Automatic adjustment of congestion control for each transport connection

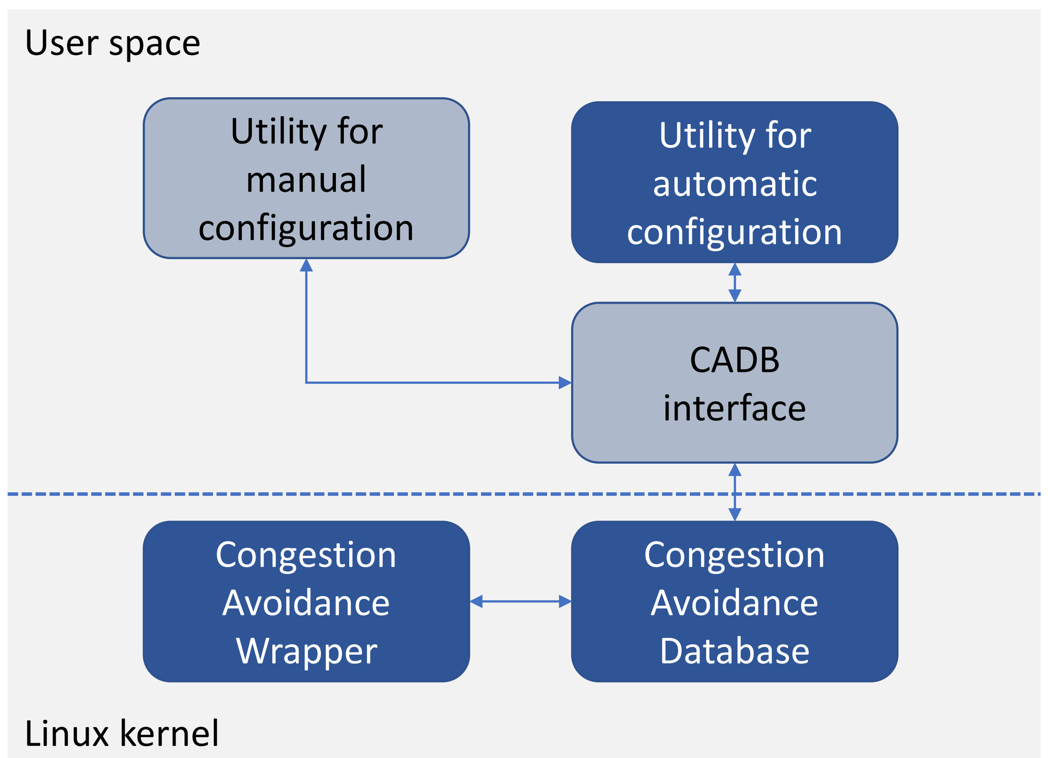
Lomonosov Moscow State University

Alexander Kovalyov
Lomonosov MSU
Moscow, Russia
cojuer@gmail.com

Introduction

Multiple congestion avoidance algorithms were developed to deal with the phenomenon of network congestion. The goal of every such algorithm is to keep transmission rate as high as possible while preventing packet drops due to link overload. Each algorithm was designed with a specific network environment in mind (throughput, delay, loss rate, etc.), and usage of inappropriate algorithm leads to tangible decrease in connection speed. However, most of distributed applications rely on default congestion avoidance algorithm as long as its customization does not cover expenses for the development. The article proposes a method to estimate quality conditions of network connection and to choose optimal congestion avoidance algorithm based on the estimation in a purely automatic way.

Implementation



Developed tool consists of 3 main components:

- Congestion Avoidance Database (**CADB**) is a Linux kernel module that stores information about suggested congestion avoidance algorithms for traffic flows of a particular type(with given source and destination IP addresses). Also it stores statistics of previous TCP sessions including delay estimation, number of acknowledgments and fraction of retransmitted packets;
- Congestion Avoidance Wrapper (**CAW**) is another Linux kernel module, which serves as a congestion avoidance algorithm and at the same time as a wrapper for some other congestion avoidance algorithm, which is called inner. CAW is able to choose appropriate inner algorithm from CADB during TCP session initialization. Wrapper has all those operations that are defined in the inner algorithm. In addition, the wrapper always defines the method *in_ack_event*, which allows to collect connection statistics for each TCP acknowledgement. When called, the wrapper can perform arbitrary operations on the socket and then call the corresponding methods of the inner algorithm. At the end of TCP session collected statistics is transferred to CADB;
- User space utility is responsible for filling in CADB. It contains a table of information about optimal algorithms for given quality conditions and a set of decision trees, where each tree is associated with one of the standard congestion avoidance algorithms. Having received the current statistics and algorithm from CADB, the utility uses corresponding decision tree to estimate current connection quality characteristics. Based on this estimation, an optimal algorithm is selected. In order to train decision trees and fill table with optimal algorithms the information is required about behavior of algorithms in different quality conditions.

The other two components shown at the picture are the utility representing command line interface for manual CADB configuration and shared library which allows user space utilities to interact with kernel module CADB.

Experimental evaluation

Experimental evaluation started with gathering information about behavior of algorithms in different network conditions. This information was required to determine the most optimal algorithm for each set of conditions and to train decision trees to estimate network conditions based on connection statistics. Multiple simulations were run in order to collect such data. Topology for each simulation consisted of two hosts (sender and receiver) and two switches connecting them. Quality parameters were configured at the link between two switches.

Exact values of simulation parameters are shown in the following table:

Parameter	**Values**
Throughput	1, 10, 100, 1000 Mb/s
Loss	10^{-5}, 10^{-4}, ... , 10^{-1}, 1%
Delay	1, 5, 10, 25, 100 ms
Jitter	0, 10, 20%
Algorithms	Westwood, Cubic, Reno, Vegas, Illinois, BBR
Scenario time	10s

After using gathered data to configure previously mentioned utility it became possible to compare developed tool to standard congestion avoidance algorithms. This part of experimental evaluation consisted of multiple simulations, each including two stages. At each stage one TCP connection was transmitting data for defined period of time. The purpose of the first stage was to collect connection statistics for given quality conditions and predetermined congestion avoidance algorithm. After the completion of the first stage, the utility was used to approximate quality conditions of the network and choose the best algorithm for those conditions. The second stage was similar to the first one except for usage of chosen algorithm instead of predetermined one.

In addition, the average connection speed was compared for two cases: with use of the predetermined algorithm and the one selected by the developed tool. The results are shown in the following table.

Algorithm	**Real gain** (%)	**Possible gain** (%)
Illinois	3.15	4.91
Vegas	10.27	12.69
Reno	10.11	12.00
Westwood	12.89	15.58
Cubic	7.92	10.08
BBR	-0.85	3.62

Also the table shows the maximum possible speed gain for the given algorithm, which is based on the average values of the connection speed obtained during data collection. As can be seen, on each of the algorithms there exists a theoretical probability to increase connection speed. And despite the fact that the proposed approach to estimating quality characteristics of the network environment is rather primitive, it allows to achieve a tangible gain on most of the tested configurations.

Conclusion

The article proposes estimation of connection quality conditions to choose optimal congestion avoidance algorithm for estimated conditions. Experimental evaluation has shown superiority of the developed tool to usage of the same congestion avoidance algorithm for various network quality conditions for almost all examined algorithms.

There are multiple topics for further research: consideration of TCP fairness, usage of more advanced estimation techniques, switching of algorithm during TCP session and real-network simulations.

Bibliography

[1] S. Mascolo, C. Casetti, M. Gerla, M. Y. Sanadidi, and R. Wang, "Tcp westwood: Bandwidth estimation for enhanced transport over wireless links," in Proceedings of the 7th annual international conference on Mobile computing and networking. ACM, 2001, pp. 287–297.
[2] C. P. Fu and S. C. Liew, "Tcp veno: Tcp enhancement for transmission over wireless access networks," IEEE Journal on selected areas in communications, vol. 21, no. 2, pp. 216–228, 2003.
[3] N. Cardwell, Y. Cheng, C. S. Gunn, S. H. Yeganeh, and V. Jacobson, "Bbr: Congestion-based congestion control," Queue, vol. 14, no. 5, p. 50, 2016.
[4] S. Ha, I. Rhee, and L. Xu, "Cubic: a new tcp-friendly high-speed tcp variant," ACM SIGOPS operating systems review, vol. 42, no. 5, pp. 64–74, 2008.
[5] K. Tan, J. Song, Q. Zhang, and M. Sridharan, "A compound tcp approach for high-speed and long distance networks," in Proceedings-IEEE INFOCOM, 2006.
[6] M. Alizadeh, A. Greenberg, D. A. Maltz, J. Padhye, P. Patel, B. Prabhakar, S. Sengupta, and M. Sridharan, "Data center tcp (dctcp)," in ACM SIGCOMM computer communication review, vol. 40, no. 4. ACM, 2010, pp. 63–74.