NF-TCP: A Network Friendly TCP Variant for Background Delay-Insensitive Applications

Mayutan Arumaithurai, Xiaoming Fu, K. K. Ramakrishnan

NF-TCP: A Network Friendly TCP Variant for Background Delay-Insensitive Applications

Mayutan Arumaithurai[1], Xiaoming Fu[1], and K.K. Ramakrishnan[2]

[1] Institute of Computer Science, University of Goettingen, Germany,
arumaithurai,fu@cs.uni-goettingen.de
[2] AT&T Labs-Research, U.S.A.,
kkrama@research.att.com

Abstract. Delay-insensitive applications, such as P2P file sharing, generate substantial amounts of traffic and compete with other applications on an equal footing when using TCP. Further, to optimize throughput, such applications typically open multiple connections. This results in unfair and potentially poor service for applications that have stringent performance objectives (including sensitivity to delay and loss). In this paper, we propose NF-TCP, a TCP variant for P2P and similar delay-insensitive applications that can afford to have communication in the "background". NF-TCP aims to be submissive to delay-sensitive applications under congestion. A major component of NF-TCP is to integrate measurement as an integral component of the congestion control framework. This enables the transport to exploit available bandwidth, so that it can aggressively utilize spare capacity.
We implemented NF-TCP on Linux and ns-2. Our evaluations of the NF-TCP Linux implementation on ns-2 show that NF-TCP outperforms other network friendly approaches (e.g., LEDBAT, TCP-LP and RAPID). NF-TCP achieves high utilization, fair bandwidth allocation among NF-TCP flows and maintains a small average queue. Our evaluations further demonstrate that with NF-TCP, the available bandwidth can be efficiently utilized.

Keywords: Network-Friendly, submissive, bandwidth-estimation

1 Introduction

Peer-to-Peer (P2P) file sharing applications form a significant part of the Internet traffic. According to a study from Ipoque [1], P2P accounted for a range between 43% and 70% of the total traffic in large parts of Europe, Middle East, Africa and South America. To optimize their throughput over a wide range of conditions, such P2P applications use TCP (which provides fairness among all coexisting flows). These applications seek to improve throughput further by opening multiple connections, thereby resulting in an unfair and potentially poor service for applications that have stringent performance objectives to meet user requirements for interactive usage. Based on user expectations and the current technology available in the Internet, applications may be broadly classified into delay-sensitive and delay-insensitive applications. Users have a higher expectation and less tolerance for delays caused to delay-sensitive applications such as video conferencing and streaming, voice over IP, and even web-browsing. On the other hand, users generally perceive applications such as software updates, "download and play" and P2P file sharing to name a few, as having lower priority.

To overcome the perceived unfair and sometimes unnecessary disadvantage that delay-sensitive applications are subjected to by traffic from background delay-insensitive applications, ISPs have resorted to throttling or even blocking of traffic from heavy users during congestion. Future trends such as congestion based charging based on approaches such as [2] also motivate the need for solutions that make background traffic more network friendly. These would aid applications to seamlessly identify congestion and facilitate delay-insensitive applications to defer the use of the network. This has the potential for a positive situation for both users and ISPs resulting in a better distribution of the network load and greater user satisfaction.

Delay-insensitive traffic needs to be both submissive during congestion periods and aggressive during non-congestion periods. Our point of view is that, it is most suitable when done as a transport layer protocol. The primary argument behind this is driven by considerations of the time-scale at which the transport layer (rather than a different layer, such as at the application) can react to the onset or the absence of congestion. In addition, congestion control and avoidance has typically required the transport protocol to place a load on network resources, create congestion, and then react to the effect of this congestion to effectively (i.e., both in terms of efficiency and fairness) use the network. This approach is in conflict with the basic goal of a network-friendly, submissive protocol. Our motivation for developing NF-TCP is based on the realization of the difficulties observed with other recent efforts to arrive at a network-friendly transport protocol. Recently, the IETF LEDBAT working group has been formed to develop such a network friendly protocol [3], which is primarily an end-host, delay-based congestion avoidance protocol. Other delay-based network friendly mechanisms include TCP-LP [4] and RAPID [5]. A limitation of these delay-based approaches is that they have to cause queuing (and potentially significant-enough queueing) to be able to operate effectively. In addition, they require high precision packet time-stamps and accurate delay measurements in the implementation to identify the onset of congestion. Delay measurements are also susceptible to noise in low latency networks and also in the presence of dynamic background traffic [6]. Additional difficulties include robustness to randomness and widely varying RTTs for competing traffic [7]. Our evaluations also highlight the fact that LEDBAT is unfriendly to standard TCP and contributes to queue buildup in high bandwidth-delay product (BDP) networks.

Efficiently utilizing available bandwidth is another consideration driving existing solutions for high BDP environments, such as High-speed TCP [8], FAST [9], Compound TCP [10], CUBIC [11], Quick-Start [12] and XCP [13]. However, part of their ability to opportunistically utilize the large bandwidths in these types of environments is due to their aggressive increase policies. Thus, they are not necessarily "submissive" or "friendly" to other delay-sensitive applications, and have the potential to cause a period of congestion. In fact, evidence of their "collateral damage" on existing TCP traffic has been documented (e.g., [14]).

In this paper, we propose a network-friendly congestion control protocol (NF-TCP), which is a TCP variant for P2P and similar background delay-insensitive applications. NF-TCP is able to be network friendly by being submissive to delay-sensitive applications under congestion. A key idea within NF-TCP is to integrate measurement as an integral component of the congestion control framework. It uses the measured value of the

available bandwidth to aggressively utilize that spare capacity during non-congestion periods, thus minimizing the reduction in throughput that it would otherwise suffer because of its submissiveness. It differs from existing approaches in two main ways:

• NF-TCP is a network-friendly, submissive variant of TCP by depending on Explicit Congestion Notification (ECN) [15] based **network feedback**. We propose a simple but efficient enhancement to the standard Active Queue Management (AQM) routers by configuring it to use a lower threshold to begin marking or dropping of packets belonging to NF-TCP flows. NF-TCP exploits this early marking scheme to reliably identify incipient congestion and have delay-insensitive applications aggressively defer their load on a congested network.
• NF-TCP is designed to exploit information provided by an available **bandwidth measurement component** that is carefully crafted to rapidly obtain the estimate. Based on this estimate, NF-TCP utilizes spare capacity when the network is uncongested in an aggressive but informed manner, thereby allowing NF-TCP to compensate for the loss of throughput during the submissive phase. To the best of our knowledge, this is the first work that incorporates a **separate** bandwidth measurement mechanism into a congestion control framework. Existing approaches estimate available bandwidth either by placing load or by switching to a rate based sending of data packets. We propose a novel ECN-enhanced bandwidth estimation mechanism (ProECN) to guide NF-TCP to aggressively utilize the bandwidth during non-congestion periods.

Our LANMAN 2010 [16] work described the philosophy and overall approach of the NF-TCP. This paper provides the details of the protocol (with significant improvements and enhancements) and analyzes it extensively. Our results are primarily based on measurements of our NF-TCP Linux-based implementation that was ported to ns-2 using the Linux TCP implementation tool [17] to enhance our understanding of the NF-TCP behavior in large scale as well as to compare our approach to several other alternatives.

2 Design Objectives

The main goal of our work is to develop a network friendly protocol that addresses the challenges described above. Our solution is built based on the following requirements:

Requirement-I: Be submissive to standard TCP when encountering network congestion This is to ensure that packets of delay-insensitive applications do not result in substantial queueing that can impact existing or newly arriving TCP flows. This buffer occupancy results in an increase in latency or drop rate for delay-sensitive applications. The submissiveness also enables standard TCP flows to utilize the available capacity. For NF-TCP to be submissive, it is essential for it to detect the onset of congestion earlier than standard TCP.

During congestion periods when the queue is building up, we get the following condition for NF-TCP using the rate deterministic model ($T_N \propto \frac{a}{\sqrt{p}}$) described in [18]:

$$\frac{a_N}{\sqrt{p_N}} \rightarrow 0 \tag{1}$$

where T_N is the throughput of an NF-TCP flow, p_N is the loss rate of the NF-TCP flow and a_N is the rate of increase of the NF-TCP flow. Therefore to meet the conditions of the equation, either NF-TCP's increase factor (a_N) should be close to zero or the loss rate of NF-TCP (p_N) should be very high.

Requirement-II: Ability to saturate available bandwidth as fast as possible in the absence of other TCP flows An NF-TCP flow must be capable of aggressively capturing available bandwidth during non-congestion periods without having a negative impact on co-existing TCP flows. Moreover, in the presence of other NF-TCP flows, the bandwidth should be equally shared among all flows.

3 NF-TCP Design

In this section we present the design of NF-TCP. We begin by describing how NF-TCP is able to be submissive to standard TCP by detecting congestion early and reliably. Next, we explain how NF-TCP uses a separate bandwidth estimation mechanism to utilize spare bandwidth aggressively during non-congestion periods.

3.1 Be submissive to standard TCP when encountering network congestion

NF-TCP's network friendly congestion control is achieved by taking advantage of a congestion detection mechanism that detects incipient congestion earlier than standard TCP. NF-TCP exploits the availability of Active Queue Management (AQM) routers that are configured to use a lower threshold to begin marking or dropping of packets belonging to NF-TCP flows. NF-TCP is designed to use the standard ECN bits and can, for example, use the low-priority DSCP code point [19] to identify an NF-TCP flow. The aim is to ensure that packets of network friendly applications do not contribute to queue build-up, which results in higher latencies for delay-sensitive traffic. The standard AQM mechanism is slightly modified to provide feedback for NF-TCP based on ECN on time-scales of an RTT. ECN-unaware routers can drop NF-TCP packets earlier to indicate the onset of congestion. Next we illustrate how a modified RED queue [20] is used for this purpose.

Table 1. The used RED queue parameters for evaluation

	NF-TCP	Standard TCP
Min-threshold (pkts)	75 packets (= 1% of queue size)	6800
Max-threshold (pkts)	3750 packets (= 50% of total queue size)	7500

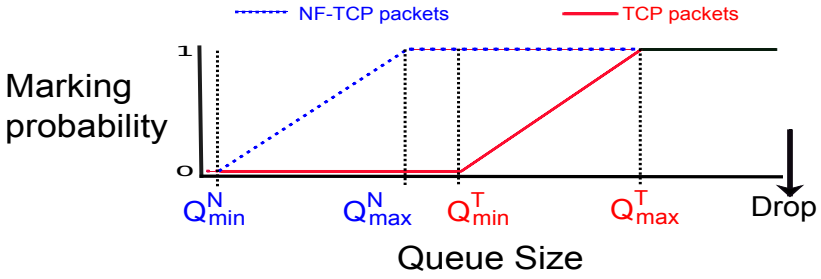


Fig. 1. A router queue performing network friendly marking

Modified RED queue for NF-TCP The modified RED queue consists of different parameters for NF-TCP compared to standard TCP. To detect the onset of congestion as early as possible while ensuring that we do not respond to truly short-term transients, we set the marking threshold values in the RED queue much lower than that for standard TCP. Fig. 1 illustrates the setting of the queue threshold values. The early marking and ECN for feedback to the source enables early reaction to the onset of congestion. The queue thresholds are based on the same mechanism as for a RED queue, except that the MinThreshold for NF-TCP flows (Q^N_{min}) is set much lower. The MaxThreshold for NF-TCP flows (Q^N_{max}) is set to about half of the buffer size. This ensures that once the "average" queue (based on an exponentially weighted moving average (EWMA)) begins to build up, NF-TCP packets are probabilistically marked (p). All NF-TCP packets are marked or dropped when the average queue size exceeds MaxThreshold, Q^N_{max}. Note that this mechanism uses a FIFO queue and is therefore different from the approach adopted by a priority queue [21]. This ensures that the delay-insensitive applications receive timely feedback and are therefore able to become submissive on their own.

Congestion window decrease On detecting congestion via network feedback, NF-TCP reduces its sending window to yield to higher priority applications. Note that the NF-TCP flow does not differentiate between the existence or non-existence of standard TCP and therefore is designed to decrease its congestion window as follows:

$$\text{CONGESTION\_DETECTED} : w \leftarrow w - b * w. \tag{2}$$

With a small value of b, NF-TCP flows will take longer to attain fairness among themselves. A high b on the other hand potentially results in low link utilization. Our evaluations are based on a value of b = 12.5%, similar to that proposed in [22]. It strikes a balance between being submissive to standard TCP, and improving the link utilization; and fairness in the presence of only NF-TCP flows.

3.2 Ability to saturate available bandwidth as fast as possible in the absence of other TCP flows

NF-TCP explores the use of a novel combination of bandwidth measurement and congestion control. The bandwidth estimation mechanism guides the decision of the congestion control framework in the appropriate timescale to opportunistically use spare bandwidth. This feature is especially applicable for a network friendly transport that needs to be both submissive during congestion and aggressive during non-congestion periods. NF-TCP uses bandwidth estimation to estimate the available bandwidth. This estimate is used as a target value, up to which the flow can increase its rate, as long as it has not received an ECN marking. This enables NF-TCP to be opportunistic in using available bandwidth resulting in throughput optimization as well as increased network utilization without causing congestion. The NF-TCP bandwidth estimation mechanism separates the measurement process for obtaining the available bandwidth from the congestion control. Whenever an estimate is not available, NF-TCP continues to use the standard conservative increase of 1 packet per RTT to be able to achieve fairness among NF-TCP flows.

Probing based on ECN (ProECN) We propose an ECN complemented probing mechanism that is based on PathChirp [23]. Similar to PathChirp, ProECN uses a series of packets that have an exponentially reducing inter-packet spacing to measure a wide range of available bandwidths. The sender sends this stream of packets to simulate an increasing sending rate and utilizes self-induced congestion to identify available bandwidth. ProECN differs from pathChirp by using an ECN complemented approach to measure the available bandwidth instead of depending only on increasing delay estimates. For this purpose we use a modified AQM queue that is able to perform instantaneous marking instead of the traditional EWMA based RED marking. The modified AQM queue identifies probe packets by the DSCP bit set in the header and marks them if the instantaneous queue size is greater than 1, to indicate self-induced congestion.

RAPID [5] also uses a PathChirp like mechanism to perform a rate-based transmission in which all data packets are part of a continuous logical group of N probe packets; NF-TCP on the other hand uses ProECN only for probing and employs a window based transmission for the data packets.

An NF-TCP flow generates two kinds of packets: normal data and probe packets. The bandwidth-estimation module starts after the first RTT on receiving an acknowledgment for the initial data packets, and only if there is no loss or ECN markings received. This is to ensure that newly starting flows do not contribute to congestion caused by the probes. The N probe packets are sent with varying inter-packet spacing so that we can measure available bandwidth in the range of $MinRate$ to $MaxRate$, such that the probes rate (r_i) is given by:

$$r_i = MinRate * (SF)^{i-1} \tag{3}$$

$$MaxRate = MinRate * (SF)^{N-2} \tag{4}$$

$MinRate$ is the minimum probe rate, $MaxRate$ is the maximum probe rate and N is the number of probe packets. The ratio of successive packet inter-spacing times within a chirp is the spread factor SF.

The estimated available bandwidth ($\mathrm{BW_{est}}$ (bps)) is described by:

$$\mathrm{BW_{est}} = \begin{cases} \mathrm{MinRate} * \mathrm{SF_{N-1}}, \\ \quad \text{if } \mathrm{BW_{Avail}} > \mathrm{Maxrate} \\ \mathrm{MinRate} * \mathrm{SF_{N-k}}, \\ \quad \text{if } \mathrm{Minrate} < \mathrm{BW_{Avail}} < \mathrm{Maxrate} \\ 0, \quad \text{if } \mathrm{BW_{Avail}} < \mathrm{Minrate} \end{cases} \tag{5}$$

k is the first packet in the series that arrives with an ECN marking and/or at a time greater than all the previous packets and $\mathrm{BW_{Avail}}$ is the actual available bandwidth on the link.

The NF-TCP bandwidth-estimation mechanism sends the probe packets with varying inter-packet gaps to emulate a range of sending rates. When the sending rate is higher than the available bandwidth, the probe packets undergo self-induced congestion. This results in the packets having to wait in the queue and thus being ECN marked. The ECN marking acts as a reliable and early indicator of a packet having to wait in the queue and therefore enables the bandwidth-estimation module to obtain a reliable

estimate of the available bandwidth. The ECN marking complements the delay-based approach of PathChirp since it exploits feedback received from the intermediate routers instead of having to depend only on delay measurements that could have high noise in low latency networks. With this enhancement, a source is able to identify excursion segments (a period of increasing delays) more accurately. The actual analysis and the heuristics utilized are similar to those described in [23] to account for bursty traffic.

Dynamic probing rate adjustment NF-TCP's bandwidth estimation module is designed to dynamically adjust the measurement probe rate to get an estimate of the available bandwidth quickly, while limiting the overhead introduced in the network. This ensures that the probing mechanism can function efficiently in networks ranging from low BDP networks to high BDP networks. Currently, by design, the minimum probe rate is set to 1Mbps to prevent probing in networks with an available capacity that is lesser than 1Mbps.

Similar to RAPID [5] the bandwidth-estimation module starts probing in a slow-start manner starting with 2 packets and doubling the number of packets afterwards. The slow-start phase is exited when the size of the probe train reaches N or when the estimated bandwidth is lower than the $MaxRate$. On exiting slow-start, the probing transits into a **dynamic probing mode**. Here, the average sending rate (r_{avg}) is set to $\alpha * \mathrm{BW_{est}}$. PathChirp probe packets are limited in quantity for a particular probe event and the SF is set to a fixed value (for our evaluations we use $N = 15$ and $SF = 1.2$). On receiving an estimate of the available bandwidth, the probing mechanism is restarted after a uniformly distributed time period with a mean value equal to that of the baseRTT. The new $MinRate$ is calculated according to the last $\mathrm{BW_{est}}$ and is given by:

$$MinRate = \frac{SF^{N-1} - 1}{(N-1)(SF-1) * SF^{N-2}} * r_{avg} \tag{6}$$

Congestion window increase NF-TCP is designed to be aggressive during non-congestion periods, in order to be opportunistic in using available bandwidth. NF-TCP takes advantage of the bandwidth-estimation performed to determine the rate of increase, so as to have an informed aggressive increase mechanism. This is unlike approaches that use an aggressive increase for large BDP networks, where they potentially cause congestion before backing off. Our approach enables NF-TCP to be truly friendly to existing transport connections. The estimate of the available bandwidth allows NF-TCP flows to aggressively utilize a certain percentage of the remaining available bandwidth. The increase is limited to a factor α of the estimated available bandwidth to allow inaccuracies in the measured estimate as well as differences in the time scales. Based on evaluations, we recommned the use of $\alpha = 0.5$. On receiving an estimate of the available bandwidth ($\mathrm{BW_{est}}$), NF-TCP switches over to an aggressive increase phase wherein the congestion window is adjusted as follows:

$$\mathrm{ACK} : w \leftarrow w + \frac{\alpha * \mathrm{BW_{est}} * \mathrm{RTT}}{w * \mathrm{packet\_size}}, \text{ if } \mathrm{BW_{est}} > 0 \tag{7}$$

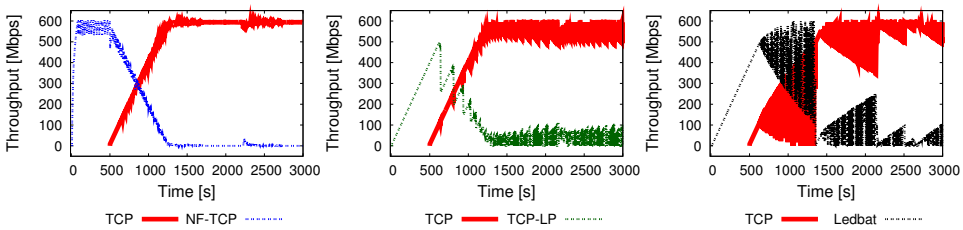
$$\mathrm{ACK} : w \leftarrow w + \frac{1}{w}, \text{ if } \mathrm{BW_{est}} = 0 \tag{8}$$

where ACK stands for acknowledgment received, RTT for round trip time in seconds and packet_size for average packet size in bits.

4 Performance Evaluation

We implemented NF-TCP on Linux kernel 2.6.31, and used the Linux TCP implementation ns-2 tool [17, 24] to import our Linux-based implementation of NF-TCP as well as the existing TCP Reno and TCP-LP onto ns-2. This enabled us to perform tests in a wide range of scale, topology and simulation time. Additionally, it also allowed us to compare NF-TCP performance against other candidate proposals such as LEDBAT and RAPID that we developed on ns-2.

We start with a single bottleneck scenario and then illustrate more sophisticated scenarios with RTT heterogeneity and multiple bottlenecks with several flows (Fig. 3). The bottleneck link routers (Router1, 2 and 3) maintain a modified RED queue with different threshold values for NF-TCP flows, and a normal RED queue for TCP flows, as shown in Table 1. Routers have a buffer capacity equal to the link BDP. We use FTP, and generate a SACK for every received data packet. Packet size 1000 bytes (including the IP header) and the initial $ssthresh$ for Reno/SACK is set to 100 packets. Bottleneck capacity is 600 Mbps and RTT is 100 ms. We refer the readers to [25] for results illustrating the performance of NF-TCP in networks with different BDPs.



(a) NF-TCP with TCP (b) TCP-LP with TCP (c) LEDBAT with TCP

Fig. 2. Single bottleneck: Instantaneous throughput of a candidate flow in presence of a TCP flow

4.1 Comparison with Other Approaches

In this section, we evaluate NF-TCP and other candidate approaches. We first focus on the performance of a single candidate (NF-TCP/LEDBAT/TCP-LP/RAPID) flow in the presence of a single standard TCP flow. We use the topology as shown in Fig. 3 with the candidate flows and the reference flows traversing the bottleneck link at Router R1.

Fig. 2 illustrates the instantaneous throughput of the network friendly flows in the presence of a competing standard TCP flow. Fig. 2(a) shows that the NF-TCP flow is able to opportunistically utilize the bandwidth in the period from 0-500s with the support of its ProECN bandwidth estimation. NF-TCP is comparable in its aggressive increase phase to the most aggressive of the alternatives, RAPID. RAPID was designed for use in high BDP networks and hence is aggressive in its startup. On the other hand, TCP-LP (Fig. 2(b)) and LEDBAT (Fig. 2(c)) are much slower in their increase and hence are unable to fully utilize the uncongested network during this time period.

From 500s onwards, as TCP increases its demand, NF-TCP quickly reduces its load, as a result of ECN marking, allowing TCP to grow its bandwidth as much as it desires. NF-TCP is thus submissive to TCP. TCP is not impacted after time t=1200s. In this

particular case, RAPID is also submissive. Again, TCP-LP and LEDBAT are much less submissive, yielding bandwidth to TCP more slowly. Further, once TCP nearly attains its full window at 1200 secs, TCP-LP and LEDBAT impact the TCP flow to different extents. In fact, when co-existing with LEDBAT, TCP continually experiences significant loss and reduced throughput, and thus incurs both additional delay and lower throughput. Thus, this experiment demonstrates both the capabilities of NF-TCP: to be opportunistic in its use of available bandwidth and submissiveness in the presence of TCP.

We evaluated the performance of a LEDBAT flow in the presence of a standard TCP flow. It is true that LEDBAT flows are network friendly to standard TCP flows as reported in [26], however only under *low BDP* scenarios. When the bottleneck bandwidth becomes higher (i.e., 200Mbps and more) and the buffer size is set equivalent to the link BDP (more realistic with higher speed links), our results show that LEDBAT flows are no longer friendly to TCP flows. Fig. 2(c) demonstrates that LEDBAT is more aggressive than NF-TCP (and TCP-LP) during congestion periods. Fig. 5(a) shows that, as the queue builds up, the base-delay stored by LEDBAT increases (Fig. 5(b)). This is due to the resetting of the baseRTT every 2-10 minutes and results in LEDBAT increasing its throughput. In short, the results demonstrate that LEDBAT does not satisfy the requirement of a network friendly protocol to maintain low queues and being submissive to TCP over a reasonable wide range of system parameters.

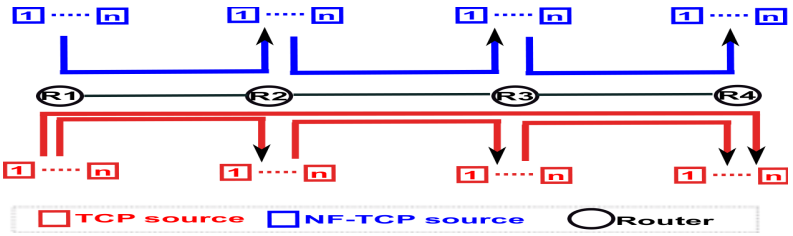
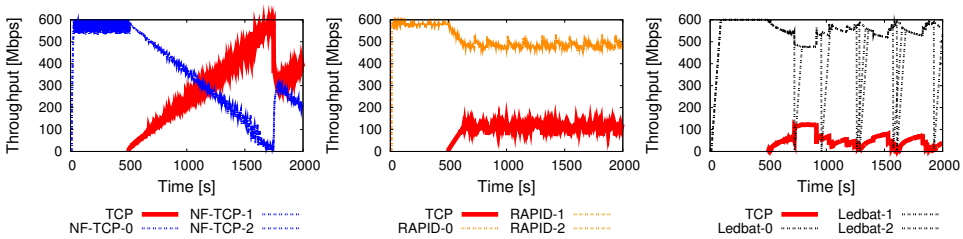


Fig. 3. Multi-hop topology with multiple bottlenecks

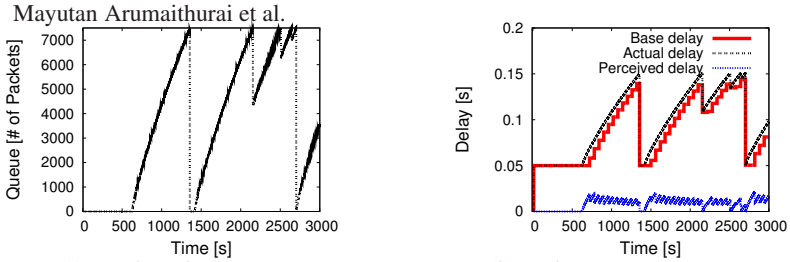


(a) NF-TCP vs TCP (b) RAPID vs TCP (c) LEDBAT vs TCP

Fig. 4. Multiple bottlenecks: Instantaneous throughput of candidate flows in the presence of a TCP flow (RTT of candidate flows = 1/3 RTT of TCP flow)

4.2 Fairness among NF-TCP Flows

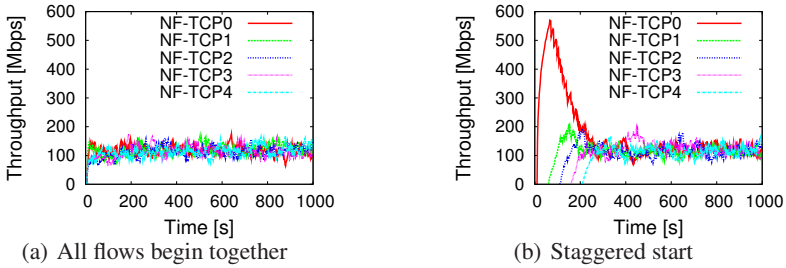
To study the fairness among NF-TCP flows, we choose the following two scenarios: a) 5 NF-TCP flows started at the same time, b) 5 NF-TCP flows starting one after another every 50 secs. Fig. 6(a) and Fig. 6(b) show that the 5 NF-TCP flows are fair to one another. Fig. 6(b) also shows that since the NF-TCP flow (NF-TCP0) that started at



(a) Bottleneck queue (b) Delay measurements

Fig. 5. LEDBAT is unfriendly: Causes substantial delay

time zero did not have any competing flow, it was able to utilize the available bandwidth completely and yield its share of the bandwidth when the other flows start. A decrease factor of 12.5% instead of the standard 50% makes it longer to achieve fairness but ensures that the overall link utilization is still high. With different RTTs, NF-TCP is also able to achieve a fairness similar to that achieved by standard TCP under the same scenario.



(a) All flows begin together (b) Staggered start

Fig. 6. Fairness among NF-TCP flows in a single hop scenario

4.3 Candidate Flows with Multihop, Varying RTTs

We now look at more realistic scenarios where the TCP flow traverses multiple hops going over several congested routers. They compete with flows that have different RTTs. For this purpose, we set up the testbed as shown in Fig. 3 with three 'bottleneck' links. The RTT on the longest path from R1 to R4 is three times the RTT on the shorter single hop paths. We perform evaluations with a TCP flow and competing candidate flows traversing one of R1, R2 and R3. The candidate flows are started at 0s and the TCP flow is started at 500s.

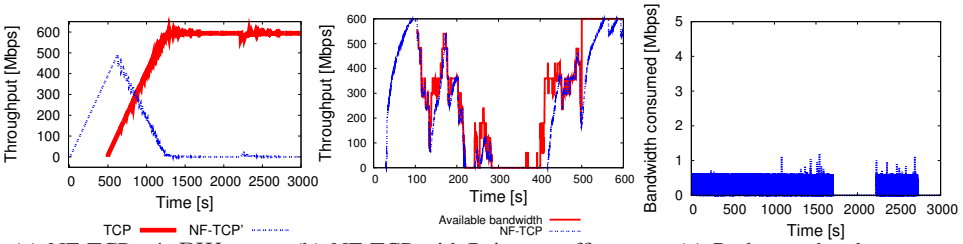
Fig. 4(a) illustrates that although NF-TCP has a much shorter RTT, it is friendly towards TCP flows while also opportunistically utilizing the spare bandwidth. RAPID (Fig. 4(b)) and LEDBAT (Fig. 4(c)) are both not submissive. For RAPID, this is due to the combination of its aggressive nature and its complete reliance on delay-based probing to measure available bandwidth.

4.4 NF-TCP in the Presence of Standard TCP

ProECN dynamic bandwidth estimation We evaluate the performance of ProECN bandwidth estimation tool within NF-TCP with a varying measurement range. Probes are sent about once per 2 RTTs. Fig. 7(c) illustrates that it is able to provide NF-TCP with an estimate of the available bandwidth while having an average probe throughput of about 0.6Mbps. The minimum rate that can be probed is limited to 1Mbps to prevent

congestion when network capacity is less than 1Mpbs. With ProECN, NF-TCP should ideally switch off during congestion periods, as confirmed by our experiments: in the period ranging from 1800-2200s and 2700s and beyond, when the NF-TCP becomes completely submissive.

NF-TCP vs UDP cross traffic We evaluate the ability of NF-TCP to opportunistically utilize available bandwidth in the presence of UDP cross traffic (rate generated from a Poisson distribution). Fig. 7(b) illustrates that NF-TCP opportunistically get close to the available bandwidth and then resorts to a slower increase. This results in better link utilization and also allows it to be friendly to other flows.



(a) NF-TCP w/o $\mathrm{BW}_{\mathrm{est}}$ (b) NF-TCP with Poisson traffic (c) Probe overhead

Fig. 7. Need for bandwidth estimation and the low overhead it produces

5 Related Work

Given the emergence of P2P-like delay-insensitive traffic, recent developments have attempted to provide means to support such applications. P4P [27] and the IETF Application-Layer Transport Optimization (ALTO) [28] protocol exploit the use of dedicated servers to assist in the selection of peers, to avoid overloading any specific part of the network. Such application layer solutions, when available, could complement NF-TCP, since they function at different time-scales.

To relax the dependency on dedicated servers and to react to instantaneous congestion levels in the networks, delay-based transport layer mechanisms such as TCP-LP [4] and LEDBAT [3] have been developed. NF-TCP differs from LEDBAT and TCP-LP as it depends on an ECN feedback for early notification and aggressively utilize bandwidth during non-congestion periods. Another aspect is in applying bandwidth measurement mechanisms to aid congestion control, such as RAPID [5]. However, to obtain an accurate measurement of the available bandwidth, RAPID is tightly coupled with PathChirp-based probing [23]. TCP Westwood [29] uses Agile probing bandwidth estimation mechanism to repeatedly reset the ssthresh value. In contrast to RAPID and TCP Westwood, NF-TCP employs a probing-based measurement scheme in addition to a window-based transmission of data packets. There is a clear separation of a lightweight measurement framework from the data transmission and flow control. Since delay-based schemes are known to be prone to transmission errors, NF-TCP introduces an ECN complemented probing instead of the pure delay-based probing approach of RAPID.

DC-TCP [30] is a new congestion control proposal developed for data center environments. Both DC-TCP and NF-TCP aim to maintain low buffers albeit by different mechanisms. DC-TCP requires the intermediate queues to perform instantaneous marking whereas NF-TCP uses an EWMA based mechanism to allow it to function better

in a heterogeneous and dynamic environment. Moreover, DC-TCP does not support aggressive start and is not designed to be submissive to TCP.

Approaches such as VCP [31], MLCP [32], BMCC [33], XCP [13], RCP [34], and rate feedback in Quick-Start [12] are based on intermediate routers providing more extensive feedback, but more importantly are not meant to be submissive to other flows.

6 Summary

In this paper, we presented a network friendly TCP variant, NF-TCP, that allows delay-insensitive applications to be submissive to standard TCP flows during congestion periods. Additionally NF-TCP exploits a novel combination of adaptive measurement of available bandwidth and the traditional window based congestion control to efficiently utilize network capacity. Our extensive evaluations illustrated that NF-TCP meets the requirements of a network friendly transport protocol and outperforms other candidate approaches in a wide range of network scenarios. NF-TCP contributes very little to the queueing at bottlenecks. We are currently experimenting with NF-TCP in a real testbed of Linux routers and PCs to further demonstrate its system performance. We believe that NF-TCP is viable and practical as an efficient network friendly protocol for delay-insensitive applications.

7 Acknowledgements

We would like to thank Fabian Glaser for his help with the implementation and the anonymous reviewers for their insightful comments.

References

1. "Internet Study 2008/2009," http://www.ipoque.com/resources/internet-studies/internet-study-2008_2009.
2. T. Moncaster, L. Krug, M. Menth, J. Araujo, S. Blake, and R. Woundy, "The Need for Congestion Exposure in the Internet," IETF, Internet-Draft draft-moncaster-conex-problem-00, Mar. 2010, work in progress.
3. S. Shalunov and G. Hazel, "Low Extra Delay Background Transport (LEDBAT)," IETF, Internet-Draft draft-ietf-ledbat-congestion-00.txt, Jul. 2010, work in progress.
4. A. Kuzmanovic and E. Knightly, "TCP-LP: a distributed algorithm for low priority data transfer," in *Proc. INFOCOM*, 2003.
5. V. Konda and J. Kaur, "RAPID: Shrinking the Congestion-Control Timescale," in *Proc. INFOCOM*, 2009.
6. S. Biaz and N. H. Vaidya, "Is the round-trip time correlated with the number of packets in flight?" in *Proc. IMC*, 2003.
7. R. Jain, "A delay-based approach for congestion avoidance in interconnected heterogeneous computer networks," *SIGCOMM Comput. Commun. Rev.*, 1989.
8. S. Floyd, "HighSpeed TCP for Large Congestion Windows," RFC 3649, Dec. 2003.
9. C. Jin, D. Wei, and S. Low, "FAST TCP: motivation, architecture, algorithms, performance," in *Proc. INFOCOM*, 2004.
10. K. Tan and J. Song, "A compound TCP approach for high-speed and long distance networks," in *Proc. INFOCOM*, 2006.
11. S. Ha, I. Rhee, and L. Xu, "CUBIC: a new TCP-friendly high-speed TCP variant," *SIGOPS Oper. Syst. Rev.*, 2008.

12. S. Floyd, M. Allman, A. Jain, and P. Sarolahti, "Quick-Start for TCP and IP," RFC 4782, Jan. 2007.
13. D. Katabi, M. Handley, and C. Rohrs, "Congestion control for high bandwidth-delay product networks," in *Proc. SIGCOMM*, 2002.
14. L. Stewart, G. Armitage, and A. Huebner, "Collateral damage: The impact of optimised tcp variants on real-time traffic latency in consumer broadband environments," in *Proc. Networking*, 2009.
15. K. Ramakrishnan, S. Floyd, and D. Black, "The Addition of Explicit Congestion Notification (ECN) to IP," RFC 3168, Sep. 2001.
16. M. Arumaithurai, X. Fu, and K. K. Ramakrishnan, "NF-TCP: Network Friendly TCP," in *Proc. LANMAN*, 2010.
17. "A Linux TCP implementation for NS2," http://netlab.caltech.edu/projects/ns2tcplinux/ns2linux/index.html.
18. F. Baccelli, G. Carofiglio, and S. Foss, "Proxy Caching in Split TCP: Dynamics, Stability and Tail Asymptotics," in *Proc. INFOCOM*, 2008.
19. J. Babiarz, K. Chan, and F. Baker, "Configuration Guidelines for DiffServ Service Classes," RFC 4594, Aug. 2006.
20. S. Floyd and V. Jacobson, "Random Early Detection Gateways for Congestion Avoidance," *IEEE/ACM Trans. on Netw.*, Jan. 1993.
21. A. Demers, S. Keshav, and S. Shenker, "Analysis and simulation of a fair queueing algorithm," in *Proc. SIGCOMM*, 1989.
22. K. K. Ramakrishnan and R. Jain, "A binary feedback scheme for congestion avoidance in computer networks with a connectionless network layer," in *Proc. SIGCOMM*, 1988.
23. V. J. Ribeiro, R. H. Riedi, R. G. Baraniuk, J. Navratil, and L. Cottrell, "PathChirp: Efficient Available Bandwidth Estimation for Network Paths," in *Proc. Passive and Active Measurement Workshop*, 2003.
24. D. X. Wei and P. Cao, "NS-2 TCP-Linux: an NS-2 TCP implementation with congestion control algorithms from Linux," in *Proc. WNS2*, 2006.
25. "Network friendly transport for delay-insensitive background traffic," http://www.net.informatik.uni-goettingen.de/research_projects/nft.
26. D. Rossi *et al.*, "News from the Internet congestion control world," *CoRR*, vol. abs/0908.0812, 2009.
27. H. Xie, R. Yang, A. Krishnamurthy, Y. Liu, and A. Silberschatz, "P4p: Portal for (p2p) applications," in *Proc. SIGCOMM*, 2008.
28. J. Seedorf and E. Burger, "Application-Layer Traffic Optimization (ALTO) Problem Statement," RFC 5693, Oct. 2009.
29. K. Yamada, R. Wang, M. Y. Sanadidi, and M. Gerla, "Tcp westwood with agile probing: Dealing with dynamic, large, leaky pipes," in *Proc. ICC*, 2004.
30. M. Alizadeh, A. Greenberg, D. Maltz, J. Padhye, P. Patel, B. Prabhakar, S. Sengupta, and M. Sridharan, "DCTCP: Efficient Packet Transport for the Commoditized Data Center," in *Proc. SIGCOMM*, 2010.
31. Y. Xia, L. Subramanian, I. Stoica, and S. Kalyanaraman, "One More Bit is Enough," *IEEE/ACM Transactions on Networking*, 2008.
32. I. Qazi and T. Znati, "On the Design of Load Factor based Congestion Control Protocols for Next-Generation Networks," in *Proc. INFOCOM*, 2008.
33. I. Qazi, T. Znati, and L. Andrew, "Congestion Control using Efficient Explicit Feedback," in *Proc. INFOCOM*, 2009.
34. N. Dukkipati, N. McKeown, and G. Fraser, "RCP-AC: Congestion Control to make flows complete quickly in any environment," in *Proc. High-Speed Networking Workshop*, 2006.