基於 P4 Hash 演算法的網路流量平衡

Network Traffic Balance Based on P4's Hash algorithm

第十組

r09921072 劉哲瑞

r09921088 李昕耘

r09942138 馮景泰

Introduction:

關於 Internet 的連線，時常會發生網路突然出現斷網的情況，常常是因為是傳輸量太大導致 link failure，對於網路流量及狀況回復這個議題是非常重要的，畢竟在有些地方對於網路延遲或網路不順會給使用者造成很大的困擾，我們希望能從中找到解決的方法。

SDN 是一種很方便的架構，他能有效且方便的對網路做控制，像是可以用 SDN 對網路做拓樸分析，並且運用一些控制工具，ex: POX, NOX, Floodlight, OpenDaylight 等等，再從獲取到的網路拓樸並對其做 path Algorithm 由於 SDN 相比傳統網路更具有便利性，所以我們想使用 SDN 這項技術去試著解決改善 Traffic balance 這個問題。

Problem Definition:

在傳統網路中只能走 shortest path，並沒有辦法控制 packet 走其他路徑，如此一來可能導致嚴重的網路壅塞，再者，許多單位都有設置流量限制

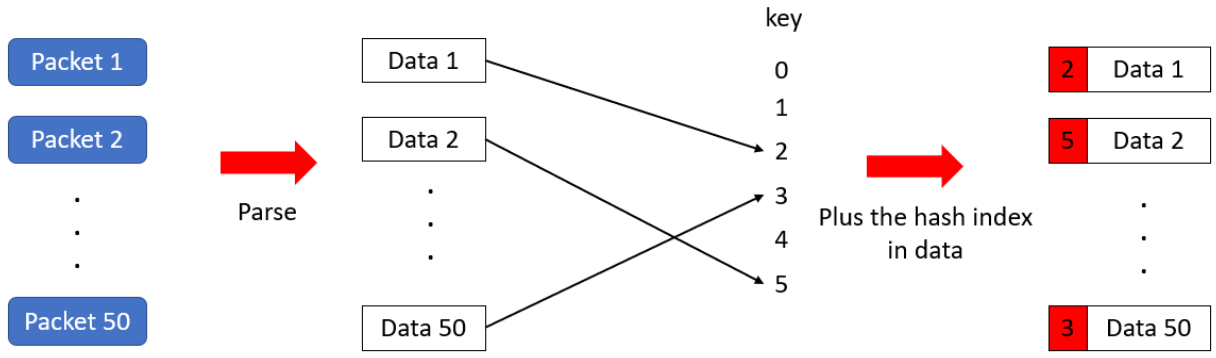
problem 1: 如何讓封包走不同路徑以避免網路壅塞

problem 2: 如何設定 load balance 比例，即走不同路徑的比例

Methodology:

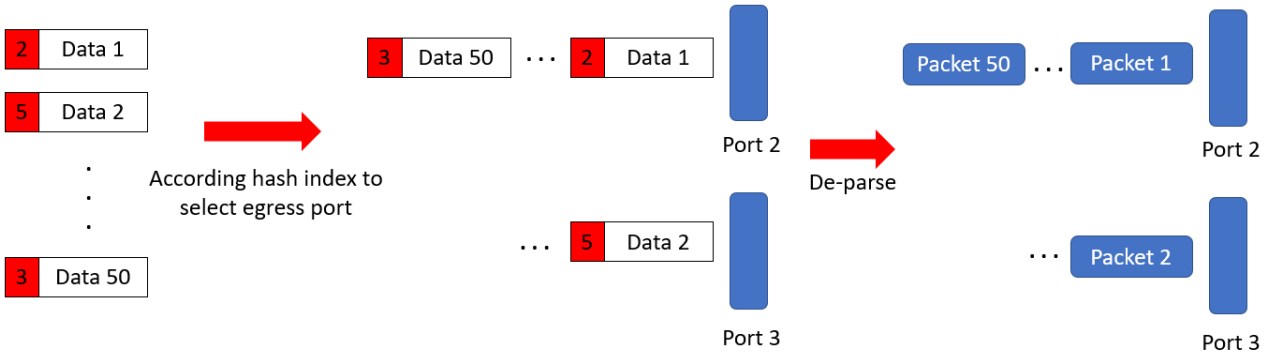
在 Openflow 中可以做到把流量平分給 output port，在 P4 中甚至能做到自定義 output 到各 port 的流量比例: 可以使用 P4 提供的 hash function 來設定分配 load balance 的比例。我們的解決方法則是決定使用 P4 的 hash function 來控制流量以相對得比例走第二短(hop 數第二少的)的路徑，來減少全部封包都走最短路徑所發生的網路壅塞情況。

Implementation:



圖[1] 交換機入口機制

應用了我們機制的交換機在入口(ingress port)收到封包後，會根據封包內的資訊(ip_srcAddr, ip_dstAddr, tcp_srcPort…)來映射到 0 ~ 5 其中一個值，並將此值加入 data 中作為 hash index，如圖[1]所示。

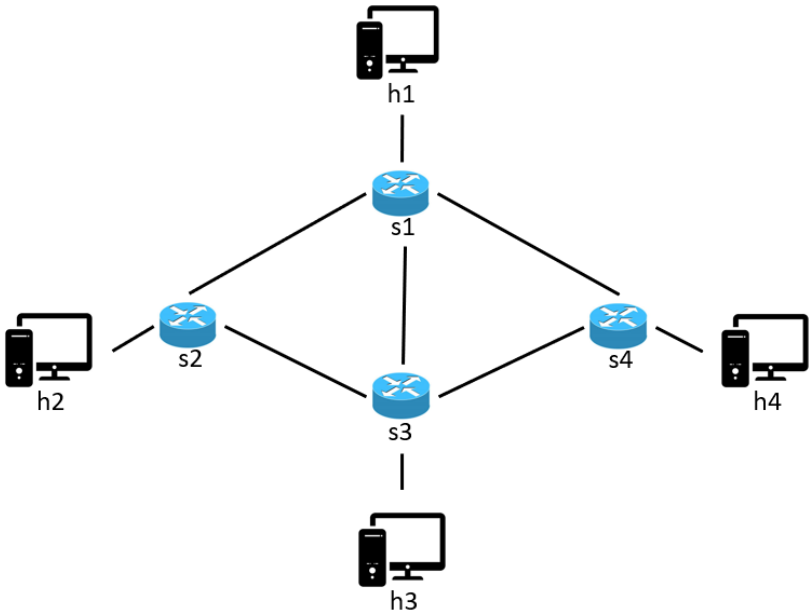


圖[2] 交換機出口機制

而在交換機出口，我們會根據 hash index 的值選擇要由哪個 egress port 的發送封包，我們此處設定 index 為 5 的封包由 port3 發送，而其他 index 的值則由 port2 發送。決定好從哪個 egress port 發送後，再將 tcp, ip, ethernet 的 header 接上對應的 data。

透過 p4 提供的 hash function 使我們對封包加上 index，並且我們將根據 5:1 的比例將封包分送至 port2 及 port3 進行傳輸，來達到 load balancing 的目的。

Experiment Setup

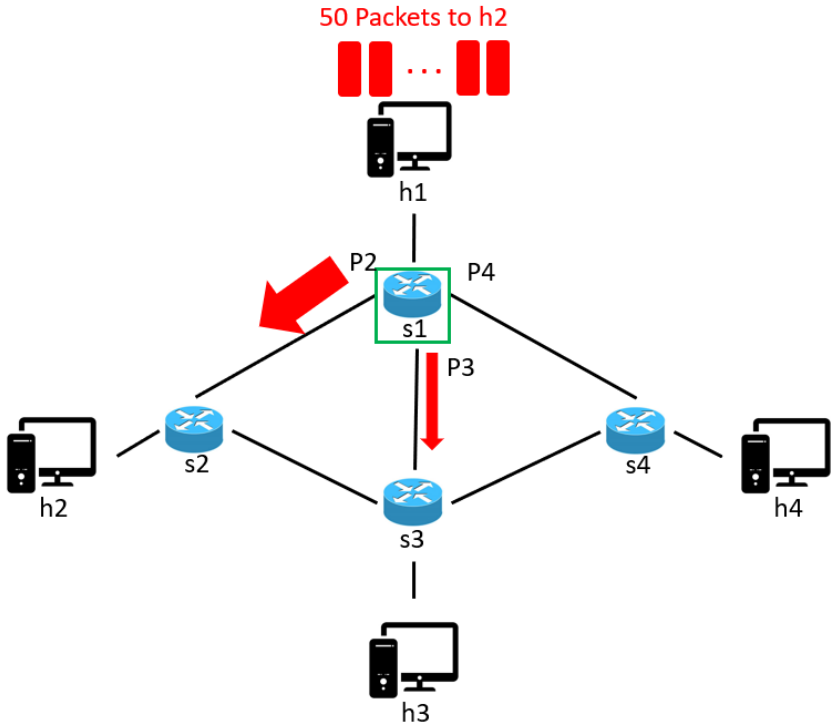


圖[3] 實驗模擬情境

```
s1 -> gRPC port: 50051
s2 -> gRPC port: 50052
s3 -> gRPC port: 50053
s4 -> gRPC port: 50054
**********
h1
default interface: eth0 10.0.1.1        08:00:00:00:01:01
**********
**********
h2
default interface: eth0 10.0.2.2        08:00:00:00:02:02
**********
**********
h3
default interface: eth0 10.0.3.3        08:00:00:00:03:03
**********
**********
h4
default interface: eth0 10.0.4.4        08:00:00:00:04:04
**********
```

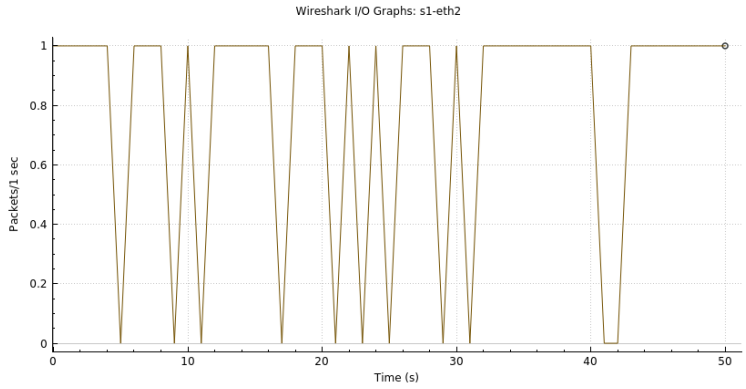
圖[4] 模擬情境中的裝置訊息

圖[3]為我們實驗場景的網路拓樸，圖[4]則為拓樸中每個裝置的訊息，其中我們的實驗方式如圖[5]所示，我們將從 h1 發送 50 個封包到 h2，其中我們的機制將會應用在 s1 上，他將會根據我們分配給不同出口的 hash index 比例，來達到分流的效果。我們將會在 s1-eth2(port2)和 s1-eth3(port3)上，使用 wireshark 觀察從這兩個出口發送的封包數，另外我們也會觀察 s1-eth4(port4)，來驗證我們的機制只會走最短和次短路徑，亦即不會有封包從 port4 出口送出。

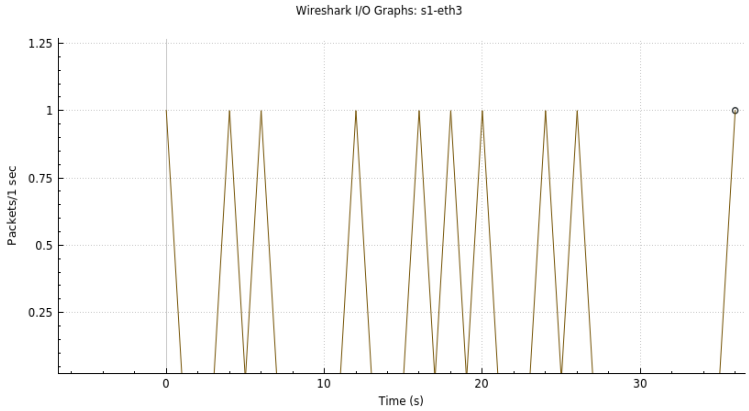


圖[5] 實驗方法

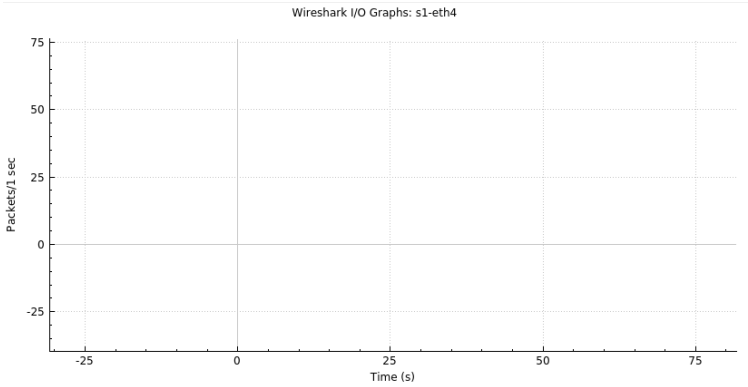
Experiment Result



圖[6] s1-eth2 發出的封包數



圖[7] s1-eth3 發出的封包數



圖[8] s1-eth4 發出的封包數

從圖[6]和圖[7]中可以發現大部分的封包都由 port2 出口發出，其中有 10 個封包由 port3 出口發出，跟我們預計要分配的比例 5:1 非常相近。另外從圖[8]可以發現沒有封包由 port4 發出。由此可知，我們的機制可以實現封包依照使用者分配比例由不同出口發送封包，且走的路徑為最短及次短的路徑。

Presentation Link:
https://www.youtube.com/watch?v=hsYFLTrVibg

Related Work:

[1] Jun Li, Xiangqing Chang,Yongmao Ren, Zexin Zhang, Guodong Wang, "An Effective Path Load Balancing Mechanism Based on SDN ", 2014 IEEE 13th International Conference on Trust, Security and Privacy in Computing and Communications.

[2] Mao Qilin, Shen WeiKang, "A Load Balancing Method Based on SDN", 2015 IEEE 7th International Conference on Measuring Technology and Mechatronics Automation