

To Appear in Infocom 2005

TCP vs. TCP: a Systematic Study of Adverse Impact of Short-lived TCP Flows on Long-lived TCP Flows

Shirin Ebrahimi-Taghizadeh

Ahmed Helmy

Sandeep Gupta

Department of Electrical Engineering
University of Southern California
Los Angeles, USA

{sebrahim; helmy}@usc.edu, sandeep@poisson.usc.edu

Abstract— While earlier studies have pointed out that short-lived TCP flows (mice) may hurt long-lived TCP flows (elephants) in the long term, they provide insufficient insight for developing scenarios leading to drastic drop in throughputs of long-lived TCP flows. We have systematically developed TCP adversarial scenarios where we use short-lived TCP flows to adversely influence long-lived TCP flows. Our scenarios are interesting since, (a) they point out the increased vulnerabilities of recently proposed scheduling, AQM and routing techniques that further favor short-lived TCP flows, and (b) they are more difficult to detect when intentionally found to target long-lived TCP flows.

We systematically exploit the ability of TCP flows in slow-start to rapidly capture greater proportion of bandwidth compared to long-lived TCP flows in congestion avoidance phase, to a point where they drive long-lived TCP flows into timeout. We use simulations, analysis, and experiments to systematically study the dependence of the severity of impact on long-lived TCP flows on key parameters of short-lived TCP flows – including their locations, durations, and numbers, as well as the intervals between consecutive flows. We derive characteristics of pattern of short-lived flows that exhibit extreme adverse impact on long-lived TCP flows. Counter to common beliefs, we show that targeting bottleneck links does not always cause maximal performance degradation for the long-lived flows. In particular, our approach illustrates the interactions between TCP flows and multiple bottleneck links and their sensitivities to correlated losses in the absence of ‘non-TCP friendly’ flows and paves the way for a systematic synthesis of worst-case congestion scenarios.

While randomly generated sequences of short-lived TCP flows may provide some reductions (up to 10%) in the throughput of the long-lived flows, the scenarios we generate cause much greater reductions (>85%) for several TCP variants (Tahoe, Reno, New Reno, Sack), and for different packet drop policies (DropTail, RED).

Keywords-TCP, short-lived flow, long-lived flow, DoS

I. INTRODUCTION

TCP carries 95% of today's Internet traffic and constitutes 80% of the total number of flows in the Internet [6]. A large majority of TCP flows are short-lived. The main distinction between short-lived and long-lived TCP flows (also called mice and elephants, respectively) is how the congestion window grows. Short-lived TCP flows spend most of their lifetime in the slow start phase when the congestion window is

increased exponentially. Long-lived TCP flows also start in the slow start phase, but they spend most of their lifetime in the congestion avoidance phase in which they perform Additive Increase Multiplicative Decrease (AIMD) congestion control.

In the context of fairness, it has been shown that long-lived TCP flows are disproportionately affected by non-TCP flows (e.g., UDP), since UDP flows use more than their fair share of the bandwidth compared to co-existing TCP flows. It is implicitly assumed that since TCP flows are friendly to other TCP flows, they cannot cause significant losses. Hence, most research on adverse impact on TCP has focused on non TCP-friendly malicious flows, such as UDP. Furthermore previous studies on interactions between short-lived and long-lived TCP flows have not shown sustained patterns of significant losses at congested routers. In other words, TCP self-sabotage has not been carefully studied.

In this study, we focus on the adverse impact of the interaction between patterns of short-lived TCP flows and long-lived TCP flows. Our work significantly departs from prior studies in several ways. First we use short-lived TCP flows (**not** UDP flows) to destructively affect long-lived TCP flows. This allows us to consider (a) scenarios where short-lived flows are malicious, i.e., designed to intentionally disrupt long-lived flows, as well as (b) scenarios where the short-lived flows are normal flows that coincidentally adversely affect the long-lived flows. Second, in contrast to previously studied scenarios, we show that scenarios using short-lived flows at bottleneck links do not necessarily cause maximum loss of performance for the long-lived flows. Finally, we derive rules to identify locations and durations of short-lived flows, and intervals between them that cause significant loss of throughput for long-lived flows. Our work is the first one to study and generate scenarios in which short-lived TCP flows target long-lived TCP flows so as to drastically affect their performance.

We evaluate the effectiveness of our scenarios by measuring the reduction in throughput of long-lived TCP flows. Simulation results show more than 85% reduction for various TCP flavors (Tahoe, Reno, New Reno and Sack) and different packet drop policies (DropTail, RED).

The scenarios where the short-lived flows are normal flows that severely affect long-lived flows are useful for better characterizing worst-case performance of TCP. This can be

especially useful in cases requiring satisfaction of QoS guarantees. These scenarios can also help obtain better estimates of average-case performance of TCP.

The remainder of this paper is organized as follows. Section II reviews the related work. Section III reviews TCP congestion control mechanisms. Section IV compares various options to create Denial of Service (DoS) attacks using UDP flows with adversarial scenarios using short-lived TCP flows. Section V describes our approach to create effective adversarial scenarios using short-lived TCP flows. In section VI, we explain the case studies and discuss the simulation results. Next we discuss test-bed experiments and the results in section VII. Finally, section VIII concludes this paper.

II. RELATED WORK

A lot of research has been done to develop separate models for mice [6] and elephants [2] in order to predict their performance. Padhye et al. have developed an analytical model for the steady state throughput of a bulk transfer TCP flow as a function of loss rate and round trip time [2]. This model captures the behavior of TCP's fast retransmit mechanism as well as the effect of TCP's timeout mechanism. Mellia et al. [6] have proposed an analytical model to predict TCP performance in terms of the completion time for short-lived flows. Various active queue managements [3, 4] and routing schemes [5] are proposed to ensure fairness between short-lived and long-lived flows, especially under competition for bandwidth when links operate close to their capacity. Guo and Matta proposed to employ a new TCP service in edge routers [3]. In this architecture, TCP flows are classified based on their lifetime and short-lived flows are given preferential treatment inside the bottleneck queue so that short connections experience less packet drop rate than long connections. They have shown that preferential treatment is necessary to improve response time for short-lived TCP flows, while ensuring fairness and without hurting the performance of long-lived flows. Additionally Kantawala and Turner have studied the performance improvements that can be obtained for short-lived TCP flows by using more sophisticated packet schedulers [4]. They have presented two different packet-drop policies in conjunction with a simple fair queuing scheduler that outperform RED and Blue packet-drop policies for various configurations and traffic mixes. Furthermore, Vutukury and Garcia-Luna-Aceves have proposed a heuristic and an efficient algorithm for QoS-routing to accommodate low startup latency and high call acceptance rates, which is especially attractive for the short-lived flows [5]. Moreover Jin et al have developed a new version of TCP, called FAST TCP [8] in which they use queuing delay in addition to packet loss as a congestion measure. This allows a finer-grained measure of congestion and helps maintain stability as the network scales up. Meanwhile FAST TCP employs pacing at the sender to reduce burstiness and massive losses. It also converges rapidly to a neighborhood of the equilibrium value after loss recovery by dynamically adjusting the AIMD parameters with more aggressive increase and less severe decrease as congestion window evolves.

However, in the event that short-lived TCP flows form a particular pattern and effectively influence long-lived TCP flows on their shared links, all such improvements in favor of

short-lived TCP flows will considerably aggravate the situation and drastically affect the performance of long-lived TCP flows.

The work of Kuzmanovic and Knightly [7], who investigated a class of low-rate UDP denial of service attacks (DoS) which are difficult for routers and counter-DOS mechanisms to detect, is closest to the work described in this paper. They have developed low-rate DoS traffic patterns using short-duration bursts of UDP flows. Through a combination of analytical modeling, simulation and Internet experiments, they have shown that such periodic low-rate attacks are highly successful against both short-lived and long-lived TCP flows.

Our work breaks new ground in that it presents a novel approach to synthesize adversarial scenarios where short-lived TCP flows maliciously throttle co-existing long-lived TCP flows under competition for bandwidth. Since most mechanisms to detect malicious traffic focus on non-TCP flows, our adversarial scenarios will remain largely undetected. Even when one considers non-malicious situations, our scenarios help better estimate worst-case and average-case TCP performance.

Table I shows the percentage reduction in throughput of long-lived flows when attacked by UDP flows [7]. The table also shows that an adversarial scenario using a carefully selected sequence of short-lived flows achieves nearly equal reduction in throughput. In the remainder of the paper we describe the methodology that we followed to design such scenarios.

III. BACKGROUND ON TCP

The Transmission Control Protocol (TCP) is developed as a highly reliable, end-to-end, window-based protocol between hosts in computer networks. Modern implementations of TCP contain four intertwined algorithms: slow start, congestion avoidance, fast retransmit, and fast recovery. When a new TCP connection is established, TCP enters slow start where congestion window ($cwnd$) evolves exponentially. On each acknowledgement for new data, $cwnd$ is increased by one segment. At some point the capacity of the network is reached and packet losses are experienced at congested routers. There are two indications of packet loss: (a) the expiration of the timeout timer, and (b) the receipt of duplicate ACKs. If three or more duplicate ACKs are received, it is considered an indication that a segment has been lost. TCP then performs a retransmission without waiting for a retransmission timer to expire. Subsequently, congestion avoidance is performed instead of slow start. This is the fast recovery algorithm that

Table I: Comparing UDP attacks with adversarial scenarios using short-lived TCP flows

Type of malicious flows	Long-lived TCP flows throughput degradation
UDP constant bit rate flows	Up to 100%
UDP short bursts with $P=1$ sec	> 90% [7]
Random mix of TCP short-lived and long-lived flows	Up to 10%
Specific pattern of short-lived TCP flows with $P=1$ sec	> 85%

allows high throughput under moderate congestion, especially for large windows. In the congestion avoidance phase, the congestion window evolves linearly rather than exponentially.

However, if packet losses are detected by the timeout mechanism, TCP sets the slow start threshold to half of the current congestion window, reduces the congestion window to one, retransmits the missing segment, performs slow start to the new threshold and then enters congestion avoidance. During congestion avoidance regime, congestion window is either increased by one per round trip time or one per window (Additive Increase); and if a packet loss is detected by receiving three or more duplicate ACKs, the congestion window (cwnd) is reduced to half its current size (Multiplicative Decrease). (Hence the name AIMD.)

Thus, TCP congestion control mechanisms run on two timescales, one short and one long: Round Trip Time and Retransmission Time Out, respectively. Allman and Paxson have experimentally shown that TCP nearly obtains maximum throughput if there exists a lower bound of one second for RTO [11]. Moreover, they found that in order to achieve the best performance and ensure that the congestion is cleared, all flows are required to have a minimum timeout of one second.

IV. BANDWIDTH SHARING

In the context of bandwidth sharing, long-lived TCP flows are shown to hurt short-lived TCP flows in terms of end-to-end delay and consequently throughput. Various scheduling schemes are proposed in favor of short-lived TCP flows in order to speed up the transfer of data and avoid long queuing delays.

In a random mixture of short-lived and long-lived TCP flows, such as LAN or WAN traffic, short-lived TCP flows are large in number but only use a small portion of link capacity. Therefore performance of long-lived TCP flows is not seriously affected by sharing the link capacity with co-existing short-lived TCP flows. In fact, long-lived TCP flows have long been known to hurt short-lived TCP flows in terms of throughput and end-to-end delay, since the long-lived flows occupy most of the buffer space and hence create large queuing delays for the short-lived flows which only have a few packets to send.

As can be seen, constant bit rate UDP flows can cause the most harm to long-lived flows since they can send at a higher rate and never back off. However, they are more prone to detection. The authors in [7] suggest a way to modify the UDP flows into short bursts to create effective DoS attacks that are less prone to detection.

In this paper, we study the interaction between long-lived TCP flows and short-lived TCP flows and generate scenarios where short-lived TCP flows significantly impact long-lived TCP flows when used in an adversarial manner. Our scenarios will not be detected by existing detection mechanisms (which focus on UDP attacks). Under non-adversarial circumstances, our scenarios help to identify worst-case performance of long-lived TCP flows in the presence of short-lived TCP flows.

When short-lived TCP flows are used in adversarial scenarios against long-lived TCP flows, both groups of flows have dynamics. However short-lived flows must utilize as much bandwidth as possible, even though they share many characteristics with long-lived flows (such as self clocking, backing off and going to time out). Therefore several parameters must be considered, including the influence period, the duration of each short-lived flow, the total number of short-lived flows that participate in an adversarial scenario, as well as the number of coexisting short-lived flows and locations of the targeted links.

In a deterministic adversarial scenario, intuitively one might prefer to target the bottleneck link in order to maximally disrupt the throughput of long-lived flows. However, as we will show, since short-lived flows also have dynamics and adapt to network conditions, the target location must not be limited to bottleneck links.

Recall that TCP congestion window grows at different rates in different phases of TCP congestion control. Specifically, during slow start a TCP connection opens its congestion window more aggressively vs. during congestion avoidance when AIMD is performed. Short-lived TCP flows spend most of their lifetime in the slow start phase when the congestion windows are increased exponentially. Long-lived TCP flows also start from slow start phase, however they stay in the congestion avoidance phase for most of their lifetime during which they perform AIMD.

In this paper, we exploit interactions between short-lived and long-lived flows and generate scenarios where carefully chosen malicious short-lived TCP flows attempt to deny network capacity to long-lived TCP flows by taking advantage of TCP timeout mechanism.

V. PROPOSED APPROACH

Consider an illustrative adversarial scenario with a single long-lived TCP flow that passes through a bottleneck link along its path (Fig. 1). Initially, the long-lived TCP flow is in the congestion avoidance phase and performs AIMD. We assume that the maximum congestion window is limited by the bottleneck capacity. Now assume that a malicious user creates severe congestion on a link along the path of the long-lived flow by sending multiple short-lived TCP flows that can be visualized as a series of spikes (e.g., see Fig. 2). During each burst, when the total traffic generated by the short-lived flows and the single long-lived flow exceeds the capacity of that link, packet losses induced are sufficient to force the long-lived flow to back off for an RTO of one second [11]. Suppose the RTO timer expires at time= t_1 . At this time the congestion window is set to one, the value of RTO is doubled and packet transmission is resumed by retransmission of the packet lost at time= t_1 . Now if the same pattern of short-lived flows repeats between time= t_1 and time= t_1+2RTT such that the retransmitted packet is also lost, the sender of the long-lived TCP flow now has to wait for $2RTO$ seconds until the retransmission timer expires. As a result the long-lived TCP flow repeatedly enters the retransmission time out phase, which is doubled every time retransmission of a lost packet fails. Consequently, the long-lived flow obtains nearly zero throughput. As the result of this

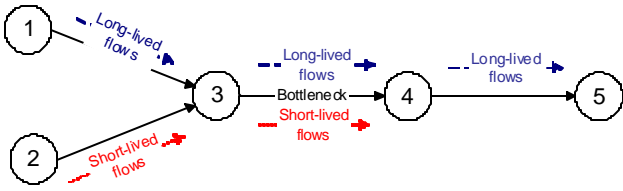


Fig. 1: Example of an adversarial scenario

congestion scenario, the throughput of the long-lived TCP flow is reduced by almost 100%.

However, the short-lived TCP flows (unlike UDP attacks) are also affected by the congestion since their window evolution is subject to the TCP slow-start rules. Hence the efficacy of short-lived TCP flows in conducting the above scenario is unclear. Let us define the following adversarial scenarios. Fig. 2 depicts the periodically injected short-lived flows. Each spike is a short-lived TCP flow in slow start and its congestion window grows exponentially from 1 to W_{ij} before it either hits congestion and enters timeout or is terminated. In general, the maximum achieved window size W_{ij} may not be a power of two. Let M_{ij} denote the last value of the congestion window that is a power of two and r_{ij} stand for the amount of data (in bytes) that is transferred in a partial window afterwards. Also let C denote the capacity of the targeted link, T_L the aggregate throughput of the long-lived flows, d_{ij} the duration of the spike - which is the time it takes for a short-lived flow to time out or be terminated, R the average rate of the spikes in a period of P sec and N_{ij} the total number of packets sent in a spike. In general, there can be n groups of m spikes in one period with a time gap g_{ij} between successive spikes and another time gap G at the end of each periodic interval before the next interval starts. In order to force the long-lived flows to time out, the overall throughput of all flows (short-lived and long-lived) should exceed the targeted link capacity such that many packets are lost from the corresponding window of data. Therefore the average rate of short-lived flows in a period should satisfy the following condition:

$$R + T_L > C \quad (1)$$

Since short-lived flows are in slow start and the congestion window evolves exponentially, it takes $(\log_2 M_{ij}) \times RTT_{ij}$ to transmit the full window and t_{ij} to send the partial window. Equation (2) gives d_{ij} in terms of M_{ij} , RTT_{ij} and t_{ij} .

$$d_{ij} = (\log_2 M_{ij}) \times RTT_{ij} + t_{ij} \quad (2)$$

Also the total number of data packets sent in each spike (N_{ij}) can easily be found from M_{ij} and r_{ij} as shown in (3).

$$N_{ij} = 2 \times M_{ij} - 1 + r_{ij} \quad (3)$$

Equation (4) gives the period of the influence interval in terms of the durations of spikes and the time gaps.

$$P = \sum_{i=1}^n (d_{ij} + g_{ij}) + G \quad (4)$$

Thus the average rate of the short-lived flows in a period ' P ' is the sum of the throughput of each row in Fig. 2. The

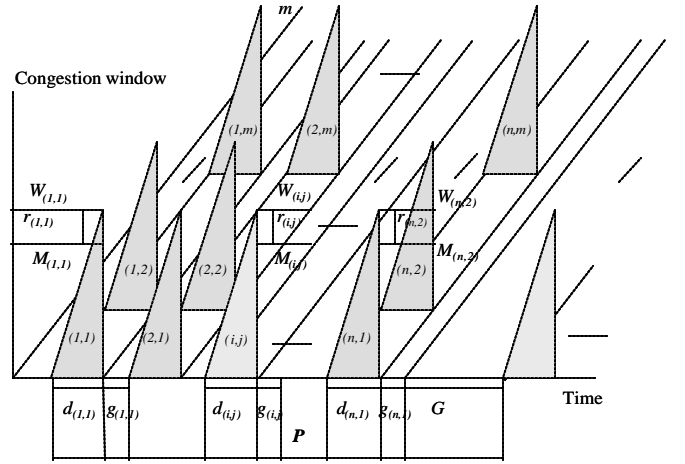


Fig. 2: General pattern of short-lived flows

throughput of each row is the amount of data transmitted in a period by the spikes in that row divided by the period. Equation (5) shows the average rate of the short-lived flows in a period.

$$R = \sum_{j=1}^m \frac{\sum_{i=1}^n N_{ij}}{P} = \sum_{j=1}^m \frac{\sum_{i=1}^n (2 \times M_{ij} - 1 + r_{ij})}{\sum_{i=1}^n (d_{ij} + g_{ij}) + G} \quad (5)$$

In order to satisfy the condition in (1), R should be maximized. Therefore the time gaps in Fig. 2 should be removed, i.e. $g_{ij} = G = 0$. Using these values, we get:

$$R = \sum_{j=1}^m \frac{\sum_{i=1}^n 2 \times M_{ij} - 1 + r_{ij}}{\sum_{i=1}^n d_{ij}} \quad (6)$$

Also it seems that increasing n would augment the summation in the numerator of Equation (6) and consequently boost R . However it should be noted that the value of M_{ij} inversely depends on n . In other words, increasing n means placing more non-overlapping groups of short-lived flows in a period, which results in smaller spikes (both in height and width) and consequently smaller R . However if there is only one group of short-lived flows in a period, they will have the entire period to open and grow their congestion window to the value allowed by the spare capacity of the targeted link. Since the flows in this group are fully overlapped, R is the sum of their throughputs. Hence increasing m will increase R until condition in (1) is satisfied at $m = m^*$ in (7). Conversely further increasing m will result in smaller R , since the short-lived flows will start competing with each other. Hence we put $n = 1$ and bound m to m^* in (6). By substituting in (1) we get:

$$R = \sum_{j=1}^{m^*} \frac{2 \times M_{1j} - 1 + r_{1j}}{d_{1j}} > C - T_L \quad (7)$$

Now recall from the illustrative adversarial scenario that the effective time interval is of the order of RTT of the long-lived

flows. Therefore condition in (1) on the ‘average’ rate seems rather conservative. Fig. 3 shows a single fully overlapped group of short-lived flows in a period. As can be seen from this Figure, most of the bandwidth of short-lived flows is concentrated around the peak of the spikes. In other words, it suffices to have R_{eff} exceed the spare capacity of the targeted link, provided that the buffers are already full. However, it takes $d \cdot T_{eff}$ to fill the buffers along the targeted path. Suppose the initial queue size in the buffer is Q_0 and the maximum buffer size is Q . Then the time it takes to completely fill the buffers is:

$$d - T_{eff} = \frac{Q - Q_0}{R' + T_L - C}, \quad (8)$$

where R' is the throughput of the short-flows during this time interval. Equation (9) shows R' in terms of d , r' and M' for a single group of m short-lived flows, similar to (7) for R .

$$R' = \sum_{j=1}^{m=m} \frac{2 \times M'_{1j} - 1 + r'_{1j}}{d_{1j} - T_{eff_{1j}}}. \quad (9)$$

The modified condition is therefore:

$$R_{eff} + T_L > C, \quad (10)$$

where R_{eff} is the throughput of the short-lived flows during T_{eff} . However by this time the buffers of the targeted link(s) are nearly full, therefore the short-lived flows can send at most one more round of packets, i.e., at most one full window before they start losing packets. Since the short-lived flows are still in slow start phase, the next full window will be at most twice as large as the last full congestion window. Therefore R_{eff} for a single group of m short-lived flows is:

$$R_{eff} = \sum_{j=1}^{m=m} \frac{W'_{1j}}{T_{eff_{1j}}}. \quad (11)$$

Naturally it follows that the effective interval is T_{eff} . As mentioned before in the case of single long-lived flow, T_{eff} is in the order of the RTT of the long-lived flow. In a heterogeneous environment with multiple long-lived flows with different round trip times, an analogous argument suggests that T_{eff} should be greater than or equal to all the round trip times of the long-lived flows. Hence most of the long-lived flows (ideally all of them) are forced to timeout simultaneously for at least RTO seconds. In this case RTO is the minimum retransmission time out among the heterogeneous long-lived flows. Obviously during the timeout phase, the condition in (8) does not hold. Furthermore, the interval between successive T_{eff} 's, which is ideally of the order of RTO , gives the short-lived flows a chance to gain more energy, increase their overall rate and prepare to influence long-lived flows during T_{eff} .

We take the idea of creating such sustained patterns of severe congestion and explore it in several dimensions. We investigate the scalability of such scenarios in terms of the number of long-lived flows. Furthermore, we study the effects of the temporal distribution of malicious short-lived flow. Additionally, we investigate the spatial distributions of various adversarial scenarios on multiple links in an attempt to locate the most vulnerable targets. Ultimately, we suggest the most effective settings for the pattern of short-lived flows during the

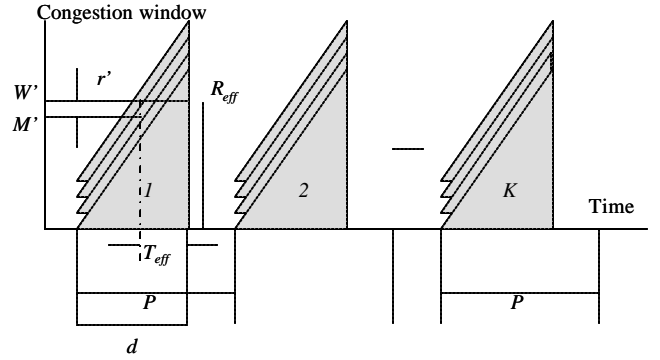


Fig. 3: Effective pattern of short-lived flows

Table II: Effect of adversarial scenarios using short-lived flows on throughput of long-lived flows

Period of Influence Interval, P	Throughput degradation		
	$d=0.5$	$d=0.75$	$d=1$
0.5 sec	> 75%	> 80%	> 80%
1 sec	> 80%	> 85%	> 85%
1.5 sec	> 75%	> 80%	> 85%
2 sec	> 65%	> 65%	> 65%
2.5 sec	> 45%	> 55%	> 55%

influence interval that maximizes its effects on the performance of the long-lived TCP flows. Table II summarizes the results obtained for various settings. Our simulation results indicate that in a random mix of traffic where the target location is also randomly selected among all the links shared by short-lived and long-lived flows, the throughput degradation for long-lived flows is less than 10%. The details will be explained in the next section. Since the steady state performance of long-lived TCP flows (such as FTP transfers that are used in the simulation of congestion scenarios) is characterized by their throughput, we consider the percentage reduction in overall throughput of long-lived flows as the evaluation metric.

VI. SIMULATIONS

In this section we explore the impact of adversarial scenarios using short-lived TCP flows on the performance of long-lived TCP flows.

We designed a series of detailed adversarial scenarios to answer the following questions. As the number of long-lived TCP flows increases, how should the pattern of the adversarial scenario formed by short-lived TCP spikes change in order to maintain the same near-zero throughput for the long-lived TCP flows? What links should be the targets of short-lived TCP adversarial scenarios to achieve the most aggregate throughput reduction for the long-lived TCP flows? In an arbitrary topology some long-lived TCP flows may share one or more bottlenecks but all may not share the same bottlenecks. In a large-scale scenario, how are the period and duration of short-lived TCP spikes determined? The first group of our simulations is designed for a chain topology to study the effects of aggregation of homogeneous (in terms of RTT) long-lived

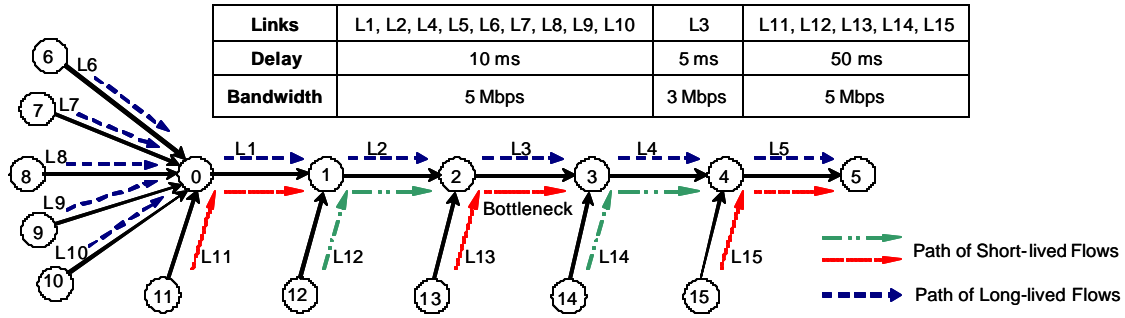


Fig. 4: Single bottleneck topology

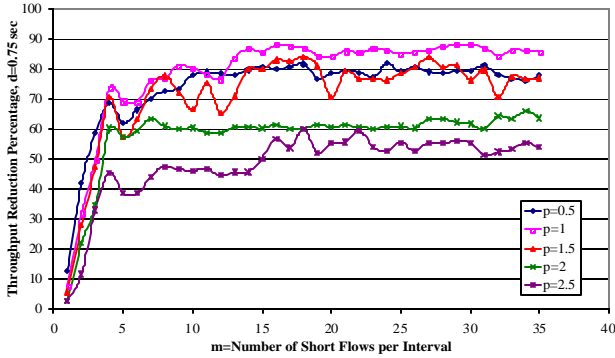


Fig. 5: Effect of influence period on %reduction for $d=0.75$ sec

TCP flows in the event of an adversarial scenario created by short-lived TCP flows. The second group of our scenario simulations is designed for an arbitrary topology (produced using random topology generator) to investigate the effects of aggregation of heterogeneous (in terms of RTT) long-lived TCP flows with multiple bottleneck links. In general, we refer to the link(s) with minimum unused capacity as the bottleneck(s).

A. Single bottleneck topology

Here we describe the simulated adversarial scenarios on the single bottleneck topology (depicted in Fig. 4) simulated using NS-2 [12]. In this topology, five groups of long-lived TCP flows share a chain of links and each of these links is also shared with a group of short-lived flows. Link L3 is the bottleneck link with a bandwidth of 3 Mbps and one-way propagation delay of 5msec. Initially, the long-lived TCP flows are in the congestion avoidance phase. We assume that maximum congestion windows of all TCP connections are limited by network capacity. Short-lived TCP flows are periodically injected on links L1 to L5 for five successive intervals (periods) according to the pattern depicted in Fig. 2. There are 5 sets of concurrent short-lived TCP flows in each time slot. All short-lived TCP flows that are in a set have the same source and destination. Each set is identified by one of the following pairs of source and destination nodes: (11, 1), (12, 2), (13, 3), (14, 4), (15, 5). For instance, Set 3 is the group of short-lived flows that start at node 13 and end at node 3. We measure the aggregate throughput reduction percentage of the long flows and plot it vs. m , the number of concurrent short-

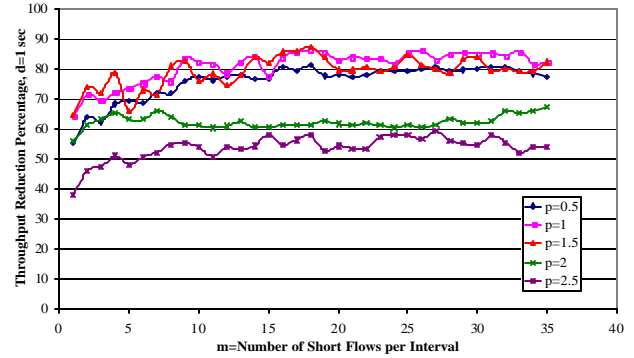


Fig. 6: Effect of influence period on %reduction for $d=1$ sec

lived flows in a set sent in a time slot. We also measure the overall throughput of short-lived flows and plot it against m . We try to identify the effective values for parameters involved in the adversarial scenario pattern, i.e., d , p , m and n . Preliminary observations indicate that temporal distribution of such scenarios on these links is less effective as compared to simultaneous scenarios on the targeted links. Therefore throughout the rest of adversarial scenarios, malicious short-lived flows are spatially distributed on multiple links but temporally concurrent. In this case d refers only to the slow start duration.

1) Configuration of short-lived flows: duration, period and aggregation

Here, we present a baseline set of simulations to identify the best settings for parameters of the short-lived flows. We verify the findings from our analytical model for short-lived flows through extensive simulations with different parameter settings for duration, period and aggregation of short-lived flows.

In Fig. 5 and Fig. 6, all the short-lived flows are terminated after $d=0.75$ sec and $d=1$ sec, respectively. The influence period is changed from 0.5 to 2.5 sec and the throughput reduction percentage of long-lived flows are plotted vs. m , for $n=1$ (Fig. 5 and Fig. 6). It is worth mentioning that when $d > P$, there is no gap between successive groups of short-lived flows, whereas $d < P$ means that there is some gap between successive groups of short-lived flows. In this set of simulations, the TCP version for both long and short flows is Reno and the packet drop policy at all buffers is DropTail. As can be seen in these

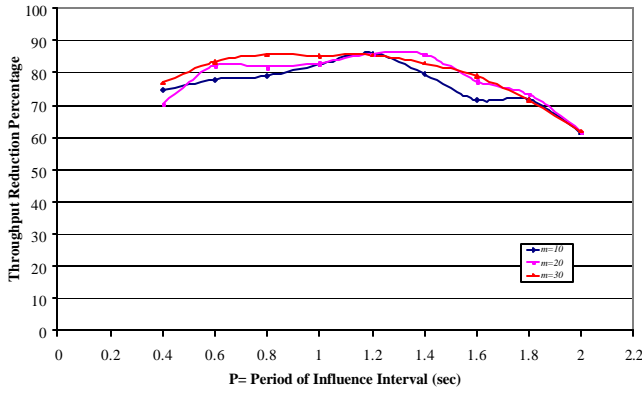


Fig. 7: Frequency response of the long-lived flows for $d=1$

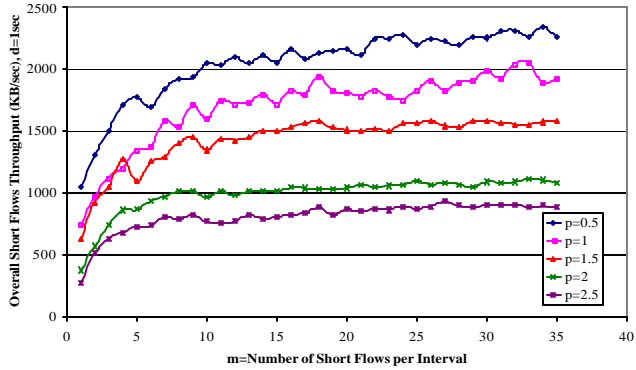


Fig. 8: Overall throughput of short-lived flows

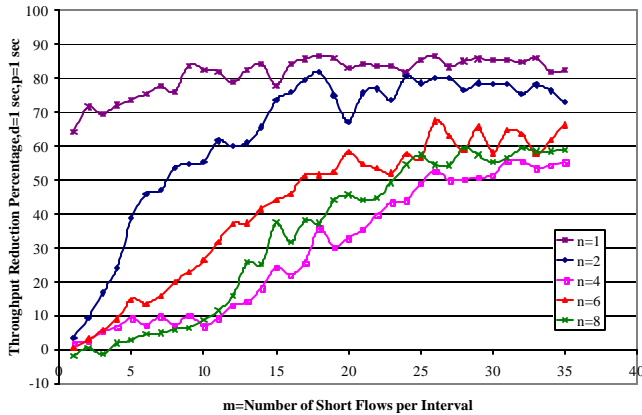


Fig. 9: Effect of n on throughput reduction percentage

Figures, regardless of the value of d the highest percentage reduction corresponds to the influence period of one second. Interestingly, this is the default value for TCP Retransmission Time Out value (RTO). In other words, when the influence period matches the RTO of the TCP long flows, the throughput degradation is maximized. As the influence period increases, the gap between expiration of the retransmission timers of the long flows and the subsequent influence interval increases and consequently long flows obtain higher throughputs.

In Fig. 5 and Fig. 6, for $m < 15$ most of the short-lived flows are already in the time out phase, when they are terminated at $d=1$ sec. For $m < 15$ the most effective scenario corresponds to

$d=1$ sec and $p=1$ sec. In this case, even a low-rate stream of malicious flows can significantly reduce the throughput of the long-lived flows. For higher values of m , the percentage reduction is almost equal for $d=0.75$ sec and $d=1$ sec. The time it takes for the short-lived flows in these scenarios to enter retransmission time out varies between about 0.75 sec and 1 sec. For $m > 15$ and $d=1$ sec, however, most of the short-lived flows are still in slow start, competing with long-lived flows for 0.25 sec longer than when $d=0.75$ sec, and obviously at a higher rate. As a result, more and more long-lived flows are forced to time out for $m < 15$. As m increases more short-lived flows enter time out thus the throughput reduction percentage for long-lived flows saturates and increasing d from 0.75 sec to 1 sec does not make much difference.

As indicated by the shape of the curves, increasing the rate of malicious flows does not monotonically increase the percentage reduction in the throughput long-lived flows. Once the overall throughput of the long-lived and malicious short-lived flows reaches the capacity of the corresponding shared links, increasing the rate of the malicious short-lived flows does not further reduce the throughput of the long-lived flows.

Fig. 7 depicts the frequency response of the long-lived TCP flows in adversarial scenarios using short-lived TCP flows in terms of percentage throughput reduction vs. period of the influence interval for several m (the number of concurrent short-lived flows per interval). The most degradation in throughput of the long-lived flows occurs when period of the influence interval is close to TCP null frequency, which is identified to be the minimum retransmission timeout, i.e., 1 sec. However, it is worth mentioning that TCP null frequency is practically more than 1 sec, since all flows will not enter time out exactly at the same time. In [7], Kuzmanovic and Knightly have developed a simple model to capture the frequency response behavior of a long-lived TCP flow under UDP short-burst attacks. Equation (12) gives their model for normalized throughput T_L of a single long-lived TCP flow in terms of the attack period P and the minimum retransmission timeout:

$$T_L = \frac{\left\lfloor \frac{\min RTO}{P} \right\rfloor P - \min RTO}{P} \quad (12)$$

Although long-lived TCP flows in our scenarios suffer from the same null frequency, their frequency response at other frequencies does not follow the trend in [7].

Fig. 8 shows the overall throughput of the short-lived flows vs. m , for different influence periods.

Apparently, for a fixed number of short-lived flows, as period of the influence interval increases, time gap between successive influence intervals becomes larger. Consequently the average influence rate decreases.

Fig. 9 shows the effect of n on the performance of long-lived flows. During each influence period, n consecutive groups of m simultaneous short-lived flows are injected on the targeted links. Now, in order to fit more such groups in one period, i.e., to increase n , d must decrease, i.e., short-lived flows must be terminated sooner. Consequently the overall rate of the short-flows will decrease and fewer long-lived flows will

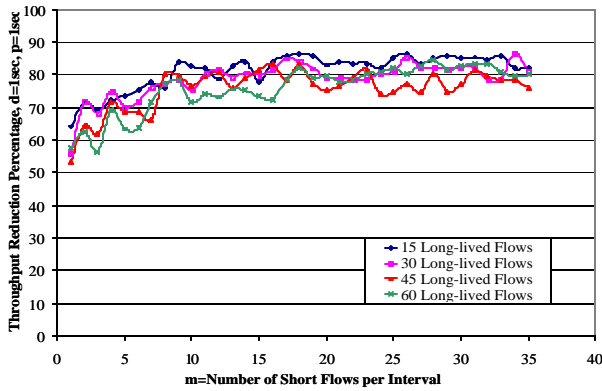


Fig. 10: Effect of aggregation of long-lived flows

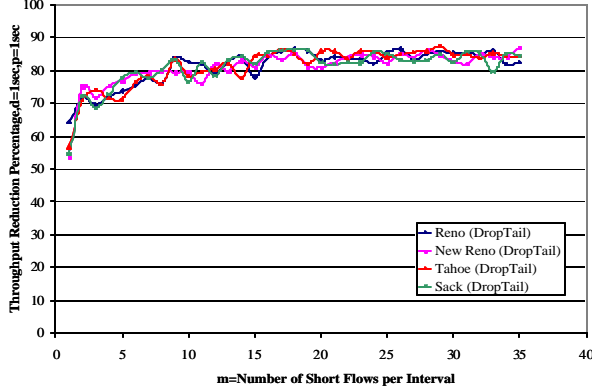


Fig. 11: Comparing various TCP flavors, DropTail

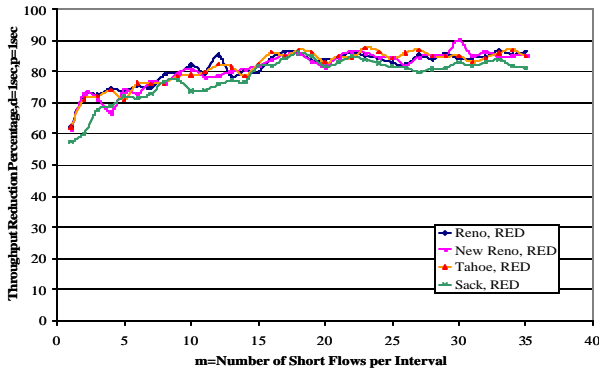


Fig. 12: Comparing various TCP flavors, RED

time out. As can be seen, at $n=1$, percentage throughput reduction for the long-lived flows is maximized.

According to the observations, it seems that $d=1$ sec, $p=1$ sec and $n=1$ are thus far the most effective settings. The rest of simulation results are obtained with these settings.

2) Aggregation of long-lived flows

In this set of simulations we study the scalability of our adversarial scenarios in terms of long-lived flows and evaluate the effectiveness of our adversarial scenarios on large aggregations of long-lived flows.

Fig. 10 depicts the effect of aggregation of homogeneous long-lived flows. As the number of long-lived flows increases, the throughput reduction percentage slightly decreases.

However the adversarial scenarios are still highly successful and severely degrade the throughput of long-lived flows.

3) TCP variants and packet drop policies

We further explore the effect of using several TCP variants for different packet drop policies.

Fig. 11 shows the simulation results for different TCP variants with DropTail packet drop policy. The TCP variants are modifications of the original TCP algorithm to help TCP flows survive multiple packet drops from a window or within an RTT and to help maintain a high throughput under moderate congestion. However simulation results indicate that all variants are significantly affected by the adversarial scenarios. This is because when many short-lived flows, all in slow-start, compete with long-lived TCP flows that are in congestion avoidance, so many packets from the window are lost that even these improvements fail to make up for vulnerabilities inherent to TCP. Recently, the authors in [13] proposed randomization of RTO to reduce the effectiveness of the short burst UDP attacks [7]. We have not yet tested our scenarios against this method.

Fig. 12 demonstrates the same effect for the above TCP variants with RED packet drop policy. The main objective of employing RED in routers is to prevent global synchronization of TCP flows. It has been shown in [7] that RED is unable to avoid the synchronization effects if the source of synchronization is an external malicious source such as DoS attacks. Interestingly, we observe that even if the source of synchronization is TCP itself, RED fails to avoid the synchronization of TCP flows, cutting their window in half or entering the time out almost simultaneously. Since long-lived TCP flows have the same minimum retransmission time out value, when period of the influence interval matches this value, it repeatedly induces a synchronization effect by forcing the long-lived flows to enter the time out at about the same time and exit the time out nearly together.

4) Location

Although the impact of target location is not explicitly captured in our analytical model for short-lived flows, it is of great importance to the outcome of the adversarial scenario. Through this set of simulations, we identify the most vulnerable target locations in order to effectively disrupt throughput of the long-lived TCP flows.

Fig. 13 depicts the effect of adversarial scenarios on different links in Fig. 4 for the following settings ($d=1$ sec, $p=1$ sec, $n=1$, and $m=30$). The horizontal axis shows the targeted links between node (a) and node (b). For instance '2to4' means targeting links L3 and L4 that are between node 2 and node 4. As can be seen in the first five bars, the percentage throughput reduction increases as more links are targeted. The 5th bar shows the largest percentage throughput reduction when all 5 links (L1 to L5) are targeted. The next 4 bars correspond to the influence intervals on links between node 1 and node 2, 3, 4 and 5, respectively. Again increasing the targeted path length increases the throughput degradation of the long-lived flows.

The next bar shows effect of targeting the bottleneck link. Surprisingly, the bottleneck link turns out to be more robust in these scenarios.

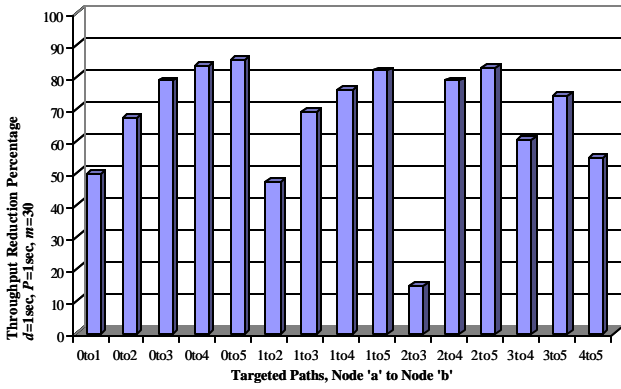


Fig. 13: Comparing effects of targeting different links from node 'a' to node 'b'

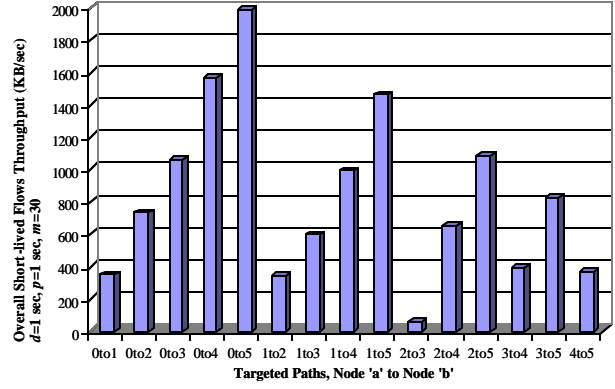


Fig. 14: Overall throughput of short-lived flows targeting different links from node 'a' to node 'b'

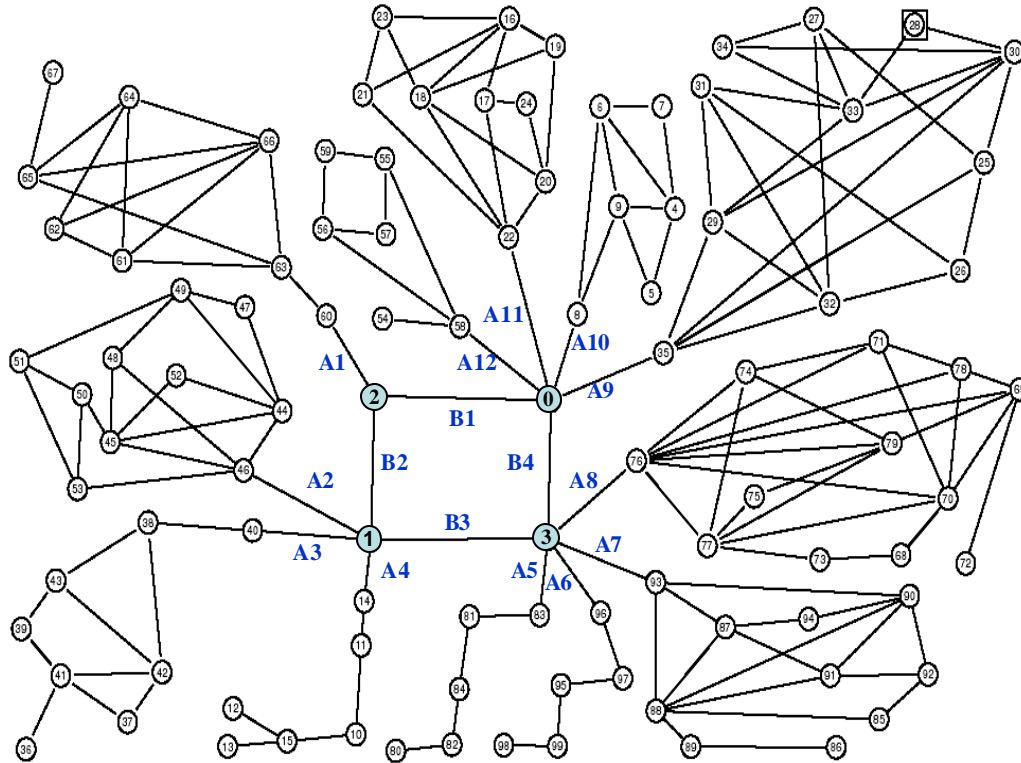


Fig. 15: Multiple bottleneck topology

The key reason for this behavior goes back to the definition of the bottleneck link. The bottleneck link is a link with the least unused capacity. The effectiveness of our scenarios highly depends on the ability of malicious short-lived flows to force the long-lived flows to enter timeout. When there is little capacity left on the targeted link, short-lived flows are not able to open their congestion window to a level where they inject a sufficient number of packets on the link to create many correlated packet losses from long-lived flows. Thus fewer long-lived flows will enter retransmission timeout and the adverse impact on them will not be maximal. Simulations show a mere 25% throughput degradation when only the bottleneck link is targeted. However, targeting the bottleneck link and other links increases the effectiveness of our scenarios to more than 80%.

It is also observed that targeting links closer to the destination is slightly more effective because packets that are dropped on the last links have already traveled the previous links, added to the traffic and contributed to the congestion on those links.

Fig. 14 shows the overall throughput of the short-lived flows when different links are targeted. Obviously targeting more links using the same pattern requires more short-lived flows. As a result the overall rate of short-lived flows increases when more links are targeted. Finally, when the bottleneck link is targeted, the overall rate of the short-lived flows is minimal for the reason mentioned above.

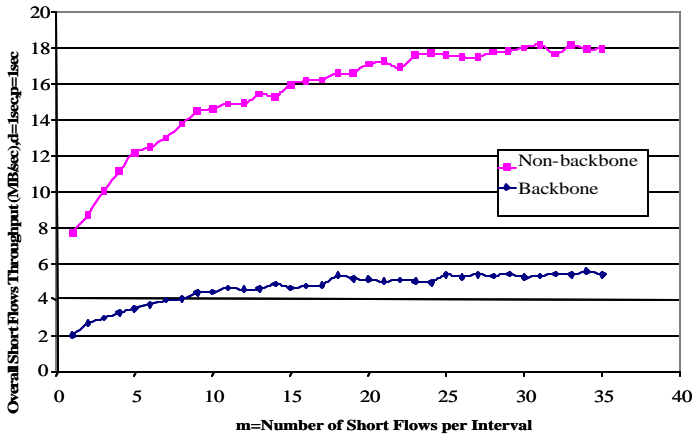


Fig. 16: Comparing throughput of short-lived flows targeting backbone and non-backbone links

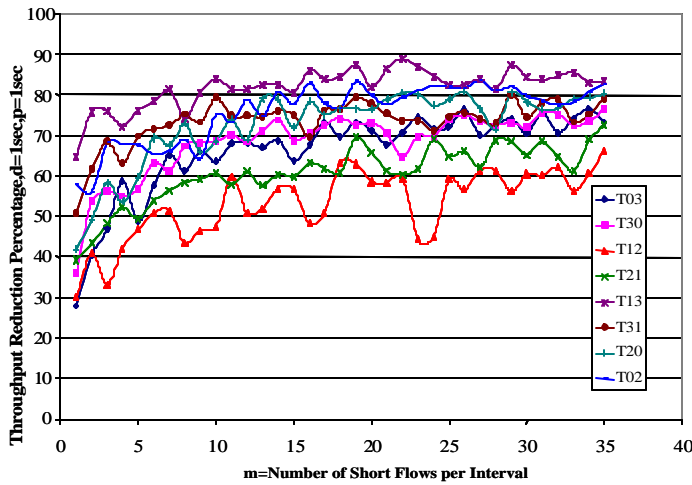


Fig. 17: Effects of targeting backbone links

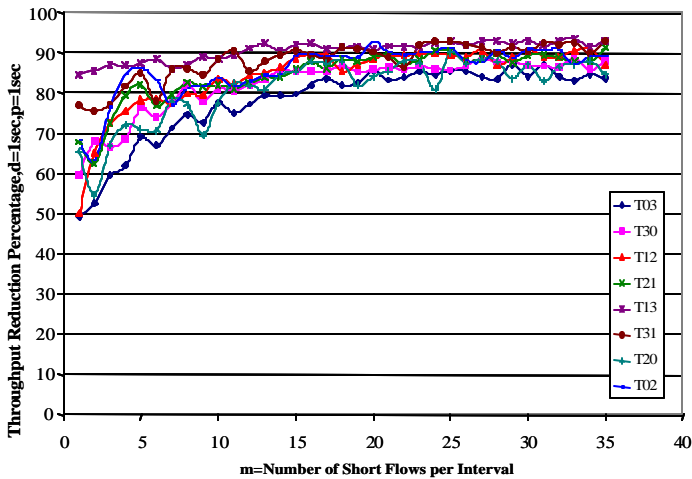


Fig. 18: Effects of targeting non-backbone links

B. Multiple bottleneck topology

In this section we describe the adversarial scenarios on an arbitrary topology (Fig. 15) simulated in NS-2. In this topology, we identify four duplex links B1 to B4 as the bottleneck links. 48 groups of 3 long-lived flows originate and

end in different stubs. These flows are further categorized into eight larger groups according to the bottleneck links they share. The percentage throughput reduction for each of these groups, T_{ij} , is measured and plotted vs. m . ‘ i ’ and ‘ j ’ denote the nodes incident on the bottleneck link that is shared by each group. For instance, T_{03} is percentage reduction in the throughput of all the flows that pass through link B4 on the direction from 0 to 3. All links have a bandwidth of 15Mbps and the propagation delays vary from 1 to 49 msec. Again, we assume that maximum congestion windows of all TCP connections are limited by network capacity, and before the influence interval of short-lived flows starts, the long-lived TCP flows are in the congestion avoidance phase. The simulation results obtained here are consistent with those obtained from the single bottleneck topology simulation.

1) RTT-Heterogenous long-lived flows

We further use this arbitrary topology to investigate the effects of aggregation of heterogeneous long-lived TCP flows (in terms of *RTT*) with multiple bottleneck links. Fig. 17 and Fig. 18 show the simulation results for two adversarial scenarios. In the first scenario short-lived TCP flows are periodically injected on links B1 to B4 in both directions using the pattern shown in Fig. 2. Note that there are two groups (T_{02} and T_{20}) that pass through two bottleneck links. The throughput reduction percentage for each of the other six groups is inversely proportional to the unused capacity at the corresponding bottleneck link. This unused capacity of the link depends on the bandwidth delay product of the link, the buffer size and the number of flows passing through that link. As can be seen from Fig. 17, the more capacity left on a link, the more short-lived flows will be able to utilize this capacity. On the other hand, the throughputs of the two groups that pass through two bottleneck links are more degraded since they are targeted on two links. This is of course consistent with the results obtained from the chain topology. As more links are targeted, we notice an increased reduction in throughput of the long-lived flows.

In the second scenario, 12 duplex links (A1 to A12) that are not the bottleneck links are targeted by the same pattern of short-lived flows. Fig. 18 shows the percentage throughput reduction of the 8 above-mentioned groups vs. m . It is worth mentioning that in this scenario each of the flows in these groups passes through two targeted links. For $m < 15$, the effectiveness of our scenarios is highly dependent on the unused capacity of the targeted links. However as m increases, more and more short-lived flows are injected on these links such that many correlated packet losses are created. Consequently the throughput degradation varies between 80% and 90% in a quasi-stable manner. Comparing the two scenarios, we conclude that the second scenario (non-backbone links) is more effective since the percentage throughput reduction for each group of long-lived flows in this scenario is about 5% to 30% higher. However Fig. 16 shows that the overall rate of the short-lived flows in the second scenario is almost 3 times as high as that in the first scenario, which is consistent with prior observations from the single bottleneck topology. Recall that in the second scenario 12 duplex links are targeted whereas in the first scenario only 4 duplex links are targeted. The results obtained from this set of simulations

indicate that the proposed influence pattern is highly successful in heterogeneous-RTT environments with multiple bottlenecks and a high level of multiplexing.

VII. EXPERIMENTS

In order to verify the findings from simulations, we set up a test-bed with the topology shown in Fig. 19 to run several experiments. Table III shows the required tools and utilities on the test-bed machines for the experiments. Links L1 (the bottleneck) and L2 are set to 32Mbps and 40Mbps respectively. The corresponding buffers are limited to 200 packets. The link propagation delays are less than 1msec. All other links have a bandwidth of 100Mbps. A total of 50 concurrent long-lived flows are created by downloading 50 copies of a 4MB file from web servers W1 and W2 to client C3. After 10 seconds, client C3 starts creating 10 sets of short-lived flows. To do so, every one second, C3 download 200 concurrent copies of a small file from web server W4. In the second experiment, C3 creates the same number of long-lived flows and C2 creates the short-lived flows in the same manner but from web server W3. In the third experiment both C2 and C3 create shot-lived flows by downloading multiple copies of the small file from web servers W3 and W4, respectively, while C3 creates the long-lived flows. In the fourth experiment, no short-lived flows are created. Traces of all downloads are collected using tcpdump. We measure throughput of the long-lived flows when the short-lived downloads are in process (Experiments 1, 2 and 3) and compare it with the case where only long-lived flows exist (Experiment 4). Fig. 20 shows the result of the above three experiments for two packet drop policies: RED and DropTail. As can be seen in this Fig., targeting L1, which is the bottleneck link, as compared to targeting L2 is less effective for both packet drop policies. This observation is completely consistent with the simulation results. However, the amount of reduction in the throughput of the long-lived flows depends on the ability of short-lived flows to force them to timeout. This is also dependant on the spare capacity on the targeted links. Therefore if the targeted link is highly loaded, such as a bottleneck link, the short-lived flows will not be able to open their congestion window much in order to inject enough packets to fill up and overflow the buffers and create many packet losses for the long-lived flows. Also as the experiment results show, targeting more links has a more severe impact on the throughput of the long-lived flows (targeting L1 and L2).

VIII. CONCLUSIONS

In this paper, we presented a systematic study of scenarios where short-lived TCP flows severely impact long-lived TCP flows. The scenarios generated by us take advantage of the facts that (i) short-lived TCP flows spend most of their life in the slow start whereas long-lived TCP flows are in congestion avoidance for most of their life, (ii) a TCP flow in slow start can rapidly capture a greater proportion of resources than another in congestion avoidance, and (iii) via consecutive influence intervals, short-lived TCP flows can drive the long-lived flows into timeout.

We have used a combination of modeling and systematic simulations to determine that large reductions (>85%) in the

Table III: Setup of the test-bed machines

Machine	Function	OS	Tools
W1,W2 W2,W3	Web server	Linux	Httpd,tcpdump,zebra
C1	Client	FreeBSD	ALTO, zebra
C2	Client	FreeBSD	ALTO, zebra
C3	Client	Linux	Httpd,tcpdump,zebra

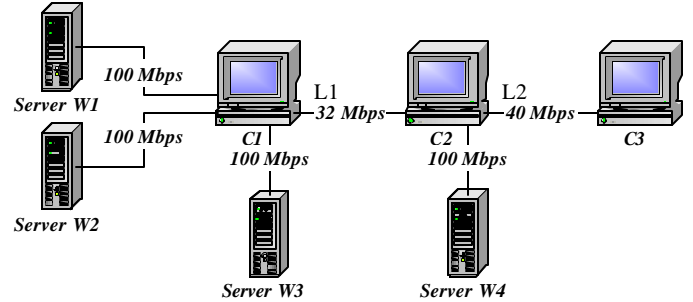


Fig. 19: Topology of the test-bed for the experiments

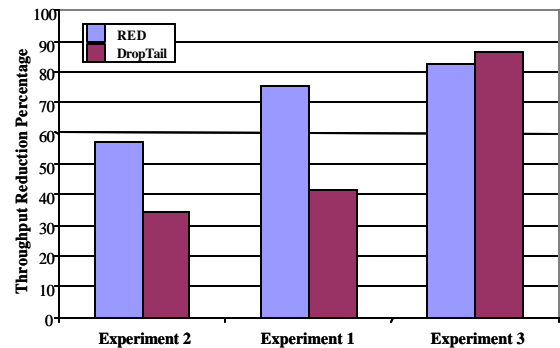


Fig. 20: Results of the experiments 1, 2 and 3

throughput of long-lived TCP flows can be obtained using short-lived TCP flows with the following characteristics:

- Short-lived flows must follow each other back-to-back, until the long-lived flows are driven into timeout. Subsequent sequence of influence intervals should start after a suitable period of time.
- The ideal duration of each adversely-affecting short-lived flow must be 1sec, a parameter related to the RTO of TCP.
- Targeting the bottleneck link alone does not always lead to the largest percentage throughput reduction. This is due to the fact that if the bottleneck link is heavily congested, then the malicious short-lived TCP flows cannot capture sufficient bandwidth.

We conducted experiments and were able to validate scenarios where targeting the bottleneck did not lead to significant percentage throughput reduction. This verified that the above conclusion is not an artifact of the simulation inaccuracies.

Randomly generated scenarios (where the numbers, durations, and locations of, as well as the intervals between, short-lived flows are all selected randomly) cause less than

10% reduction in the throughput of long-lived flows. In contrast, the scenarios generated using our heuristic approach that takes advantage of above results provide greater than 80% reduction in throughput. They do so even when the number of long-lived flows is large. Our scenarios use numbers of short-lived flows such that the ratio of short-lived and long-lived flows is always within normally expected limits.

Interestingly, the scenarios we generate achieve greater than 85% reduction in throughput for a number of TCP variants (Tahoe, Reno, New Reno, and Sack) and for different packet drop policies (DropTail, RED).

Our results demonstrate that even TCP-friendly flows, if carefully orchestrated, can severely degrade throughput of long-lived TCP flows. They also demonstrate that the AQMs and packet schedulers designed to give even higher preference to short-lived flows will make long-lived TCP flows even more vulnerable to such adversarial scenarios.

Presently, we are analyzing large-scale simulations to identify probabilities of chance occurrences of such scenarios in normal large-scale simulations. We propose to develop a mixed simulation strategy for large-scale networks taking into account characteristics of our worst-case scenarios as well as discrete models that capture key details (e.g., queue sizes, protocol events), which can be used to systematically generate worst-case congestion scenarios.

ACKNOWLEDGMENT

We would like to extend our appreciation and acknowledgment for the support from DARPA, NSF and NASA. We also thank Shshank Sharma for helping with the setup of the test-bed and experiments.

REFERENCES

[1] J. Postel, "Transmission control protocol," RFC 793,

September 1981

- [2] J. Padhye, V. Firoiu, D. Towsley, and J. Kurose, "Modeling TCP Throughput: A Simple Model and Its Empirical Validation," ACM SIGCOMM, August 1998.
- [3] L. Guo and I. Matta, "The War between Mice and Elephants," ICNP'2001, November 2001.
- [4] A. Kantawala and J. Turner, "Queue Management for Short-Lived TCP Flows in Backbone Routers," Proc. of High-Speed Symposium, Globecom 2002.
- [5] S. Vutukury and J.J. Garcia-Luna-Aceves, "WiNN: An Efficient Method for Routing Short-Lived Flows," Proc. IEEE ICT 2003, February 2003.
- [6] M. Mellia, I. Stoica, and H. Zhang, "TCP Model for Short Lived Flows," IEEE Communications Letters, February 2002.
- [7] A. Kuzmanovic and E. Knightly, "Low-Rate TCP-Targeted Denial of Service Attacks," ACM SIGCOMM August 2003.
- [8] C. Jin, D. X. Wei, and S. H. Low, "FAST TCP: motivation, architecture, algorithms, performance," IEEE Infocom, March 2004.
- [9] V. Jacobson, "Congestion Avoidance and Control," ACM SIGCOMM 1988.
- [10] K. Fall and S. Floyd, "Simulation-based Comparisons of Tahoe, Reno, and SACK TCP," ACM Computer Communications Review, vol. 26, no. 3, July 1996.
- [11] M. Allman and V. Paxson, "On estimating end-to-end network path properties," ACM SIGCOMM, September 1999
- [12] L. Breslau et al. , "Advances in network simulation," IEEE Computer, Vol. 33, May 2000
- [13] Guang Yang, Mario Gerla and M. Y. Sanadidi, "Randomization: Defense against Low-rate TCP-targeted Denial-of-Service Attacks," ISCC 2004