SDNFV Project2 Report

Part 1

總共有2個。

Match Fields	Actions	Timeout Values
IN_PORT=2	Output Port=1	0
IN_PORT=1	Output Port=2	0

Part 2

ARP

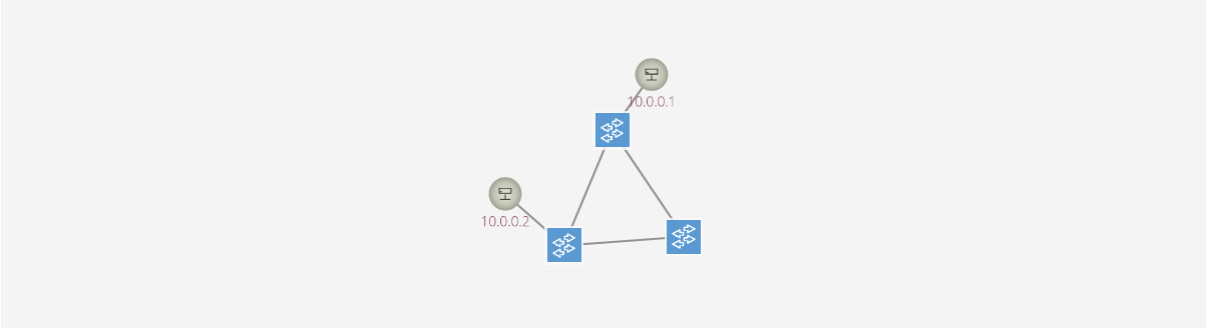
```
mininet> h1 arping h2
ARPING 10.0.0.2
42 bytes from c6:0e:71:2c:50:e6 (10.0.0.2): index=0 time=8.186 usec
42 bytes from c6:0e:71:2c:50:e6 (10.0.0.2): index=1 time=13.315 usec
42 bytes from c6:0e:71:2c:50:e6 (10.0.0.2): index=2 time=12.836 usec
42 bytes from c6:0e:71:2c:50:e6 (10.0.0.2): index=3 time=12.597 usec
42 bytes from c6:0e:71:2c:50:e6 (10.0.0.2): index=4 time=12.627 usec
42 bytes from c6:0e:71:2c:50:e6 (10.0.0.2): index=5 time=12.544 usec
```

IPv4

```
mininet> h1 ping h2
PING 10.0.0.2 (10.0.0.2) 56(84) bytes of data.
64 bytes from 10.0.0.2: icmp_seq=1 ttl=64 time=0.413 ms
64 bytes from 10.0.0.2: icmp_seq=2 ttl=64 time=0.102 ms
64 bytes from 10.0.0.2: icmp_seq=3 ttl=64 time=0.135 ms
64 bytes from 10.0.0.2: icmp_seq=4 ttl=64 time=0.099 ms
64 bytes from 10.0.0.2: icmp_seq=5 ttl=64 time=0.051 ms
64 bytes from 10.0.0.2: icmp_seq=6 ttl=64 time=0.082 ms
64 bytes from 10.0.0.2: icmp_seq=7 ttl=64 time=0.100 ms
64 bytes from 10.0.0.2: icmp_seq=8 ttl=64 time=0.100 ms
64 bytes from 10.0.0.2: icmp_seq=9 ttl=64 time=0.265 ms
64 bytes from 10.0.0.2: icmp_seq=10 ttl=64 time=0.043 ms
64 bytes from 10.0.0.2: icmp_seq=11 ttl=64 time=0.037 ms
64 bytes from 10.0.0.2: icmp_seq=12 ttl=64 time=0.113 ms
64 bytes from 10.0.0.2: icmp_seq=13 ttl=64 time=0.103 ms
```

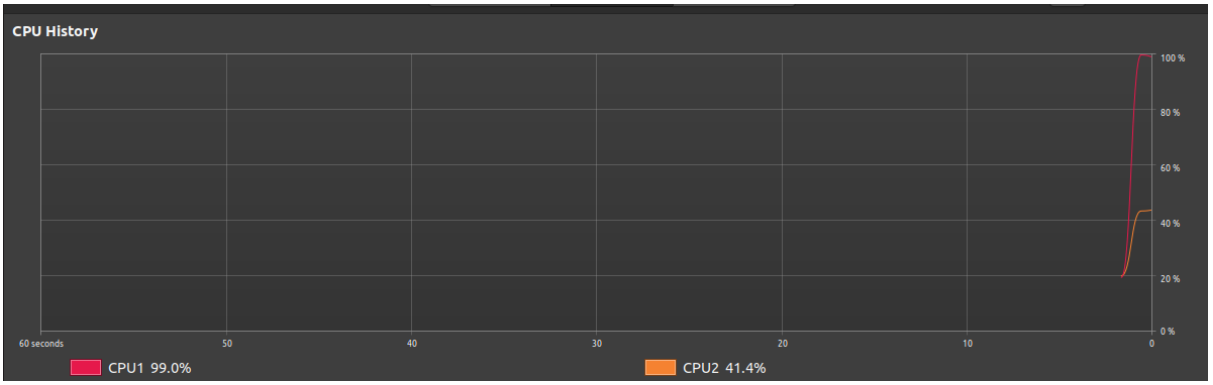
Part 3

使用了下面這個拓樸圖，並給予每個switch ”OUTPUT” : “ALL”的flow rule。



接著嘗試讓H1去ping H2：
$ mininet> h1 ping h2

觀察到的現象：CPU使用率突然飆升到接近100%。



在上面的拓樸圖中出現了路由迴路，s1 -> s2, s2 -> s3, s3 -> s1，導致了broadcast storm。

Part 4

當啟動reactive forwarding時，如果從h1送出封包給h2，第一個封包會被送往controller，並由reactive forwarding app協助判斷如何進行forwarding。

學到或解決了什麼?

在這個project中，我學到了openflow封包的觀察、如何安裝自定義flow rules、broadcast storm的原因與現象，以及reactive forwarding的行為。