TRƯỜNG ĐẠI HỌC CÔNG NGHỆ - ĐẠI HỌC QUỐC GIA HÀ NỘI

Khoa Điện tử - Viễn thông

BÁO CÁO CUỐI KÌ

CHỦ ĐỀ: RYU CONTROLLER – QOS

Môn học : Mạng điều khiển mềm

Đỗ Mạnh Toàn – 20020250

Trần Phương Lâm – 20021546

Ninh Hải Bình – 20021495

Nguyễn Quang Khải – 20021544

Đỗ Mạnh Linh – 20021547

Vũ Huy Tùng - 20021598

TÓM TẮT: Nghiên cứu này giới thiệu và đánh giá các chức năng quan trọng trong Mạng điều khiển mềm (SDN) gồm chức năng Ryu, QoS và Traffic Engineering dựa trên QoS. Ryu là một controller SDN mã nguồn mở được phát triển tích hợp cho các ứng dụng quản lý mạng trong môi trường SDN. Nó cung cấp một nền tảng linh hoạt để phát triển các ứng dụng và dịch vụ SDN, hỗ trợ giao thức OpenFlow và các giao thức khác để quản lý các thiết bị mạng, đồng thời nâng cao tính tự động hóa và mở rộng của hệ thống mạng. Bên cạnh đó QoS là một trong những yếu tố quan trọng nhằm đảm bảo hiệu suất, độ tin cậy và ổn định của mạng. Trong SDN, QoS giúp kiểm soát và phân bổ tài nguyên mạng như băng thông, độ trễ, chất lượng cuộc gọi, v.v... Nó giúp kiểm soát và ưu tiên cho các dòng dữ liệu khác nhau dựa trên nhu cầu, chất lượng và giảm độ trễ dữ liệu trong mạng, tối ưu hóa hiệu suất tổng thể. Còn Traffic Engineering là quá trình điều hướng và phân bổ tài nguyên mạng để đáp ứng các yêu cầu về hiệu suất và đảm bảo QoS.

Keyword: SDN, QoS, Traffic Engineering, Traffic shaping, Traffic policing, Ryu Controller, OVS, INSERV, DIFFSERV

MỤC LỤC

Các hình vẽ và bảng

I. Giới thiệu

1.1. Mục tiêu nghiên cứu

Mục tiêu của nghiên cứu về chủ đề Ryu controller và QoS là tìm hiểu và áp dụng các kỹ thuật điều khiển và quản lý chất lượng dịch vụ (QoS) trong mạng SDN (Software-Defined Networking) sử dụng Ryu controller. Mạng SDN là một kiến trúc mạng được phát triển gần đây với nhiều ưu điểm, bao gồm tính linh hoạt, dễ dàng quản lý, cập nhật và giám sát. Trong mạng SDN, Ryu controller được sử dụng để điều khiển và quản lý các thiết bị mạng.

1.2. Phương pháp nghiên cứu

Tìm hiểu và đánh giá các công cụ và kỹ thuật điều khiển mạng SDN hiện có, đặc biệt là Ryu controller. Nghiên cứu các phương pháp đánh giá QoS trong mạng SDN, bao gồm cách đo lường các thông số về băng thông, độ trễ và độ tin cậy của mạng. Xây dựng mô hình mạng SDN sử dụng Ryu controller, bao gồm cấu hình các switch và flow entries để hỗ trợ QoS. Thực hiện thử nghiệm và đánh giá hiệu suất của hệ thống, bao gồm các thông số về QoS, tốc độ truyền dữ liệu, độ trễ và độ tin cậy

1.3. Đóng góp của nghiên cứu

Nghiên cứu về Ryu Controller và QoS đóng góp rất lớn cho việc phát triển các hệ thống mạng thông minh hiệu quả. Việc áp dụng QoS trong mạng là rất quan trọng để đảm bảo chất lượng dịch vụ cho người dùng và tối ưu hóa sự sử dụng tài nguyên mạng. Phương pháp nghiên cứu sử dụng trong việc thực hiện nghiên cứu này có thể bao gồm việc sử dụng các kỹ thuật mô phỏng, thực nghiệm trên mạng thực tế, hoặc sử dụng các công cụ phân tích dữ liệu để phân tích hiệu quả của hệ thống QoS. Tùy thuộc vào mục tiêu cụ thể của nghiên cứu, phương pháp nghiên cứu có thể được lựa chọn sao cho phù hợp nhất. Kết quả của nghiên cứu có thể giúp các nhà quản trị mạng và các nhà phát triển ứng dụng hiểu rõ hơn về cách triển khai QoS trên mạng và đảm bảo chất lượng dịch vụ tốt nhất cho người dùng. Nghiên cứu này cũng có thể đóng góp cho việc phát triển các hệ thống mạng thông minh và cải thiện hiệu suất của chúng.

II. Chi tiết báo cáo nghiên cứu

2.1. Những nghiên cứu liên quan

Trong phần này, nhóm chúng em sẽ nói về các nghiên cứu trước đây và các nghiên cứu đã được thực hiện trong Ryu Controller: QoS. Các nghiên cứu này

chứng minh khả năng của Ryu như một bộ điều khiển Mạng điều khiển mềm (SDN) để quản lý tài nguyên mạng và cải thiện Chất lượng dịch vụ (QoS) cho nhiều ứng dụng mạng khác nhau:

Nghiên cứu [1] nêu cách áp dụng QoS trong SDN và đề cập đến việc tích hợp các tính năng QoS trong Ryu. So sánh hiệu năng của Ryu controller với các bộ điều khiển SDN khác trong việc áp dụng QoS. Các kịch bản hoạt động và kết quả thử nghiệm được đạt được từ việc áp dụng QoS trong SDN thông qua Ryu controller.

Nghiên cứu này [2] nói về nói về việc đánh giá hiệu năng của các chỉ số trong SDN sử dụng Ryu Controller. Bài báo đánh giá và so sánh hiệu năng của các chỉ số QoS như băng thông, độ trễ, tổn thất gói dữ liệu, v.v. trong mạng SDN. Nghiên cứu còn mô tả các phương pháp thí nghiệm được sử dụng để đánh giá hiệu năng của các chỉ số QoS và thảo luận về kết quả thu được, giúp người đọc hiểu được ưu nhược điểm của từng chỉ số trong SDN, đặc biệt khi sử dụng Ryu Controller.

Mục đích của nghiên cứu [3] là để cải thiện QoS trong mạng SDN thông qua việc tối ưu hóa công tác định tuyến và quản lý lưu lượng. Các tác giả đã sử dụng cơ chế Hierarchical Token Bucket (HTB) để phân loại và giới hạn lưu lượng truy cập trong hệ thống giúp cân bằng tài nguyên mạng giữa các ứng dụng, người dùng và dịch vụ một cách hiệu quả.Với Ryu Controller, người dùng có thể dễ dàng tùy chỉnh và quản lý chính sách định tuyến, quyết định định tuyến dựa trên nhu cầu QoS của ứng dụng và người dùng, hỗ trợ triển khai và tích hợp cơ chế HTB vào hệ thống SDN.

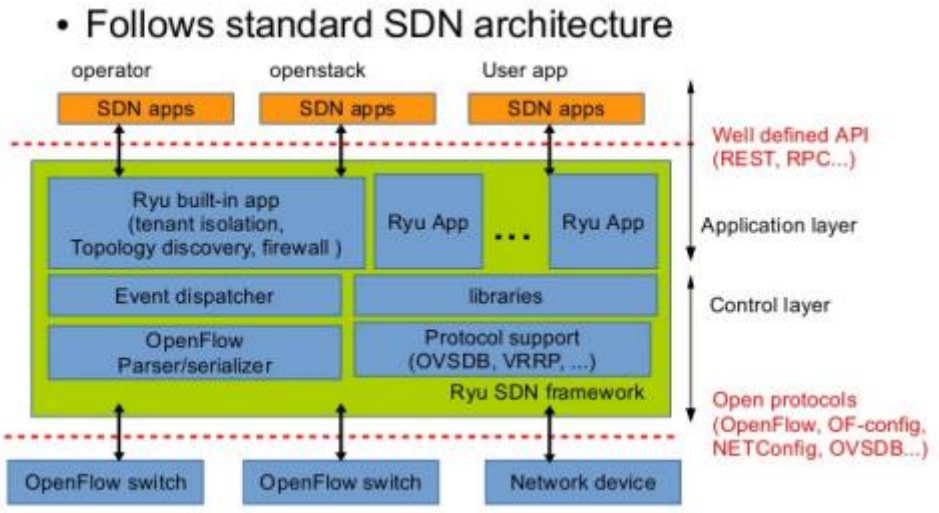
Tóm lại, các nghiên cứu này đều tập trung vào việc đánh giá, tối ưu và cải thiện chất lượng dịch vụ (QoS) trong mạng Software Defined Networking (SDN) sử dụng Ryu Controller. Họ xem xét các chỉ số QoS như băng thông, độ trễ, tổn thất gói dữ liệu và áp dụng các cơ chế phân loại và giới hạn lưu lượng truy cập để cân bằng tài nguyên mạng và đảm bảo QoS cho các ứng dụng, người dùng và dịch vụ. Các nghiên cứu này đưa ra các phương pháp thử nghiệm và so sánh hiệu năng của các bộ điều khiển SDN, đưa ra những ưu và nhược điểm của Ryu Controller trong việc áp dụng QoS trong SDN.

2.2. Ryu Controller

2.2.1: Tổng quan

Ryu Controller là một SDN Controller mở, được thiết kế để tăng tính linh hoạt của mạng bằng cách giúp dễ dàng quản lý và điều chỉnh cách xử lý lưu lượng. Nói chung, SDN Controller là bộ não của môi trường SDN, truyền thông tin xuống các Switches và Routers với các Southbound API, và đến các ứng dụng và logic nghiệp vụ với các Northbound API. Ryu Controller được NTT(Nippon Telegraph

and Telephone Corporation) hỗ trợ và cũng được triển khai trong các trung tâm dữ liệu đám mây của NTT [4].



Hình 1. Kiến trúc Ryu SDN Controller

Ryu Controller cung cấp các thành phần phần mềm, với các giao diện chương trình ứng dụng (API) được xác định rõ ràng, giúp các nhà phát triển dễ dàng tạo các ứng dụng điều khiển và quản lý mạng mới. Cách tiếp cận thành phần này giúp các tổ chức tùy chỉnh triển khai để đáp ứng các nhu cầu cụ thể của họ; các nhà phát triển có thể nhanh chóng và dễ dàng sửa đổi các thành phần hiện có hoặc triển khai thành phần của riêng họ để đảm bảo mạng bên dưới có thể đáp ứng nhu cầu thay đổi của ứng dụng của họ.

- Ryu Libraries

Ryu hỗ trợ một số thư viện và nhiều giao thức southbound. Liên quan đến các giao thức southbound, Ryu cũng hỗ trợ Giao thức quản lý cơ sở dữ liệu Open vSwitch (OVSDB), OF-Config, NETCONF, Sflow, Netflow và các giao thức bên thứ ba khác. Các giao thức Sflow và Netflow có thể được sử dụng để đo lưu lượng mạng bằng cách sử dụng các phương pháp khác nhau như lấy mẫu và tổng hợp gói. Các thư viện của bên thứ ba bao gồm liên kết Open vSwitch Python, thư viện cấu hình Oslo và thư viện Python cho NETCONF client. Với sự trợ giúp của thư viện của Ryu, nhà phát triển mạng có thể phân tích và xây dựng một số gói giao thức, như VLAN, MPLS, v.v.

- OpenFlow Protocol and controller

Khung Ryu bao gồm bộ điều khiển nội bộ và giao thức OF, một trong những giao thức southbound được hỗ trợ. Ryu hỗ trợ giao thức OpenFlow từ phiên bản 1.0 đến phiên bản 1.4. Bảng dưới đây tóm tắt các thông báo giao thức OpenFlow và API tương ứng của bộ điều khiển Ryu.

Controller to switch message	Asynchronous message	Symmetric message	Structures
• Handshake • switch-config • flow-table-config modify/read state • queue-config • packet-out, barrier • role-request	• Packet-in • flow-removed port-status • Error.	• Hello • Echo-Request & Reply • Error • experimenter	• Flow-match
• send_msg API and packet builder APIs	• set_ev_cls API and packet parser APIs	• Both Send and Event APIs	

Trong kiến trúc Ryu, bộ điều khiển OpenFlow là một trong những nguồn sự kiện nội bộ có thể quản lý các switch và sự kiện. Ngoài ra, Ryu còn bao gồm một thư viện giải mã và mã hóa giao thức OpenFlow.

- Managers and Core-processes

Thành phần thực thi chính trong kiến trúc Ryu là trình quản lý Ryu. Trong thời gian chạy, trình quản lý Ryu tạo một trình nghe có thể kết nối với các Switch OpenFlow. Khi nó được chạy, nó sẽ lắng nghe địa chỉ IP được chỉ định và cổng được chỉ định (6633 theo mặc định). Sau đó, bất kỳ Switch OpenFlow nào cũng có thể kết nối với trình quản lý Ryu.

Trình quản lý ứng dụng là một trong những thành phần chính cho tất cả các ứng dụng Ryu vì chúng cần kế thừa chức năng từ lớp RyuApp của Trình quản lý ứng dụng. Thành phần quy trình cốt lõi trong kiến trúc bao gồm nhắn tin, quản lý sự kiện, quản lý trạng thái trong bộ nhớ, v.v. Trong kiến trúc Ryu, giao diện lập trình ứng dụng (API) Northbound được minh họa ở lớp trên cùng, nơi các phần bổ trợ được hỗ trợ có thể giao tiếp với Ryu's Hoạt động của OF.

- Ryu Northbound API

Ở lớp API, Ryu hỗ trợ rộng rãi giao diện REST cho các hoạt động OpenFlow của nó. Ryu cũng bao gồm một plug-in Openstack Neutron hỗ trợ cả cấu hình lớp phủ dựa trên VLAN và GRE. Ngoài ra, nhà nghiên cứu có thể dễ dàng tạo API REST bằng cách sử dụng khung kết nối máy chủ web và ứng dụng bằng Python có tên là WSGI.

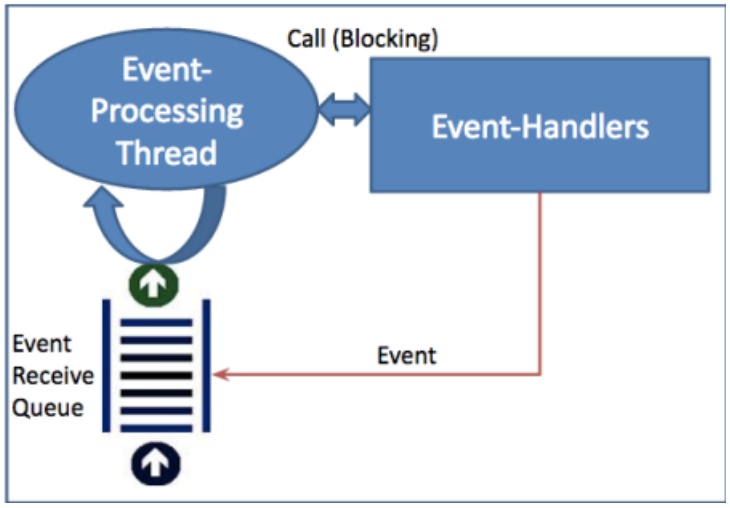
- Ryu Application

Ứng dụng Ryu là một trong những yếu tố thiết yếu vì logic điều khiển và hành vi được xác định trong đó. Nhiều ứng dụng đã được bao gồm trong Ryu framework chẳng hạn như cấu trúc liên kết, Simple_switch, tường lửa, bộ định tuyến, v.v. Mặc dù các ứng dụng Ryu được triển khai và cung cấp nhiều chức năng khác

nhau, nhưng chúng hoạt động như các thực thể đơn luồng. Trước đây, các ứng dụng Ryu gửi các sự kiện không đồng bộ cho nhau.

Mỗi ứng dụng Ryu thường có hàng đợi nhận riêng cho các sự kiện có thể xảy ra, đặc biệt là FIFO để duy trì đúng thứ tự điều hành của các sự kiện. Hơn nữa, mỗi ứng dụng thường bao gồm một luồng để xử lý đúng các sự kiện từ hàng đợi. Vòng lặp chính của luồng đưa ra các sự kiện từ hàng đợi nhận và gọi bộ xử lý sự kiện phù hợp. Do đó, trình xử lý sự kiện được gọi một cách tự nhiên trong ngữ cảnh của chuỗi xử lý sự kiện, hoạt động theo kiểu chặn, nghĩa là khi một trình xử lý sự kiện được cấp quyền quản lý, sẽ không có sự kiện bổ sung nào cho ứng dụng Ryu được xử lý cho đến khi quá trình quản lý được thực hiện trả lại.

Kiến trúc chức năng của ứng dụng Ryu được giới thiệu trong hình dưới đây:



2.2.2 Ryu Controller phù hợp như thế nào trong môi trường SDN

Mã nguồn của Ryu Controller được lưu trữ trên GitHub và được quản lý và duy trì bởi cộng đồng Ryu mở. OpenStack, chạy cộng tác mở tập trung vào phát triển hệ điều hành đám mây có thể kiểm soát tài nguyên điện toán, lưu trữ và kết nối mạng của một tổ chức, hỗ trợ triển khai Ryu với tư cách là Network Controller.

Được viết hoàn toàn bằng Python, tất cả mã của Ryu đều có sẵn theo giấy phép Apache 2.0 và mở cho mọi người sử dụng. Ryu Controller hỗ trợ các giao thức quản lý mạng NETCONF và OF-config, cũng như OpenFlow, một trong những tiêu chuẩn truyền thông SDN đầu tiên và được triển khai rộng rãi nhất. Nó cũng hỗ trợ các tiện ích mở rộng của Nicira.

Quản trị viên IT viết các ứng dụng cụ thể giao tiếp với Ryu Controller về cách quản lý Switches và Routers. Ryu Controller có thể sử dụng OpenFlow hoặc các giao thức khác để tương tác với Forwarding Plane(Switches và Routers) nhằm sửa đổi cách mạng sẽ xử lý các luồng lưu lượng. Nó đã được thử nghiệm và chứng nhận để hoạt động với một số Openflow Switches, bao gồm Open vSwitch(OVS) và các dịch vụ từ Centec, Hewlett Packard, IBM và NEC.

2.3 Chức năng của QoS trong SDN

2.3.1. QoS Model

QoS có thể được phân loại thành hai mô hình chính là:

• Dịch vụ tích hợp (Integrated Service)

• Dịch vụ khác biệt (Differentiated Service)

a, Dịch vụ Tích hợp (Integrated Service)

Cisco network xác định Dịch vụ Tích hợp (còn được gọi là IntServ) là một mô hình dịch vụ đa năng có thể đáp ứng nhiều yêu cầu QoS khác nhau, trong đó ứng dụng yêu cầu một loại dịch vụ cụ thể từ mạng trước khi gửi dữ liệu. Yêu cầu cho dịch vụ có thể xử lý yêu cầu về băng thông và độ trễ bởi ứng dụng cần được hoàn thành và xác nhận trước khi dữ liệu có thể được truyền trên mạng. Với việc triển khai QoS IntServ, mạng được cấu hình để duy trì trạng thái theo dòng lưu lượng (per flow state) trong các thiết bị mạng và sau đó sử dụng các công cụ QoS như phân loại (classification), đánh dấu (marking), kiểm soát lưu lượng (policing), tạo hình (shaping), xếp hàng (queuing) và lập lịch (scheduling) để xử lý các gói tin.

IntServ sử dụng giao thức đăng ký tài nguyên (Resource Reservation Protocol - RSVP) để đặt trước băng thông và độ trễ yêu cầu trên mạng cho mỗi phiên kết nối từ một điểm cuối đến điểm cuối khác. Tuy nhiên, vấn đề chính của IntServ là yêu cầu các thay đổi cơ bản trong hạt nhân của mạng do sử dụng trạng thái theo dòng lưu lượng trong mạng. Số lượng các luồng yêu cầu tăng dẫn đến việc cấu hình mạng trở nên phức tạp và đắt đỏ cho các thiết bị định tuyến để cung cấp IntServ. Để giải quyết vấn đề này, đã tạo ra một công nghệ mới được gọi là Dịch vụ Khác biệt (Differentiated Service - DiffServ).

b, Dịch vụ Khác biệt (Differentiated Service - DiffServ)

DiffServ sử dụng mô hình dịch vụ đa năng cũng có thể đáp ứng và đáp ứng yêu cầu QoS khác nhau. Khác với IntServ, DiffServ cung cấp dịch vụ có tính mở rộng mà không cần đăng ký tài nguyên và trạng thái theo dòng lưu lượng. Điều này có nghĩa là nó không cần xác nhận thông tin về băng thông hoặc độ trễ từ các định tuyến trước khi gửi gói tin. DiffServ chỉ xử lý các định tuyến cạnh và không cần cấu hình phức tạp tại core, mặc dù định tuyến cạnh phải có thể chuyển tiếp các gói tin bằng cách sử dụng các ưu tiên và phân loại của mỗi gói tin.

Một trong những cách để thiết lập QoS trong DiffServ bao gồm cài đặt bit trong tiêu đề IP của địa chỉ nguồn và đích. DiffServ cũng sử dụng các công cụ QoS như phân loại, đánh dấu, giám sát, hình dạng và hàng đợi thông minh. Khác với IntServ, mô hình DiffServ khả thi hơn trong tổ chức và hiệu quả chi phí hơn. Nó cũng phục vụ tốt hơn trong cung cấp ứng dụng quan trọng cho nhiệm vụ hơn

IntServ. DiffServ sử dụng Điểm Mã Dịch vụ Khác biệt (Differentiated Service Code Point - DSCP) để đánh dấu trên tiêu đề IP của mỗi gói tin, điều này có thể được áp dụng bởi quản trị mạng trên các định tuyến cạnh. Việc đánh dấu và phân loại DSCP loại bỏ chi phí chờ đợi và xác nhận băng thông của IntServ trước khi hoạt động.

2.3.2. Tóm lược về công cụ QoS

Bộ công cụ QoS bao gồm các công cụ phân loại, công cụ đánh dấu, công cụ kiểm soát và công cụ định hình. Một lời giải thích ngắn gọn về mỗi danh mục bộ công cụ được đưa ra dưới đây:

1. **Phân loại (Classification)** là thuật ngữ dùng cho việc phân tích flow để xác định loại lưu lượng mà flow thuộc về và đưa ra quyết định dựa trên kết quả của việc phân tích. Quyết định được thực hiện được gọi là đánh dấu(marking).

2. **Việc đánh dấu(making)** được thực hiện sau khi các flow đã được phân tích, các gói (packets) sau đó được đánh dấu tại bộ định tuyến xâm nhập cạnh vào trong mạng lưới.

3. **Kiểm soát (policing)** là hành động drop gói bất cứ khi nào tài nguyên mạng được phân bổ đã bị vượt quá.

4. **Định hình (shaping)** liên quan đến việc làm chậm lưu lượng trên mạng để tối đa hóa việc sử dụng mạng được phân bổ tài nguyên. Không giống như kiểm soát, định hình sẽ làm chậm quá trình truyền hoặc nhận gói thay vì drop gói. Việc làm chậm lưu lượng được thực hiện chủ yếu bằng cách xếp hàng đợi các gói và sau đó chuyển tiếp chúng khi kết thúc.

2.3.3. Tổng quan về Traffic Shaping và Traffic Policing

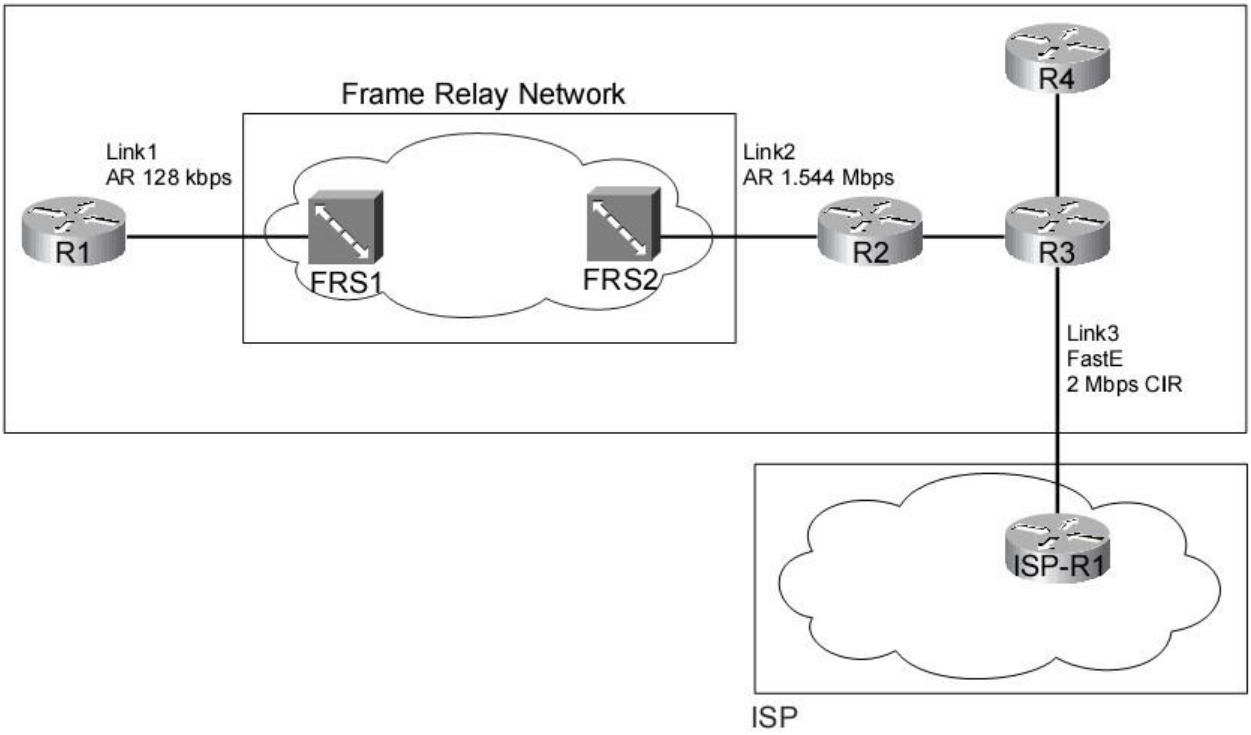
Traffic Shaping (TS) và Traffic Policing (TP) là một phần quan trọng của chất lượng dịch vụ QoS. TS và TP dùng để đo tốc độ truyền hoặc nhận dữ liệu. TS là công cụ điều hòa lưu lượng, giúp cho các gói tin được gửi theo đúng tốc độ đã cấu hình. Nói cách khác, khi gói tin đi ra khỏi router thì tốc độ truyền tổng thể của gói tin này không vượt qua một giới hạn đã định nghĩa. Để đảm bảo tốc độ truyền của các gói tin không vượt qua giới hạn đã định nghĩa, TS làm giảm tốc độ của các gói tin này bằng cách đưa chúng vào các hàng đợi điều hòa (các hàng đợi này khác với các hàng đợi mềm trên các cổng của router). Sau đó, router mới tiếp tục gửi các gói tin trên theo đúng tốc độ đã định nghĩa.

TP là công cụ khống chế lưu lượng. TS đo tốc độ truyền của gói khi gói tin vào và ra một cổng của router. Nếu tốc độ truyền vượt quá tốc độ đã cấu hình, router sẽ loại bỏ đủ số lượng gói tin sao cho tốc độ đã cấu hình không bị vượt qua. Hoặc router cũng có thể đánh dấu các gói tin sao cho các gói tin này có thể bị loại bỏ về sau. Ví dụ, ISP cung cấp dịch vụ và cho khách hàng có thể gửi các gói tin

với một tốc độ x (tốc độ cam kết giữa ISP và khách hàng). Tuy nhiên, nếu khách hàng gửi các gói tin với tốc độ 10x thì TP sẽ làm rớt các gói tin đó.

TS ngăn chặn khách hàng sử dụng vượt quá bandwidth cho đã thỏa thuận (kể cả trong trường hợp ISP vẫn đủ khả năng quản lý những lưu lượng này).

TS và TP thường xảy ra ở biên giữa giữa 2 loại mô hình mạng. Ví dụ, xét mô hình mạng sau:



Hình 2: Ví dụ mô hình mạng minh họa cho Policing và Shaping

Theo hình 2, mình họa mô hình mạng của một công ty A. TS và TP xảy ra ở các biên mạng giữa:

- R1 với FRS1 (link1).

- FRS2 với R2 (link 2).

- R3 với ISP-R1 (link 3).

Trong các biên mạng trên thì đều có một sự thõa thuận về tốc độ truyền dữ liệu giữa công ty A với dịch vụ Frame Relay và ISP. R2 sử dụng đường truyền 1.544Mbps (T1) và R1 sử dụng đường truyền 128Kbps tới mạng Frame Relay (FRS2 và FRS1). Tuy nhiên, tốc độ CIR giữa R1 và R2 có thể chỉ là 64Kbps. Điều tương tự khi R3 kết nối với ISP bằng đường fastEthernet (100Mbps).

Trong khi đó, ISP chỉ cung cấp đường truyền 2Mbps theo thõa thuận. R1, R2, R3 sẽ gửi dữ liệu ra một cổng với tốc độ lần lượt là 128Kbps, 1.544Mbps, 100Mbps (đúng với tốc độ vật lý). Do đó, việc gửi các gói tin vượt quá tốc độ cho phép xảy ra. Lúc này, có thể sẽ làm tăng độ delay và packet loss. TS và TP sẽ giúp

ngăn chặn các hiện tượng nêu trên. Việc sử dụng TS và TP được thể hiện cụ thể trong bảng sau:

	Traffic Policing	**Traffic Shaping**
Lý do sử dụng	Nếu một mạng gửi gói tin vượt quá giới hạn cho phép. TP được sử dụng để đảm bảo dữ liệu được gửi đúng tốc độ thõa thuận bằng cách đánh rớt các gói tin. Từ đó, ngăn chặn sự quá tải với lưu lượng truy cập quá nhiều.	Lý do đầu tiên sử dụng TS là do TP. Thay vì để TP đánh rớt các gói tin thì TS có thể làm chậm và đưa các gói tin vào hàng đợi đảm bảo các gói tin này đi ra khỏi hàng đợi với tốc độ đã cấu hình từ trước. Lý do thứ hai là để giải quyết hiện tượng nghẽn ngõ ra (egress blocking). Hiện tượng nghẽn ngõ ra xảy ra khi router gửi dữ liệu vào một mạng Frame Relay hoặc mạng ATM. Hiện tượng nghẽn ngõ ra thường liên quan đến độ delay và packet loss. Với TS, delay và packet loss có thể tránh được hoặc giảm thiểu đến mức nhỏ nhất. Bằng cách đưa các gói tin này vào hàng đợi mềm của thiết bị cung cấp dịch vụ (Frame Relay Switch hoặc ATM router) trước khi gửi đi. Bởi vì các kỹ thuật hàng đợi mềm được cấu hình tại router cung cấp QoS tốt hơn.
Vị trí sử dụng.	Thông thường, TP thực hiện chức năng của mình khi có gói dữ liệu đi vào thiết bị biên của một mạng. Khi các gói dữ liệu đi ra khỏi mạng đó, TP cũng được hỗ trợ nhưng ít hơn.	TS là luôn thực hiện chức năng của mình khi có gói dữ liệu được chuyển theo chiều đi ra. Thông thường, TS được thực hiện trên các gói dữ liệu ra khỏi một router và đi vào một mạng khác. Nơi này có thể là biên giữa router đó và một mạng WAN multiaccess, hoặc có thể là một liên kết đến ISP.

Bảng 1: Vị trí và Cách sử dụng TP và TS

2.3.4. Triển khai QoS trong SDN

Như trong mạng truyền thống, QoS cũng có thể được thực hiện trong SDN. SDN sử dụng giao thức OpenFlow và OpenvSwitch (OVS) để thực hiện. OVS hỗ trợ kiểm soát lưu lượng truy cập Linux (tc) mà nó sử dụng để giới hạn tốc độ. Nó cũng sử dụng Linux queueing disciplines (qdisc) để quản lý băng thông và Hierarchical Token Bucket (HTB). HTB giúp kiểm soát băng thông ra ngoài trên một liên kết nhất định. Khi QoS được triển khai, HTB đảm bảo mỗi hàng đợi trong cài đặt QoS có một số tài nguyên được phân bổ.

Hai cách chính để sử dụng QoS với OVS trong SDN là kiểm soát và định hình. OVS hỗ trợ định hình lưu lượng truy cập lối ra từ một switch. Lưu lượng có thể ở dạng giá trị DSCP được phân bổ cho mỗi hàng đợi trên QoS. OVS cũng hỗ trợ kiểm soát lưu lượng truy cập xâm nhập vào switch. Tóm lại, SDN sử dụng OVS để triển khai QoS bằng cách lập chính sách kiểm soát dựa trên giới hạn tốc độ xâm nhập trên mỗi giao diện bằng cách sử dụng qdisc hoặc bằng định hình dựa trên hàng đợi và mức độ ưu tiên trên mạng liên kết sử dụng HTB.

2.4. Traffic Engineering dựa trên QoS trong SDN

2.4.1. Một vài lý thuyết nền tảng

a, Openflow

OpenFlow là một giao thức truyền thông mạng phổ biến được sử dụng trong Sự định tuyến phần mềm và mảng mạng định tuyến (SDN). Nó được phát triển bởi ONF (Open Networking Foundation) với mục tiêu đơn giản hóa việc cấu hình và quản lý mạng, giảm thiểu chi phí và tạo điều kiện cho sự linh hoạt và sáng tạo trong việc triển khai các ứng dụng mạng.

OpenFlow hoạt động bằng cách phân tách luồng truyền thông giữa tầng quản lý điều khiển (Control Plane) và tầng chuyển mạch truyền tin (Data Plane) trong thiết bị mạng như bộ định tuyến, công tắc mạng. Nhờ đó, điều khiển mạng có thể tập trung vào một máy tính, giúp quản lý hệ thống hiệu quả hơn.

Bảng luồng dữ liệu OpenFlow (OpenFlow Flow Table)

Bảng luồng dữ liệu là một bảng tra cứu có các trường tương ứng và hành động và được xử lý như pipeline. Khi khung dữ liệu vào cổng, nó được xử lý bởi Bảng 0 bằng mục nhập luồng tương ứng có độ ưu tiên cao nhất. Mục nhập luồng này sẽ chứa một tập hành động có thể xuất khung dữ liệu đến một cổng cụ thể, áp dụng các hành động hoặc gửi khung dữ liệu đến một bảng khác. Trong trường hợp bảng không tìm thấy tương ứng, khung dữ liệu sẽ bị bỏ qua bởi switch. Bảng luồng dữ liệu bao gồm các trường sau.

Loại	Mô tả
Match Fields	Tiêu chí khớp cho các khung (frames). Bao gồm dữ liệu header và thông tin metadata. Các trường khớp được đặt trên bảng luồng (flow table) để xác định gói tin mà một hành động sẽ được thực hiện. Điều này bao gồm thông tin 5-tuple và một số tiêu chí bổ sung khác cũng có thể được sử dụng.
Độ ưu tiên (Priority)	Độ ưu tiên cho tiêu chí khớp. Các khớp xảy ra theo thứ tự độ ưu tiên. Hữu ích cho việc định nghĩa các mục ngoại lệ và các mục mặc định trong đường ống bảng luồng.
Bộ đếm (Counters)	Đếm số lượng khớp
Hướng dẫn (Instructions)	Xác định những gì sẽ được thực hiện với khung sau khi khớp; có một hoặc nhiều hướng dẫn này.
Thời gian chờ (Timeouts)	Xác định thời gian một luồng có thể tồn tại trong switch. Thời gian chờ mềm xác định bao lâu luồng sống nếu không có khung khớp. Thời gian chờ cứng xác định bao lâu luồng sống bất kể số lượng khớp.
Cookie	Trường được định nghĩa bởi controller. Không được sử dụng trong xử lý gói tin nhưng hữu ích để lọc thống kê luồng.

Bảng 2: Bảng luồng dữ liệu OF

b, SDN Controller cho mặt phẳng điều khiển (SDN Controller for the Control Plane)

Trong kiến trúc SDN, bộ điều khiển hoạt động như não của mạng và nó là nơi chứa Mặt điều khiển như được miêu tả trong Hình 3.1. Bộ điều khiển là một phần mềm đóng vai trò là trung tâm điều khiển tập trung giám sát mạng và qua đó các ứng dụng có thể truy cập và quản lý mạng. Khi nói rằng bộ điều khiển là trung tâm của mạng, điều này chỉ có ý nghĩa tập trung về mặt logic. Phần mềm bộ điều khiển thường được triển khai trên một máy chủ có hiệu suất cao, nhưng để phân phối tải hoặc đảm bảo tính khả dụng cao và sự bền vững, có thể có nhiều máy chủ được kết nối với nhau theo các topo khác nhau.

Bộ điều khiển có trách nhiệm thực hiện các nhiệm vụ sau:

• Phát hiện thiết bị: bộ điều khiển quản lý việc phát hiện các switch và thiết bị người dùng cuối và quản lý chúng.

• Theo dõi topologia mạng: bộ điều khiển khám phá các liên kết nối các thiết bị trong mạng và duy trì một cái nhìn về các tài nguyên cơ bản.

• Quản lý luồng: bộ điều khiển duy trì một cơ sở dữ liệu phản ánh các mục luồng được cấu hình trong các switch nó quản lý.

• Theo dõi thống kê: bộ điều khiển thu thập và lưu trữ các thống kê theo mức luồng từ các switch.

Cần nhấn mạnh rằng bộ điều khiển không điều khiển mạng theo bất kỳ cách nào và cũng không thay thế bất kỳ thiết bị mạng nào. Ngay cả chức năng chuyển mạch hoặc định tuyến cơ bản cũng phải được cung cấp bởi các ứng dụng cụ thể sử dụng bộ điều khiển để tiếp cận mạng. Giao tiếp với các thiết bị mạng được thực hiện thông qua một southbound interface, trong đó Open SDN khuyến khích sử dụng giao thức OpenFlow. Các giao diện này được sử dụng để cấu hình và quản lý các switch và để nhận thông điệp từ chúng. Kết nối được thực hiện thông qua một kênh an toàn và tùy thuộc vào cài đặt, có thể được mã hóa hoặc không được mã hóa.

Ứng dụng giao tiếp với bộ điều khiển thông qua northbound interface. Qua giao diện này, chúng lấy thông tin về mạng và gửi các yêu cầu của mình, trong khi bộ điều khiển sử dụng nó để chia sẻ thông tin về các sự kiện xảy ra. Tùy thuộc vào việc triển khai, giao diện có thể là cấp thấp, cung cấp truy cập thống nhất vào từng thiết bị riêng lẻ, hoặc là cấp cao, trừu tượng hóa phần lớn lớp bên dưới và đại diện cho mạng như một tổng thể. Hiện nay không có tiêu chuẩn cho northbound interfacevà mỗi bộ điều khiển triển khai API riêng của mình - có thể là Java API, Python API, REST API hoặc bất cứ cái gì khác. Thiếu tiêu chuẩn

northbound interface này hiện nay làm cho việc tạo các ứng dụng độc lập với bộ điều khiển trở lên khó khăn.

+) Software Switch cho Data Plane

Bảng 3 tóm tắt các lệnh Software Switch mà được sử dụng để thử nghiệm và phát triển dịch vụ mới.

Switch	Ngôn ngữ sử dụng	Mô tả
Open vSwitch	C/Python	Ngăn xếp OpenFlow được sử dụng như một Switch ảo và được chuyển sang nhiều nền tảng phần cứng
Ofsoftswitch13	C/C++	Thực hiện chuyển đổi phần mềm không gian người dùng

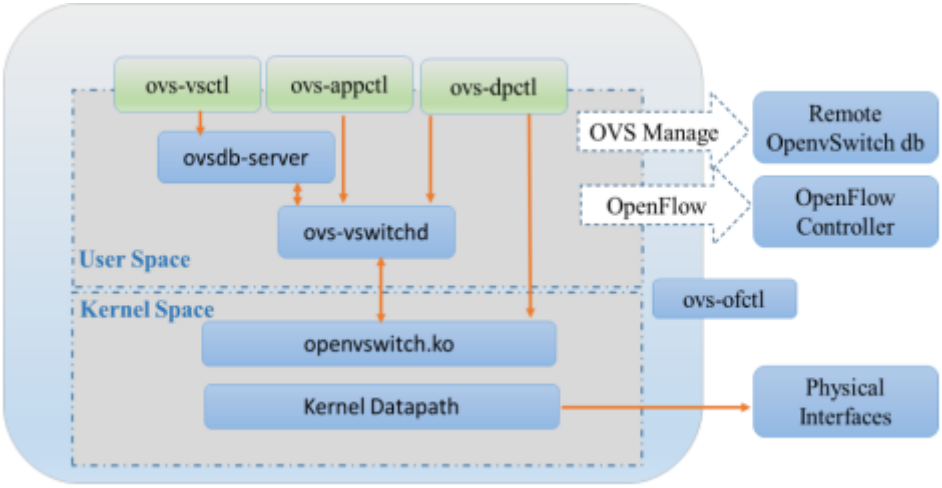
Bảng 3 OpenFlow Software Switches phổ biến

+) Open vSwitch

OpenFlow có thể được triển khai ở cấp độ phần mềm hoặc cấp độ phần cứng trên các thiết bị chuyển tiếp trong data plane. Cụ thể hơn, nhiều nhà cung cấp mạng nổi tiếng như Cisco, Juniper, IBM và HP, hỗ trợ OpenFlow, bằng sản phẩm chuyên dụng hoặc chạy một Openflow software switch trên các switches của họ. Open vSwitch là một trong những software switches được triển khai để hỗ trợ OpenFlow, có thể được cài đặt để kích hoạt OpenFlow.

Open vSwitch là một software switch đa lớp nhằm hoạt động như một switch ảo. Open vSwitch hỗ trợ tất cả các phiên bản OpenFlow từ 1.0 đến 1.5 cũng như GRE tunneling, hàng đợi, v.v. Cốt lõi của Open vSwitch là daemon chuyển đổi (ovs-vswitchd). Daemon này theo dõi các truy vấn thống kê và quản lý luồng nội bộ trên switch, đồng thời xử lý giao tiếp với các thiết bị và dịch vụ bên ngoài. Để quản lý và cấu hình, song song với khả năng kết nối với Openflow controller , có thể cấu hình và điều khiển Open vSwitch thông qua ovs-vsctl và ovs-ofctl. Ovs-vsctl là công cụ dòng lệnh để định cấu hình ovs vswitchd bằng cách cung cấp giao diện cho cơ sở dữ liệu cấu hình của nó, trong khi ovs-ofctl là công cụ dòng lệnh để giám sát và quản trị Open vSwitch.

Hơn nữa, màn hình ovsdb là một công cụ để xem các Flow Table và databases của Open vSwitch. Như được minh họa trong Hình 3 ovsdb-server dựa vào giao thức Quản lý OVS để giao tiếp với database Open vSwitch từ xa (databases được duy trì bởi Open vSwitch để lưu trữ thông tin của nó). Không giống như các flow entries trong switch cấu hình OVSDB được giữ nguyên ngay cả sau khi Switch khởi động lại.



Hình 3. Kiến trúc Open vSwitch đơn giản

+) OfSoftSwitch (CPqD)

OfSoftSwitch13 (CPqD) là một switch khác được sử dụng rộng rãi trong cộng đồng nghiên cứu. Đây là một switch thử nghiệm được phân nhánh từ triển khai SoftSwitch của Ericsson Traffic với những thay đổi trong forwarding plane để hỗ trợ OF1.3. Ofsoftswitch13 đang chạy trong không gian người dùng và nó cũng hỗ trợ nhiều phiên bản OpenFlow.

Ofsoftswitch13 hỗ trợ nhiều tính năng OpenFlow nhưng gần đây nó đã gặp phải một số vấn đề về khả năng tương thích với các phiên bản Linux mới nhất (Ubuntu 14.0 trở lên) và hỗ trợ nhà phát triển cũng bị đình trệ. Nó được đi kèm với các công cụ sau để kiểm soát và quản lý mặt phẳng dữ liệu:

+ OfDatapath: Triển khai switch
+ OfLib: Một thư viện để chuyển đổi sang/từ OF1.3 wire formats.
+ DPCTL: Công cụ điều khiển để cấu hình switch
+ OfProtocol: Kênh giao tiếp an toàn với Controller

Tất cả điều này làm cho nó trở thành một sự thay thế hoàn toàn cho OVS. Tuy nhiên, các tác giả của switch cho biết như sau: “Mặc dù thực tế là switch vẫn phổ biến đối với những nhà phát triển đang cố gắng thực hiện các thay đổi của riêng mình đối với OpenFlow, hỗ trợ hiện đang dựa trên “best-effort”. Hiện tại, có rất nhiều lời phàn nàn về hiệu suất giảm sút, các tính năng bị hỏng và các vấn đề cài đặt.”. Switch vẫn có một trong những hỗ trợ tốt nhất cho các tính năng OF1.3 trong số các soft-switch hiện có, đặc biệt là các tính năng tùy chọn như meter tables v.v., khiến nó trở thành một lựa chọn hấp dẫn để bắt tay vào sử dụng. Ngoài ra, soft switch hỗ trợ tiện ích quản lý có tên là Data Path Control (Dpctl), để điều khiển trực tiếp Openflow switch bao gồm thêm và xóa luồng, thống kê chuyển đổi truy vấn và sửa đổi cấu hình Flow Table.

+) Loại bỏ luồng và trục xuất

Có thể xóa các luồng khỏi Controller bằng 3 cách: : theo yêu cầu của controller,hết hạn hoặc thông qua cơ chế loại bỏ switch. Cơ chế hết hạn của Flow switch xác định thời gian chờ quá lâu và thời gian chờ không hoạt động. Thời gian chờ không hoạt động gây ra trục xuất khi và chỉ khi không có frames nào được nhìn thấy trong khoảng thời gian đó. Thời gian chờ quá lâu gây ra việc trục xuất bất kể có được nhìn thấy hay chưa. Controller cũng có thể gửi thông báo DELETE gây ra việc loại bỏ luồng. Cơ chế loại bỏ Flow Switch cho phép switch loại bỏ các luồng để lấy lại tài nguyên. Sau khi xóa, một thông báo FLOW REM có thể được gửi tùy chọn nếu cờ SEND FLOW REM được đặt trong flow entry. Thông báo này được sử dụng để thông báo cho Controller luồng đã bị loại bỏ để Controller có thể giữ số liệu thống kê hoặc đưa ra quyết định dựa trên thông tin này.

c, Iperf:

Iperf là một công cụ đo hiệu năng rất hữu ích để đo lường băng thông tối đa có sẵn giữa hai nút. Nó được sử dụng trong các kết nối TCP (Transport Control Protocol) và UDP (User Datagram Protocol), thông qua việc điều chế các thông số khác nhau. Chủ yếu, Iperf được sử dụng để đo lường luồng mạng giữa hai nút. Nó phải được cài đặt trên cả hai nút, sau đó phải được khởi động dưới dạng máy chủ trên một nút và dưới dạng máy khách trên nút còn lại. Quá trình truyền sẽ chỉ mất vài giây và sau đó bạn sẽ thấy băng thông.

Các công việc sau có thể được thực hiện bằng cách sử dụng Iperf:

- Đo lường băng thông
- Trong một mạng client-server, khách hàng tạo ra một dòng UDP với băng thông cụ thể (BW)
- Đo lường Packet loss
- Đo lường sự chệch lệch (jitter)
- Làm việc trong môi trường multicast.

2.4.2. Cung cấp QoS trong mạng truyền thống

Cung cấp QoS qua Internet là cần thiết để đảm bảo hiệu suất cao cho các ứng dụng khác nhau. Có hai cách tiếp cận chính để hỗ trợ QoS vào mạng dựa trên IP:

+ Dự trữ tài nguyên: Dự trữ tài nguyên là một trong những cách tiếp cận để cung cấp đảm bảo QoS từ đầu đến cuối trên mỗi luồng bằng cách phân bổ tài nguyên băng thông mạng để đảm bảo QoS cho một luồng cụ thể (ví dụ: phiên truyền phát video). Theo yêu cầu QoS của ứng dụng, tài nguyên mạng được phân bổ và tuân theo chính sách quản lý băng thông.

+ Ưu tiên: Ưu tiên là một trong những cách tiếp cận để phân loại lưu lượng mạng và phân bổ tài nguyên mạng theo tiêu chí chính sách quản lý băng thông. Để kích hoạt QoS, các phần tử mạng dành ưu tiên cho các phân loại được xác định là có nhiều yêu cầu khắt khe hơn.

Ngay cả các ứng dụng khác nhau chạy trên cùng một hệ thống phân loại cũng có thể có các yêu cầu QoS khác nhau với các giá trị tham số khác nhau. Hơn nữa, một số tham số QoS này có thể phụ thuộc vào thời gian nhưng có thể không độc lập lẫn nhau. Có một số giao thức và thuật toán QoS khác nhau để đáp ứng nhu cầu về các loại QoS khác nhau này:

+ ReSerVation Protocol (RSVP): RSVP là một giao thức tầng vận chuyển có thể cung cấp tín hiệu để cho phép đặt trước tài nguyên mạng. Bằng cách sử dụng RSVP, người nhận bắt đầu đặt trước tài nguyên mạng bằng cách gửi tin nhắn và lượng đặt trước đảm bảo thỏa mãn các ràng buộc về băng thông, thời gian và kích thước bộ đệm. Dọc theo đường dẫn đã chọn đến người gửi, đặt trước được thiết lập ở trạng thái mềm và đặt các tài nguyên cần thiết sang một bên. Trạng thái mềm được người nhận làm mới định kỳ nếu không nó sẽ hết thời gian hủy đặt trước. RSVP thường được sử dụng trên cơ sở mỗi luồng và cũng được sử dụng để dự trữ tài nguyên cho các tập hợp.

+ Differentiated Services (DiffServ): DiffServ được phát triển để cung cấp một cách đơn giản và thô sơ để phân loại và ưu tiên các tập hợp luồng lưu lượng mạng. Ở bước đầu tiên, DiffServ phân loại tất cả các luồng thành một số lớp giới hạn và xác định một "hành vi per-hop" khác nhau cho mỗi lớp. Để xác định các hành vi per-hop, phải mất 6 bit từ vùng Type-of-Service (TOS) trong IP Header. Sau đó, một hồ sơ lưu lượng truy cập nhất định được tạo bởi khách hàng và trả nó cho nhà cung cấp mạng. Các gói khách hàng được đánh dấu bởi các Edge Routers. Khi các gói đến Core Router, Core Router sẽ biết phải làm gì bằng cách xem mã 6-bit hành vi per-hop từ IP Header.

+ Multi-Protocol Labeling Switching (MPLS): MPLS là công nghệ chuyển tiếp dữ liệu cung cấp khả năng quản lý băng thông cho các tập hợp luồng thông qua điều khiển định tuyến mạng theo nhãn trong Packet Headers.

+ Subnet Bandwidth Management (SBM): SBM là sơ đồ báo hiệu cho phép phân loại và ưu tiên ở lớp Data Link (Lớp 2) trên các mạng IEEE 802 được chia sẻ.

2.4.3. Cung cấp QoS trong SDN

Dù cho tính hiệu quả của QoS-guarantee do IntServ và DiffServ cung cấp, đảm bảo QoS vẫn là một thách thức trên quy mô lớn. Thách thức này về cơ bản khái niệm hóa để quản lý tài nguyên và kiểm soát lưu lượng truy cập. Kiến trúc Internet hiện tại dựa trên các giao thức mạng phân loại chạy trên các phần tử mạng (ví dụ:

Switches và Routers). Việc sử dụng các giao thức phân loại và điều phối các thay đổi trong các mạng thông thường vẫn cực kỳ phức tạp để cấu hình và lập chính sách trên phần cứng mạng cơ bản nhằm kích hoạt nhiều dịch vụ từ định tuyến và chuyển mạch lưu lượng. Việc duy trì trạng thái của một số thiết bị mạng và cập nhật chính sách trở nên khó khăn hơn khi các chính sách ngày càng tinh vi được triển khai thông qua một tập hợp các lệnh cấu hình cấp thấp có giới hạn trên phần cứng mạng thông thường. Do đó, các cấu hình sai thường xuyên như điều kiện lưu lượng thay đổi đòi hỏi phải can thiệp thủ công nhiều lần để cấu hình lại mạng, tuy nhiên, các công cụ có sẵn có thể không đủ tinh vi để cung cấp đủ mức độ chi tiết và tự động hóa nhằm đạt được cấu hình tối ưu.

2.4.4. Hỗ trợ QoS trong các phiên bản OpenFlow khác nhau

OpenFlow đã hỗ trợ khái niệm QoS ngay từ đầu. Tuy nhiên, sự hỗ trợ đã bị hạn chế. Khi các phiên bản mới của OpenFlow xuất hiện trong nhiều năm, mỗi bản phát hành mới của OpenFlow đều mang đến các tính năng mới hoặc cập nhật các tính năng hiện có. Trong tiểu mục này, chúng tôi tóm tắt những gì thay đổi từng đặc tả OpenFlow được thực hiện liên quan đến các tính năng QoS. Các phiên bản đầu tiên của OpenFlow OF1.0 - OF1.1 được hỗ trợ hàng đợi với tỷ lệ tối thiểu. OF1.2+ bắt đầu hỗ trợ hàng đợi với cả tỷ lệ tối thiểu và tối đa. Hàng đợi OpenFlow có sự hỗ trợ rộng rãi trên diện rộng. Hầu hết các triển khai chuyển đổi phần mềm phổ biến (ví dụ: OVS và CPqD OfSoftSwitch) và nhiều nhà cung cấp phần cứng (ví dụ: HP 2920 và Pica8 P-3290) đều hỗ trợ hàng đợi OpenFlow.

Openflow	Tính năng cụ thể
1.0	Enqueue action, thuộc tính tốc độ tối thiểu cho hàng đợi và các trường tiêu đề mới
1.1	Kiểm soát nhiều hơn đối với VLA và MPLS
1.2	Thuộc tính tốc độ tối đa cho hàng đợi và hàng đợi truy vấn của Controller từ các Switches
1.3	Giới thiệu meter table, giới hạn tỷ lệ và tính năng giám sát tỷ lệ
1.4	Giới thiệu một số tính năng giám sát
1.5	Thay thế meter action thành meter instruction

Bảng 4. Các tính năng liên quan đến QoS trong các phiên bản OpenFlow khác nhau

OF1.3 đã giới thiệu khái niệm meter tables để đạt được QoS chi tiết hơn trong các mạng OpenFlow. Trong khi các hàng đợi kiểm soát tốc độ đầu ra của lưu lượng, các meter tables có thể được sử dụng để theo dõi tốc độ của lưu lượng trước khi xuất. Nói cách khác, hàng đợi kiểm soát tốc độ đi ra và meter table có thể được sử dụng để kiểm soát tốc độ đi vào của lưu lượng truy cập. Điều này làm cho hàng đợi và meter table bổ sung cho nhau. OpenFlow Switches cũng có khả năng đọc và ghi các bit Type of Service (ToS) trong IP Header . Nó là một trường có thể

được sử dụng để khớp với một Packet trong một flow entry. Tất cả các tính năng này cùng nhau cho phép quản trị viên mạng triển khai QoS trong mạng của họ.

Sau đây là các tính năng liên quan đến QoS trong các phiên bản OpenFlow [4]:

1. Hàng đợi

Như đã đề cập trước đó, giao thức OpenFlow đã mô tả hàng đợi giới hạn tốc độ tối thiểu trong OF1.0 và hàng đợi giới hạn tốc độ tối thiểu và tối đa trong OF1.2. Theo đặc tả OpenFlow, OpenFlow sử dụng hàng đợi trên các Switches nhưng không xử lý việc quản lý hàng đợi trên nó. Việc quản lý hàng đợi trên switch diễn ra bên ngoài giao thức OF và bản thân giao thức OF chỉ có thể truy vấn số liệu thống kê hàng đợi từ switch. Có hai giao thức để cung cấp tác vụ quản lý hàng đợi (tạo, xóa và thay đổi) trong các Switches hỗ trợ OpenFlow: OF-Config và OVSDB. Bên cạnh đó, có một hàng đợi tiêu chuẩn quản lí được cung cấp bởi bất kỳ OpenFlow Controller nào.

2. OVSDB

OF-Config và OVSDB là hai giao thức Southbound để điều khiển hoạt động của các thiết bị chuyển tiếp ngoài các quyết định chuyển tiếp. Cụ thể, OVSDB quản lý các hoạt động chuyển mạch như tạo đường hầm, chuyển đổi trạng thái cổng, cấu hình hàng đợi và quản lý QoS. OVSDB sử dụng nhiều bảng để quản lý Open vSwitch. Các bảng này bao gồm bảng luồng, bảng cổng, bảng NetFlow và các bảng khác. Tương tự, nó duy trì các bảng cho QoS và Hàng đợi. Trong khi hầu hết các bảng khác là bảng gốc của lược đồ OVSDB, nghĩa là bảng và các mục nhập của nó sẽ không tự động bị xóa nếu không thể truy cập được. Do đó, QoS và các bảng đợi tồn tại và có thể được thay đổi độc lập, cho dù chúng có được tham chiếu bởi một cổng hay không. Bảng cổng có liên quan đến bảng QoS và bảng giao diện. Mối quan hệ với bảng giao diện là bắt buộc, nghĩa là mỗi cổng phải được liên kết với một giao diện. Tuy nhiên, mối quan hệ với QoS là tùy chọn. Một cổng có thể tồn tại mà không có cài đặt QoS kèm theo. Một cổng có thể có một bảng QoS có thể có nhiều hàng đợi được gán cho nó.

Khi QoS và hàng đợi đã được thiết lập trên một Switch các luồng có thể được chuyển hướng đến một hàng đợi cụ thể bằng cách sử dụng hành động thiết lập hàng đợi của OpenFlow. Hành động này sẽ chuyển tiếp các luồng phù hợp với tiêu chí phù hợp vào hàng đợi được đề cập. Nếu có nhiều hơn một luồng đi qua Switch cùng một lúc, tốc độ tổng hợp của các luồng sẽ được kiểm soát ở đầu ra theo tốc độ tối thiểu và tốc độ tối đa được xác định bởi hàng đợi. Chúng ta hãy tìm hiểu sâu hơn một chút về cách hàng đợi được triển khai trong giao thức OpenFlow và OVS.

Đặc tả OpenFlow nêu các thuộc tính sau về hàng đợi:

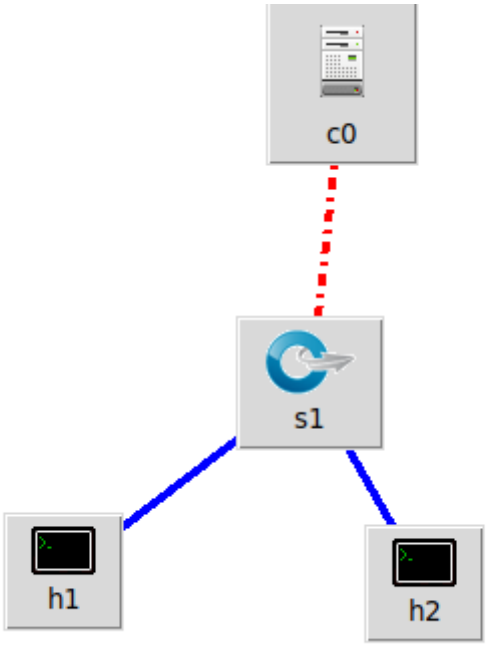
+ Tốc độ tối thiểu: Tốc độ dữ liệu tối thiểu được đảm bảo cho một hàng đợi. Dung lượng được chia sẻ theo tỷ lệ dựa trên tỷ lệ tối thiểu của mỗi hàng đợi. Khi tốc độ tối thiểu được đặt, Switch sẽ ưu tiên hàng đợi để đạt được tốc độ tối thiểu đã nêu. Nếu có nhiều hơn một hàng đợi trong một cổng, với tổng tốc độ tối thiểu cao hơn dung lượng của liên kết, tốc độ của tất cả các hàng đợi đó sẽ bị phạt.

+ Tốc độ tối đa: Tốc độ dữ liệu tối đa có thể được phép cho một hàng đợi. Nếu tốc độ thực tế của các luồng lớn hơn tốc độ tối đa của hàng đợi đã nêu, thì Switch sẽ trì hoãn các Packets hoặc loại bỏ chúng để đáp ứng tốc độ tối đa.

Mặc dù các thông số kỹ thuật của OpenFlow đề cập đến các nguyên tắc này cho các Switch tương thích với OpenFlow, nhưng việc triển khai Switch để nhận ra các tính năng này là tùy thuộc vào quá trình triển khai. Open vSwitch, dựa trên Linux, sử dụng chương trình Traffic Control(TC) của Linux để triển khai hàng đợi.

2.5. Môi trường thực hành:

Sử dụng một topo gồm 1 controller, 1 switch và 2 host như hình sau:



2.6. Đánh giá:

2.6.1. INSERV - PER-FLOW QoS:

- Cài đặt hàng đợi và flow entry cho các switch:

Queue ID	Max rate	Min rate
0	500Kbps	-
1	(1Mbps)	800Kbps

(Priority)	Destination address	Destination port	Protocol	Queue ID	(QoS ID)
1	10.0.0.1	5002	UDP	1	1

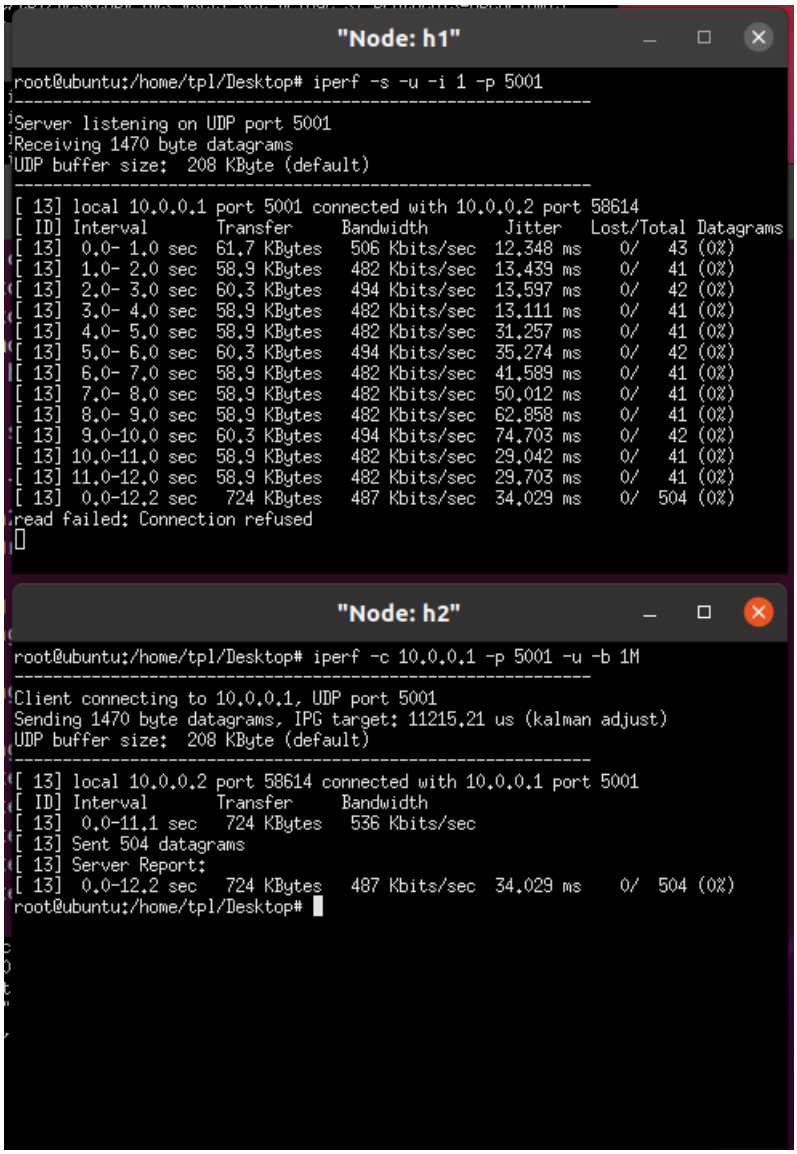
- Đo băng thông:

Xterm h1 và h2: Sử dụng iperf

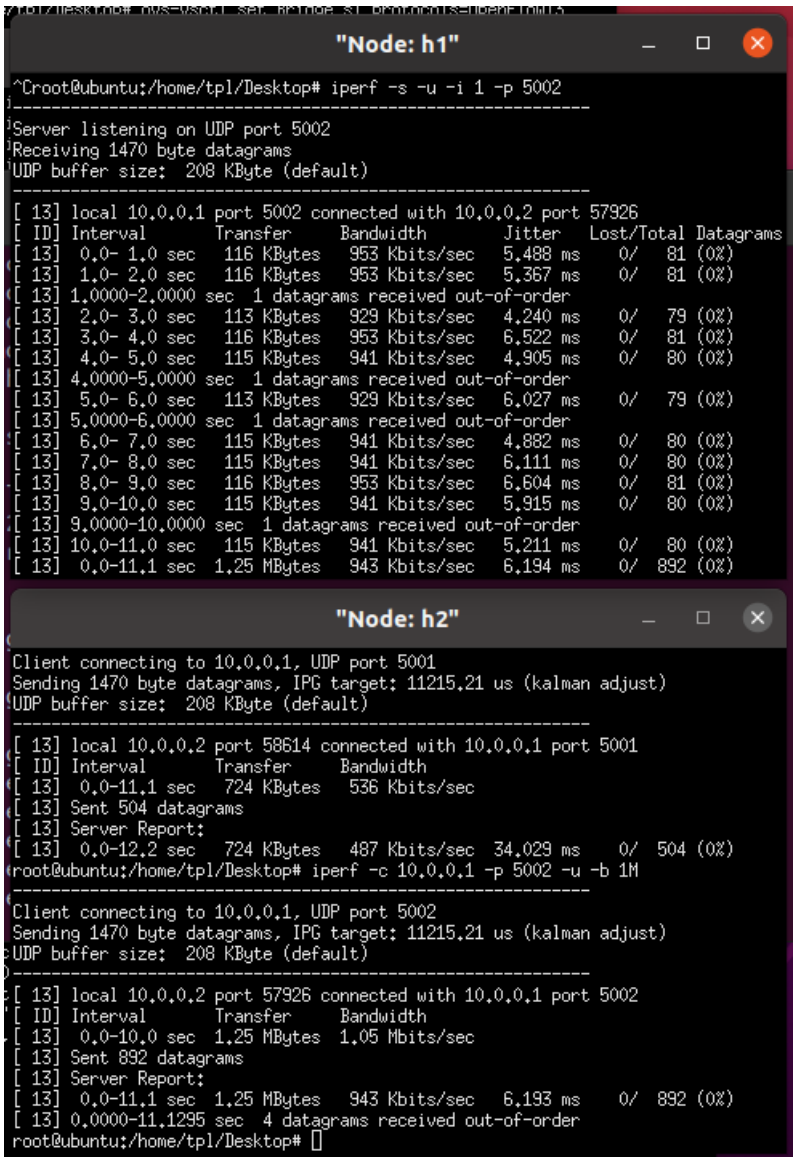
h1: server nhận port 5001 và 5002 (giao thức UDP)

h2: client gửi 1mbps UDP traffic đến port 5001 và 5002

Port 5001:



Port 5002:



- Kết luận:

Kết quả trên cho thấy lưu lượng gửi đến cổng 5001 được điều chỉnh với tốc độ tối đa 500Kbps và lưu lượng gửi đến cổng 5002 được đảm bảo băng thông 800Kbps như băng thông tối thiểu đã cài đặt trước đó.

Ta thấy phần mô tả ví dụ về hoạt động của QoS theo luồng đã được trình bày. Bằng cách thiết lập các ưu tiên khác nhau cho các luồng và phân bổ chúng vào các hàng đợi khác nhau với các đảm bảo băng thông khác nhau, QoS cho mỗi luồng được đảm bảo. Ví dụ cũng cho thấy cách phân bổ băng thông và lập lịch được thực hiện bởi các switch OpenFlow bằng cách sử dụng các luật lệ điều khiển luồng. Loại hoạt động QoS theo luồng này có thể được sử dụng để đảm bảo QoS cho các loại lưu lượng khác nhau trong một mạng, chẳng hạn như các ứng dụng thời gian thực, phát trực tuyến đa phương tiện và lưu lượng cố gắng tốt nhất.

2.6.2. DIFFSERV:

- Cài đặt hàng đợi và flow entry cho các switch:

Queue ID	Max rate	Min rate	Class
0	1Mbps	-	Default
1	(1Mbps)	200Kbps	AF3
2	(1Mbps)	500Kbps	AF4

Priority	DSCP	Queue ID	(QoS ID)
1	26(AF31)	1	1
1	34(AF41)	2	2

(Priority)	Destination address	Destination port	Protocol	DSCP	(QoS ID)
1	172.16.20.10	5002	UDP	26(AF31)	1
1	172.16.20.10	5003	UDP	34(AF41)	2

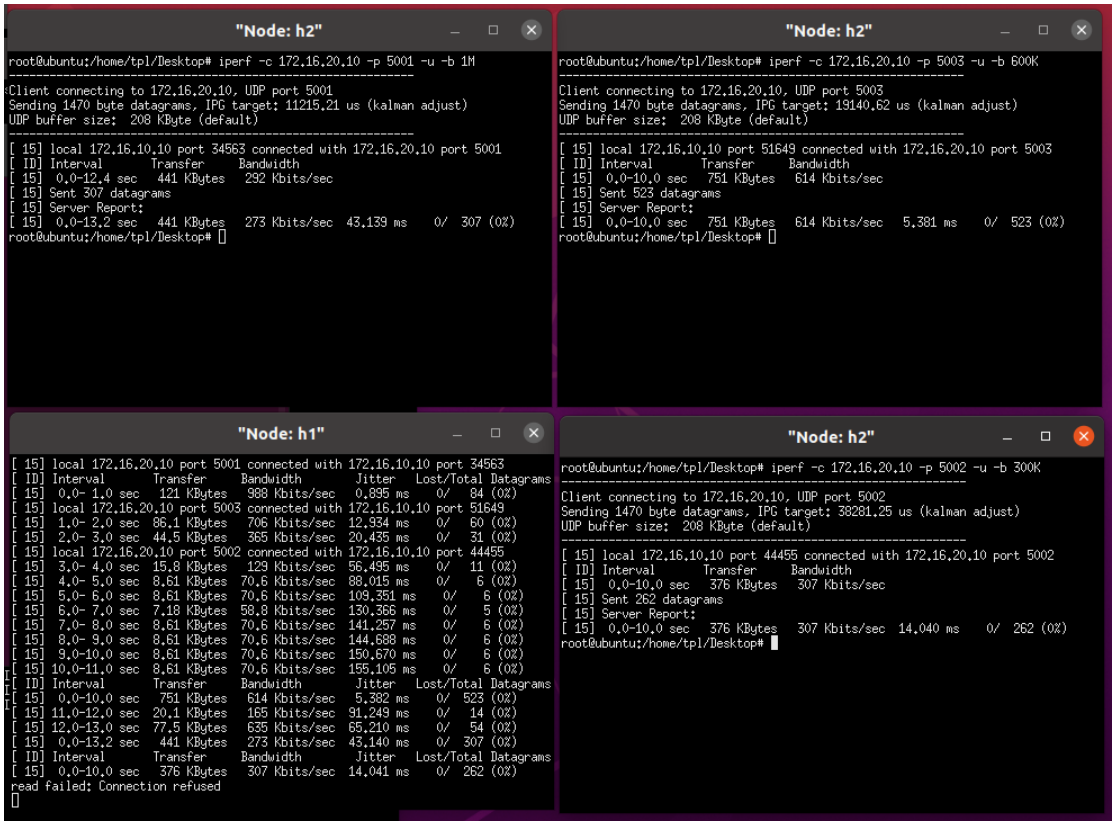
- Đo băng thông

Sử dụng iperf: h1 (server) nhận tin từ các port port 5002, 5001, 5003 bằng giao thức UDP

h2 (client) gửi: 1mbps UDP traffic port 5001

300kbps UDP traffic port 5002

600kbps UDP traffic port 5003



- Kết luận:

Kết quả trên cho thấy:

Lưu lượng dữ liệu được đánh dấu với AF41 (gửi đến cổng 5003) được đảm bảo băng thông là 500Kbps, và lưu lượng dữ liệu được đánh dấu với AF31 (gửi đến cổng 5002) được đảm bảo băng thông là 200Kbps. Trong khi đó, băng thông của lưu lượng truyền thông theo phương thức "best-effort" bị giới hạn trong khi lưu lượng dữ liệu được đánh dấu với lớp AF đang truyền thông. Như vậy, chúng ta đã có thể xác nhận rằng có thể thực hiện QoS bằng cách sử dụng mô hình DiffServ.

Với việc thực hiện thành công của phương pháp này, chúng ta có thể kết luận rằng việc triển khai QoS trên mạng thông qua DiffServ model là hoàn toàn khả thi và hiệu quả. Nó giúp cho việc ưu tiên và phân phối băng thông trên mạng được thực hiện một cách nhanh chóng và chính xác hơn, giúp đảm bảo chất lượng dịch vụ cho các ứng dụng và dịch vụ quan trọng trên mạng.

3. Tài liệu tham khảo

[1] Adebayo Oluwaseun Adedayo, B. T. (n.d.). *QoS Functionality in Software Defined Network.*

[2] Rani, H. B. (2021). Performance Evaluation of QoS metrics in Software Defined Networking.

[3] Shuangyin Ren, Quanyou Feng, Wenhua Dou. (n.d.). *An End-to-End QoS Routing on Software Defined Network.*

[4] Saleh Asadollahi, Bhargavi Goswami, Mohammed Sameer. (n.d.). *Ryu Controller's Scalability Experiment on.*

[5] Thazin, N. (2019). *QoS-BASED TRAFFIC ENGINEERING IN SOFTWARE.*

[6] Ryu project team. *RYU SDN Framework.*