SDN Project1 Answer Sheet

Name: 陳正宗 Student ID: 0756823 Department: 資訊組

|  |
| --- |
|  |
| **1.**  **Ans:**   |  |  |  | | --- | --- | --- | | **Layer** | **Description** | **Content** | | Physical | Frame | 170 bytes on wire (1360 bits), 170 bytes (1360 bits) on interface 0 | | Data Link | Ethernet II | Src: 00:00:00:00:00:00, Dst: 00:00:00:00:00:00 | | Network | Internet Protocol Version 4 (IPV4) | Src: 127.0.0.1, Dst: 127.0.0.1 | | Transport | Transmission Control Protocol (TCP) | Src Port: 6653, Dst Port:53970, Seq:1265, Ack:7777, Len:104 | |
| **2.**  **Ans:**   |  |  |  | | --- | --- | --- | | **Priority** | **Match** | **Action** | | 40000 | ETH\_TYPE: bddp | Output: Controller | | 40000 | ETH\_TYPE: lldp | Output: Controller | | 40000 | ETH\_TYPE: arp | Output: Controller | | 5 | ETH\_TYPE: ipv4 | Output: Controller | |
| **Ans:**    **由練習題可知, 在沒有開onos fwd的狀況下, 封包不會自動轉送, 必須手動新增flow rules.**  **題目為s1, h1 and h2, 由練習題可知, 練習題只新增了一條flow rule, 封包由h1送給h2時, h2可以收到h1封包, 卻因為h2到h1沒有flow rule, 沒辦法將封包回覆給h1, 於是需要新增一條flow rules讓兩台hosts 能互相通訊.**  **Flow Rules,**   |  |  |  |  | | --- | --- | --- | --- | | **File name** | **deviceId** | **IN\_PORT** | **OUTPUT** | | flows\_s1-1\_0756823.json | of:0000000000000001 | 1 | 2 | | flows\_s1-2\_0756823.json | of:0000000000000001 | 2 | 1 |   **flows\_s1-1\_0756823.json**  {  "priority": 50000,  "timeout": 0,  "isPermanent": true,  "deviceId": "of:0000000000000001",  "treatment": {  "instructions": [  {  "type": "OUTPUT",  "port": "2"  }  ]  },  "selector": {  "criteria": [  {  "type": "IN\_PORT",  "port": "1"  }  ]  }  }  **flows\_s1-2\_0756823.json**  {  "priority": 50000,  "timeout": 0,  "isPermanent": true,  "deviceId": "of:0000000000000001",  "treatment": {  "instructions": [  {  "type": "OUTPUT",  "port": "1"  }  ]  },  "selector": {  "criteria": [  {  "type": "IN\_PORT",  "port": "2"  }  ]  }  } |
|  |
| **Ans**:  如題目, 由topo.py新增s1, s2, s3, 且新增flow rules, 讓s1, s2, s3形成一個迴路, 當h1 ping h2, h1從IN\_PORT 向外送出ARP廣播封包 詢問h2在哪, 由於封包不斷在迴圈內進行傳送, 形成廣播風暴  CPU: 觀察cpu的狀況由原本50%使用率升高至75%~99%之間  Network: 再用wireshark 觀察ARP封包, ARP封包上升的速度非常快, 幾乎每秒可新增上萬筆封包紀錄  **Flow Rules,**   |  |  |  |  | | --- | --- | --- | --- | | **File name** | **deviceId** | **IN\_PORT** | **OUTPUT** | | flows\_s1-1\_0756823.json | of:0000000000000001 | 2 | 1 | | flows\_s1-2\_0756823.json | of:0000000000000001 | 3 | 1 | | flows\_s2-1\_0756823.json | of:0000000000000002 | 1 | 2 | | flows\_s3-1\_0756823.json | of:0000000000000003 | 1 | 2 |   C:\Users\Admin\Pictures\PicPick\20200323_154040.png  **Topo\_0756823.py**  from mininet.topo import Topo  class Project2\_Topo\_0756823( Topo ):  def \_\_init\_\_( self ):  Topo.\_\_init\_\_( self )  # Add hosts  h1 = self.addHost( 'h1')  h2 = self.addHost( 'h2')  # Add switches  s1 = self.addSwitch( 's1' )  s2 = self.addSwitch( 's2' )  s3 = self.addSwitch( 's3' )  # Add links  self.addLink( s1, s2 )  self.addLink( s2, s3 )  self.addLink( s1, s3 )  self.addLink( h1, s1 )  self.addLink( h2, s2 )  topos = { 'topo\_0756823': Project2\_Topo\_0756823 }  **flows\_s1-1\_0756823.json**  {  "priority": 50000,  "timeout": 0,  "isPermanent": true,  "deviceId": "of:0000000000000001",  "treatment": {  "instructions": [  {  "type": "OUTPUT",  "port": "1"  }  ]  },  "selector": {  "criteria": [  {  "type": "IN\_PORT",  "port": "2"  }  ]  }  }  **flows\_s1-2\_0756823.json**  {  "priority": 50000,  "timeout": 0,  "isPermanent": true,  "deviceId": "of:0000000000000001",  "treatment": {  "instructions": [  {  "type": "OUTPUT",  "port": "1"  }  ]  },  "selector": {  "criteria": [  {  "type": "IN\_PORT",  "port": "3"  }  ]  }  }  **flows\_s2-1\_0756823.json**  {  "priority": 50000,  "timeout": 0,  "isPermanent": true,  "deviceId": "of:0000000000000002",  "treatment": {  "instructions": [  {  "type": "OUTPUT",  "port": "2"  }  ]  },  "selector": {  "criteria": [  {  "type": "IN\_PORT",  "port": "1"  }  ]  }  }  **flows\_s3-1\_0756823.json**  {  "priority": 50000,  "timeout": 0,  "isPermanent": true,  "deviceId": "of:0000000000000003",  "treatment": {  "instructions": [  {  "type": "OUTPUT",  "port": "2"  }  ]  },  "selector": {  "criteria": [  {  "type": "IN\_PORT",  "port": "1"  }  ]  }  } |
|  |
| **Ans:**  **Data Plane,**   |  |  |  |  | | --- | --- | --- | --- | | **Packet** | **Layer** | **Src** | **Dst** | | OFPT\_PACKET\_IN | DataLink (Ethernet II) | 00:00:00:00:00:00 | 00:00:00:00:00:00 | | Network (IPV4) | 127.0.0.1 | 127.0.0.1 | | Transport (TCP/UDP) | **Port: 56406** | **Port: 6653** | | OFPT\_PACKET\_OUT | DataLink (Ethernet II) | 00:00:00:00:00:00 | 00:00:00:00:00:00 | | Network (IPV4) | 127.0.0.1 | 127.0.0.1 | | Transport (TCP/UDP) | **Port: 6653** | **Port: 56406** |   **Flow Rules**   |  |  |  | | --- | --- | --- | | **Priority** | **Match** | **Action** | | 40000 | ETH\_TYPE: bddp | Output: Controller | | 40000 | ETH\_TYPE: lldp | Output: Controller | | 40000 | ETH\_TYPE: arp | Output: Controller | | **5** | **ETH\_TYPE: ipv4** | **Output: Controller** |   當開啟 org.onosproject.fwd 後,  Flow planes: 在flow rules內新增一筆 **ETH\_TYPE: ipv4** && **Output:Controller**的紀錄  Data planes: 當h1 ping h2時, 可以發現OFPT\_PACKET\_IN的封包是透過TCP傳送資料至controller(6653), 再由controller(6653) 將封包OFPT\_PACKET\_OUT向Dst轉送 |