Survey on fast network congestion detection and avoidance

Luis Cordero
University of Technology and Engineering Careers
Lima, Perú
luis.cordero@utec.edu.pe

Anthony Guimarey
University of Technology and Engineering Careers
Lima, Perú
anthony.guimarey@utec.edu.pe

Abstract

Daily, packets are transmitted in multiple devices, accessing different nodes to their destination, potentially causing network congestion. In this work, we adapt a solution provided by the authors of paper *Fast network congestion detection and avoidance using P4*. The logic behind it make a clear idea of the power of controllers to manage congestion in forward nodes.

Keywords: congestion detection, congestion avoidance, networks

1 Introduction

Networks have provided an easy solution for the transport of information and data. However, like any transport system, it can become congested and generate serious problems such as packet loss due to overflow or a long delay in the delivery of information. For this reason, networks have been evolving, not just physically, like fiber optics or 5G internet, but even in techniques and algorithms to avoid and properly manage congestion.

Network problems will always remain constant since the amount of devices per person connected to a network is increasing and more data is being transported trough it. This is why the techniques need to be efficient, even when a large burst of data comes trough the network it must not represent a problem at all. In this aspect, we will evaluate the effectiveness of avoiding congestion and separately how the algorithm faces an ongoing congestion. Congestion control mechanisms of traditional transport protocols such as TCP detect congestion at the sender node and modify the sending rate accordingly [1]. We want to compare this technique to a solution that does not modify the sending rate but the package route to control the problem.

The objective of this experimentation is to see how this solution works, how efficient it is compared to other control mechanisms and its limits. It is expected that the results obtained show a significant difference when using this method, as well as see how close the theory behind it is.

2 Concepts

In this section we explain the concepts of congestion control, congestion avoidance and transport protocols. These concepts are related and are the core for this project. The concepts of congestion control and avoidance manage the problem in different situations and ways.

2.1 Transport protocols

A transport-layer protocol provides logical communication between application processes running on different hosts [2]. One of those protocols is the Transmission Control Protocol (TCP) which is the one that we must be focusing on during this project. It carries most Internet traffic, so performance of the Internet depends to a great extent on how well TCP works. Performance characteristics of a particular version of TCP are defined by the congestion control algorithm it employs [3].

2.2 Congestion Control

When a congestion is in progress, a response from any part of the network must occur to solve it or at least try to mitigate as much as possible. The changes to TCP include a limited transmit mechanism for transmitting new packets upon receipt of one or two duplicate acknowledgments, and a SACK-based mechanism for detecting and responding to unnecessary fast retransmits or retransmit timeouts [5]. This means that, the way that TCP manages a congestion is limiting the send rate of new packages since the host is receiving messages that the information that it is sending is taking more time that it should to get to its destination.

2.3 Congestion Avoidance

Congestion avoidance mechanisms allow a network to operate in the optimal region of low delay and high throughput, thereby, preventing the network from becoming congested. The key component of any congestion avoidance scheme is the algorithm used by the users to increase or decrease their load [6].

3 Methodology

The experiment in the research had been done in the language P4. However, in this work, we decided to adapt it to the programming language Python and an emulation environment called Mininet. This language offers the ability to gather and export important packet meta-data, like timestamps from different stages of processing, directly from the data-plane while the data-packets are being processed. To complete our task, we use a hierarchical architecture to detect and avoid congestion. Fig. 1 shows a central controller

that configures the latency thresholds and other parameters, and a local congestion control module at the P4 switches, which monitors the state of all the low-latency flows. All the work is based on the project made by the authors of the paper, so we adapt it to our needs.

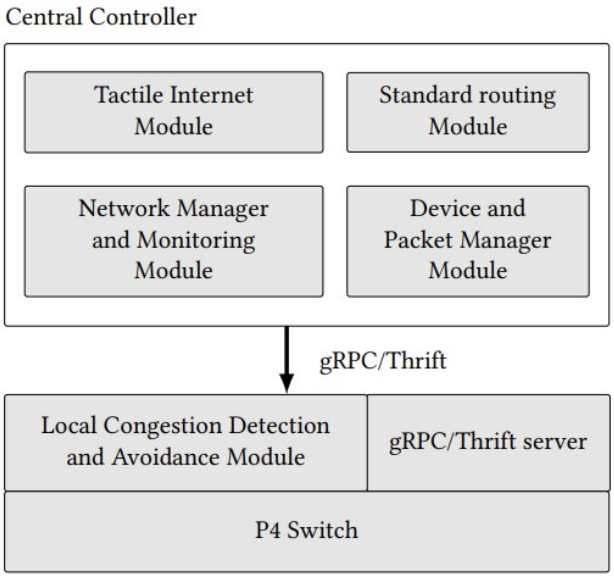

Figure 1. Hierarchical design of the control plane.

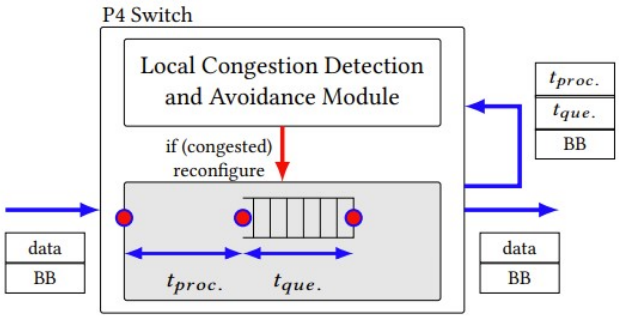

Figure 2. Detection of congestion in the data-plane.

Local Congestion Detection and Avoidance module, see Fig. 2, is developed to monitor the processing and queuing delays. This module works based on whether the critical latency flow increases and there is a possible congestion scenario, it switches the traffic to a backup path if it exists or sends a signal to the previous node on the primary path that is congested. Therefore, it should no longer forward any more packets belonging to that flow. So, in order to achieve our goal, we have two options:

1. Add both entries (primary and backup) to a table and decide which rule to apply based on some meta-data stored in the registers
2. Send packets or packet digest notifications to a local listener that tracks the flows and acts as a small local control-plane.

The first solution is applicable to all P4 compatible hardware and all processing is done entirely in the data plane. However, more register and table entries are needed to maintain an accurate flow state in the data plane. Also, updating registers per packet can lead to race conditions when packets from the same flow are processed in parallel [1].

The other solution is use packet copies or packet digests, a local listener module which makes local routing decisions based on received data. The problem is the additional time it takes to transmit the digest packet and process it at the local control module. In addition, the local control application module is likely to be overloaded, if the speed in which digest notifications are generated is very high [1].

4 Implementation

The network topology selected is shown in Figure 3. The main files are in github.

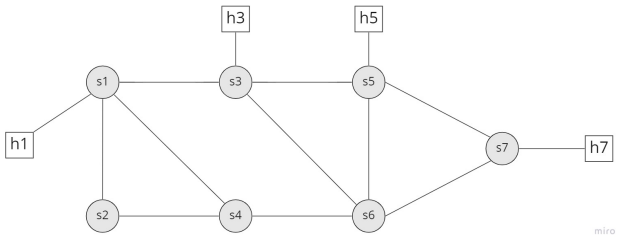

Figure 3. Network topology

References

[1] Turkovic, Belma, Fernando Kuipers, Niels van Adrichem and Koen Langendoen. "Fast Network Congestion Detection and Avoidance Using P4". *Proceedings of the 2018 Workshop on Networking for Emerging Applications and Technologies*, July 2018, 45-51. https://doi.org/10.1145/3229574.3229581.

[2] Kurose, James F., and Keith W. Ross. "Introduction and Transport-Layer Services". *Computer Networking a Top-down Approach*, 8th edition, 182. Hoboken: Pearson, 2022.

[3] Afanasyev, Alexander, Neil Tilley, Peter Reiher, and Leonard Kleinrock. "Host-to-Host Congestion Control for TCP." *IEEE Communications Surveys; Tutorials 12*, no. 3 (2010): 304–42. https://doi.org/10.1109/surv.2010.042710.00114.

[4] Jain, R, and K.K. Ramakrishnan. "Congestion Avoidance in Computer Networks with a Connectionless Network Layer: Concepts, Goals and Methodology". *[1988] Proceedings. Computer Networking Symposium*,September 2, 1988, 199. https://doi.org/10.1109/cns.1988.4990.

[5] Floyd, S. "A Report on Recent Developments in TCP Congestion Control". *IEEE Communications Magazine* 39, no. 4 (April 2001): 84–90. https://doi.org/10.1109/35.917508.

[6] Chiu, Dah-Ming, and Raj Jain. "Analysis of the Increase and Decrease Algorithms for Congestion Avoidance in Computer Networks". *Computer Networks and ISDN Systems* 17, no. 1 (June 10, 1989): 1–14. https://doi.org/10.1016/0169-7552(89)90019-6.