomalous events per time unit. The time unit is 100 ms long. According to MRA, both these processes are stationary: moreover, from our measurements, stationarity usually holds for long flows in general. Figure 11 depicts the log-scale diagram for both the considered time series. The log-scale diagram is a flexible tool to evaluate the power spectra, and it has good properties once dealing with LRD process, and it provides excellent H estimates [26]. The diagram reports the process correlation estimate: a horizontal line is symptomatic of an uncorrelated process, while linear slope for large scales is indication of high correlation. Figure 11 shows that the traffic generated by the TCP flow is *highly* correlated for the whole flow lifetime. This has been previously observed and is well known [26]. The slope of the curve leads to an estimation of the parameter $H \simeq 0.74$; the time scale of the *RTT* is visible as the "bump" around 400 ms in the left-hand side of the curve and represents an intrinsic time periodicity for the flow dynamic related to the RTT. Indeed, the *RTT* has been recognized as a natural time cutoff scale i) below which traffic is highly irregular and uncorrelated, and ii) above which traffic is long-term correlated. These results confirm similar findings in the literature [29,30].

Considering the anomalous event process (solid dots), the plot clearly states that no

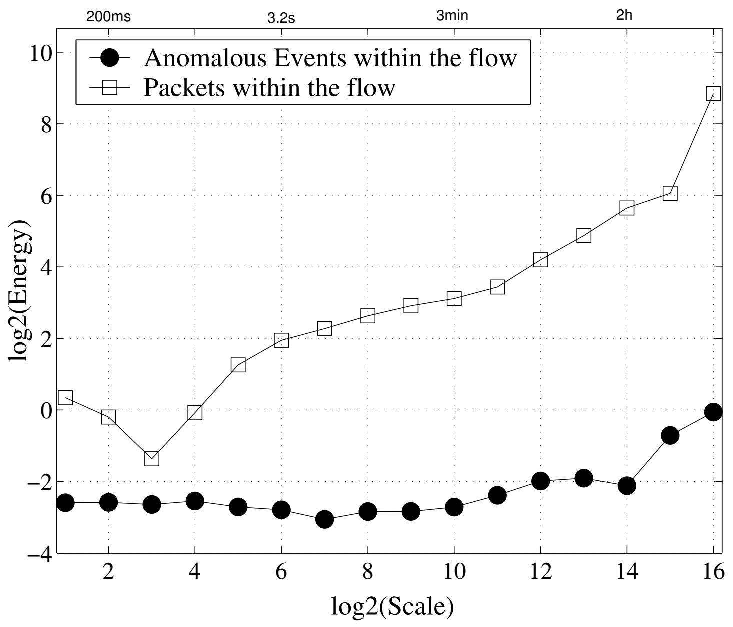

Fig. 11. Energy plots of the time series of i) number of segments and ii) number of anomalies.

correlation is present. The fact that a single flow suffers from uncorrelated anomalies seems a counter-intuitive results, since we already noticed that the *aggregated* anomaly exhibits LRD phenomena. A possible intuition of this may stem from the fact that anomalies observed from a *single* flow are sampled from the aggregate on larger time scales (in the order of RTT) so that they are perceived as uncorrelated.

7 Conclusions and Discussion

In this paper we defined, identified and studied the TCP anomalies, i.e., out-of-sequence segments, by extending and refining the heuristic originally proposed in [7]. We applied our technique to traffic traces collected at both network edges and core links. By studying the statistical properties of these phenomena, some interesting aspects of TCP were observed. First, though the absolute amount of anomalies is highly dependent on the link load, both its relative amount over the total traffic and the anomaly breakdown are almost independent from the current load. This is a quite interesting finding that clashes with the usual assumption that the loss probability increases with load.

Second, when considering the aggregation of traffic flows at any Internet link, the arrival process of anomalies is highly correlated. On the contrary, when considering individual flows, the suffered anomalies are almost i.i.d.. This is an important observation that validates the quite popular assumption that packet drops can be modeled by i.i.d. processes.

Third, *RTO* is the most preferred mechanism to identify packet loss by the TCP sender. Flows are often too short to allow Fast Retransmit to kick in. Novel congestion control and recovery mechanisms should cope with this packet loss identification scheme.

Fourth, network reordering can be large on some paths, while negligible on other paths. When present, reordered packets are separated by few ms, i.e., an amount of time that is negligible with respect to the flow RTT. This observation has consequences that extend beyond TCP, e.g., on multimedia protocols that must cope with network reordering.

Finally, the proposed heuristic has been proved to be reliable and conservative. It can be adopted to monitor traffic in real time, giving network operators deep insight into the offered performance.

References

[1] D. Rubenstein, J. F. Kurose, and D. F. Towsley, "Detecting shared congestion of flows via end-to-end measurement," *IEEE/ACM Transactions on Networking*, vol. 10, no. 3, pp. 381395, 2002.

[2] V. Padmanabhan, L. Qiu, and H. Wang, "Server-based inference of Internet performance," *in IEEE INFOCOM'03*, San Francisco, CA, USA, April 2003.

[3] E. Brosh, G. Lubetzky-Sharon, Y. Shavitt, "Spatial-Temporal Analysis of passive TCP Measurements." *in IEEE INFOCOM'05*, Miami, FL, USA, March 2005.

[4] P. Benko and A. Veres, "A Passive Method for Estimating End-to-End TCP Packet Loss," *in IEEE GLOBECOM'02*, Taipei, TW, November 2002.

[5] N. Fonseca and M. Crovella, "Bayesian Packet Loss Detection for TCP," *in IEEE INFOCOM'05*, Miami, FL, USA, March 2005.

[6] M. Allman, W. Eddy and S. Ostermann, "Estimating Loss Rates With TCP," *ACM Performance Evaluation Review*, 31(3), December 2003.

[7] S. Jaiswal, G. Iannaccone, C. Diot, J. Kurose, D. Towsley, "Measurement and classification of out-of-sequence packets in a tier-1 IP backbone", *IEEE/ACM Transaction on Networking*, V.15, N.1, pp.54-66, February 07.

[8] S.Rewaskar, J.Kaur, F.D.Smith, "A passive state-machine approach for accurate analysis of TCP out-of-sequence segments", *ACM SIGCOMM Computer Communication Review archive*, V.36, N.3, pp:51-64, July 2006.

[9] M. Allman, V. Paxson and W. Stevens, "TCP Congestion Control," *IETF Request For Comment (RFC2581)*, Standards Track, April 1999.

[10] M.Mellia, M.Meo, L.Muscariello, "TCP anomalies: identification and analysis", *Tyrrenian International Workshop on Digital Communication - TIWDC*, Sorrento, Italy, July, 4-6 2005.

[11] L. Muscariello, M. Meo, M. Mellia and D. Rossi, "Passive Measurement of TCP Anomalies," *in IEEE International Conference of Communication (ICC'06)*, Istanbul, Turkey, June 2006.

[12] G.Iannaccone, C.Diot, I.Graham, and N.McKeown, "Monitoring very high speed links," *ACM Internet Measurement Workshop*, San Francisco, CA, November 2001.

[13] M. Mellia, R. Lo Cigno and F. Neri, "Measuring IP and TCP behavior on edge nodes with Tstat," *Computer Networks*, Vol. 47, No. 1, pp. 1-21, Jan. 2005.

[14] V. Paxson and M. Allman "Computing TCP's Retransmission Timer" *IETF Request For Comment (RFC2988)*, Standards Track, November 2000.

[15] J.C.R. Bennett, C.C. Partridge and N. Shectman, "Packet reordering is not pathological network behavior," *IEEE/ACM Transactions on Networking*, Vol. 7, N. 6, pp.789-798, Dec. 1999.

[16] M. Mathis, J. Mahdavi, S. Floyd, A. Romanow, "TCP Selective Acknowledgment Options", *IETF Request For Comment (RFC2018)*, Standards Track, October 1996.

[17] S. Hemminger, "Network Emulation with NetEm," *Linux Conference AU (LCA'05)*, Aug. 2005.

[18] GARR Network, `http://www.noc.garr.it/mrtg/RT.TO1.garr.net`

[19] M. Mellia, M. Meo and C. Casetti, "TCP Smart Framing: a Segmentation Algorithm to Reduce TCP latency," *IEEE/ACM Transactions on Networking*, Vol. 13, No. 2, pp. 316-329, Apr. 2005.

[20] J. C. Hoe, "Improving the Start-up Behavior of a Congestion Control Scheme for TCP," *In ACM SIGCOMM'96*, August 1996.

[21] D. MacDonald and W. Barkley, "Microsoft Windows 2000 TCP/IP Implementation Details," *Microsoft Technical Report*, `http://www.microsoft.com/technet/itsolutions/network/deploy/depovg/tcpip2k.mspx`

[22] "Description of Windows 2000 and Windows Server 2003 TCP Features," *Microsoft Technical Report 224829 (rev. 5.0)*, November 2004, `http://support.microsoft.com/kb/q224829/`

[23] "Microsoft Windows CE 3.0 TCP Retransmission Behavior," *Microsoft Technical Report*, April 2004, `http://msdn.microsoft.com/library/default.asp?url=/library/en-us/wcecomm/html/_wcesdk_TCP_Retransmission_Behavior.asp`

[24] AbileneIII packet trace, `http://pma.nlanr.net/Special/ipls3.html`

[25] Tstat Web Page, `http://tstat.tlc.polito.it`

[26] P. Abry and D. Veitch. "Wavelet Analysis of Long-Range Dependent Traffic," *IEEE Transaction on Information Theory*, Vol. 44, No. 1, pp:2-15, Jan. 1998.

[27] D. Veitch and P. Abry, "A statistical test for the time constancy of Scaling Exponents," *IEEE Transactions on Signal Processing*, Vol. 49 No. 1, pp. 2325-2334, Oct. 2001.

[28] Darryl Veitch Home Page, Wavelet Estimation Tools, `http://www.cubinlab.ee.mu.oz.au/~darryl`

[29] D. R. Figueiredo, B. Liu, V. Misra and D. F. Towsley,"On the autocorrelation structure of TCP traffic," *Computer Networks*, Vol. 40, No. 3, pp. 339-361, 2002.

[30] H. Jiang and C. Dovrolis, "Why is the Internet traffic bursty in short (sub-*RTT*) time scales?," *in ACM SIGMETRICS'05*, Banff, CA, Jun. 2005.