High short-term bit-rates from TCP flows *

Khushboo Shah
University of Southern California
Dept. of Electrical Eng.
Los Angeles, CA-90089
khushboo@usc.edu

Stephan Bohacek
University of Delaware
Dept. of Electrical and Computer Eng.
Newark, DE-19711
bohacek@eecis.udel.edu

Abstract

Micro-bursts from TCP flows are investigated. The chip-rate is introduced and used to quantify the short-term bit-rate of TCP flows. This paper examines packets with chip-rates above the 90^{th} percentile. The examination is performed over time scales ranging from 244 μs to 125 ms. Two issues are addressed, the impact and the causes of the micro-bursts. It is found that the packets with high chip-rate experience an elevated probability of burst losses. For example, the probability of a burst loss is up to 10 times larger for packets sent in micro-bursts. Furthermore, in some settings, these packets experience higher loss rate in general. It is also found that micro-bursts cause an increase in queuing delay. The causes of these micro-bursts are investigated. One finding is that at short-time scales, ACK clocking, which should reduce micro-bursts, is not functioning correctly. For example, in some cases, most of the packets contained in micro-bursts are ACKed at a rate that is less than half of the data rate.

1 Introduction

There are two principle causes of bursty network traffic, the number of connections sending data is bursty and the bit-rates are bursty [24]. The former is well understood and is related to the long tailed file distribution [27], [8]. On the other hand, bit-rates have received less attention [24], [22], [13]. This paper examines the bursts at short time-scales. We call these bursts *micro-bursts* to distinguish them from well-studied burstiness that result from user behavior and file sizes [27]. In [7], the authors noticed that users and applications induce self-similarity at time-scales of several hundreds of millisecond and above. In this paper, we focus strictly on smaller time-scales, 244 μs to 125 ms. We further distinguish the micro-bursts from the bursty behavior of TCP that has been investigated in [8]. In [8], pseudo-self-similarity was found to be caused by TCP, specifically from exponential backoff, time-out, and slow-start. While much of this previous work focuses on the correlation structure, here the bursts themselves are studied. It is found that subtle aspects of TCP and its implementation result in the micro-bursts observed here. Furthermore, the impact of these micro-bursts is investigated.

Numerous papers have indicated that micro-burst may be detrimental to network performance. As a result, many papers and RFCs propose different techniques to reduce or eliminate micro-bursts. For example, SCTP utilizes a scheme that limits the number of back-to-back transmissions [25]. Furthermore, there has been a large amount of effort into the development of TCP pacing schemes that limit bursts (e.g., see [1] and references therein). Such research was based on the intuition that bursts of packets can have negative effects on the sender's flow and the network. However, the existence, impact, and causes of TCP induced micro-bursts in real networks has never been verified.

This paper meets this need. First, *the impact of the micro-burst* is studied. It is shown that the micro-burst can result in dramatic increase in loss probability, especially, the probability of burst-losses. Further, evidence is provided that burst-losses impact not only the flow that causes the burst-losses but also other flows in the network. It is observed that these micro-bursts also impact queueing delay. Second, *the causes of micro-bursts* are investigated. It is found that ACK clocking is not functioning at short time-scales. This strongly suggests that ACK clocking should not be relied upon to limit the data rate at short time-scales and that TCP pacing be deployed, especially on servers with high-speed access links.

While average data rate is relatively straightforward to

*This work was prepared through collaborative participation in the Collaborative Technology Alliance for Communications and Networks sponsored by the U.S. Army Research Laboratory under Cooperative Agreement DAAD19-01-2-0011. The U.S. Government is authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright notation thereon.

Proceedings of the 13th IEEE International Symposium on Modeling, Analysis, and Simulation of Computer and Telecommunication Systems (MASCOTS'05)
1526-7539/05 $20.00 © 2005 **IEEE**

define, short-term bit-rates are not. We introduce a metric called *chip-rate* to quantify the short-term sending rate. The chip-rate of a packet (byte) at a time-scale T is defined to be the maximum number of packets (bytes) sent over any time interval of duration T that contains the packet (byte), divided by T (see Section 3). The chip-rate is defined in relation to a time-scale. While there has been some work examining the relevant time-scale for queues [15], there has been no investigation as to what the relevant time-scale of sending rates is. Due to the wide variety of RTTs that are observed in the Internet, and due to the dependence of TCP on the RTT, it is difficult, and likely misleading, to select a single time-scale. Thus, we examine four time-scales from $244\mu s$ to 125ms. Although $244\mu s$ is a very short time-scale, it is still a relevant time-scale. For example, as will be shown, packets that are sent fast at this time scale are subject to dramatically elevated probability of drops and bursts of drops, as well as an increase in RTT.

Due to space limitations, this version of the paper only presents plots from one dataset, and some issues are only discussed briefly. A more complete version of the paper is available [23]. This version of the paper will be occasionally referred to throughout the paper.

The remainder of the paper is organized as follows. Section 2 discusses the data used. Section 3 provides the definition of data chip-rate and ACK chip-rate as well as some other definitions. Section 4 discusses the chip-rates of TCP flows. Section 5 discusses the impact of micro-bursts, specifically, RTT and packet loss. Section 6 discusses the origins of micro-bursts. Section 7 discusses some related work. And finally, Section 8 provides some concluding remarks.

2 Data Description

The work presented here is based on packet traces collected from OC-48 links from two different US backbone ISPs between San Jose, CA to Seattle, WA. The traces are described in Table I. D04S packet traces contain 44 bytes of each packet while the D14N/S packet traces contain up to 84 bytes, depending upon the size of TCP options in the packet. We extracted out TCP-SACK flows that are bidirectional with respect to our observation point from datasets D14N/S and D12N/S as is commonly done [10]. We found that out of 4.1 M bidirectional TCP flows 2.1M flows were TCP-SACK for D14N/S dataset and out of 5 M bidirectional TCP flows 2.7 M flows were TCP-SACK for D12N/S dataset. These TCP-SACK flows form D14-SACK and D12-SACK datasets. This version of the paper presents analysis mostly from D14-SACK. The analysis of the D04S and D12-SACK datasets can be found in the technical report [23].

3 Terminology and Definitions

In this section, we introduce the terminology and definitions used throughout this paper.

3.1 Chip-Rate

The main focus of this paper is on short-term bit-rates. In order to make the idea short-term bit-rates more precise, we define the *chip-rate* of a packet (or byte) at a time-scale T to be the maximum number of bits sent over any time interval of duration T that contains the packet (or byte), divided by T. Values of T considered in this paper are 244μs, 1.9531ms, 15.6125ms, and 125ms. For brevity, we write only the three significant digits of these values or 2^{-k}s. Figure 1 illustrates how the chip-rate of a packet (and all the bytes in the packet) is determined. A key feature of the chip-rate is that all the packets (or bytes) that belong to a micro-burst of packets that have equal interarrival times will have the same chip-rate (the chip-rate may be the same if the inter-arrival time is nearly the same). To not be distracted by the packet size, we will often speak of the chip-rate of a byte and insist that each byte within a packet has the same chip-rate. While this precise definition of chip-rate appears to be new, the idea is not new. For example, [22] examines what they call the peak rate of a flow which is what we call the maximum chip-rate of a flow.

There are several reasons to examine short-term bit rate with chip-rates. If a flight of packets has a chip-rate of R arrives at a non-empty queue with outgoing link speed B, and no other flows are sharing the router, then the resulting added delay at the end of the flight is, $(B - R) \times t$ where the length of the flight of packet is t. For such a calculation, the chip-rate is more useful than a moving average bit-rate. As shown in Figure 1, if a micro-burst of packets has the same interarrival time, the chip-rate will correctly show a constant bit-rate during the micro-burst, but the moving average will show an artificially smaller bit-rate in the beginning and end of the micro-burst. Counting the number of packets sent during a RTT (i.e., the congestion window) is essentially the chip-rate but at a time-scale of the RTT. However, as shown in [15], the time scales for queuing is often smaller than the RTT. Furthermore, as will be made clear, a congestion window's worth of packets that is spread over the entire RTT is far different from a congestion window's worth of packets sent in a small burst.

This paper closely examines "heavy-hitter" bytes. Specifically, we examine the packets and bytes with chip-rates larger than the 90^{th}, 95^{th}, and 99^{th} percentiles as defined in Table 2. Such packets (bytes) will be referred to as *90-fast*, *95-fast*, and *99-fast packets (bytes)* respectively. When the exact percentile is not important and when a general property of packets (bytes) in any of these percentiles

Proceedings of the 13th IEEE International Symposium on Modeling, Analysis, and Simulation of Computer and Telecommunication Systems (MASCOTS'05)
1526-7539/05 $20.00 © 2005 IEEE

Table 1. Bulk sizes of OC-48 datasets (IP layer byte counts)

Set	Bb	Date	Day	Start	Dur	Dir	Src.IP	Dst.IP	Flows	Packets	Bytes	Aver.Util.	Ut.%
D04S	1	2002-08-14	Wed	10:00	60 m	Sbd (1)	317 K	1819 K	29.2 M	426.7 M	286.1 G	636.9 Mbps	25.6
D14N	2	2004-04-28	Wed	19:29	122 m	Nbd (1)	1686 K	5752 K	79.0 M	1148 M	795 G	869.4 Mbps	34.9
D14S	2	2004-04-28	Wed	19:29	122 m	Sbd (0)	357 K	12439 K	33.6 M	265.4 M	95.7 G	104.6 Mbps	4.2
D12N	2	2004-03-17	Wed	09:59	122 m	Nbd (1)	1562 K	6438 K	83.0 M	1455 M	1092 G	1194 Mbps	48.0
D12S	2	2004-03-17	Wed	09:59	122 m	Sbd (0)	282 K	8706 K	25.6 M	272.8 M	124.1 G	135.6 Mbps	5.4

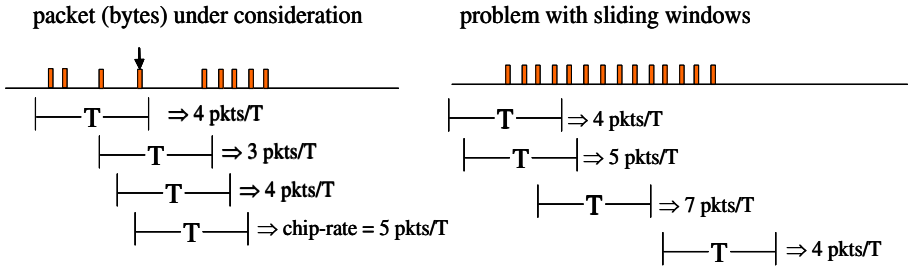

Figure 1. The left-hand figure shows how the chip-rate for the indicated packets (and the bytes in the packet) is defined. In this case four intervals are considered. Each interval contains the packet under consideration. The maximum number of packets in such intervals is 5. Assuming that the packets are the same size, then the chip-rate for the packet under consideration is 5×packet size$/T$. The right-hand figure shows the bit-rates assigned to packets in a burst if a sliding window approach is used. The first packet would receive a bit-rate of 4×packet size$/T$ as indicated. The packets in the middle of the burst would receive a bit-rate of 7×packet size$/T$. The chip-rate for each packet in the burst would be 7×packet size$/T$.

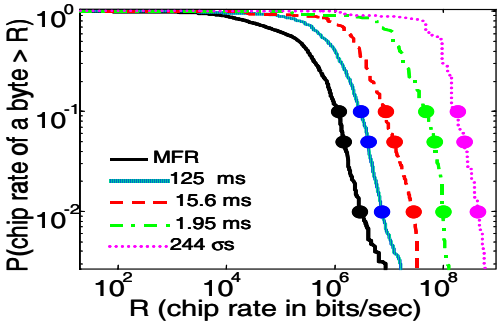

Figure 2. The figure shows CCDF of mean flow rate MFR and Chip-Rates.

is being discussed, we use the generic term *fast packets (bytes)*.

3.2 ACK chip-rate

While much of the paper is focused on the rate that data packets are sent, Section 6 examines whether the sender sends packets at a higher chip-rate than the packets are acknowledged (ACKed). To this end, we employ the *ACK chip-rate* at a time-scale T, where the ACK chip-rate of a packet (byte) is the maximum number of packets (bytes) ACKed during a time interval of length T that contains the acknowledgment for the packet (byte), divided by T. Since this examination focuses only on TCP-SACK, the ACK chip-rate for out-of-order arrivals can be determined. As a result, the ACK chip-rate in the vicinity of a dropped packet can be determined.

Besides insisting on TCP-SACK, a few other implementation dependent issues need to be addressed to fully define the ACK chip-rate. If delayed ACKs are used, then a single ACK packet will acknowledge all the bytes in two data packets. This affects the ACK chip-rate. While RFC 1122 states that delayed ACKs "SHOULD" ACK at most two packets, some implementations will ACK a large number of packets if the data packets arrive in a small amount of time. Since these ACKs are noncompliant and skew the relationship between ACK rate and data rate in an obvious way, we do not include the ACK chip-rate for those packets that are ACKed by packets that ACK more than two data packets. Thus, we restrict our attention to the case where ACK clocking should work correctly.

Table 2. Percentile of MFR and Chip-Rates

Set D14	90^{th} percentile (Mbps)	95^{th} percentile (Mbps)	99^{th} percentile (Mbps)
MFR	1.18	1.47	2.9
2^{-3} sec	3.056	4.22	7.6
2^{-6} sec	8.65	12.60	28.42
2^{-9} sec	49.1	69.7	102.2
2^{-12} sec	188.8	245.66	441.7

4 TCP Chip-Rates

Figure 2 shows the distributions of the chip-rates at time-scales 125ms, 15.6ms, 1.95ms, and 244μs as well as the distribution of the mean flow rate (MFR). The MFR is defined as the file size divided by the flow duration, and is included as a point of reference. The distributions are the byte weighted. Thus, if a byte is selected at random[1] from the link measured, the distribution of chip-rate assigned to this byte is the one shown in the figure. The distribution of the MFR is also byte weighted.

The distribution of the MFR gives the impression that data is sent at a relatively slow rate. Indeed, Table 2 shows that the 99$^{\text{th}}$ percentile for the D14-SACK dataset is around 3Mbps. At long time-scales, the chip-rates show relatively small values while at short time-scales the chip-rates are very high. For example, at the 244μs time-scale, the 99$^{\text{th}}$ percentile of the D14-SACK dataset is 440Mbps. The comparison of MFR and chip-rate at 244μs, illustrates the bursty nature of TCP, indeed, the peak bit-rate is 100 times more than the average bit-rate. While bytes sent above the 99^{th} percentile are, of course, rare, one can expect that one out of every 100 bytes to be part of such a micro-burst.

While 440Mbps is very high bit-rate, 244μs is a very short period of time. Indeed, a micro-burst of only 9 packets within 244μs will result in this chip-rate in the 99^{th} percentile. On one hand, 9 packets seems to be a small number of packets, while on the other hand, 440Mbps is quite fast. In order to understand the relevance of the 244μs time-scale, one must examine the impact of burst as this time-scale. Such an investigation is carried out in Section 5 where it is shown that even a small number of high chip-rate packets have a significant impact.

5 The Impact of Micro-Bursts

In this section we examine the impact of these fast bytes on the queuing delay and loss probability. Furthermore, we examine the impact of the burst drops on other flows. It will be shown that packets with higher chip-rate experience a significantly larger probability of burst drops and, in some settings, experience a higher overall loss probability. Furthermore, packets with higher chip-rate experience elevated queuing delay. And finally, we will see that after a burst drop, the queuing delay is reduced for up to a second.

We focus on downstream RTT (DSRTT) and downstream loss probability. With SACK blocks, these DSRTT and loss can be accurately determined even during fast retransmit. Thus, the focus is on the downstream part of the connection, from the measurement point to the receiver and back to the measurement point.

5.1 Downstream Loss Probability, High Chip-Rates, and Bursts

Since sending packets in a micro-burst might lead to a burst of drops, we consider drop events where possibly multiple packets are lost. For example, if a sequence of L packets are lost, we will define this as a drop event of length L. However, it was found that there are sequence of packets where not every packet is dropped, but where most of the packets are dropped. Thus, we extend the definition of a drop event to be a sequence of packets where 70% of the packets are dropped. With a slight abuse of terminology, we define a *drop event of length* L to be a drop event where L packets are lost (thus L out of a sequence of $L/0.7$ packets are dropped). The 70% threshold was used because it was noticed that a large number of drop events had a few packets that were not dropped. Experimenting with several thresholds found that threshold between between 60% and 80% perform similarly, with the 70% selected as a compromise.

The top plots in Figure 3 show the probability that a packet was dropped and was in a drop event of length at least L. This figure shows this probability for all packets (shown with the dashed black line) as well as for the packets with high chip-rates. Note that the probability of a packet being dropped and being in a drop event of length at least 1 is simply the packet loss probability. By examining the plots for $L = 1$, it can be seen that the impact of sending packets with high chip-rates has a mixed impact on the loss probability. At the 125 ms time-scale, the fast packets have lower loss probability, whereas other time-scales and percentiles show no change in the loss probability except for the 99-fast packets at the 244 μs which have a loss probability that is twice as high as the nominal loss probability. More startling is that at all time-scales, the fast packets have a significantly (up to ten times) higher probability of being in a large drop event than the unconditional packets have.

In Section 6.1, it is shown that a significant fraction of fast packets are sent after either a long lull where no packets were sent or after a non-MSS (maximum segment size) sized packet. Non-MSS sized packets and lulls may be due to the transport layer emptying the sending buffer and being

[1] Of course, hardware network traffic monitors typically sample packets. However, the selection of a packet at random should be weighted by the packet size. Thus, we simply select bytes.

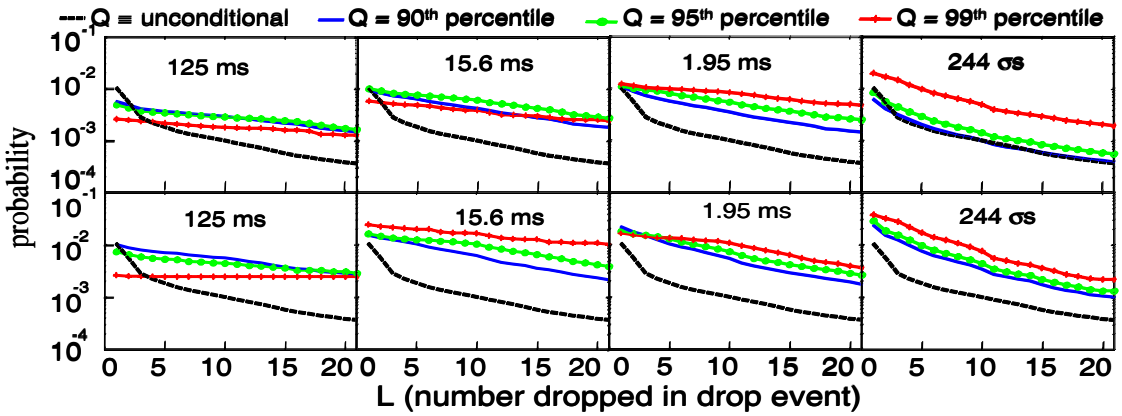

Figure 3. L versus $P\left(\text{pkt dropped and pkt in drop event of length } \geq \text{L} \mid \text{Q}\right)$. **The top figure shows the probability of burst losses for different chip-rates and time-scales. The lower plots only considers packets with high chip rates where the high chip rates occures after a non-MSS sized packet**

forced to wait for data from the application layer. When the application layer finally does deliver the data, the transport layer may send a burst of data.

The impact of such fast micro-bursts are shown in the lower plots of Figure 3. These results are for the micro-bursts that follow a non-MSS sized packet. The results for micro-bursts that follow lulls are similar [23]. Note that at all time-scales except 125 ms, the loss probability experienced by fast packets sent after a non-MSS sized packet is significantly higher than it is for unconditional packets. For example, the loss probability of a packet that is 99-fast at the 244μs time-scale is 5%, which is 5 times larger than the unconditional loss probability. While 5 times larger is clearly significant, a loss probability of 5% is especially bad since time-out becomes a serious problem with such a high loss probability[2]. Figure 3 also shows that the probability of experiencing a large drop event is greatly amplified when packets are sent with high chip-rates after non-MSS sized packets. Recall that if a large number of packets are lost, then the TCP flow might time-out resulting in an increase in the time it takes to complete the file transfer. Time-outs can add a significant amount of time to the transfer time when small files are sent. In [23], it is shown how small file transfers account for a non-negligible amount of the packets that are sent with fast chip-rates.

The slope of the curves in Figure 3 indicates the conditional probability of experiencing a burst of losses given a single loss occurs. It can be seen that in many cases, the slope for the packets with high chip-rate is much less than it is for the unconditional packets. For example, in the case of 99-fast packets at the 1.95ms time-scale, we find the 66% of the packet losses are part of a burst loss of at least 10 packets. This should be compared to the unconditional case which shows a steep drop which corresponds to only 1% of all losses being in a micro-burst of at least 10 losses.

Remark 1 *Figure 3 clearly shows that the 99-fast packets at the 244μs time-scale experience a drastic increase in both loss probability and probability of experiencing bursts of losses. In Section 4 it was remarked that the 244μs time-scale is very small and that only 9 packets sent in 244μs results in a micro-burst of packets in the 99^{th} percentile. While 9 packets may seem like a small number, Figure 3 shows while small in number, the high chip-rate of these packets has a large impact on packet losses.*

Figure 4 shows the fraction of losses that the fast packets received out of the total losses observed. Here the 90-fast packets from all time-scales are combined. These fast packets make up 18% of all packets observed. Thus, if losses were distributed uniformly, these fast packets would receive 18% of the losses. On the other hand, these are the fast packets, and if TCP is performing as hoped, meaning sending packets at high rates only when the loss probability is small, then one might expect that these 18% of the packets would receive less than 18% of the losses. Indeed, this is the case, these 18% of the packets get 12% of all the losses. However, when considering loss events of length greater than 1, it is seen that these fast bytes get more than their fair share of losses. Indeed, over 50% of the packets in loss events with length greater than 16 were experienced by these fast packets.

[2]In [5], we found that if RTT is 35 ms, then a loss probability of 1% results in flows being in time-out 1% of the time, while a loss probability of 5% results in the flows being in time-out 50% of the time.

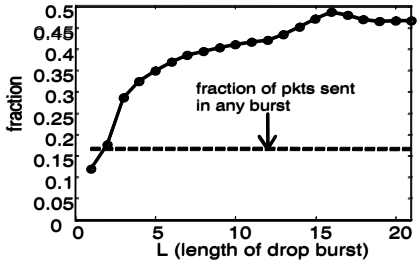

Figure 4. Fraction of All Losses that Fast Bytes Experienced.

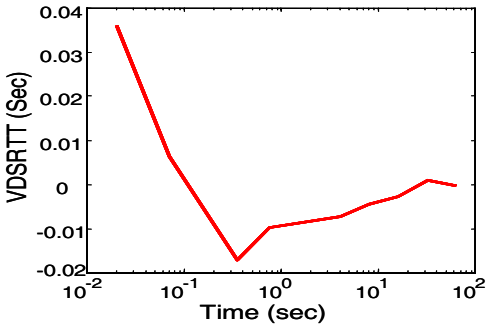

Figure 5. Average variation in DSRTT (sec) for a burst of 5 packets or more vs Time (sec).

5.2 The Impact of Bursts of Losses on other Flows

It is clear that high chip-rates result in a greatly elevated probability of burst losses. While bursts of loss will lead to a reduction in the senders' sending rate, and perhaps time-out, such burst losses may have an impact on other flows. Figure 5 shows the average variation in the DSRTT at different times after a burst of 5 or more drops. That is, for each burst of 5 or more losses, we compute $DSRTT_t - E\left(DSRTT\right)$ where $DSRTT_t$ is the value of DSRTT t seconds after the loss. More formally, we define time steps T_k. For each drop event during flow i, we define $J\left(i,k,l\right)$ to be the set of packets that were sent around time step T_k (i.e., between time $\left(T_k + T_{k-1}\right)/2$ and $\left(T_k + T_{k+1}\right)/2$) after first packet loss of the l^{th} burst loss event. $DSRTT_{i,j}$ is the DSRTT of the j^{th} packet sent by flow i. The average variation in the DSRTT at time step T_k is

$$VDSRTT_k = \frac{1}{\sum_{i\in BDF}\sum_{l\in BD(i)}\sum_{j\in J(i,k,l)}} \times \sum_{i\in BDF}\sum_{l\in BD(i)}\sum_{j\in J(i,k,l)}\left(DSRTT_{i,j} - \overline{DSRTT}_i\right),$$

where BDF is the set of flows with burst drops, $BD\left(i\right)$ is the set of burst drops experienced by the i^{th} flow, and $\overline{DSRTT}_i$ is the average downstream RTT of flow i.

We see that 20 ms after the first drop of the burst of losses, the average DSRTT is about 35 ms above its long-term average. However, the average DSRTT rapidly decreases. About 400 ms after the loss event, the average DSRTT decreases and reaches a minimum of nearly 20 ms less than its nominal value. After reaching this minimum value the average DSRTT slowly increases back to its long-term average value.

Figure 5 shows that after a burst of losses, the queuing delay is lower than its long-term average. This decrease in the queuing delay could be due to the flow under observation decreasing its sending rate in response to the loss. Alternatively, it could be from both the flow under observation decreasing its sending rate and other flows that also experienced drops during this event decreasing their sending rate. Under the first scenario, the impact of the burst loss on other flows is restricted to a lower queuing delay, while in the second scenario, the burst loss causes other flows to also have losses. Without direct measurement of the queues where these burst of drops took place, it is not possible to make definitive statements about the course of events that cause this variation in DSRTT. However, it is clear that the burst losses have an impact on other flows.

5.3 Downstream RTT and High Chip-Rate Micro-Bursts

While losses are one result of sending packets in a micro-burst, an increase in queueing delay is another potential result. One might expect that the first packet in a micro-burst would experience a typical downstream RTT (DSRTT), but as the micro-burst continues, queues begin to fill and subsequent packets experience larger DSRTT. Thus, in a micro-burst, it is expected that the last packet of the micro-burst will experience the largest DSRTT. To investigate this possibility, we determine the DSRTT of the last packet of a micro-burst and compare it to the DSRTT experienced by the average packet in the flow[3]. Specifically, we compare the average DSRTT of all packets in the flow to the DSRTT of the last packet in a sequence of packets where each packet in the sequence had a chip-rate above 90$^{\text{th}}$, 95$^{\text{th}}$, or 99$^{\text{th}}$ percentile.

Figure 6 shows the expected value of the ratio of the DSRTT of the last packet in a micro-burst and the mean DSRTT for packets in the flow. The marker is the average

[3]If an ACK is delayed and there are two packets ACKed, then the DSRTT can only be esitmated for the second packet. If the last packet of the burst is dropped, we consider the last non-dropped packet of the burst.

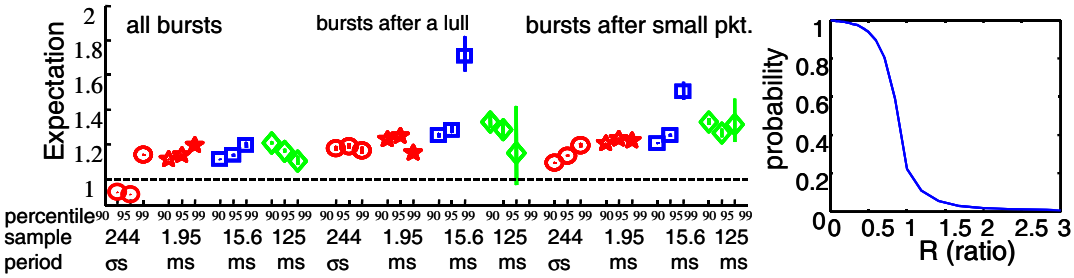

Figure 6. The left-hand figure shows the $E\left(\frac{\text{DSRTT at the end of the burst}}{\text{mean DSRTT}}\right)$ for different percentiles and different sample periods. The right-hand plot shows R versus $P\left(\frac{\text{DSRTT}}{\text{mean DSRTT}} > R\right)$. This plot shows the variation of the downstream RTT.

ratio and the vertical bars indicate the 95% confidence interval that was found via bootstrapping [3]. As is the case for packet losses, we consider micro-bursts that occur after a lull or after a packet that is smaller than the MSS was sent. The figure indicates that when considering all micro-bursts, there is an increase in the DSRTT for the 125ms, 15.6ms, and 1.95ms time-scale. However, at the 244μs the packets that come after a micro-burst of 90-fast or 95-fast packets tend to experience a slightly lower DSRTT. On the other hand, when one focuses on the micro-bursts that occur after a non-MSS sized packet or after a lull, the DSRTT is found to increase in all cases except for the packets that were in the 99$^{\text{th}}$ percentile at the 125ms time-scale. In this last case, the confidence interval includes 1, hence no conclusions can be made.

While fast bytes result in an increase in the DSRTT, the change seems rather small. On the other hand, it is unclear what constitutes a large change in DSRTT. In [2] it was found that RTT was fairly stable while [11] found that RTT wildly varies. Figure 6 (the right-hand plot) also shows the CCDF of the ratio of DSRTT for all bytes. We see that large variations in DSRTT are quite rare. For example, the ratio exceeds 1.2 less than 10% of the time, and yet Figure 6 (the left-hand plot) shows that the packets that are at the end of a micro-burst typically experience an increase of RTT of this magnitude. Thus, it can be concluded that the bytes with high chip-rate do indeed impact the DSRTT in a significant way.

The conclusion that chip-rate does affect RTT may appear to be at odds with other research. However, other research did not focus on bytes with high chip-rates. In [14], micro-congestion at a single router was studied. There it was argued that single flow has little impact on congestion. The observations presented here cannot comment on the dominating causes of queuing or the frequency with which fast packets cause congestion, but simply that the fastest packets do cause queuing. On the other hand, we have shown that periods of extreme congestion where many packets are dropped, are caused by a relatively small number of fast packets. Indeed, 18% of all packets experience 50% of the drop events that included 16 or more drops (see Section 5). [14] did not examine such extreme congestion.

In [4] it was shown that TCP's congestion window size is not related to RTT. However, in [4] all flows were considered and here we do not consider the window size but focus on chip-rate. While chip-rate and window size are related, we have observed that it is in general not true that a congestion window's worth of packets are sent in a single micro-burst, but it is sometimes the case and can result in high chip-rates. In [**?**] it was shown that the estimate of RTT at the beginning of the flow is significantly less than the average RTT during the flow. One possible cause of the increase in RTT is that the flow itself causes the RTT to increase. Thus, this result is in agreement with the findings presented here.

6 The Causes of Micro-Bursts

In this section the causes of high chip-rates are investigated. There are three obvious possible causes, high-speed connections that support high-speed chip-rates, packet compression where widely spaced packet are bunched together at a queue, and a loss of ACK clocking that causes the packets to be sent in a burst. Of these, the third can be examined and is the focus of this section.

ACK clocking has long been considered an important part of TCP as it forces the sender to not send packets faster than the bottleneck link speed [9]. The idea behind ACK clocking is that the sender can only send packets as fast as the ACKs arrive and ACKs arrive only as fast as the receiver sends them which is only as fast as the bottleneck transmits packets. The loss of ACK clocking can result in TCP micro-bursts. There are many possible causes that can result in loss of ACK clocking (e.g. see [12]). The main causes cited are

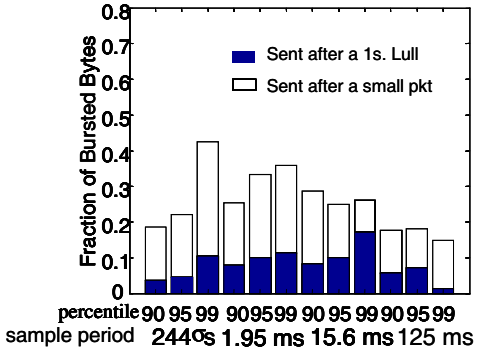

Figure 7. The figure shows some of the causes of bursts.

temporary data starvation of the transport layer, ACK compression [16], TCP's Slow-Start, lost ACKs, and reordering of packets. An important and often ignored cause of loss of ACK-clocking is the burst processing of packet arrivals by a processor sharing in operating system.

6.1 Loss of Ack-Clocking due to Data Starvation

One cause of loss of ACK clocking is temporary data starvation of the transport layer. This occurs when the transport layer runs out of data to send and hence does not send data when ACKs arrive. As a result, the number of bytes "in flight" becomes smaller than the congestion window size. Once data is provided to the transport layer, the difference between the congestion window size and the number of in flight packets is sent in burst of back-to-back packet transmissions. Such starvation could occur because of a busy server or because the server has completed one file transfer and is waiting for the client application to make the next request.

There are two ways to detect transport layer starvation. First, if there are no packets sent for an extended period of time and then a micro-burst is sent, it is likely that the micro-burst is caused by the transport layer being starved. In this analysis an "extended period of time" is defined as $\max(3 \times RTT, 1 \text{ second})$. Note that RFC 2581 says that the sender should collapse the congestion window if packets are not sent after 1 second. A second way to detect the transport layer being starved of data is when packets smaller than MSS are sent. Since TCP will always send full sized packets if possible, non-MSS sized packets indicate that the transport layer did not have data to send. Hence. these micro-bursts due to the application starving the transport layer can be detected.

The left-hand plot in Figure 7 shows the fraction of bytes sent in micro-bursts that followed a lull or, if they did not follow a lull but followed a non-MSS sized packet (thus the bytes that followed a lull may have also followed a non-MSS sized packet). At the 244 μs time-scale, we see that over 40% of the 99-fast bytes are from micro-bursts that follow either a lull or a non-MSS sized packet. At the other time-scales, the bytes after such lulls or non-MSS sized packets make-up from 15% to 35% of the fast bytes.

6.2 Breakdown of ACK-clocking at small time-scales

To further investigate causes of micro-bursts, the fast packets that follow a suspected data starvation episode are removed from the set of fast packets and the remaining packets are investigated. Specifically, this section examines whether ACK clocking was functioning correctly when these packets traversed the network. To examine ACK clocking, the ratio of the data chip-rate and the ACK chip-rate of the ACK that acknowledges the data packet is examined. (The ACK chip-rate is defined in Section 3.) Figure 8 shows the CCDF of this ratio. In the ideal case, this ratio would be one; ACKs are returned at the exact rate that data is sent. It can be seen that at the 125 ms time-scale, this ratio is not far from the ideal case with 70% of the fast packets being ACKed at the data rate. Furthermore, if the ACK chip-rate does not match the data chip-rate, the difference is not very large.

On the other hand, the 244 μs time-scale case is quite different; only 20% of the 90-fast and 95-fast bytes are ACKed at the data chip-rate and about 40% of the bytes are ACKed at a rate that is less than half the data chip-rate. The 99-fast bytes are the most extreme with only 15% of the bytes being ACKed at or above the data rate and 60% of the bytes being ACKed at rate that is less than half the data chip-rate.

The agreement between the ACK chip-rate and data chip-rate at 125 ms time-scale indicates that ACK clocking is mostly functioning (recall that the bursts that follow a data starvation episode have been removed). Thus, the bursts at this time-scale is likely due to fast end-to-end connections. Note that this result is in agreement with [13] which studied time-scales from 100 ms to 1 s and argued that at these time-scales ACK clocking is responsible for bursts. However, at smaller time-scales, it is obvious that ACK clocking is not functioning correctly and a lack of ACK clocking is a major source of bursts.

Without instrumenting the sender and receiver, we are forced to conjecture about the reason for the mismatch in ACK chip-rate and data chip-rate depicted in Figure 8. While ACK compression [16] is a well-known cause, OS processor sharing is less discussed, and yet likely a significant cause of micro-bursts. For example, Linux does not support preemptive task swapping, but gives processes 10

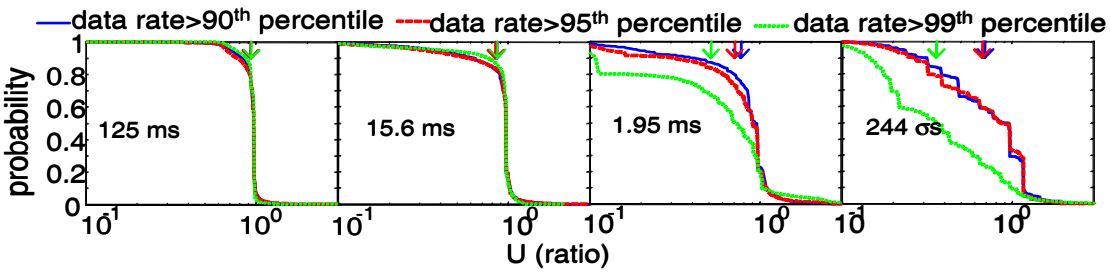

Figure 8. Y-axis shows the probability of ratio of ACK Chip-Rate to Data Chip-Rate. The arrows indicate the mean value. $P\left(\frac{\text{Ack chip-rate}}{\text{data chip-rate}} > U \mid \text{data chip-rate } > Q \text{ and burst is not due to server}\right)$

ms time-slices [18] based on priority[4]. Thus, if the sender is busy and the transport layer is the highest priority process waiting to be served, then the transport layer will typically wait 5 ms before servicing the packet. During some long, high-priority tasks such as memory management, the transport layer may have to wait far longer before processing packets. In any case, a large number of packets may arrive before the transport layer receives its time-slice. When the transport layer finally does gain control, a burst of packets is sent at the line rate. Note that this is the typical behavior of today's OSs and can only be avoided by utilizing a specialized real-time OS.

As a final note, one possible cause of the data chip-rate exceeding the ACK chip-rate is that TCP's slow-start phase allows packets to be sent twice as fast as the ACK arrival rate (if delayed ACK is used, the data rate will be less than twice the ACK rate). However, in [23] it is shown that slow-start is not an important contributor to micro-bursts.

7 Related Work

Research in network traffic characterization has taken several directions. One area that has received an enormous amount of attention is the correlation properties of packet arrivals [26]. Much of this statistical work examines the burstiness of traffic in terms of the variance of the aggregate. In contrast, this paper directly examines packets with high chip-rate. While packets with high chip-rate are related to bursty traffic, they are not the same. For example, in [17], a flow is considered to be a "porcupine" if the packets within a burst have a high chip–rate and the time between the bursts is large, i.e., if the variance of the chip-rate is large, the flow is a porcupine. Here, we do not focus on the time between bursts, but simply focus on the chip-rates of packets in relation to the chip-rate of all other packets and in relation to the ACK chip-rate. In other words, this paper examines the tail of the arrival process, not the correlation structure. The correlation structure and the tail of the stationary distribution are distinct characteristic of a process.

[13] examines burstiness in terms of correlation and focuses on burstiness at moderate time-scales. Specifically, [13] examines time-scales between 100 ms and 1s, whereas this paper considers 244μs to 125 ms. In [13], it is shown that network traffic has a periodic component around the average RTT and that bursts are due to ACK clocking. As discussed above, this investigation finds ACK clocking to be correctly functioning at 125 ms, but not at smaller time-scales.

As is done in this paper, [14] studies congestion. In [4] the impact of TCP's window size on RTT has been investigated. See Section 5 for a comparison of conclusions from these papers and the ones made here. There has been a relatively small amount of work on packet loss [21], [19], [6], but these papers did not examine the relationship between packet loss and bursts.

The general conclusion that burstiness is bad is made in [20], or more specifically, [20] shows the difficulties that arise from self-similarity. Besides the difference between self-similarity and bursts discussed above, the key difference between [20] and the work presented here is that this paper examines bursts at far smaller time scales. Furthermore, [20] focuses on burstiness that results from the application layer (e.g., file size distribution and user behavior) and from the aggregation of many flows. This paper examines bursts that appear to be caused by subtle transport layer issues such as ACK clocking. Furthermore, the bursts studied here are from a single flow, not an aggregation of flows.

[4]Recall that a hardware interrupts are preemptive. Such interrupts will preform some low level processing and call a software interrupt. Transport layer functions are called through such software interrupts. The time when a software interrupt is handled is decided by the OS and depends on relative priority and is nonpreemptive.

8 Conclusions

An examination of micro-burst was presented. Two areas were addressed. First, the impact of micro-bursts was examined. It was found that packets that make up a micro-burst have a significantly increased probability of experiencing burst of losses. Packets that are sent in micro-bursts after a non-MSS sized packet or after a lull, experience even further elevated probability of burst losses as well as an increase in the loss probability in general. Evidence is also presented that these burst drops impact other flows in the network. And finally, it was shown that packets sent with high chip-rates experienced higher downstream RTT.

The second area addressed was the causes of micro-bursts. It was shown that these micro-bursts often occur after the transport layer was starved for data by an application. Furthermore, at small time-scales, it was found that micro-bursts are not from fast connections as the chip-rate of the data packets is considerably higher than the chip-rate of the ACKs.

Acknowledgment

The authors would like to acknowledge Andre Broido (CAIDA) for his support and invaluable feedback. We would also like to thank K. C. Claffy, Colleen Shannon, and Ken Keys at CAIDA for their help and support for data processing and collection.

Disclaimer

The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the Army Research Laboratory or the U. S. Government.

References

[1] A. Aagarwal, S. Savage, and T. Anderson. Understanding the performance of TCP pacing. *Proc.IEEE INFOCOM*, 2000.

[2] J. Aikat, J. Kaur, F. D. Smith, and N. K. Jeffay. Variability in TCP round-trip times. *IMW*, 2003.

[3] B. Efron and R. J. Tibshirani. *Introduction to the Bootstrap*. Chapman and Hall, New York, 1993.

[4] S. Biaz and N. H. Vaidya. Is the round-trip time correlated with the number of packets in flight? *IMC*, 2003.

[5] S. Bohacek and K. Shah. TCP's throughput and timeout – steady-state and time-varying dynamics. *Globecom*, 2004.

[6] J. C. Bolot. End-to-end packet delay and loss behavior in the Internet. *ACM SIGCOMM*, 1993.

[7] A. Feldmann, A. C. Gilbert, and W. Willinger. Data networks as cascades: Investigating the multifractal nature of internet WAN traffic. *SIGCOMM*, pages 42–55, 1998.

[8] L. Guo, M. Crovella, and I. Matta. How does TCP generate pseudo-self-similarity? *MASCOTS*, 2001.

[9] V. Jacobson. Congestion avoidance and control. *ACM SIGCOMM Conference of Communications Architectures and Protocols*, pages 314–329, 1988.

[10] S. Jaiswal, G. Iannaccone, C. Diot, J. Kurose, and D. Towsley. Measurement and Classification of Out-of-Sequence Packets in a Tier-1 IP Backbone. In *IEEE Infocom*, 2003.

[11] S. Jaiswal, G. Iannaccone, C. Diot, J. F. Kurose, and D. Towsley. Inferring TCP connection characteristics through passive measurements. *INFOCOM*, 2004.

[12] H. Jiang and C. Dovrolis. Source-level IP packet bursts: Causes and effects. *IMC*, 2003.

[13] H. Jiang and C. Dovrolis. Why is the Internet traffic bursty in short time scales? *sigmetrics*, 2005.

[14] D. Veitch, K. Papagiannaki and N. Hohn. Origins of micro-congestion in an access router. *PAM*, 2004.

[15] R. Cruz, K. Papagiannaki and C. Diot. Network performance monitoring at small time scales. *IMC*, 2003.

[16] L. Zhang S. Shenker and D. Clark. Observations on the dynamics of a congestion control algorithm: The effects of two-way traffic. *SIGCOMM*, 1991.

[17] K. Lan and J. Heidemann. On the correlation of internet flow characteristics. *Tech. Rep. ISI-TR-574*, July 2003.

[18] R. Love. Linux kernel development. *Novell Press*.

[19] J. Kurose, M. Yajnik, S. Moon and D. Towsley. Measurement and modeling of the temporal dependence in packet loss. *INFOCOM*, 1999.

[20] K. Park, G. Kim, and M. Crovella. On the effect of traffic self-similarity on network performance. *Winter Simulation Conference*, 1997.

[21] V. Paxson. End-to-end internet packet dynamics. *IEEE/ACM Transactions on Networking*, 7, 1999.

[22] S. Sarvotham and R. Riedi and R. Baraniuk. Connection-level analysis and modeling of network traffic. *in Proceedings of IEEE/ACM SIGCOMM Internet Measurement Workshop*, 2001.

[23] K. Shah and S. Bohacek. On the burstiness of TCP traffic at backbone routers. *University of Delaware, Tech. Rep available at http://www.eecis.udel.edu/ bohacek/ChipRates/TechReport.pdf*, 2004.

[24] K. Shah, S. Bohacek, and A. Broido. On the tail of arrival process. *SPIE ITCOM*, 2004.

[25] R. Stewart, Q. Xie, K. Morneault, C. Sharp, H. Schwarzbauer, T. Taylor, M. K. I. Rytina, L. Zhang, and V. Paxson. Stream control transmission protocol (SCTP). *RFC 2960*, 2000.

[26] M. Taqqu. W. Willinger and A. Erramilli. A bibliographical guide to self-similar traffic and performance modeling for modern high-speed networks. *Stochastic Networks*, pages 339–366, 1996.

[27] W. Willinger, M. S. Taqqu, R. Sherman, and D. V. Wilson. Self-similarity through high-variability: statistical analysis of Ethernet LAN traffic at the source level. *IEEE/ACM Transactions on Networking*, 5(1):71–86, 1997.

[28] Y. Zhang, L. Breslau, V. Paxson and S. Shenker. On the characteristics and origins of internet flow rates. *In Proceedings of ACM SIGCOMM*, pages 161–174, 2002.