H-TCP Implementation in ns-3

[Extended Abstract] *

Amir Modarresi*, Siddharth Gangadhar*, Truc Anh N. Nguyen*,
and James P.G. Sterbenz*†§
*Information and Telecommunication Technology Center
Department of Electrical Engineering and Computer Science,
The University of Kansas, Lawrence, KS 66045, USA
‡School of Computing and Communications (SCC) and InfoLab21, Lancaster University, LA1 4WA, UK
§Department of Computing, The Hong Kong Polytechnic University, Hung Hom, Kowloon, Hong Kong
{amodarresi, siddharth, annguyen, jpgs}@ittc.ku.edu
www.ittc.ku.edu/resilinets

ABSTRACT

Along with the evolution of the Internet, high bandwidth-delay product (BDP) network environments are becoming more common for data transfer. However, TCP, as the most popular transport protocol in the Internet, is not able to fully utilize the available resources in these connections. Conventional TCP has been proven to react conservatively, especially in the presence of packet losses. Multiple variants have been proposed to address this issue and Hamilton TCP (H-TCP) is one such variant. H-TCP implements a loss-based congestion control algorithm that modifies the congestion avoidance and the fast retransmit/recovery states to allow for the protocol to achieve better throughput in high BDP instances. In this paper, we present our implementation of the H-TCP protocol in the open source network simulator (ns-3) and validate the model. We also perform multiple experiments with H-TCP comparing it against existing ns-3 TCP algorithms in varying scenarios which are available in the full version of this paper.

Categories and Subject Descriptors

I.6 [**Simulation and Modeling**]: General, Model Development, Model Validation and Analysis; C.2.2 [**Computer-Communication Networks**]: Applications— *Transport protocol*

Keywords

TCP, NewReno, Westwood, HighSpeed, Hamilton, congestion control, ns-3 network simulator, performance evaluation

1. INTRODUCTION

In order to enhance the poor behavior of conventional TCP, many variants have been designed and implemented. Various mechanisms have been included such as additional additive factors in the additive increase (AI) phase to increase link utilisation. Each variant relies on specialised congestion control algorithms by using network parameters such as RTT and try to deduce the optimal sending rate of the protocol. Existing variants include TCP Hybla [2], Scalable TCP [4], and HighSpeed TCP [3]. Much work has been focused on allowing these algorithms to not only use the large pipes, but also behave fairly in a shared environment, particularly with NewReno still widely used.

H-TCP [6][5] is one of the loss based algorithm that intended for high BDP environments. The objective of this paper is to present our implementation of H-TCP in ns-3, an open source network simulator [1] and validate it against the results in the original paper [6]. In addition to implementation and validation, we perform multiple experiments that compare H-TCP to other variants in varying scenarios that include both congestion and corruption. The implementation, validation and experiments of this protocol has been done in the development version of ns-3 (ns-3-dev), available between version 3.24 and 3.25, that has gone through a major overhaul of the TCP framework.

2. H-TCP IMPLEMENTATION

We implement the H-TCP protocol in the development version of ns-3 that contains the new TCP framework. As part of this section, we explain TCP class interactions briefly, followed by the H-TCP architecture.

2.1 TCP Class Interaction in ns-3

TCP functionality in ns-3 is provided by a set of classes interacting with each other. TCP sockets interact with TCP protocols to pass data segments to IP modules [1]. TCP protocols are located between the socket and IP layer. After ns-3.24, a major overhaul was performed in the structure and relationship among modules. A new class, `TcpSocketState`, was introduced to keep track of the common attributes and state of sockets like congestion window size and slow start threshold. These attributes had been defined in `TcpSocketBase` in older versions. Another class, called `TcpCongestionOps`, was designed as a base class to control the congestion operations. On the other hand, class `TcpNewReno` was set as the parent class of other TCP protocols including

*A full version of this paper is available as *H-TCP Implementation in ns-3* at `www.ittc.ku.edu/resilinets/reports/h-tcp-2016.pdf`

Wns3 '16 June 15, Seattle, United States.

DOI:

TCP HighSpeed, Hybla and Westwood inherited from `TcpCongestionOps`. H-TCP follows the same relationship and inheritance from `TcpNewReno`.

2.2 H-TCP Implementation in ns-3

H-TCP [6][5] is a variant designed for networks with high BDP. It is compatible with standard TCP, while it can switch to more aggressive behavior in high bandwidth networks. H-TCP considers elapsed time since last congestion. Then it calculates factor α as a function of last congestion time. The value of α starts larger, if congestion has not occurred. This value increases the size of the congestion window every time an ACK is received. Therefore, the protocol can utilise the bandwidth faster than standard TCP resulting in higher throughput and link utilization.

H-TCP does not change the slow start process. However, α is used in the congestion avoidance process to adjust the congestion windows size. If δ is the elapsed time in seconds since the last congestion for a specific flow, then the α can be calculated as a function of δ. Moreover, in order to be compatible with standard TCP, H-TCP changes its mode from standard to enhanced mode, when δ is greater than or equal to a threshold such as δ_l.

The Backoff coefficent β is calculated based on maximum and minimum RTT since the last congestion and throughput of the flow. The details can be found in the full version of the paper.

`HTCP` is an inheritance of `TcpNewReno`. Since the slow start process is the same as standard TCP, we do not define any new method for slow start. Moreover,`HTcp::CongestionAvoidance` is defined to modify `m_cWnd` based on H-TCP protocol. `HTcp::CongestionAvoidance` uses `AlphaFunction` method to calculate the correct value of α. `Htcp::PktsAcked` is used to keep RTT_{min}, RTT_{max} and throughput for each congestion period. We use `HTcp:: GetSsthresh` to adjust `ssthresh` value based on calculated β.

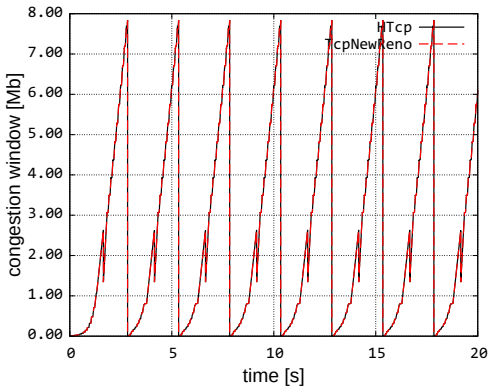

Figure 1: Slow start and fast recovery validation

3. VALIDATION

In order to validate our implementation, the first topology consists of a single sender and receiver connected by a router. Our second topology is the popular dumbbell topology with multiple senders and receivers separated by a pair of routers. In all of our experiments, traffic generation is unidirectional and from the sender to the receiver. The TCP socket buffer sizes, along with the TCP delayed ACK and its timeout are unmodified from ns-3's default parameters. In our experiments, we start the initial flow at 4s. We set the router queue size to be BDP for all scenarios. Some of the validation results for various part of the H-TCP algorithm are illustrated in Figures 1,2 and 3.

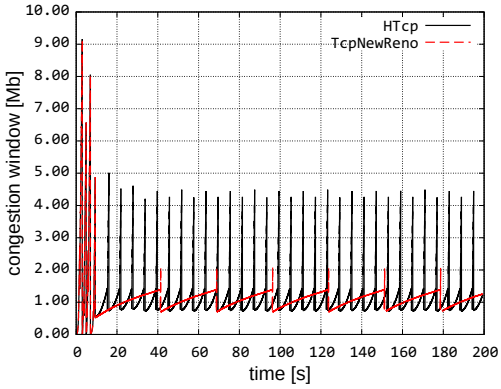

Figure 2: Validation of all operational states

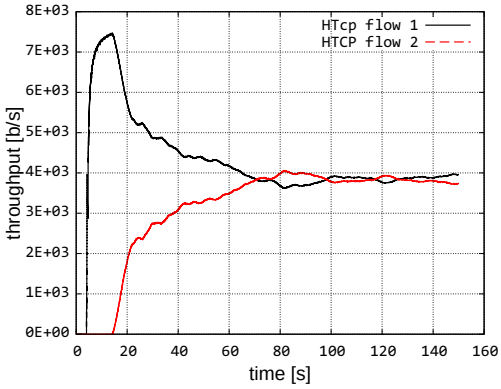

Figure 3: Friendliness of two H-TCP flows

4. CONCLUSION

In this paper, we explain the implementation of the H-TCP protocol. We validate each state of our protocol and compare its behavior against the original paper. Furthermore, we compare the performance of the protocol with other available protocols in ns-3 in the full version of this paper. The results show that, H-TCP is able to fill up the pipe much faster than NewReno, and performs in the range of other high speed protocols such as HighSpeed and Hybla.

5. REFERENCES

[1] The ns-3 Network Simulator. http://www.nsnam.org.

[2] C. Caini and R. Firrincieli. TCP Hybla: a TCP enhancement for heterogeneous networks. *International journal of satellite communications and networking*, 22(5):547–566, 2004.

[3] S. Floyd. HighSpeed TCP for Large Congestion Windows. RFC 3649 (Experimental), Dec. 2003.

[4] T. Kelly. Scalable tcp: Improving performance in highspeed wide area networks. *SIGCOMM Comput. Commun. Rev.*, 33(2):83–91, Apr. 2003.

[5] D. Leith and R. Shorten. online.

[6] D. Leith and R. Shorten. H-TCP protocol for high-speed long distance networks. In *Proc. PFLDnet*, 2004.