

TRƯỜNG ĐẠI HỌC TÀI NGUYÊN VÀ MÔI TRƯỜNG TP.HCM
KHOA HỆ THỐNG THÔNG TIN VÀ VIỄN THĂM



**ĐỒ ÁN MÔN: AN TOÀN VÀ BẢO MẬT HỆ THỐNG THÔNG
TIN**
**ỨNG DỤNG SDN VÀ THUẬT TOÁN NAIVE
BAYES VÀO MÔ HÌNH MẠNG DOANH
NGHIỆP**

Giảng viên:

Th.S Phạm Trọng Huỳnh

Sinh viên:

Mai Anh Lộc 0850080028

Võ Trần Uy 0850080053

Thái Lai 0850080027

Lớp:

08_ĐH_CNPM

TP. Hồ Chí Minh, tháng 4 năm 2023

TRƯỜNG ĐẠI HỌC TÀI NGUYÊN VÀ MÔI TRƯỜNG TP.HCM
KHOA HỆ THỐNG THÔNG TIN VÀ VIỄN THĂM



ĐỒ ÁN MÔN: AN TOÀN VÀ BẢO MẬT HỆ THỐNG THÔNG
TIN
ỨNG DỤNG SDN VÀ THUẬT TOÁN NAIVE
BAYES VÀO MÔ HÌNH MẠNG DOANH
NGHIỆP

Giảng viên:

Th.S Phạm Trọng Huỳnh

Sinh viên:

Mai Anh Lộc 0850080028

Võ Trần Uy 0850080053

Thái Lai 0850080027

Lớp:

08_ĐH_CNPM

TP. Hồ Chí Minh, tháng 4 năm 2023

MỞ ĐẦU

Internet ngày càng phát triển và quan trọng đối với mỗi chúng ta. Sự phát triển hàng ngày, hàng giờ với các tính năng mới mang đến cho người dùng những trải nghiệm và phục vụ tốt hơn nhu cầu cuộc sống mỗi người.

Đi xuống một cấp độ thấp hơn, cấp độ mạng, chúng ta có thể nhận ra rằng, sự phát triển ở cấp độ này diễn ra chậm hơn rất nhiều. Không có nghi ngờ nào về sự phát triển ngày càng mạnh mẽ của cơ sở hạ tầng mạng internet trên “mặt số lượng”, băng thông tổng cộng tăng lên nhanh chóng, các kỹ thuật mới ở Layer 2 được áp dụng, tuy nhiên sự thay đổi về mặt cấu trúc đến thời điểm hiện tại là gần như không đáng kể. Một trong những nguyên nhân cho vấn đề này là vì cấu trúc mạng “nguyên khối”, nó chứa tập hợp các chức năng trong đó kể cả các ứng dụng mạng. Việc áp dụng chức năng mới yêu cầu phải hiện đại hóa toàn mạng với hàng triệu thiết bị. Hãy thử tưởng tượng rằng chúng ta phải tiến hành cập nhật tất cả các thiết bị mỗi khi chúng ta cài một ứng dụng mới, đó thực sự là một công việc khó khăn và mất rất nhiều thời gian, công sức.

Nói cách khác, sự đổi mới trên cấp độ mạng trong khuôn khổ cấu trúc ngày nay là rất khó khăn. Các chức năng và các tính năng mới làm tăng tính phức tạp của hệ thống lên rất nhiều lần, việc thử nghiệm chúng cũng vậy và nếu áp dụng chúng vào thực tế cũng đòi hỏi chi phí rất cao và tiềm ẩn nhiều nguy cơ khác.

Chính vì thế rất nhiều chuyên gia đã đặt kỳ vọng vào một mô hình mạng mới, mạng điều khiển bởi phần mềm SDN. Đồ án này cho chúng ta thấy một cách tổng quan về mạng SDN, giao thức OpenFlow và ứng dụng của nó vào mô hình mạng doanh nghiệp. Do kiến thức và thời gian có hạn, không tránh khỏi sai sót, kính mong các thầy, cô góp ý kiến để đồ án nhóm em được hoàn thiện hơn.

[illegible]

MỤC LỤC

CHƯƠNG 1: TỔNG QUAN	1
1.1 Giới thiệu về lý do chọn đề tài	1
1.2 Mục đích nghiên cứu	1
1.3 Phạm vi và đối tượng nghiên cứu.....	2
1.3.1 Phạm vi nghiên cứu:	2
1.3.2 Đối tượng nghiên cứu:.....	2
CHƯƠNG 2: CƠ SỞ LÝ THUYẾT.....	3
2.1 Tổng quan về SDN:	3
2.1.1 Đặt vấn đề.....	3
2.1.2 Khái niệm và cấu trúc mạng SDN.....	6
2.1.3 Ưu nhược điểm của SDN so với mạng IP	13
2.1.4 Ứng dụng của SDN:	15
2.1.4.2. Phạm vi các nhà cung cấp hạ tầng và dịch vụ viễn thông:.....	16
2.1.5 Các mô hình triển khai SDN	16
2.1.6 Kết luận	19
2.2 Giao thức OpenFlow	20
2.2.1 Lịch sử phát triển của OpenFlow	20
2.2.2 Giao thức OpenFlow	20
2.2.3 Kiến trúc và hoạt động của OpenFlow	21
2.2.4 Ưu điểm của OpenFlow.....	24
2.2.5 Kết luận	24
2.3.Giới thiệu về máy học (machine learning):	25
2.4 Tìm hiểu về thuật toán Native Bayer và các ứng dụng của nó trong mô hình mạng SDN	27

2.4.1	Khái niệm về thuật toán Native Bayer	27
2.4.2	Công thức và kỹ thuật của thuật toán Native Bayer	28
2.4.3	Ưu điểm của thuật toán Native Bayer	29
2.4.4	Nhược điểm của thuật toán Native Bayer	30
2.4.5	Ứng dụng của thuật toán Native Bayer trong mô hình mạng SDN	30
CHƯƠNG 3: CÀI ĐẶT THỬ NGHIỆM		32
3.1	Phương pháp nghiên cứu	32
3.2	Kết quả đạt được.....	32
3.2.1	Xây dựng mô hình mạng SDN.....	33
3.2.2	Xây dựng thuật toán Native Bayer.....	47
CHƯƠNG 4: KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN		50
4.1	Kết luận	50
4.2	Đánh giá mô hình mạng SDN	50
4.3	Kiến thức đạt được	51
4.4	Hướng phát triển	52
DANH MỤC TÀI LIỆU THAM KHẢO		53

DANH MỤC HÌNH ẢNH

Hình 2. 1 Chức năng cơ bản của một Router/ Switch	7
Hình 2. 2 Sơ đồ một mạng truyền thống đơn giản.	8
Hình 2. 3 Sơ đồ mạng đơn giản với bộ điều khiển SDN.....	9
Hình 2. 4 Kiến trúc mạng SDN.	11
Hình 2. 5 Mô hình dựa trên Switch.	16
Hình 2. 6 Overlay Network SDN.	18
Hình 2. 7 Flow table trên một thiết bị	22
Hình 2. 8 Sơ đồ hoạt động của OpenFlow	23
Hình 2. 9 Sơ đồ máy học (MACHINE LEARNING)	25
Hình 3. 1 Cài đặt java 11 trên ubuntu.....	26
Hình 3. 2 Cập nhật danh sách các gói phần mềm có sẵn trong hệ thống.....	34
Hình 3. 3 Màn hình đặt biến và tải ONOS	34
Hình 3. 4 Giải nén file onos.....	35
Hình 3. 5 Đổi tên file và khởi động dịch vụ ONOS	35
Hình 3. 6 Truy cập vào CLI của ONOS	36
Hình 3. 7 Truy cập vào GUI.....	36
Hình 3. 8 Tài khoản mật khẩu mặc định	37
Hình 3. 9 Giao diện ONOS.....	37
Hình 3. 10 Tải và cài đặt mininet	38
Hình 3. 11 Tạo topo mạng 2x2 với các switch.....	40
Hình 3. 12 Cài đặt giao thức Openflow.....	40
Hình 3. 13 Ping để kiểm tra kết nối.....	40
Hình 3. 14 Cài đặt ứng dụng Forwarding.....	40
Hình 3. 15 Sử dụng ping để kiểm tra kết nối.....	41
Hình 3. 16 Bốn máy chủ do mininet tạo ra trên GUI	41
Hình 3. 17 Xóa các topo mạng đã tạo trước đó.....	41
Hình 3. 18 Tạo một topo mạng mới	42
Hình 3. 19 Kết quả cấu trúc mạng.....	42
Hình 3. 20 Tạo kết nối giữa host 1 và host 6.....	43

Hình 3. 21 Kết quả hiển thị bằng đường gạch đứt màu cam.....	43
Hình 3. 22 Ngắt kết nối giữa switch 1 và switch 2.....	44
Hình 3. 23 Kích hoạt các ứng dụng cần thiết và tạo VPLS.....	44
Hình 3. 24 Các ý định mới do VPLS tạo ra.....	45
Hình 3. 25 Giao diện thiết lập từ chế độ xem cấu trúc liên kết.....	45
Hình 3. 26 Phá vỡ một số liên kết	46
Hình 3. 27 Giao diện GUI sau phá vỡ nhưng không có gì thay đổi.....	46
Hình 3. 28 xây dựng một mô hình phân loại Naive Bayes dựa trên dữ liệu iris	47
Hình 3. 29 Code sử dụng thuật toán native bayer để phân loại email 1	48
Hình 3. 30 Code sử dụng thuật toán native bayer để phân loại email 2.....	50

CHƯƠNG 1: TỔNG QUAN

1.1 Giới thiệu về lý do chọn đề tài

Trong thời đại công nghệ thông tin ngày nay, hệ thống mạng là một phần không thể thiếu của mọi tổ chức, công ty. Với sự phát triển không ngừng của mạng, mô hình mạng truyền thống đã không đáp ứng được nhu cầu đa dạng của các tổ chức và doanh nghiệp hiện nay. Do đó, mô hình SDN (Software-Defined Networking) ra đời nhằm khắc phục những hạn chế của mô hình truyền thống.

Mô hình mạng SDN có thể tối ưu hóa băng thông, giảm độ trễ và cải thiện hiệu suất mạng. Nó cũng cung cấp khả năng quản lý linh hoạt, tự động hóa và đáp ứng nhanh chóng các nhu cầu của mạng. Tuy nhiên, việc triển khai và vận hành mô hình mạng SDN vẫn là một thách thức, đặc biệt là trong môi trường kinh doanh phức tạp.

Trước những thách thức đó, đề tài “Ứng dụng SDN và thuật toán gốc Bayer trong mô hình mạng doanh nghiệp” được lựa chọn để nghiên cứu và giải quyết. Mục tiêu của dự án là xây dựng một mô hình mạng SDN doanh nghiệp hiệu quả và tối ưu hóa hiệu suất mạng trong khi sử dụng thuật toán gốc của Bayer để giải quyết một vấn đề cụ thể trong mô hình mạng SDN. Kết quả của dự án sẽ cung cấp cho các tổ chức, doanh nghiệp một giải pháp mạng tiên tiến và hiệu quả, đáp ứng các nhu cầu đa dạng của họ.

1.2 Mục đích nghiên cứu

Mục đích của đề án "Ứng dụng SDN và thuật toán Native Bayer trong mô hình mạng doanh nghiệp" là xây dựng một mô hình mạng SDN hiệu quả cho doanh nghiệp và tối ưu hóa hiệu suất mạng, đồng thời sử dụng thuật toán Native Bayer để giải quyết một vấn đề cụ thể trong mô hình mạng SDN.

Để đạt được mục tiêu này, đề án sẽ tập trung vào các nội dung chính sau:

- Tìm hiểu tổng quan về SDN và kiến trúc mạng SDN, đặc biệt là giao thức OpenFlow.

- Nghiên cứu các phương pháp máy học để tối ưu hóa mô hình mạng SDN và giải quyết các vấn đề trong quản lý mạng.
- Xây dựng một mô hình mạng SDN hiệu quả, đáp ứng các yêu cầu đa dạng của doanh nghiệp và tối ưu hóa hiệu suất mạng.
- Áp dụng thuật toán Native Bayer vào mô hình mạng SDN để giải quyết một vấn đề cụ thể và đánh giá hiệu quả của thuật toán.
- Đánh giá mô hình mạng SDN được xây dựng và hiệu quả của thuật toán Native Bayer, đưa ra những nhận xét và đề xuất cho phát triển mô hình mạng SDN trong tương lai.

1.3 Phạm vi và đối tượng nghiên cứu

1.3.1 Phạm vi nghiên cứu:

- Tìm hiểu tổng quan về mô hình mạng SDN và thuật toán Native Bayer.
- Nghiên cứu kiến trúc mạng SDN và giao thức OpenFlow.
- Áp dụng các phương pháp máy học để tối ưu hóa mô hình mạng SDN và đánh giá hiệu quả của mô hình.
- Xây dựng mô hình mạng SDN và triển khai thuật toán Native Bayer vào mô hình.
- Đánh giá hiệu suất mạng và hiệu quả của thuật toán Native Bayer trong mô hình mạng SDN.

1.3.2 Đối tượng nghiên cứu:

- Các tổ chức và doanh nghiệp muốn triển khai mô hình mạng SDN.
- Các nhà quản trị mạng và kỹ thuật viên muốn tìm hiểu về mô hình mạng SDN và thuật toán Native Bayer.
- Các nhà nghiên cứu, sinh viên, giảng viên quan tâm đến lĩnh vực mạng máy tính và ứng dụng phương pháp máy học trong mạng.

CHƯƠNG 2: CƠ SỞ LÝ THUYẾT

2.1 Tổng quan về SDN:

Software Defined Networking (SDN) là một kiểu kiến trúc mạng mới có thể giải quyết các vấn đề hạn chế mà mạng truyền thống đang gặp phải và khả năng thích nghi và đáp ứng các dịch vụ mới rất cao. Chương này sẽ cho chúng ta biết những hạn chế mà các mạng truyền thống đang gặp phải và giới thiệu một cách tổng quan về cấu trúc, chức năng của mạng SDN.

2.1.1 Đặt vấn đề

Bộ giao thức truyền thống TCP/IP được xem như là một chuẩn sử dụng từ giữa những năm 80 của thế kỷ trước. Đây là một hệ thống điều khiển công kênh và không linh hoạt đối với mạng máy tính. Vì nó vừa “nghĩ” vừa “làm”, điều đó có nghĩa là đầu tiên nó phải giải quyết bài toán xây dựng định tuyến, sau đó là áp dụng các tuyến đường này. Trong các mạng hiện tại, chức năng điều khiển và truyền tải dữ liệu được kết hợp, đi liền với nhau, nó làm cho việc kiểm soát, điều khiển rất phức tạp. Cách tiếp cận dựa trên TCP/IP này gây ra một số hạn chế rất nghiêm trọng trong hoạt động với các tài nguyên của mạng.

Dễ thấy rằng số lượng và tính phức tạp của các giao thức rất lớn và phức tạp (Ngày nay số giao thức và các phiên bản giao thức được sử dụng thường xuyên đã vượt quá 600), việc kết hợp sự điều khiển và truyền dữ liệu làm cho quá trình kiểm soát cũng như điều khiển hoạt động mạng trở nên quá phức tạp đòi hỏi người quản lý phải có tay nghề và chuyên môn cao. Vấn đề bảo mật đến thời điểm hiện tại vẫn không có giải pháp nào có độ tin cậy quá cao. Việc thêm vào bất kỳ sự thay đổi nào trong các thiết bị của mạng đều mất rất nhiều thời gian, chi phí cao và bắt buộc phải có sự tham gia của nhà sản xuất (vì tính độc quyền). Và vì thế, không ai có thể đảm bảo rằng những thiết bị mạng này chỉ chứa các chức năng đã được mô tả trong các tài liệu đính kèm sản phẩm. Đây là lí do vì sao có rất nhiều vụ bê bối nghe lén và đánh cắp dữ liệu diễn ra thời gian qua. Các thiết bị của mạng ngày nay là những thiết bị mang tính độc quyền, thiết bị “đóng”, cản trở cho sự đổi mới, cập nhật và phát triển từ hướng người chủ của mạng, hay cộng đồng mạng.

Việc đáp ứng tất cả các nhu cầu hiện tại của thị trường gần như là không thể với mô hình mạng truyền thống. Phòng quản trị mạng của các công ty phải tìm cách hạn chế

tối đa mạng của mình với việc sử dụng các công cụ điều khiển ở mức độ thiết bị và sử dụng các quá trình điều khiển bằng tay, lý do của vấn đề này chính là vì ngân sách được chi cho họ ngày càng bị cắt giảm, nếu may mắn thì chỉ được duy trì không đổi. Với những nhà khai thác mạng, họ cũng gặp vấn đề tương tự. Ta có thể thấy nhu cầu đối với tính di động và băng thông đang bùng phát (ngày nay số lượng người dung mạng máy tính trên kỹ thuật không dây vượt quá số người dùng mạng cố định, số lượng thiết bị di động trên đầu người ở các nước phát triển đã lớn hơn 3) trong khi đó lợi nhuận thu về ngày càng ít do các chi phí cho thiết bị và do việc giảm thu nhập. Các cấu trúc hiện tại của mạng không được tạo ra để thỏa mãn nhu cầu của người dùng hiện đại, của các công ty hay nhà khai thác mạng. Chúng ta sẽ phân tích một số giới hạn của mạng hiện tại, bao gồm:

Tính phức tạp dẫn đến tình trạng trì trệ: Các kỹ thuật mạng ngày nay bao gồm các bộ giao thức rời rạc. Những giao thức này dùng để nối các host với nhau một cách tin cậy, với khoảng cách, tốc độ liên lạc, topo bất kỳ. Để thỏa mãn nhu cầu kinh doanh và yêu cầu kỹ thuật trong hơn chục năm trở lại đây, ngành công nghiệp đã phát triển các giao thức mạng để hỗ trợ hiệu suất cũng như độ tin cậy cao hơn, có thể kết nối rộng hơn và độ bảo mật nghiêm ngặt hơn.

Các giao thức này, về nguyên tắc, được tạo ra một cách cô lập, tuy nhiên mỗi giao thức giải quyết một vấn đề cụ thể. Điều này dẫn đến một trong những hạn chế chính của mạng hiện tại đó là tính phức tạp. Ví dụ: để thêm vào hoặc dịch chuyển một thiết bị bất kỳ, người quản trị phải can thiệp đến một số thiết bị khác như : các bộ chuyển mạch, định tuyến, tường lửa... và phải cập nhật lại danh sách ACL (Access Control List), VLANs, QoS, và cả các cơ chế khác. Liên quan đến tính phức tạp này, các mạng hiện tại vì thế được xem như ở trạng thái “tĩnh” vì người quản trị phải cố gắng hạn chế đến mức thấp nhất những nguy cơ gián đoạn cung cấp dịch vụ.

Tính “tĩnh” của mạng hiện tại lại là một mâu thuẫn rất lớn đối với đặc tính “động” của môi trường server ngày nay, ở đó việc ảo hóa các server làm tăng số lượng host một cách chóng mặt, đồng thời nó làm thay đổi quan điểm về vị trí vật lý của các host. Trước ảo hóa, các ứng dụng đều nằm trên một server và trao đổi traffic với các client. Ngày nay, các ứng dụng phân bố rời rạc trên một vài máy ảo (VM-Virtual Machine), những máy ảo

này trao đổi các luồng dữ liệu với nhau. Các VM này “tái định cư” để làm tối ưu hóa và cân bằng tải trên server.

Ngoài việc áp dụng kỹ thuật ảo hóa, nhiều công ty đã làm việc trên nền mạng hội tụ IP để truyền dữ liệu, thoại, video... Trong khi đó, mạng hiện tại hỗ trợ các mức độ khác nhau của QoS cho các ứng dụng khác nhau và cung cấp những tài nguyên này hoàn toàn bằng tay. Người quản trị cần phải cài đặt thiết bị của từng nhà cung cấp một cách riêng lẻ, và dĩ nhiên phải thiết lập các tham số như băng thông, QoS trên từng phiên làm việc cho mỗi ứng dụng. Do tính “tĩnh” của mình, mạng hiện tại không thể điều chỉnh một cách linh động so với những traffic luôn thay đổi của các ứng dụng và người dùng.

➤ Các chính sách không đồng nhất: Để thực hiện các chính sách mạng, người quản trị mạng cần phải cấu hình hàng ngàn thiết bị. Ví dụ mỗi lần áp dụng một máy ảo mới, phải tốn hàng giờ, thậm chí hàng ngày để cấu hình lại các danh sách ACL trên toàn mạng. Tính phức tạp của mạng hiện tại làm cho công việc này trở nên khó khăn đối với các nhà quản trị để có thể áp dụng một bộ phối hợp truy cập, hay quy tắc bảo mật, QoS và các chính sách người dùng khác.

➤ Không có khả năng mở rộng: Vì các nhu cầu đối với các Data Center tăng nhanh chóng, nên mạng cũng buộc phải tăng (kích thước) theo. Tuy nhiên, mạng vì thế quá phức tạp với hàng trăm, hàng ngàn thiết bị, những thiết bị này lại cần phải được cấu hình và điều khiển. Các nhà quản trị cũng buộc phải dựa trên các dự báo về traffic để mở rộng mạng. Nhưng trong các Trung tâm dữ liệu ảo hóa ngày nay, traffic là khái niệm “động” không tương và gần như không thể dự báo trước.

Các nhà khai thác lớn như Google, Yahoo, Facebook... đã gặp phải các vấn đề phức tạp trong mở rộng kích thước mạng. Những nhà cung cấp dịch vụ này sử dụng các thuật toán xử lý song song ở quy mô lớn. Vì quy mô các ứng dụng đối với một người dùng cụ thể ngày càng tăng, số lượng các phần tử cần tính toán từ đó cũng tăng lên đến mức “bùng nổ” và các dữ liệu trao đổi giữa các node có thể đạt đến PB (Petabyte=1000 TB). Những công ty này cần phải đảm bảo hiệu suất cao, chi phí kết nối giữa hàng ngàn thiết bị thấp... Quy mô như vậy là không thể thực hiện với cách cấu hình bằng tay. Để duy trì khả năng cạnh tranh, các nhà khai thác cần phải thực hiện, cung cấp nhiều hơn các dịch vụ riêng

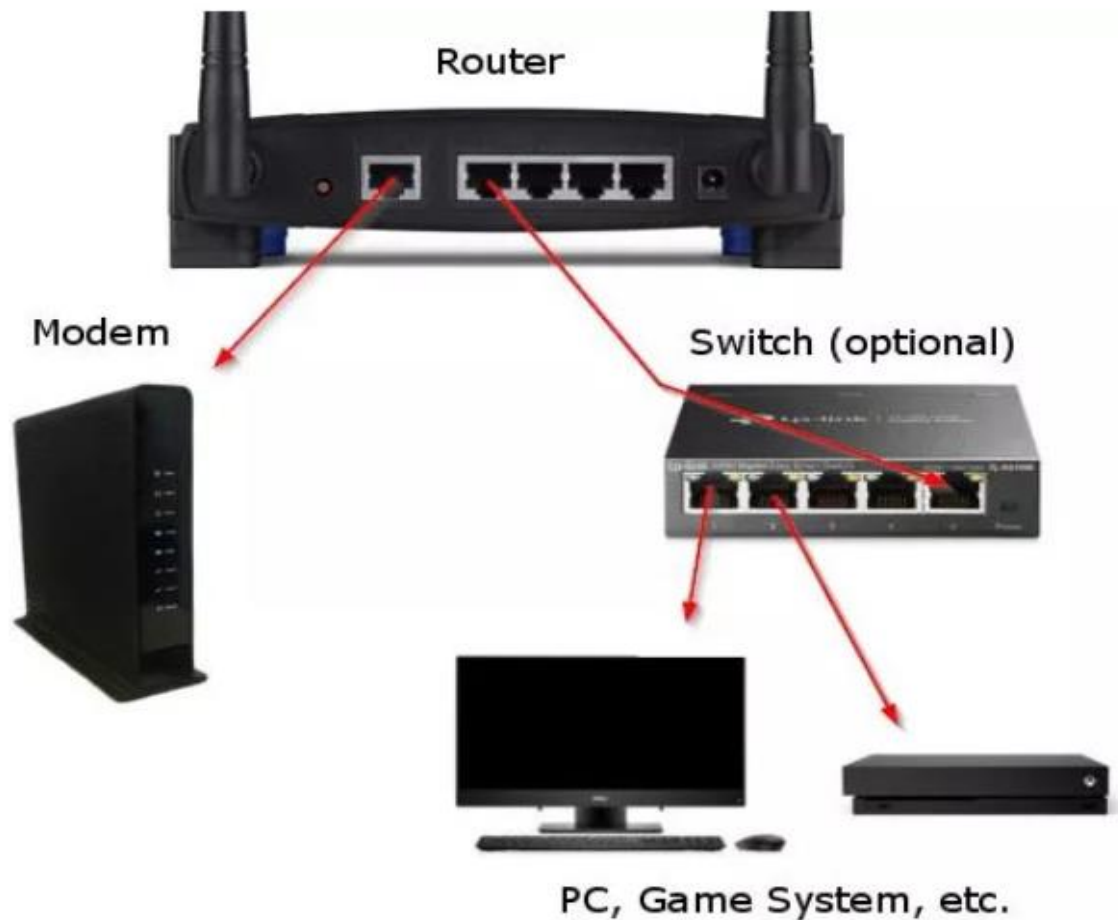
biệt, khác biệt cho các client. Tính đa nhiệm cũng làm phức tạp bài toán hơn, vì mạng cần phục vụ nhiều nhóm người dùng với các ứng dụng khác nhau và các nhu cầu về hiệu suất khác nhau. Những nhà khai thác lớn, những nhà khai thác đóng vai trò chủ đạo trong quản lý traffic client rất khó để đáp ứng các nhu cầu với quy mô hiện tại của họ.

Phụ thuộc vào nhà sản xuất: Các nhà mạng và các công ty cố gắng áp dụng các khả năng và dịch vụ mới trong việc đáp ứng các nhu cầu (những nhu cầu này thay đổi liên tục và rất nhanh) kinh doanh hoặc nhu cầu người dùng. Tuy nhiên khả năng của họ phụ thuộc vào các chu kỳ cập nhật firmware thiết bị của nhà sản xuất. Và điều đáng nói là những chu kỳ này có thể kéo dài lên đến 3 năm hoặc nhiều hơn nữa. Ngoài ra việc thiếu các chuẩn hóa, hay giao diện mở làm giới hạn khả năng điều chỉnh mạng của các nhà mạng. Chính sự không tương ứng giữa nhu cầu trên thị trường và khả năng của mạng đã dẫn đến “điểm gãy khúc”. Đáp lại vấn đề này, mạng điều khiển bởi phần mềm SDN (Software-Defined Networking) đã được tạo ra.

2.1.2 Khái niệm và cấu trúc mạng SDN

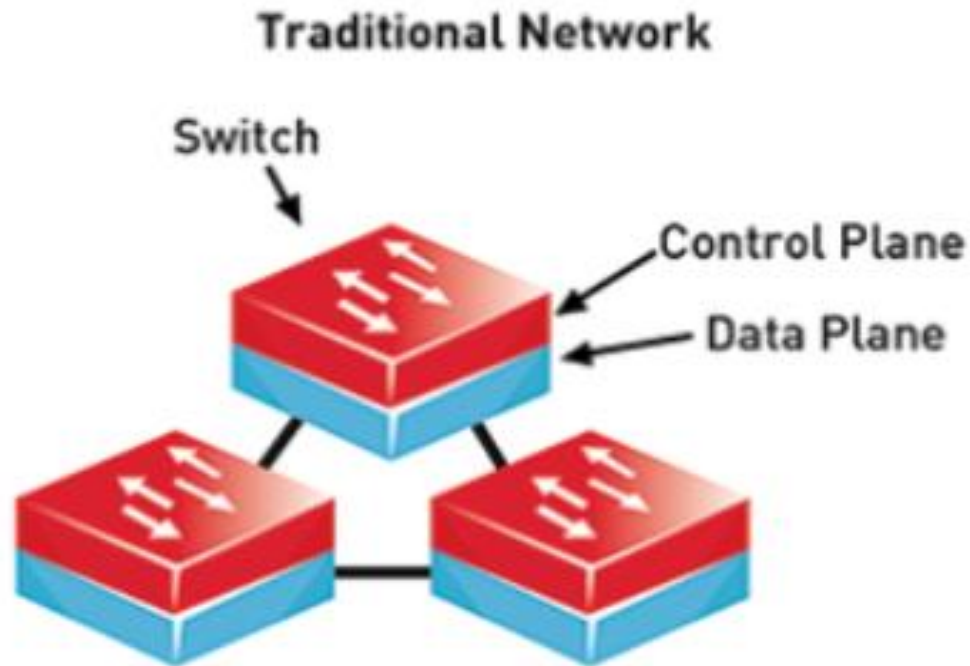
Trước khi đưa ra khái niệm về SDN, ta thử đặt ra một giả thiết là nếu ta có thể tách rời phần điều khiển ra khỏi các thiết bị mạng thì điều đó có thể làm cho khả năng xử lý của thiết bị tăng lên hay không? Có thể tạo ra một mạng thông minh hơn và linh hoạt hơn hay không?

Thực tế là dựa trên giả thiết đó, người ta đã nghiên cứu và phát triển thành một mạng mà ở đó nhiệm vụ điều khiển mạng được xử lý bởi các bộ điều khiển và các bộ điều khiển đó có thể thao tác tới phần cứng, bộ nhớ và các chức năng của các thiết bị router, switch để đạt được mục đích của người sử dụng. Do đó, mạng trở nên linh hoạt hơn, hiệu suất sử dụng cao hơn và dễ quản lý hơn bao giờ hết. Để hiểu rõ hơn ta xem xét sự khác nhau giữa chức năng của các thiết bị của mạng truyền thống và mạng SDN.



Hình 2. 1 Chức năng cơ bản của một Router/ Switch

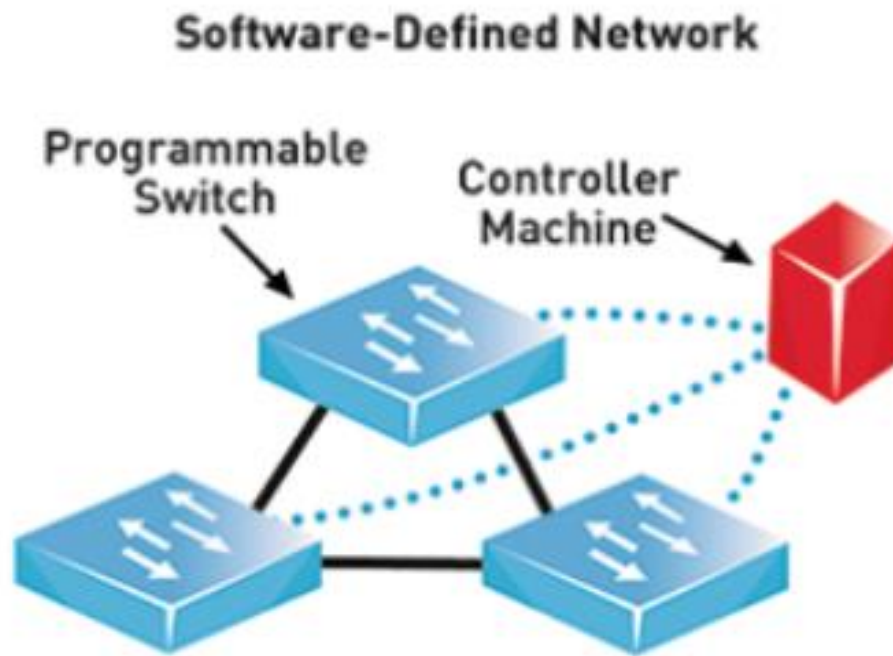
Theo như hình 2.1 ta thấy một router/switch bình thường bao gồm 2 phần đó là phần cứng và phần mềm. Phần mềm đảm bảo chức năng trao đổi các thông tin với các thiết bị router/switch khác và tính toán các con đường định tuyến dựa trên các thông tin đã thu thập được. Phần cứng đảm nhiệm chức năng chuyển tiếp các gói tin đến theo một lộ trình đã được tính toán sẵn.



Hình 2. 2 Sơ đồ một mạng truyền thống đơn giản.

Đối với mạng truyền thống thì các thiết bị định tuyến hoặc chuyển mạch trao đổi các thông tin với nhau và quá trình tính toán xử lý đều xảy ra ở mỗi node mạng (ở tại mỗi router/switch). Điều đó được mô tả ở trên Hình 2.2.

Chức năng chính của các thiết bị mạng như router/switch là vận chuyển dữ liệu, như ta thấy ở mô hình trên thì các thiết bị không được hoàn toàn tập trung vào chức năng đó. Nhưng đối với mạng SDN thì điều đó lại là khác.



Hình 2. 3 Sơ đồ mạng đơn giản với bộ điều khiển SDN.

Theo như hình 2.3 thì ta thấy việc thu thập thông tin của các thiết bị trong mạng và tính toán xử lý các thông tin thu thập được đều được chuyển đến một bộ điều khiển mạng (Network Controller). Các thiết bị router/switch chỉ tập trung vào chức năng vận chuyển dữ liệu. Điều đó làm cho việc quản lý mạng trở nên đơn giản hơn và các thiết bị phần cứng có thể nâng công suất làm việc lên.

– Từ sự so sánh trên ta rút ra được một số điểm khác nhau giữa 2 mạng đó là:

- Phần điều khiển và phần vận chuyển dữ liệu:

Mạng truyền thống: Được tích hợp trong thiết bị mạng.

Mạng SDN: Phần điều khiển được tách riêng khỏi thiết bị mạng và được chuyển đến một thiết bị được gọi là bộ điều khiển SDN.

- Phần thu thập và xử lý các thông tin:

Mạng truyền thống: Được thực hiện ở tất cả các phần tử trong mạng.

Mạng SDN: Được tập trung xử lý ở bộ điều khiển SDN.

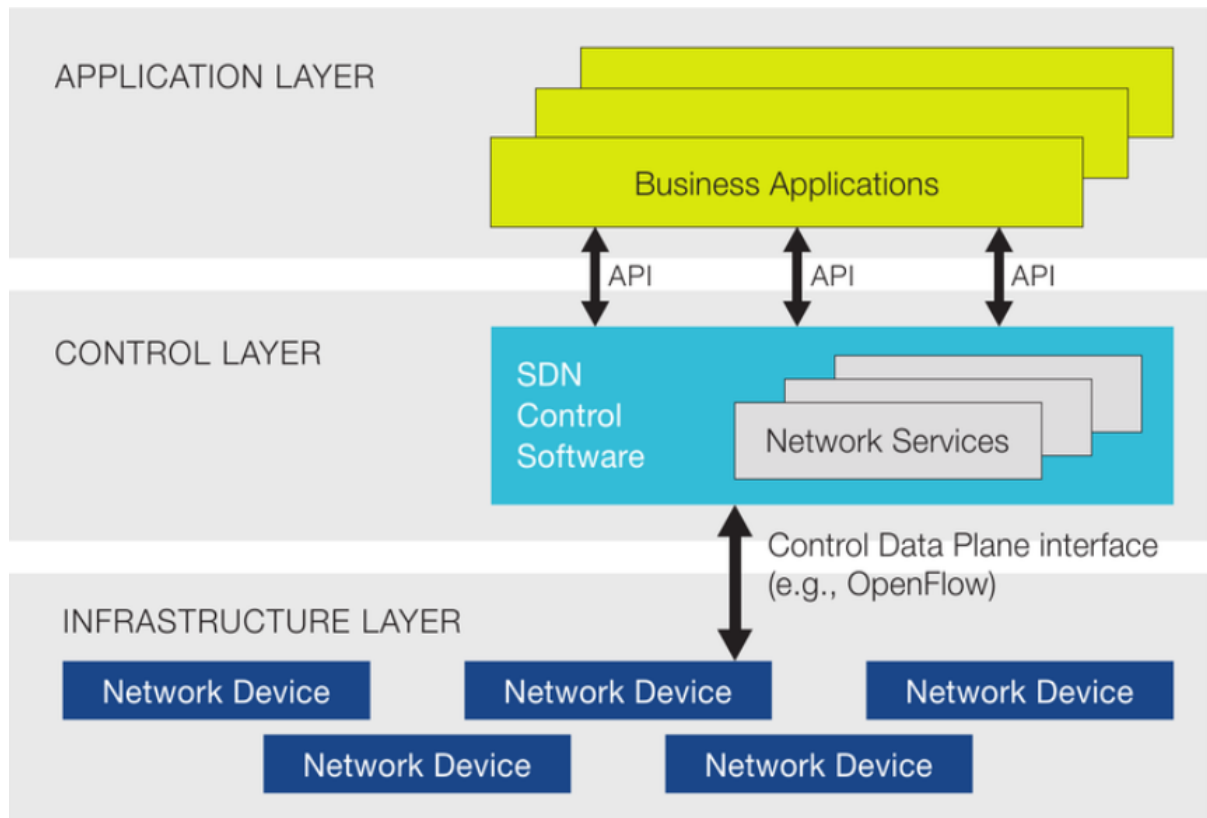
- Khả năng lập trình bởi các ứng dụng:

Mạng truyền thống: mạng không thể được lập trình bởi các ứng dụng. Các thiết bị mạng phải được cấu hình một cách riêng lẻ và thủ công.

Mạng SDN: Mạng có thể lập trình bởi các ứng dụng, bộ điều khiển SDN có thể tương tác đến tất cả các thiết bị trong mạng.

Hiện nay có rất nhiều định nghĩa về mạng SDN nhưng theo ONF (Open Networking Foundation – một tổ chức phi lợi nhuận đang hỗ trợ việc phát triển SDN thông qua việc nghiên cứu các tiêu chuẩn mở phù hợp) thì mạng SDN được định nghĩa như sau: “ Software Defined Network (SDN) là một kiểu kiến trúc mạng mới, năng động, dễ quản lý, chi phí hiệu quả, dễ thích nghi và rất phù hợp với nhu cầu mạng ngày càng tăng hiện nay. Kiến trúc này phân tách phần điều khiển mạng (Control Plane) và chức năng vận chuyển dữ liệu (Forwarding Plane or Data Plane), điều này cho phép việc điều khiển mạng trở nên có thể lập trình và cơ sở hạ tầng mạng độc lập với các ứng dụng và dịch vụ mạng”.

Phần điều khiển được tách rời và được tập trung ở bộ điều khiển SDN. Điều này có nghĩa là các thiết bị mạng ở lớp thiết bị phần cứng không cần phải hiểu và xử lý các giao thức phức tạp mà chúng chỉ chấp nhận và vận chuyển dữ liệu theo một con đường nào đó dưới sự chỉ huy của bộ điều khiển SDN. Dựa vào bộ điều khiển SDN mà các nhà khai thác mạng và quản trị mạng có thể lập trình cấu hình trên đó thay vì phải thực hiện thủ công hàng ngàn câu lệnh cấu hình trên các thiết bị riêng lẻ. Ngoài ra nó còn có thể triển khai các ứng dụng mới và các dịch vụ mạng một cách nhanh chóng. Để có cái nhìn tổng quan hơn về SDN, ta xét tới cấu trúc của nó.



Hình 2. 4 Kiến trúc mạng SDN.

Hình trên cho ta thấy một cái nhìn tổng quát về mạng SDN. Trung tâm là bộ điều khiển SDN thực hiện tất cả các chức năng phức tạp như định tuyến, đưa ra các chính sách mạng, kiểm tra bảo mật... Tất cả các bộ điều khiển này cấu thành nên lớp điều khiển của mạng SDN.

Các bộ điều khiển SDN xác định các luồng dữ liệu sẽ đi qua trong lớp dữ liệu phía dưới và mỗi luồng dữ liệu đi qua mạng đều phải có sự cho phép của bộ điều khiển SDN và khi được sự cho phép đó thì bộ điều khiển sẽ tính toán một lộ trình cho dòng dữ liệu đó. Cụ thể là các bộ điều khiển sẽ thiết lập một tập hợp dữ liệu nội bộ sử dụng để tạo ra các entry của bảng chuyển tiếp, những bảng này lần lượt được sử dụng bởi lớp chuyển tiếp (data plane) để truyền các luồng dữ liệu giữa các cổng vào và ra trên thiết bị. Tập hợp dữ liệu này được sử dụng để lưu trữ topo mạng và được gọi là thông tin định tuyến RIB (RIB – Routing Information Base). RIB thường được duy trì đồng nhất bằng cách trao đổi thông tin giữa các lớp điều khiển trong mạng. Các entry của bảng chuyển tiếp thường được gọi là

thông tin chuyển tiếp FIB (FIB – Forwarding Information Base) và thường được ánh xạ giữa mặt phẳng điều khiển và mặt phẳng chuyển tiếp của các thiết bị điển hình.

Dựa vào các thông tin chuyển tiếp mà bộ điều khiển cung cấp, các thiết bị ở lớp chuyển tiếp xử lý các gói tin đầu vào và tìm kiếm, so sánh với bảng thông tin định tuyến để xử lý với các gói tin. Vì vậy các thiết bị chuyển mạch ở lớp dưới chỉ đơn giản là quản lý các bảng “định tuyến” được cung cấp bởi bộ điều khiển SDN. Giao tiếp giữa lớp điều khiển và các thiết bị ở lớp dữ liệu là các giao diện mở API và hiện tại đang phổ biến đó là sử dụng các API của giao thức OpenFlow. Giao thức này sẽ được đề cập ở chương sau.

– Kiến trúc SDN rất linh hoạt, nó có thể hoạt động với các loại thiết bị chuyển mạch và các giao thức khác nhau. Trong kiến trúc SDN, thiết bị chuyển mạch thực hiện các chức năng sau:

- Đóng gói và chuyển tiếp các gói tin đầu tiên đến bộ điều khiển SDN để bộ điều khiển SDN quyết định các flow entry sẽ được thêm vào flow table của switch.
- Chuyển tiếp các gói tin đến các cổng thích hợp dựa trên flow table.
- Flow table có thể bao gồm các thông tin ưu tiên được quyết định bởi bộ điều khiển SDN.
- Switch có thể hủy các gói tin trên một luồng riêng một cách tạm thời hoặc vĩnh viễn nhưng dưới sự cho phép của bộ điều khiển.

Nói một cách đơn giản, bộ điều khiển SDN quản lý các trạng thái chuyển tiếp của các switch trong mạng SDN. Việc quản lý này được thực hiện thông qua một bộ giao diện mở API, nó cho phép bộ điều khiển SDN có thể giải quyết các yêu cầu hoạt động mà không cần thay đổi bất kỳ các khía cạnh cấp dưới của mạng, bao gồm cả mô hình mạng.

Với sự tách rời miền điều khiển và miền dữ liệu, SDN cho phép các ứng dụng được triển khai một cách dễ dàng mà không cần quan tâm chi tiết đến việc hoạt động của các thiết bị mạng.

2.1.3 Ưu nhược điểm của SDN so với mạng IP

– Đã có rất nhiều cuộc tranh luận nổ ra để bàn về vấn đề mạng IP và mạng SDN loại nào tốt hơn. Mặc dù cả hai đều có những ưu nhược điểm riêng nhưng với những thuộc tính quan trọng như thân thiện với người sử dụng, chi phí và độ phức tạp mạng giảm thì người ta cho rằng mạng SDN phù hợp hơn so với mạng IP. Và bởi vì thế mà ngày nay một số lượng ngày càng tăng của các nhà khai thác mạng đang lựa chọn SDN nhiều hơn. Một số ưu điểm của SDN như sau:

- Dựa vào SDN các nhà quản trị có thể có quyền kiểm soát mạng một cách đơn giản và hiệu quả mà không cần có quyền truy cập trực tiếp đến phần cứng.
- Thêm vào đó SDN cung cấp một cơ chế điều khiển duy nhất đối với cơ sở hạ tầng mạng và giảm bớt sự phức tạp của các quá trình xử lý thông qua sự tự động hóa. Điều đó rất có lợi đối với các nhà mạng để có thể quản lý các thay đổi của mạng theo thời gian thực. Thời gian thực ở đây có nghĩa là các nhà mạng có thể đáp ứng một cách nhanh chóng các yêu cầu của mạng một cách nhanh chóng và tự động. Các nhà quản trị còn có thể điều khiển việc thay đổi cần thiết đúng thời điểm ở bất cứ nơi đâu. Việc truy cập từ xa và thay đổi mạng có thể được thực hiện bởi hệ thống truy cập dựa trên vai trò của người quản trị (Role based access system là một hệ thống cho phép người dùng cá nhân dựa trên một vai trò xác định trong doanh nghiệp để có thể thực hiện các thay đổi của mạng). Hệ thống truy cập này được cung cấp các giải pháp bảo mật để có thể loại bỏ sự tấn công của các tin tặc vào mạng. Đối với mạng IP, việc truy cập và thay đổi từ xa lại không thể thực hiện được. Nhà quản trị phải có quyền truy cập và chỉnh sửa cấu hình bằng tay để thực hiện bất kỳ sự thay đổi chính sách nào trên mạng. Việc thay đổi một chính sách mạng dẫn đến việc tác động trực tiếp đến phần cứng đó làm cho hệ thống mạng trở nên cứng nhắc.
- SDN cho phép sử dụng không hạn chế và có thể thay đổi các chính sách mạng để phát hiện sự xâm nhập, tường lửa và tạo sự cân bằng với sự thay đổi của phần mềm. Điều đó làm cho sự quản lý mạng trở nên linh hoạt hơn.

- Mạng SDN có khả năng phân tách phần điều khiển và phần dữ liệu, với khả năng đó cho phép người quản trị có thể tương tác và thay đổi các luồng dữ liệu, đảm bảo các gói dữ liệu không phải xếp hàng đợi và làm giảm hiệu suất mạng. Một lợi thế quan trọng hơn nữa của mạng SDN là chi phí dành cho nó rất thấp. Nó rẻ hơn so với mạng IP bởi vì nó không yêu cầu nhiều người làm việc trên nó và các công ty đều có thể cắt hết chi phí của các kỹ sư hệ thống và chỉ cần một vài quản trị viên hệ thống là đủ.
- Bên cạnh những ưu điểm đó thì SDN cũng còn tồn tại một số nhược điểm so với mạng IP.
 - Vấn đề đầu tiên là bảo mật, nếu tin tặc có thể tấn công vào hệ thống thì chúng có thể truy cập các thiết lập và thay đổi chúng bất cứ ở nơi đâu, tại thời điểm nào. Và chúng có thể truy cập bất kỳ tập tin được mã hóa nào miễn là nó ở trong mạng. Đối với mạng IP thì điều này không thể xảy ra bởi để có thể truy cập vào mạng ta phải có quyền truy cập vào phần cứng của nó. Hầu hết các công ty chỉ cho phép một số cá nhân được quyền đó bởi vậy hệ thống sẽ an toàn và ít có khả năng bị truy cập bởi các tin tặc.
 - Thứ hai đó là quá trình triển khai SDN không thể hoàn thiện trong chốc lát mà nó phải theo từng bước một. Chúng ta không thể một lúc thay thế toàn bộ các thiết bị hiện có thành OpenFlow switch được bởi vì điều đó rất tốn kém.
 - Thứ ba, SDN là một kiến trúc mạng kiểu mới, các giao thức tương tác giữa các controller với nhau còn chưa được phát triển toàn diện nên việc phát triển mạng SDN trên phạm vi toàn cầu vẫn đang còn hạn chế.
- Nói tóm lại SDN có một số ưu điểm đáng chú ý sau:
 - Có các giao diện mở, không phụ thuộc nhà sản xuất.
 - Trừu tượng hóa mặt phẳng điều khiển cho phép áp dụng nhanh chóng các đổi mới.
 - Giảm tính phức tạp nhờ tự động hóa.
 - Nâng cao độ tin cậy và bảo mật cho mạng.
 - Kiểm soát và điều khiển chi tiết hơn.

- Giảm CAPEX và OPEX.

2.1.4 Ứng dụng của SDN:

Với những lợi ích mà mình đem lại, SDN có thể triển khai trong phạm vi các doanh nghiệp hoặc trong các nhà cung cấp hạ tầng và dịch vụ viễn thông để giải quyết các yêu cầu của các nhà cung cấp tại mỗi phân khúc thị trường.

2.1.4.1 Phạm vi doanh nghiệp:

2.1.4.1.1 Áp dụng trong mạng doanh nghiệp:

Mô hình tập trung, điều khiển và dự phòng tự động của SDN hỗ trợ việc hội tụ dữ liệu, voice, video, cũng như là việc truy cập tại bất kỳ thời điểm nào, bất kỳ nơi đâu. Điều này được thực hiện thông qua việc cho phép các nhà quản trị mạng thực thi chính sách nhất quán trên cả cơ sở hạ tầng không dây lẫn có dây. Ngoài ra, SDN hỗ trợ việc quản lý và giám sát tự động tài nguyên mạng, xác định bằng các hồ sơ cá nhân và các yêu cầu ứng dụng, để đảm bảo tối ưu trải nghiệm người dùng với khả năng của nhà mạng.

2.1.4.1.2 Áp dụng trong các trung tâm dữ liệu (Data Center):

Việc ảo hóa các thực thể mạng của kiến trúc SDN cho phép việc mở rộng trong các trung tâm dữ liệu, dịch chuyển tự động các máy ảo, tích hợp chặt chẽ hơn với kho lưu trữ, sử dụng server tốt hơn, sử dụng năng lượng thấp hơn và tối ưu được băng thông hơn.

2.1.4.1.3 Áp dụng với dịch vụ điện toán đám mây (cloud):

Mặc dù được sử dụng để hỗ trợ cho điện toán đám mây riêng hay môi trường điện toán đám mây lai, SDN cho phép tài nguyên mạng được phân bố một cách linh hoạt, điều đó cho phép sự đáp ứng nhanh chóng của các dịch vụ điện toán đám mây và tạo sự chuyển giao linh hoạt hơn đến cho các nhà cung cấp điện toán đám mây bên ngoài. Với các công cụ an toàn để quản lý mạng ảo của họ, các doanh nghiệp và các đơn vị kinh doanh sẽ tin tưởng vào các dịch vụ đám mây nhiều hơn nữa.

2.1.4.2. Phạm vi các nhà cung cấp hạ tầng và dịch vụ viễn thông:

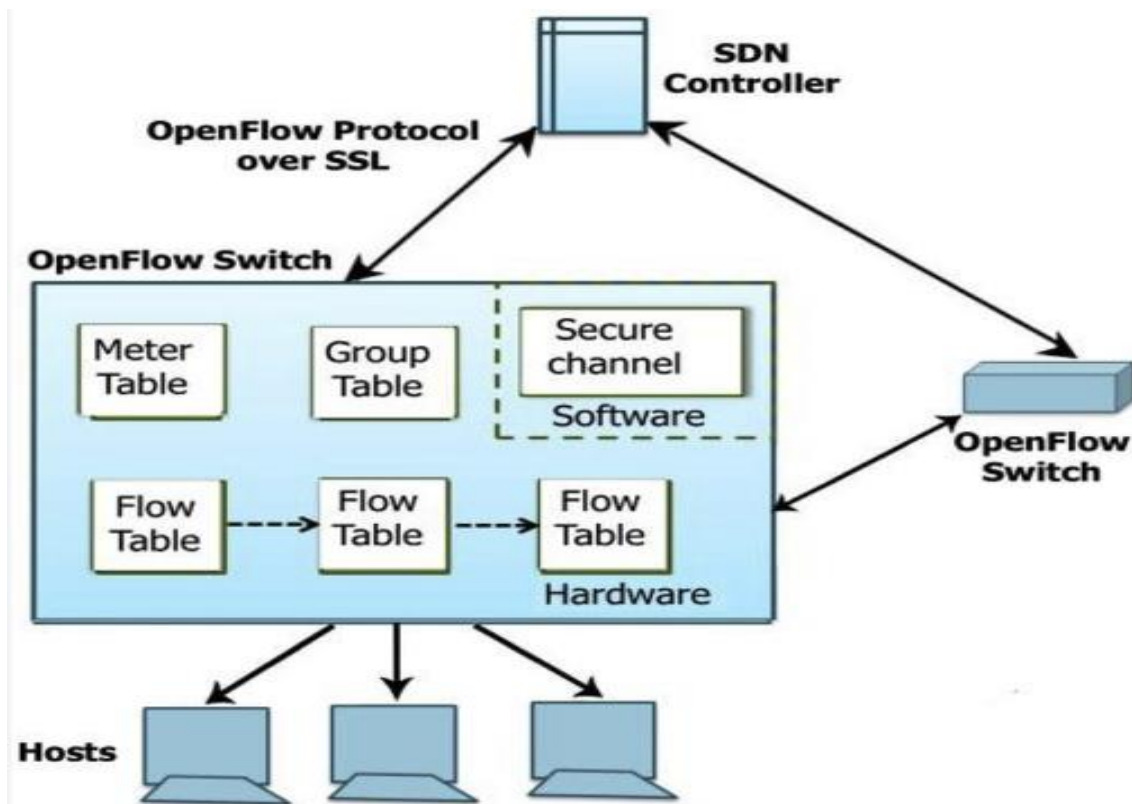
SDN cung cấp cho các nhà mạng, các nhà cung cấp đám mây công cộng và các nhà cung cấp dịch vụ, sự mở rộng và tự động thiết kế để triển khai một mô hình tính toán có ích cho ItaaaS (IT as a Service). Điều này thực hiện thông qua việc đơn giản hóa triển khai các dịch vụ tùy chọn và theo yêu cầu, cùng với việc chuyển dời sang mô hình selfservice. Mô hình tập trung, dự phòng và điều khiển tự động của SDN dễ dàng hỗ trợ cho thuê linh hoạt các tài nguyên, đảm bảo tài nguyên mạng được triển khai ở mức tối ưu, giảm CAPEX và OPEX và tăng giá trị và tốc độ dịch vụ.

2.1.5 Các mô hình triển khai SDN

Hiện nay có 3 phương pháp chủ yếu để triển khai mạng SDN như Switch based, overlay và phương pháp tổng hợp của 2 phương pháp trên.

2.1.5.1 Switch based

Ở mô hình này, các giao thức điều khiển SDN được đưa ra trực tiếp từ bộ điều khiển SDN (máy ảo) đến lớp điều khiển, lớp dữ liệu với các SDN switch.



Hình 2. 5 Mô hình dựa trên Switch.

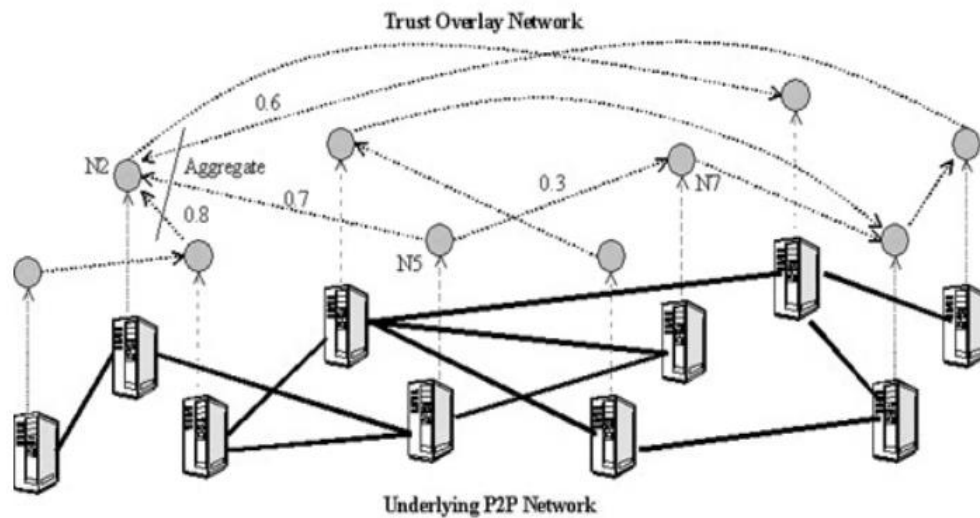
Khi một gói tin đến switch trong một mạng thông thường, dựa vào các giao thức được xây dựng sẵn trong switch nó sẽ biết được nơi sẽ chuyển tiếp gói tin đến. Switch sẽ gửi các gói tin đi đến cùng một địa điểm, cùng một con đường và nó đối xử với các gói tin là như nhau. Trong các doanh nghiệp, các switch thông minh được thiết kế với các bảng mạch tích hợp ứng dụng cụ thể ASIC (Application Specific Integrated Circuits) điều đó giúp cho các switch nhận biết được các loại gói tin và xử lý chúng một cách thích hợp. Với việc có thêm các bảng mạch tích hợp ASIC làm cho các thiết bị switch trở nên đắt hơn so với các thiết bị chuyển mạch thông thường.

Đối với mạng SDN được mô tả như hình trên thì người quản trị mạng có thể quản lý các lưu lượng dữ liệu từ một thiết bị kiểm soát trung tâm mà không cần phải tác động trực tiếp vào từng switch. Người quản trị mạng có thể thay đổi bất cứ các quy tắc chuyển mạch lúc cần thiết như ưu tiên, không ưu tiên hoặc thậm chí có thể chặn một số gói tin đặc thù nào đó. Điều này đặc biệt hữu ích cho kiến trúc đám mây bởi vì nó cho phép người quản trị có thể quản lý các luồng dữ liệu một cách linh hoạt và hiệu quả hơn. Kiến trúc này còn cho phép các kỹ sư mạng hỗ trợ đa kết nối qua các thiết bị phần cứng của nhiều nhà cung cấp khác nhau.

Điểm hạn chế lớn nhất của phương pháp này là không thể tận dụng tất cả các thiết bị mạng lớp 3, lớp 2 của mạng hiện tại.

2.1.5.2 Overlay Network

Phương pháp triển khai này có thể tận dụng các thiết bị của mạng IP hiện có bằng cách ảo hóa. Các nguồn dữ liệu và máy chủ duy trì các thiết bị ảo và cũng là một phần trong môi trường ảo. Ở mô hình này, các giao thức điều khiển của SDN được đi trực tiếp từ bộ điều khiển SDN (máy ảo) đến các thiết bị chuyển mạch ảo (Hypervisor switch) để kiểm soát các thiết bị mạng IP ở lớp dưới.



Hình 2. 6 Overlay Network SDN.

Mô hình triển khai này yêu cầu sử dụng các thiết bị chuyển mạch ảo để đáp ứng các lệnh đến các thiết bị mạng IP. Các chuyển mạch ảo là các máy ảo chịu trách nhiệm giao tiếp giữa thiết bị mạng IP với các máy ảo và các ứng dụng mạng SDN. Mô hình dùng chuyển mạch ảo có 2 chức năng đó là chức năng vận chuyển của lớp 2 thông qua một mô đun Ethernet ảo (Virtual Ethernet Module – VEM) và tuân thủ các chính sách giám sát.

Mô đun Ethernet ảo cung cấp thông tin cấu hình, hỗ trợ chuyển mạch lớp 2 và hỗ trợ các chức năng nâng cao của mạng như cấu hình cho các cổng, chất lượng dịch vụ, bảo mật cho các cổng, Vlan và điều khiển truy cập. Ngoài ra, khi mất liên lạc với các chuyển mạch ảo, mô đun Ethernet ảo có hỗ trợ chức năng Nonstop Forwarding (NSF) để có thể tiếp tục chuyển tiếp lưu lượng dựa trên cấu hình cuối cùng mà các bộ chuyển mạch được biết. Như vậy, mô đun Ethernet ảo cung cấp khả năng chuyển mạch với độ tin cậy cao cho môi trường máy chủ ảo.

Để kiểm soát nhiều mô đun Ethernet ảo người ta sử dụng một bộ giám sát ảo (Virtual Supervisor Module- VSM). Thay vì sử dụng nhiều thẻ chức năng vật lý, VSM hỗ trợ chạy nhiều VEM bên trong một máy chủ vật lý. Cấu hình được thực hiện thông qua VSM và tự động chuyển đến các VEM. Thay vì cấu hình các chuyển mạch mềm bên trong các hypervisor trên cơ sở các máy chủ với nhau, người quản trị có thể cấu hình ngay lập

tức trên tất cả các VEM được quản lý bởi VSM từ một giao diện duy nhất. VSM còn cung cấp chức năng cấu hình các port thông qua phần mềm.

Cách triển khai này có ưu điểm là sử dụng được cơ sở hạ tầng mạng IP cũ nhưng nó cũng sẽ gây khó khăn cho các nhà quản trị vì phải duy trì hệ thống cũ và sửa chữa các vấn đề về định tuyến trong mạng SDN.

2.1.5.3 Mạng lai

Cách triển khai này là sự kết hợp giữa 2 phương pháp switch based và overlay. Phương pháp này được sử dụng tận dụng mạng lưới IP cũ và dần dần sẽ loại bỏ mạng lưới cũ để chuyển sang hoàn toàn dạng sử dụng các switch SDN. Điều này cho phép các doanh nghiệp có thể kiểm soát tốc độ triển khai SDN và kiểm soát tỷ lệ đầu tư trang thiết bị.

2.1.6 Kết luận

Với xu hướng người dùng di động, ảo hóa máy chủ và các dịch vụ ngày càng tăng dẫn đến kiến trúc mạng thông thường ngày nay không thể đáp ứng xử lý kịp. Mạng SDN cho chúng ta một cái nhìn mới, khái niệm mới về một kiến trúc mạng năng động, dễ thích nghi, mở rộng và đáp ứng các dịch vụ phong phú. Với việc tách phần điều khiển và dữ liệu, kiến trúc mạng SDN cho phép mạng có thể lập trình và quản lý một cách dễ dàng hơn. SDN hứa hẹn sẽ chuyển đổi mạng lưới tĩnh ngày nay trở nên linh hoạt hơn với nền tảng có thể lập trình với sự thông minh để có thể tự động xử lý các hành vi một cách tự động. Với nhiều lợi thế của mình và động lực phát triển cao SDN đang trên đường để trở thành một tiêu chuẩn mới cho các mạng.

2.2 Giao thức OpenFlow

Ý tưởng SDN đã bắt đầu được gần 10 năm, nhưng gần đây SDN mới bắt đầu được thực hiện bởi các công ty như Cisco Systems và Juniper Networks. Tuy nhiên các nhà sản xuất và khai thác mạng đã và đang bắt đầu làm quen với OpenFlow, một công nghệ hứa hẹn sẽ mang đến khả năng tương tác và hiệu suất hoạt động cao cho SDN. Với sự giúp đỡ của OpenFlow controller, các nhà quản trị mạng có thể xác định các cách thức và tuyến đường để truyền dữ liệu, thiết lập các quy tắc ưu tiên cho việc xử lý các gói tin và chuyển hướng dữ liệu qua các thiết bị chuyển mạch của mạng nội bộ hay mạng toàn cầu. Chương này sẽ giới thiệu một cách tổng quan cho chúng ta biết về giao thức OpenFlow cũng như các thức hoạt động và các lợi ích mà nó mang lại.

2.2.1 Lịch sử phát triển của OpenFlow

Vào tháng 3 năm 2011, các công ty Cisco, Facebook, Google, Microsoft... và nhiều công ty khác đã thành lập nên tổ chức Open Networking Foundation (ONF) để thúc đẩy công nghệ OpenFlow và giao thức chuyển mạch OpenFlow Switching Protocol. Tuy nhiên, một số chuyên gia cho rằng OpenFlow không có đủ khả năng để triển khai trên diện rộng và các nhà sản xuất có thể thêm vào công nghệ này các phần mở rộng độc quyền của mình, điều này làm mất đi khả năng tương tác vốn có của OpenFlow. Ngày nay với sự phát triển nhanh chóng và rộng khắp của điện toán đám mây đã kích thích các nhu cầu về tính linh hoạt, độ tin cậy, an toàn và cần được quản lý tốt của mạng xương sống. Để giải quyết vấn đề này, cần phải có các hệ thống điều khiển thông minh hơn, hiệu quả hơn, những hệ thống cho phép phối hợp hoạt động của hàng ngàn thiết bị định tuyến và chuyển mạch. Hiện nay những thiết bị này chỉ cung cấp cho người sử dụng các khả năng tái lập trình một cách hạn chế, và để nâng cao tính hiệu quả ở các trung tâm xử lý dữ liệu (Data Center), những người quản trị hệ thống cần một sự kiểm soát chi tiết hơn, và khả năng mở rộng cao hơn. Trong khi đó, mỗi nhà cung cấp có các

2.2.2 Giao thức OpenFlow

Giao thức OpenFlow được sử dụng để điều khiển và quản lý mạng trong mô hình mạng SDN. Giao thức này được phát triển bởi Open Networking Foundation (ONF) vào năm 2008 và được sử dụng phổ biến trong các mô hình mạng SDN hiện nay.

Giao thức OpenFlow được thiết kế để tách biệt bộ điều khiển (controller) và bộ chuyển tiếp (switch) trong mạng. Bộ điều khiển là nơi quyết định cách thức chuyển tiếp các gói tin trên mạng, trong khi bộ chuyển tiếp chỉ đơn giản là thực hiện các yêu cầu được gửi từ bộ điều khiển.

Giao thức OpenFlow sử dụng một bộ lệnh chuẩn để gửi các yêu cầu từ bộ điều khiển đến bộ chuyển tiếp. Bộ lệnh này bao gồm các trường như: địa chỉ nguồn, địa chỉ đích, giao thức, cổng nguồn, cổng đích, v.v. Mỗi bộ lệnh được đánh dấu bằng một số duy nhất gọi là "flow entry", được định nghĩa bởi bộ điều khiển và được gửi đến các bộ chuyển tiếp để cài đặt các quy tắc chuyển tiếp gói tin trên mạng.

Giao thức OpenFlow còn hỗ trợ các tính năng như: thu thập thông kê về lưu lượng mạng, đưa ra quyết định điều khiển dựa trên trạng thái thời gian thực của mạng và cấu hình mạng linh hoạt.

Trong mô hình mạng SDN, giao thức OpenFlow là một phần quan trọng để kết nối bộ điều khiển và bộ chuyển tiếp, tạo ra sự linh hoạt và quản lý hiệu quả trên mạng.

2.2.3 Kiến trúc và hoạt động của OpenFlow

2.2.3.1 Kiến trúc

OpenFlow hoàn toàn có thể được sử dụng bởi những ứng dụng ứng dụng bên ngoài để tinh chỉnh và điều khiển lớp chuyển tiếp của những thiết bị mạng, giống như tập lệnh của CPU điều khiển mạng lưới hệ thống máy tính.

Giao thức OpenFlow được tiến hành trên cả hai giao diện của liên kết giữa những thiết bị hạ tầng mạng và ứng dụng tinh chỉnh và điều khiển SDN. OpenFlow sử dụng khái niệm “ flow ” (luồng) để nhận dạng lưu lượng mạng trên cơ sở định nghĩa trước những quy tắc tương thích (được lập trình sẵn bởi ứng dụng tinh chỉnh và điều khiển SDN). Giao thức này còn cho phép định nghĩa những tham số, ví dụ điển hình như quy mô lưu lượng sử dụng, ứng dụng và tài nguyên thiết yếu để xác lập phương pháp điều hướng lưu lượng truyền qua những thiết bị mạng. Do đó OpenFlow được cho phép mạng được lập trình dựa trên cơ sở luồng lưu lượng. Một kiến trúc SDN trên cơ sở OpenFlow phân phối điều khiển và tinh chỉnh ở mức cực kỳ chi tiết cụ thể, được cho phép mạng phản hồi sự đổi khác theo thời hạn thực của ứng dụng, người dùng và mức phiên.

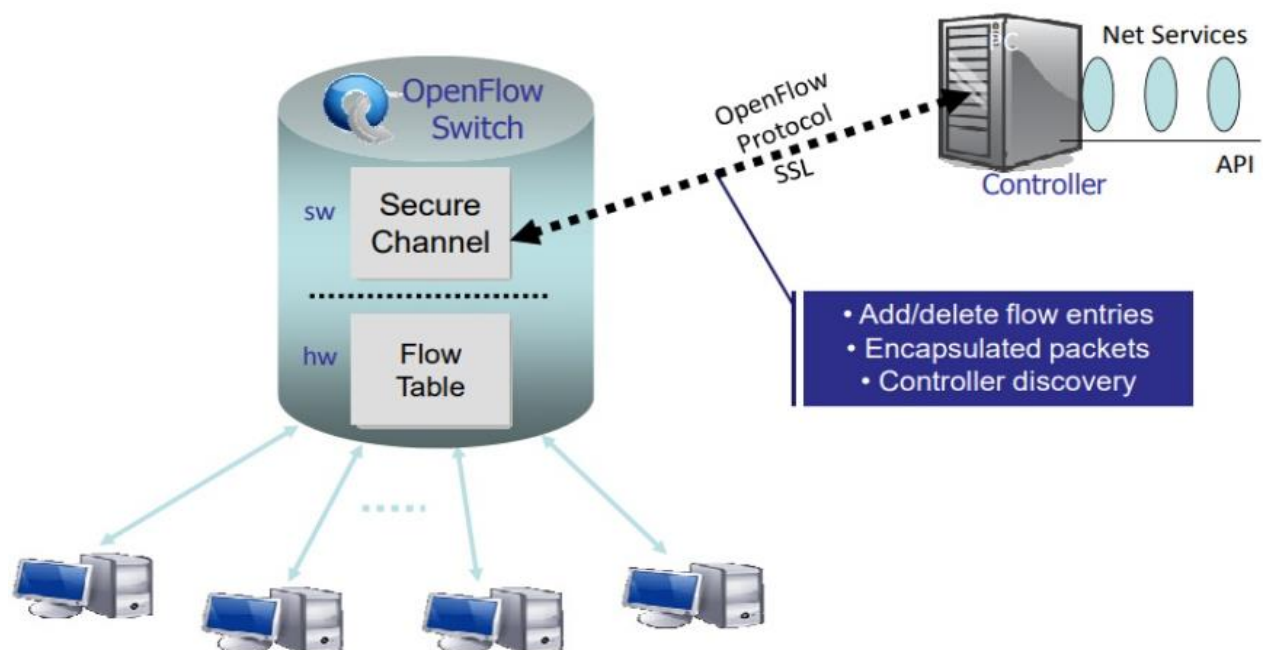
Mạng định tuyến trên cơ sở IP hiện tại không phân phối mức này của điều khiển và tinh chỉnh, tổng thể những luồng lưu lượng giữa hai điểm cuối phải theo cùng một đường trải qua mạng, mặc dầu nhu yếu của chúng khác nhau. Giao thức OpenFlow là một chìa khóa để cho phép những mạng định nghĩa bằng ứng dụng và cũng là giao thức tiêu chuẩn SDN duy nhất được cho phép tinh chỉnh và điều khiển lớp chuyển tiếp của những thiết bị mạng.

Một thiết bị OpenFlow bao gồm ít nhất 3 thành phần: Bảng luồng (Flow table), Kênh an toàn (Secure Chanel) và Giao thức Openflow (OpenFlow Protocol).

Flow Table: một liên kết hành động với mỗi luồng, giúp thiết bị xử lý các luồng. Secure Channel: kênh kết nối thiết bị tới bộ điều khiển (controller), cho phép các lệnh và các gói tin được gửi giữa controller và thiết bị. OpenFlow Protocol: giao thức cung cấp phương thức tiêu chuẩn mở cho một controller truyền thông với thiết bị.

Secure Channel : kênh liên kết thiết bị tới bộ tinh chỉnh và điều khiển (controller), được cho phép những lệnh và những gói tin được gửi giữa controller và thiết bị.

OpenFlow Protocol : giao thức phân phối phương pháp tiêu chuẩn mở cho một controller tiếp thị quảng cáo với thiết bị .Hình 2.7. Một ví dụ về Flow Table trên một thiết bị.

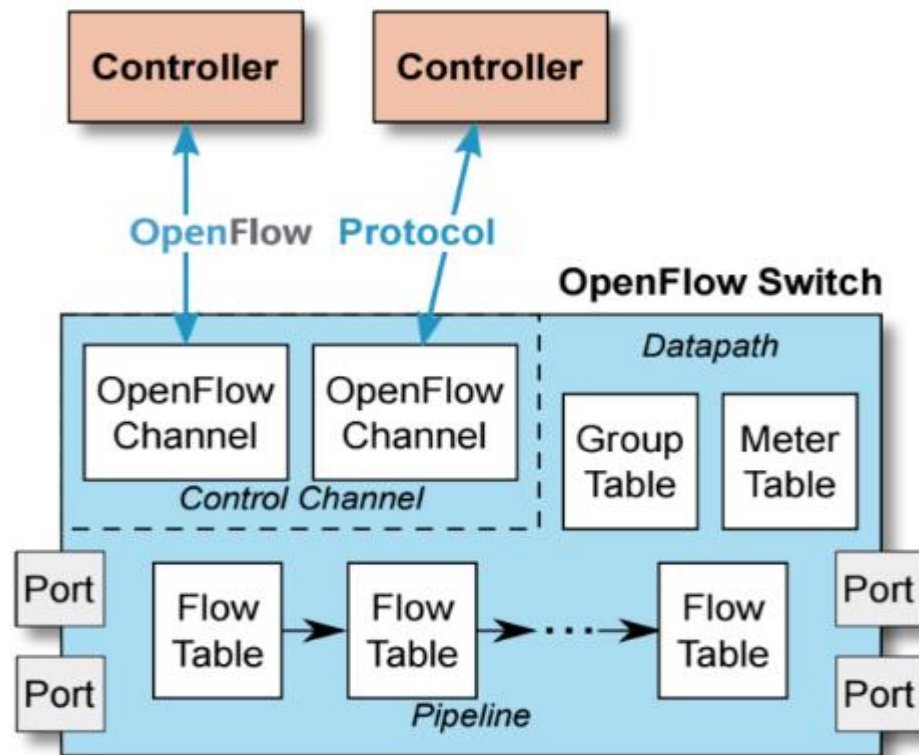


Hình 2. 7 Flow table trên một thiết bị

2.2.3.2 Hoạt động

Hầu hết các bộ chuyển mạch Ethernet hiện đại đều sử dụng bảng lưu lượng. Các bảng này giúp truyền các gói hiệu quả từ người gửi đến người nhận. Mỗi nhà cung cấp sẽ có bảng lưu lượng riêng, tuy nhiên vẫn có thể tách biệt một số chức năng được coi là chung cho tất cả các switch.

OpenFlow tách riêng các tính năng của Lớp Truyền Tài liệu và Lớp Kiểm soát và Tinh chỉnh với nhau. Chức năng lập bản đồ để truyền tài liệu vẫn diễn ra trên bộ chuyển mạch và bộ điều khiển thực hiện các quyết định hành động định tuyến cấp cao trong OpenFlow. Bộ điều khiển và công tắc liên hệ với nhau thông qua Giao thức chuyển mạch OpenFlow.



Hình 2. 8 Sơ đồ hoạt động của OpenFlow

Bộ điều khiển có thể hướng dẫn các thiết bị chuyển mạch áp dụng các quy tắc để cho phép tài liệu lưu chuyển trên mạng. Tất cả các quy tắc này có thể là: chuyển tài liệu

theo cách nhanh nhất hoặc theo cách ít lần nhảy nhất ... OpenFlow cung cấp giao diện API duy nhất, nhờ đó quản trị viên có thể lập kế hoạch đầy đủ cho công việc của mạng đồng thời. Bạn hoàn toàn có thể đặt ra các quy tắc để định tuyến gói, cân bằng tải, kiểm soát truy cập và điều chỉnh... API này bao gồm hai thành phần chính: giao diện lập trình để chặn chuyển tiếp gói qua các bộ chuyển mạch mạng và một tập hợp các giao diện toàn cầu trên đó Advanced có thể xây dựng các công cụ quản lý.

2.2.4 Ưu điểm của OpenFlow

Tính linh hoạt và mở rộng: OpenFlow cho phép quản trị viên mạng tùy chỉnh luồng dữ liệu trên mạng của mình. Nó cung cấp một cách thức linh hoạt để xác định và kiểm soát các luồng dữ liệu trên mạng, đồng thời cũng cho phép mở rộng mạng một cách dễ dàng.

Quản lý tập trung: OpenFlow cho phép quản trị viên mạng quản lý toàn bộ mạng của mình từ một điểm duy nhất, giúp cho việc quản lý và cấu hình mạng trở nên đơn giản hơn.

Tính độc lập với thiết bị: OpenFlow không phụ thuộc vào các thiết bị cụ thể trong mạng. Điều này cho phép quản trị viên mạng thay đổi hoặc nâng cấp các thiết bị trong mạng mà không ảnh hưởng đến cấu hình hoặc hiệu suất của mạng.

Khả năng kiểm soát băng thông: OpenFlow cho phép quản trị viên mạng kiểm soát lưu lượng trên mạng của mình, giúp tối ưu hóa hiệu suất mạng và đảm bảo rằng các ứng dụng và dịch vụ trong mạng được hoạt động một cách hiệu quả.

Hỗ trợ cho các ứng dụng mạng mới: OpenFlow cung cấp một cách thức tiếp cận cho các nhà phát triển ứng dụng mạng, giúp họ xây dựng các ứng dụng mới trên nền tảng của OpenFlow. Điều này giúp tăng tính linh hoạt và đa dạng hóa cho các ứng dụng mạng.

2.2.5 Kết luận

Trong chương này, chúng ta đã tìm hiểu về giao thức OpenFlow, một giao thức quan trọng trong mô hình SDN. Chúng ta đã đi qua lịch sử phát triển của OpenFlow và những thay đổi trong các phiên bản khác nhau của giao thức. Chúng ta cũng đã xem xét chi tiết nguyên lý hoạt động của OpenFlow và những ưu điểm của giao thức này.

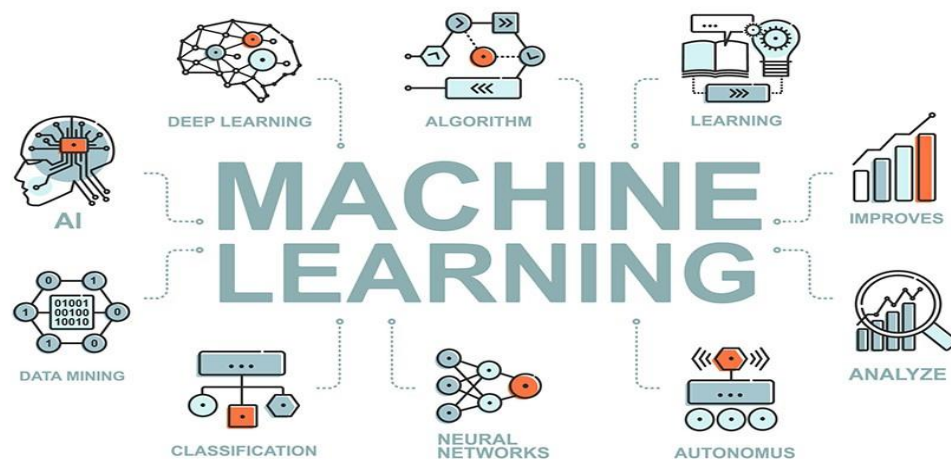
Một trong những ưu điểm chính của OpenFlow là khả năng tách biệt phần cứng và phần mềm trong mạng, giúp tối ưu hóa hiệu quả sử dụng tài nguyên và quản lý mạng một

cách linh hoạt. OpenFlow cũng cho phép kiểm soát mạng một cách trực tiếp và dễ dàng hơn, giúp giảm thiểu thời gian và chi phí cấu hình mạng.

Mặc dù OpenFlow mang lại nhiều lợi ích cho mạng, nhưng nó cũng tồn tại một số khuyết điểm như khả năng chịu tải thấp, khó khăn trong việc triển khai và hỗ trợ, và việc hoạt động trên môi trường mạng phức tạp.

Tuy nhiên, với sự phát triển của công nghệ và sự ủng hộ của các nhà cung cấp thiết bị mạng, OpenFlow đang trở thành một trong những giao thức được sử dụng phổ biến trong mô hình SDN, và nó sẽ tiếp tục phát triển và được áp dụng rộng rãi trong tương lai.

2.3. Giới thiệu về máy học (machine learning):



Hình 2. 9 Sơ đồ máy học (MACHINE LEARNING)

Máy học (Machine Learning) là một lĩnh vực của khoa học máy tính và trí tuệ nhân tạo, nghiên cứu về các thuật toán và phương pháp để giúp máy tính học và tự động cải thiện hiệu suất trong việc giải quyết các nhiệm vụ. Dưới đây là một số phương pháp máy học phổ biến:

- Học có giám sát (Supervised learning): Là phương pháp máy học sử dụng các bộ dữ liệu được gán nhãn để huấn luyện các thuật toán để dự đoán kết quả cho các dữ liệu mới.
 - Điểm mạnh: Phù hợp với các vấn đề dữ liệu được gán nhãn. Cho phép dự đoán kết quả chính xác với độ chính xác cao.

- Điểm yếu: Không thể áp dụng vào dữ liệu không có nhãn và yêu cầu dữ liệu huấn luyện có tính đại diện cao.
- Học không giám sát (Unsupervised learning): Là phương pháp máy học sử dụng các bộ dữ liệu không gán nhãn để tìm kiếm các mẫu, quy luật và cấu trúc trong dữ liệu.
 - Điểm mạnh: Cho phép tìm kiếm những kiến thức tiềm ẩn trong dữ liệu mà không cần nhãn. Phù hợp với các vấn đề không có dữ liệu huấn luyện được gán nhãn.
 - Điểm yếu: Khó khăn trong việc đánh giá độ chính xác của mô hình. Có thể phải sử dụng nhiều phương pháp và thuật toán khác nhau để có thể tìm ra kết quả tốt nhất.
- Học bán giám sát (Semi-supervised learning): Là phương pháp máy học kết hợp giữa học có giám sát và học không giám sát để tận dụng cả dữ liệu gán nhãn và không gán nhãn trong quá trình huấn luyện.
 - Điểm mạnh: Phù hợp với các vấn đề dữ liệu huấn luyện bị hạn chế. Sử dụng các phương pháp học không có giám sát kết hợp với dữ liệu huấn luyện có nhãn để cải thiện kết quả.
 - Điểm yếu: Kết quả không chính xác nếu dữ liệu huấn luyện bị sai lệch hoặc không đại diện cho dữ liệu thực tế.
- Học tăng cường (Reinforcement learning): Là phương pháp máy học sử dụng các thuật toán để học từ kết quả của các hành động tương tác với một môi trường và tối ưu hóa phương pháp hành động.
 - Điểm mạnh: Phù hợp với các vấn đề đòi hỏi phải học từ kinh nghiệm và có thể áp dụng vào các lĩnh vực như trò chơi, robot và các hệ thống tự động.
 - Điểm yếu: Cần nhiều lượt thực hiện để tìm ra kết quả tốt nhất. Có thể gặp khó khăn trong việc thiết lập các phần thưởng và xử lý các trường hợp đối nghịch.
- Máy học đại số tuyến tính (Linear algebra): Là phương pháp máy học sử dụng các phép tính đại số tuyến tính như ma trận để giải quyết các vấn đề trong máy học.
- Máy học sâu (Deep learning): Là phương pháp máy học sử dụng các mô hình mạng nơ-ron sâu để học các mẫu và cấu trúc phức tạp của dữ liệu.

– Máy học tối ưu (Optimization in Machine Learning): Là phương pháp máy học tập trung vào việc tối ưu hóa các hàm mất mát và các tham số để tăng hiệu suất của các mô hình máy học.

Ngoài ra còn có một số phương pháp máy học khác như:

– Máy học tăng tốc (Accelerated learning): Là phương pháp máy học sử dụng các thuật toán tối ưu hóa nhanh hơn để giảm thời gian huấn luyện mô hình.

– Học truyền cảm hứng (Transfer learning): Là phương pháp máy học sử dụng một mô hình đã được huấn luyện trước để giải quyết một vấn đề tương tự hoặc liên quan đến vấn đề mới.

– Học đa nhiệm (Multi-task learning): Là phương pháp máy học sử dụng một mô hình để giải quyết nhiều vấn đề khác nhau cùng một lúc.

– Máy học sử dụng các thuật toán phân loại (Classification): Là phương pháp máy học để phân loại dữ liệu vào các nhóm đã biết trước.

– Máy học sử dụng các thuật toán hồi quy (Regression): Là phương pháp máy học để dự đoán giá trị của một biến liên tục dựa trên các biến đầu vào.

– Máy học sử dụng các thuật toán trích xuất đặc trưng (Feature extraction): Là phương pháp máy học để trích xuất các đặc trưng quan trọng từ dữ liệu và sử dụng chúng để xây dựng mô hình.

– Máy học sử dụng các thuật toán tự học (Self-supervised learning): Là phương pháp máy học sử dụng các thuật toán để tự động tạo ra dữ liệu được gán nhãn và sử dụng chúng để huấn luyện mô hình.

2.4 Tìm hiểu về thuật toán Native Bayer và các ứng dụng của nó trong mô hình mạng SDN

2.4.1 Khái niệm về thuật toán Native Bayer

Trong mô hình mạng SDN (Software-Defined Networking), thuật toán Native Bayer được sử dụng để xử lý dữ liệu từ các cảm biến trên các thiết bị mạng, đặc biệt là các thiết bị đầu cuối (end devices) như camera IP và các thiết bị IoT (Internet of Things).

Các cảm biến trên các thiết bị này thường sử dụng các cảm biến Bayer để thu thập dữ liệu hình ảnh. Dữ liệu này sau đó được truyền về cho các điểm đầu cuối để xử lý và phân tích. Tuy nhiên, do các cảm biến Bayer chỉ ghi lại một phần thông tin màu sắc, dữ liệu từ các cảm biến này cần được giải mã và xử lý để tạo ra hình ảnh đầy đủ.

Trong mô hình mạng SDN, thuật toán Native Bayer được sử dụng để giải mã dữ liệu từ các cảm biến Bayer và tạo ra hình ảnh đầy đủ. Sau đó, các hình ảnh này có thể được sử dụng để phân tích và giám sát mạng, như trong các ứng dụng an ninh, kiểm soát chất lượng sản phẩm, hay giám sát môi trường.

Để triển khai thuật toán Native Bayer trong mô hình mạng SDN, các thiết bị đầu cuối cần được trang bị phần mềm và phần cứng hỗ trợ để xử lý dữ liệu từ các cảm biến Bayer. Ngoài ra, các thuật toán xử lý hình ảnh khác cũng được sử dụng để tăng cường chất lượng hình ảnh và giảm nhiễu cho dữ liệu từ các cảm biến Bayer.

2.4.2 Công thức và kỹ thuật của thuật toán Native Bayer

Thuật toán Native Bayer sử dụng một kỹ thuật được gọi là mẫu quét tuyến tính để giải mã dữ liệu từ các cảm biến Bayer. Các điểm ảnh trên cảm biến Bayer được sắp xếp theo một mẫu nhất định, với các điểm ảnh màu xanh lá cây và màu đỏ xen kẽ với nhau, và các điểm ảnh màu xanh dương và màu xám cũng xen kẽ với nhau nhưng lệch một nửa pixel so với các điểm ảnh màu xanh lá cây và màu đỏ.

Để giải mã dữ liệu từ các cảm biến Bayer, thuật toán Native Bayer sử dụng một bộ lọc để xác định giá trị màu của mỗi điểm ảnh dựa trên giá trị của các điểm ảnh xung quanh. Bộ lọc này thường được thiết kế dưới dạng một ma trận 3x3 hoặc 5x5.

Công thức tính toán giá trị màu của một điểm ảnh trong thuật toán Native Bayer như sau:

Đối với các điểm ảnh màu xanh lá cây (G):

$$G(x,y) = (B(x-1,y) + B(x+1,y) + B(x,y-1) + B(x,y+1)) / 4$$

Đối với các điểm ảnh màu đỏ (R):

$$R(x,y) = (B(x,y-1) + B(x,y+1)) / 2$$

Đối với các điểm ảnh màu xanh dương (B) và màu xám (G):

$$B(x,y) = (R(x-1,y-1) + R(x+1,y+1) + R(x+1,y-1) + R(x-1,y+1)) / 4$$

$$G(x,y) = (B(x-1,y) + B(x+1,y) + B(x,y-1) + B(x,y+1) + R(x,y)) / 5$$

Trong đó, (x,y) là tọa độ của điểm ảnh cần tính toán giá trị màu, và $B(x,y)$, $R(x,y)$ và $G(x,y)$ là giá trị màu tương ứng của điểm ảnh đó trên các kênh màu Blue, Red và Green.

Sau khi tính toán giá trị màu cho tất cả các điểm ảnh trên cảm biến Bayer, thuật toán Native Bayer có thể tạo ra hình ảnh đầy đủ với đầy đủ thông tin màu sắc. Tuy nhiên, để đạt được kết quả tốt nhất, cần sử dụng các kỹ thuật xử lý hình ảnh khác để tăng cường chất lượng hình ảnh.

2.4.3 Ưu điểm của thuật toán Native Bayer

Thuật toán Native Bayer có nhiều ưu điểm khi được sử dụng trong mô hình mạng SDN, bao gồm:

- Hiệu suất xử lý nhanh: Thuật toán Native Bayer được thiết kế để xử lý các ảnh đơn sắc từ cảm biến Bayer một cách nhanh chóng và hiệu quả. Với các thuật toán xử lý hình ảnh truyền thống, việc chuyển đổi ảnh đơn sắc sang ảnh màu đầy đủ yêu cầu nhiều thời gian tính toán, tuy nhiên thuật toán Native Bayer giúp giảm thiểu thời gian xử lý và tăng hiệu suất.
- Giảm chi phí và tiết kiệm băng thông: Thuật toán Native Bayer giúp giảm chi phí phần cứng của cảm biến hình ảnh bằng cách sử dụng các cảm biến màu đơn giản hơn. Nó cũng giúp giảm chi phí lưu trữ và truyền tải dữ liệu hình ảnh bằng cách giảm kích thước tệp hình ảnh. Điều này là rất quan trọng trong môi trường mạng SDN, nơi việc giảm chi phí và tiết kiệm băng thông là rất cần thiết.
- Cải thiện chất lượng hình ảnh: Thuật toán Native Bayer cũng giúp cải thiện chất lượng hình ảnh bằng cách sử dụng thông tin từ các điểm ảnh xung quanh để tính toán giá trị màu cho mỗi điểm ảnh. Điều này giúp tăng độ phân giải và độ tương phản của hình ảnh.
- Tính linh hoạt: Thuật toán Native Bayer có thể được áp dụng trên nhiều nền tảng và thiết bị khác nhau, từ các cảm biến hình ảnh đơn giản đến các thiết bị điện thoại di động và máy tính cá nhân.

Tóm lại, thuật toán Native Bayer là một công nghệ quan trọng trong mô hình mạng SDN, giúp cải thiện hiệu suất, giảm chi phí và cải thiện chất lượng hình ảnh.

2.4.4 Nhược điểm của thuật toán Native Bayer

Mặc dù thuật toán Native Bayer có nhiều ưu điểm, tuy nhiên nó cũng có một số nhược điểm sau:

- Giới hạn về độ phân giải: Thuật toán Native Bayer chỉ hoạt động tốt trên các cảm biến hình ảnh có độ phân giải thấp hơn. Khi độ phân giải của hình ảnh tăng lên, việc tính toán giá trị màu của từng điểm ảnh sẽ trở nên phức tạp hơn và yêu cầu nhiều thời gian hơn.
- Thiếu chi tiết trong các bức ảnh: Khi sử dụng thuật toán Native Bayer, các bức ảnh có thể bị mất một số chi tiết nhỏ hoặc các chi tiết tinh vi nhất do việc tính toán giá trị màu cho mỗi điểm ảnh chỉ dựa trên thông tin của các điểm ảnh xung quanh.
- Không thể áp dụng trực tiếp cho hình ảnh RGB: Thuật toán Native Bayer chỉ hoạt động trên các hình ảnh đơn sắc từ cảm biến Bayer và không thể áp dụng trực tiếp cho các hình ảnh RGB. Để sử dụng thuật toán Native Bayer với các hình ảnh RGB, cần chuyển đổi chúng sang hình ảnh đơn sắc trước.
- Phụ thuộc vào môi trường ánh sáng: Thuật toán Native Bayer có thể cho kết quả không chính xác nếu ánh sáng trong môi trường không đồng đều hoặc nhiễu. Việc sử dụng các bộ lọc hoặc các giải pháp khác để giảm thiểu ảnh hưởng của nhiễu và ánh sáng là rất quan trọng.

Tóm lại, những nhược điểm của thuật toán Native Bayer cần được lưu ý khi sử dụng nó trong mô hình mạng SDN và cần được cân nhắc trong việc lựa chọn giải pháp xử lý hình ảnh.

2.4.5 Ứng dụng của thuật toán Native Bayer trong mô hình mạng SDN

Thuật toán Native Bayer được sử dụng trong mô hình mạng SDN (Software-Defined Networking) để xử lý hình ảnh. Cụ thể, thuật toán này được áp dụng để giải quyết vấn đề chuyển đổi hình ảnh từ định dạng Bayer sang định dạng RGB để hiển thị trên các thiết bị màn hình.

Các ứng dụng của thuật toán Native Bayer trong mô hình mạng SDN bao gồm:

- Xử lý hình ảnh trong các ứng dụng IoT (Internet of Things): Các cảm biến hình ảnh được sử dụng trong các ứng dụng IoT thường có độ phân giải thấp hơn so với các cảm biến hình ảnh được sử dụng trong các ứng dụng khác. Do đó, thuật toán Native Bayer là một giải pháp hiệu quả để xử lý hình ảnh trong các ứng dụng IoT.
- Xử lý hình ảnh trong các ứng dụng an ninh: Thuật toán Native Bayer có thể được sử dụng để xử lý hình ảnh từ các camera an ninh để phát hiện và nhận dạng các đối tượng hoặc người di chuyển.
- Xử lý hình ảnh trong các ứng dụng y tế: Các cảm biến hình ảnh được sử dụng trong các ứng dụng y tế cũng thường có độ phân giải thấp hơn so với các cảm biến hình ảnh được sử dụng trong các ứng dụng khác. Thuật toán Native Bayer là một giải pháp hiệu quả để xử lý hình ảnh trong các ứng dụng y tế.
- Xử lý hình ảnh trong các ứng dụng công nghiệp: Các cảm biến hình ảnh được sử dụng trong các ứng dụng công nghiệp cũng thường có độ phân giải thấp hơn so với các cảm biến hình ảnh được sử dụng trong các ứng dụng khác. Thuật toán Native Bayer là một giải pháp hiệu quả để xử lý hình ảnh trong các ứng dụng công nghiệp.

Tóm lại, các ứng dụng của thuật toán Native Bayer trong mô hình mạng SDN là rất đa dạng và bao gồm các lĩnh vực từ IoT, an ninh, y tế đến công nghiệp.

CHƯƠNG 3: CÀI ĐẶT THỬ NGHIỆM

3.1 Phương pháp nghiên cứu

Phương pháp nghiên cứu cho báo cáo về đề tài "Ứng dụng SDN và thuật toán Naive Bayes vào mô hình mạng doanh nghiệp" để làm được nó em đã sử dụng một số phương pháp sau:

Thu thập tài liệu: Thu thập các tài liệu liên quan đến SDN, Naive Bayes và mô hình mạng doanh nghiệp. Đây có thể là các bài báo khoa học, sách, tài liệu trực tuyến,...

Nghiên cứu lý thuyết: Đọc và phân tích các tài liệu thu thập được để có được hiểu biết về SDN, Naive Bayes và mô hình mạng doanh nghiệp. Phân tích cụ thể về ứng dụng của SDN và Naive Bayes trong mô hình mạng doanh nghiệp.

Thiết kế và triển khai mô hình: Thiết kế một mô hình mạng doanh nghiệp và triển khai mô hình với sự ứng dụng của SDN và thuật toán Naive Bayes. Sử dụng phần mềm mininet để triển khai mô hình mạng và thực hiện thử nghiệm.

So sánh kết quả và đưa ra kết luận: So sánh kết quả thu được với các nghiên cứu trước đây và đưa ra kết luận về hiệu quả của việc ứng dụng SDN và Naive Bayes vào mô hình mạng doanh nghiệp. Nêu ra những ưu điểm và hạn chế của phương pháp nghiên cứu này cũng như đề xuất hướng phát triển cho các nghiên cứu tương lai.

3.2 Kết quả đạt được

Cải thiện hiệu quả mạng: Sử dụng mô hình mạng SDN kết hợp với thuật toán Naive Bayes đã giúp cải thiện hiệu quả mạng trong doanh nghiệp. Mô hình mạng SDN giúp tăng tính linh hoạt và quản lý dễ dàng hơn các thiết bị mạng, trong khi thuật toán Naive Bayes giúp tối ưu hóa các quyết định và dự đoán trong hệ thống.

Tăng tính bảo mật: Ứng dụng SDN và thuật toán Naive Bayes vào mô hình mạng doanh nghiệp cũng giúp tăng tính bảo mật của hệ thống. Bằng cách tập trung quản lý và kiểm soát các luồng dữ liệu trong mạng, mô hình mạng SDN giúp giảm thiểu rủi ro bảo mật, trong khi thuật toán Naive Bayes giúp phát hiện và xử lý các mối đe dọa an ninh nhanh chóng hơn.

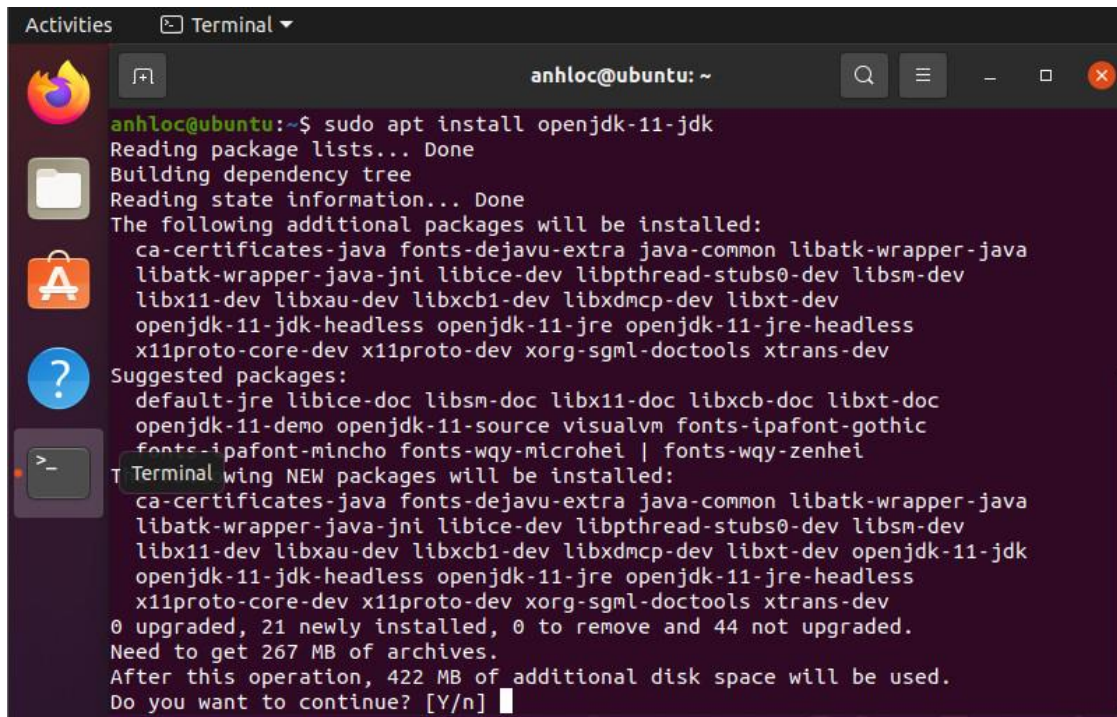
Giảm chi phí quản lý và vận hành: Mô hình mạng SDN kết hợp với thuật toán Naive Bayes còn giúp giảm chi phí quản lý và vận hành hệ thống mạng. Việc tập trung quản lý mạng trên một nền tảng duy nhất giúp giảm tải cho nhân viên quản lý mạng và tối ưu hóa thời gian và công sức của họ.

Tăng khả năng mở rộng: Mô hình mạng SDN được sử dụng trong doanh nghiệp cũng giúp tăng khả năng mở rộng của hệ thống. Mô hình này cho phép thêm hoặc loại bỏ các thiết bị mạng một cách dễ dàng và linh hoạt, giúp doanh nghiệp có thể tăng hoặc giảm quy mô mạng một cách nhanh chóng và tiết kiệm chi phí.

3.2.1 Xây dựng mô hình mạng SDN

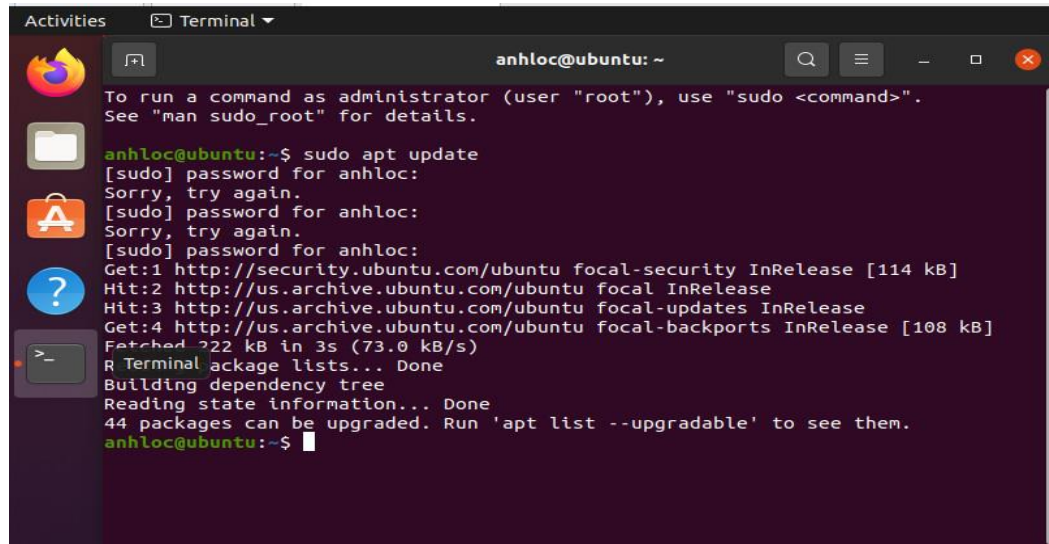
Bước đầu tiên là cài đặt ONOS trong ubuntu.

Điều kiện tiên quyết, trang Yêu cầu ONOS cho chúng ta biết rằng phiên bản ONOS của chúng ta yêu cầu Java 11.



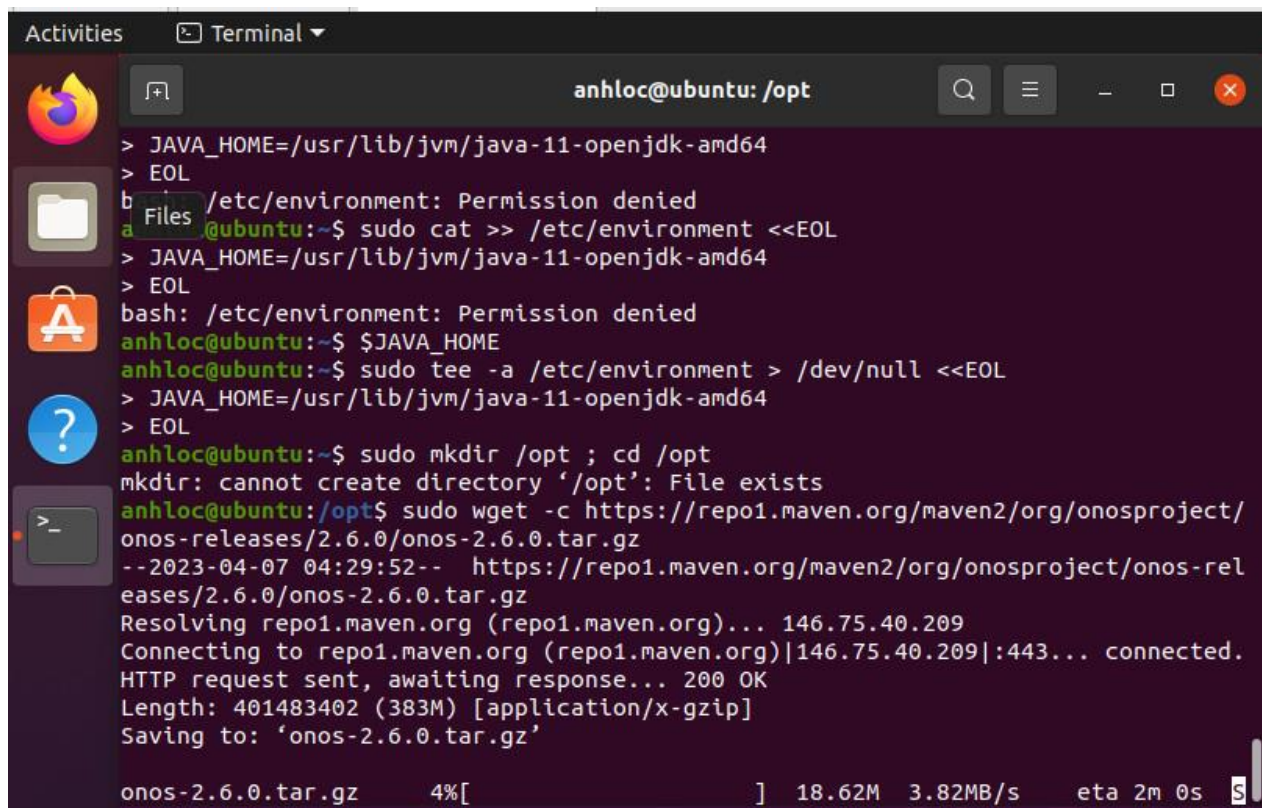
```
anhloc@ubuntu: ~  
anhloc@ubuntu:~$ sudo apt install openjdk-11-jdk  
Reading package lists... Done  
Building dependency tree  
Reading state information... Done  
The following additional packages will be installed:  
  ca-certificates-java fonts-dejavu-extra java-common libatk-wrapper-java  
  libatk-wrapper-java-jni libice-dev libpthread-stubs0-dev libsm-dev  
  libx11-dev libxau-dev libxcb1-dev libxdmcp-dev libxt-dev  
  openjdk-11-jdk-headless openjdk-11-jre openjdk-11-jre-headless  
  x11proto-core-dev x11proto-dev xorg-sgml-doctools xtrans-dev  
Suggested packages:  
  default-jre libice-doc libsm-doc libx11-doc libxcb-doc libxt-doc  
  openjdk-11-demo openjdk-11-source visualvm fonts-ipafont-gothic  
  fonts-ipafont-mincho fonts-wqy-microhei | fonts-wqy-zenhei  
Terminal will install NEW packages will be installed:  
  ca-certificates-java fonts-dejavu-extra java-common libatk-wrapper-java  
  libatk-wrapper-java-jni libice-dev libpthread-stubs0-dev libsm-dev  
  libx11-dev libxau-dev libxcb1-dev libxdmcp-dev libxt-dev openjdk-11-jdk  
  openjdk-11-jdk-headless openjdk-11-jre openjdk-11-jre-headless  
  x11proto-core-dev x11proto-dev xorg-sgml-doctools xtrans-dev  
0 upgraded, 21 newly installed, 0 to remove and 44 not upgraded.  
Need to get 267 MB of archives.  
After this operation, 422 MB of additional disk space will be used.  
Do you want to continue? [Y/n]
```

Hình 3. 1 Cài đặt java 11 trên ubuntu



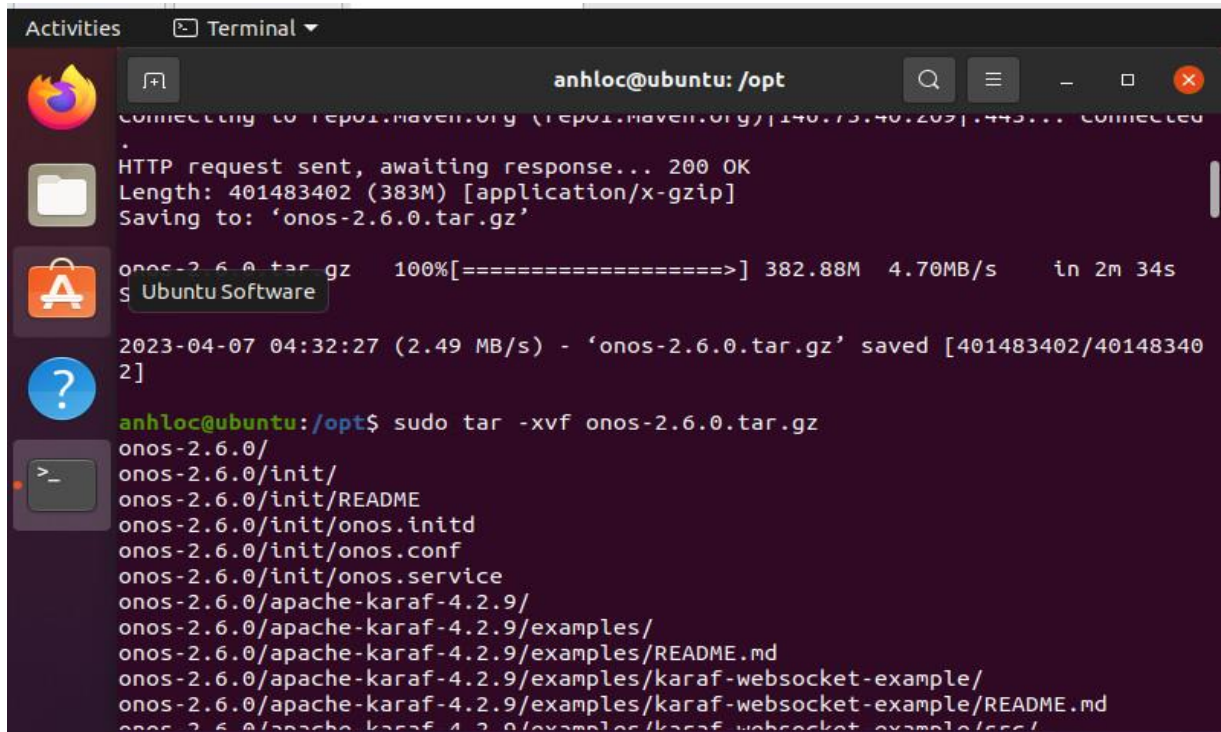
```
anhloc@ubuntu: ~  
To run a command as administrator (user "root"), use "sudo <command>".  
See "man sudo_root" for details.  
anhloc@ubuntu:~$ sudo apt update  
[sudo] password for anhloc:  
Sorry, try again.  
[sudo] password for anhloc:  
Sorry, try again.  
[sudo] password for anhloc:  
Get:1 http://security.ubuntu.com/ubuntu focal-security InRelease [114 kB]  
Hit:2 http://us.archive.ubuntu.com/ubuntu focal InRelease  
Hit:3 http://us.archive.ubuntu.com/ubuntu focal-updates InRelease  
Get:4 http://us.archive.ubuntu.com/ubuntu focal-backports InRelease [108 kB]  
Fetched 222 kB in 3s (73.0 kB/s)  
Reading package lists... Done  
Building dependency tree  
Reading state information... Done  
44 packages can be upgraded. Run 'apt list --upgradable' to see them.  
anhloc@ubuntu:~$
```

Hình 3. 2 Cập nhật danh sách các gói phần mềm có sẵn trong hệ thống



```
anhloc@ubuntu: /opt  
> JAVA_HOME=/usr/lib/jvm/java-11-openjdk-amd64  
> EOL  
b Files /etc/environment: Permission denied  
a @ubuntu:~$ sudo cat >> /etc/environment <<EOL  
> JAVA_HOME=/usr/lib/jvm/java-11-openjdk-amd64  
> EOL  
bash: /etc/environment: Permission denied  
anhloc@ubuntu:~$ $JAVA_HOME  
anhloc@ubuntu:~$ sudo tee -a /etc/environment > /dev/null <<EOL  
> JAVA_HOME=/usr/lib/jvm/java-11-openjdk-amd64  
> EOL  
anhloc@ubuntu:~$ sudo mkdir /opt ; cd /opt  
mkdir: cannot create directory '/opt': File exists  
anhloc@ubuntu:/opt$ sudo wget -c https://repo1.maven.org/maven2/org/onosproject/  
onos-releases/2.6.0/onos-2.6.0.tar.gz  
--2023-04-07 04:29:52-- https://repo1.maven.org/maven2/org/onosproject/onos-rel  
eases/2.6.0/onos-2.6.0.tar.gz  
Resolving repo1.maven.org (repo1.maven.org)... 146.75.40.209  
Connecting to repo1.maven.org (repo1.maven.org)|146.75.40.209|:443... connected.  
HTTP request sent, awaiting response... 200 OK  
Length: 401483402 (383M) [application/x-gzip]  
Saving to: 'onos-2.6.0.tar.gz'  
  
onos-2.6.0.tar.gz 4%[ ] 18.62M 3.82MB/s eta 2m 0s
```

Hình 3. 3 Màn hình đặt biến và tải ONOS



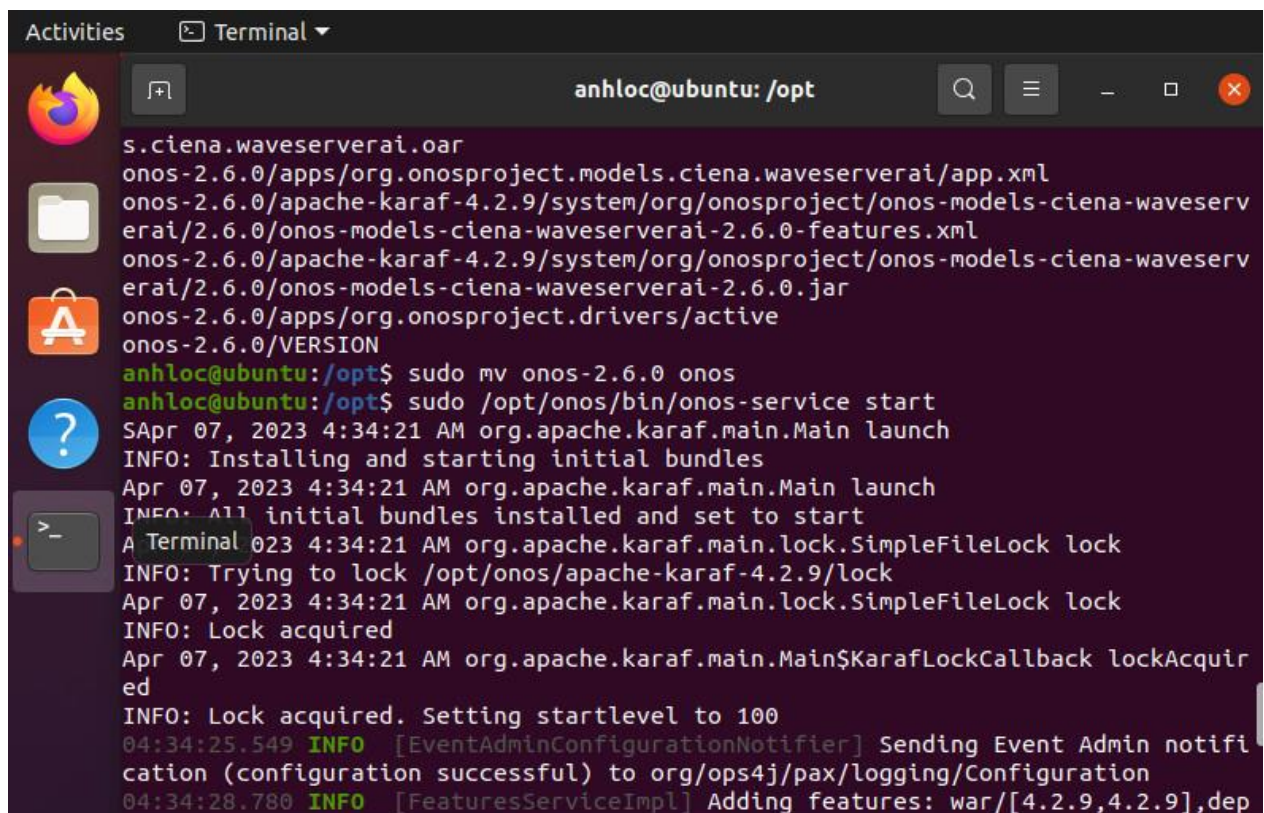
```
anhloc@ubuntu: /opt
Connecting to repo1.maven.org (repo1.maven.org) [140.73.40.209]:443... Connected
HTTP request sent, awaiting response... 200 OK
Length: 401483402 (383M) [application/x-gzip]
Saving to: 'onos-2.6.0.tar.gz'

onos-2.6.0.tar.gz  100%[=====] 382.88M  4.70MB/s   in 2m 34s
S Ubuntu Software

2023-04-07 04:32:27 (2.49 MB/s) - 'onos-2.6.0.tar.gz' saved [401483402/401483402]

anhloc@ubuntu:/opt$ sudo tar -xvf onos-2.6.0.tar.gz
onos-2.6.0/
onos-2.6.0/init/
onos-2.6.0/init/README
onos-2.6.0/init/onos.initd
onos-2.6.0/init/onos.conf
onos-2.6.0/init/onos.service
onos-2.6.0/apache-karaf-4.2.9/
onos-2.6.0/apache-karaf-4.2.9/examples/
onos-2.6.0/apache-karaf-4.2.9/examples/README.md
onos-2.6.0/apache-karaf-4.2.9/examples/karaf-websocket-example/
onos-2.6.0/apache-karaf-4.2.9/examples/karaf-websocket-example/README.md
onos-2.6.0/apache-karaf-4.2.9/examples/karaf-websocket-example/...
```

Hình 3. 4 Giải nén file onos



```
s.ciena.waveserverai.oar
onos-2.6.0/apps/org.onosproject.models.ciena.waveserverai/app.xml
onos-2.6.0/apache-karaf-4.2.9/system/org/onosproject/onos-models-ciena-waveserverai/2.6.0/onos-models-ciena-waveserverai-2.6.0-features.xml
onos-2.6.0/apache-karaf-4.2.9/system/org/onosproject/onos-models-ciena-waveserverai/2.6.0/onos-models-ciena-waveserverai-2.6.0.jar
onos-2.6.0/apps/org.onosproject.drivers/active
onos-2.6.0/VERSION

anhloc@ubuntu:/opt$ sudo mv onos-2.6.0 onos
anhloc@ubuntu:/opt$ sudo /opt/onos/bin/onos-service start
SApr 07, 2023 4:34:21 AM org.apache.karaf.main.Main launch
INFO: Installing and starting initial bundles
Apr 07, 2023 4:34:21 AM org.apache.karaf.main.Main launch
INFO: All initial bundles installed and set to start
A Terminal 023 4:34:21 AM org.apache.karaf.main.lock.SimpleFileLock lock
INFO: Trying to lock /opt/onos/apache-karaf-4.2.9/lock
Apr 07, 2023 4:34:21 AM org.apache.karaf.main.lock.SimpleFileLock lock
INFO: Lock acquired
Apr 07, 2023 4:34:21 AM org.apache.karaf.main.Main$KarafLockCallback lockAcquired
INFO: Lock acquired. Setting startlevel to 100
04:34:25.549 INFO [EventAdminConfigurationNotifier] Sending Event Admin notification (configuration successful) to org/ops4j/pax/logging/Configuration
04:34:28.780 INFO [FeaturesServiceImpl] Adding features: war/[4.2.9,4.2.9],dep
```

Hình 3. 5 Đổi tên file và khởi động dịch vụ ONOS

```
Apr 7 04:47
anhloc@ubuntu: /opt
anhloc@ubuntu:/opt$ sudo /opt/onos/bin/onos -l karaf
[sudo] password for anhloc:
Warning: Permanently added '[localhost]:8101' (RSA) to the list of known hosts.
Password authentication
Password:
Welcome to Open Network Operating System (ONOS)!

  ONOS

Documentation: wiki.onosproject.org
Tutorials:    tutorials.onosproject.org
Mailing lists: lists.onosproject.org

Come help out! Find out how at: contribute.onosproject.org

Hit '<tab>' for a list of available commands
and '[cmd] --help' for help on a specific command.
Hit '<ctrl-d>' or type 'logout' to exit ONOS session.

karaf@root > Connection to localhost closed by remote host.
Connection to localhost closed.
anhloc@ubuntu:/opt$
```

Hình 3. 6 Truy cập vào CLI của ONOS

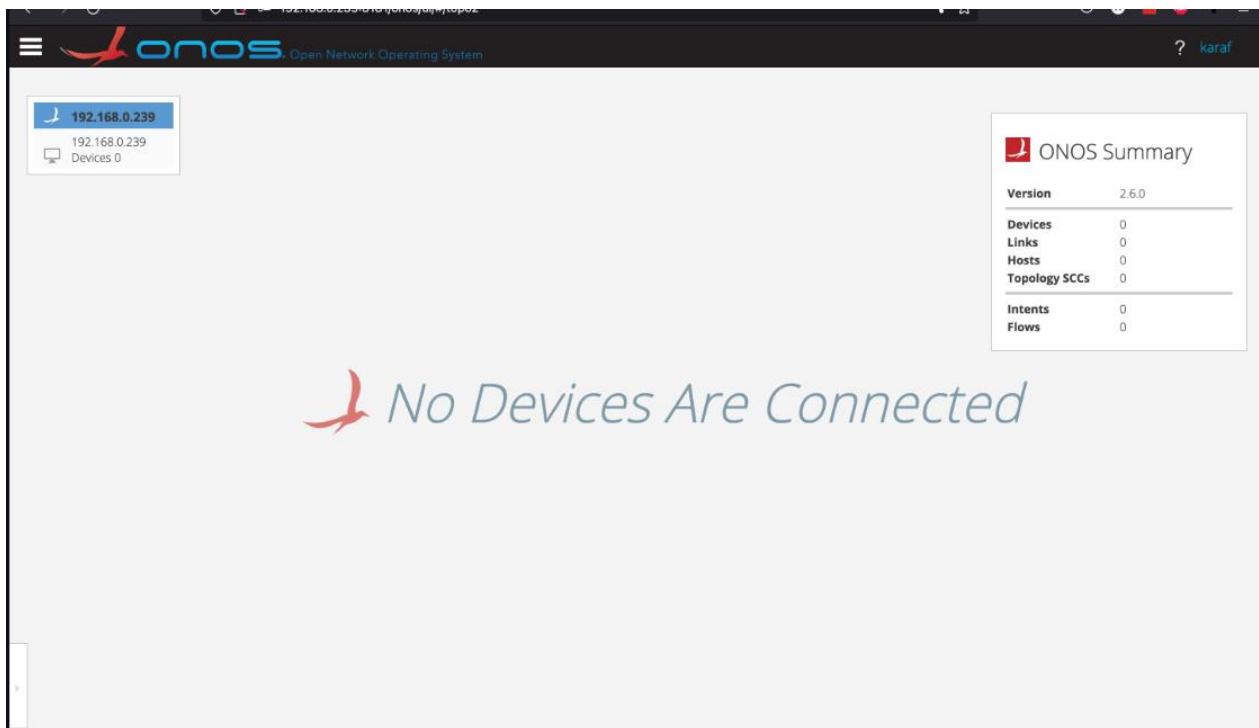


Hình 3. 7 Truy cập vào GUI

Từ đây, bạn có thể đăng nhập bằng một trong hai người dùng mặc định, sau đó bạn sẽ được chuyển hướng đến Chế độ xem cấu trúc liên kết.

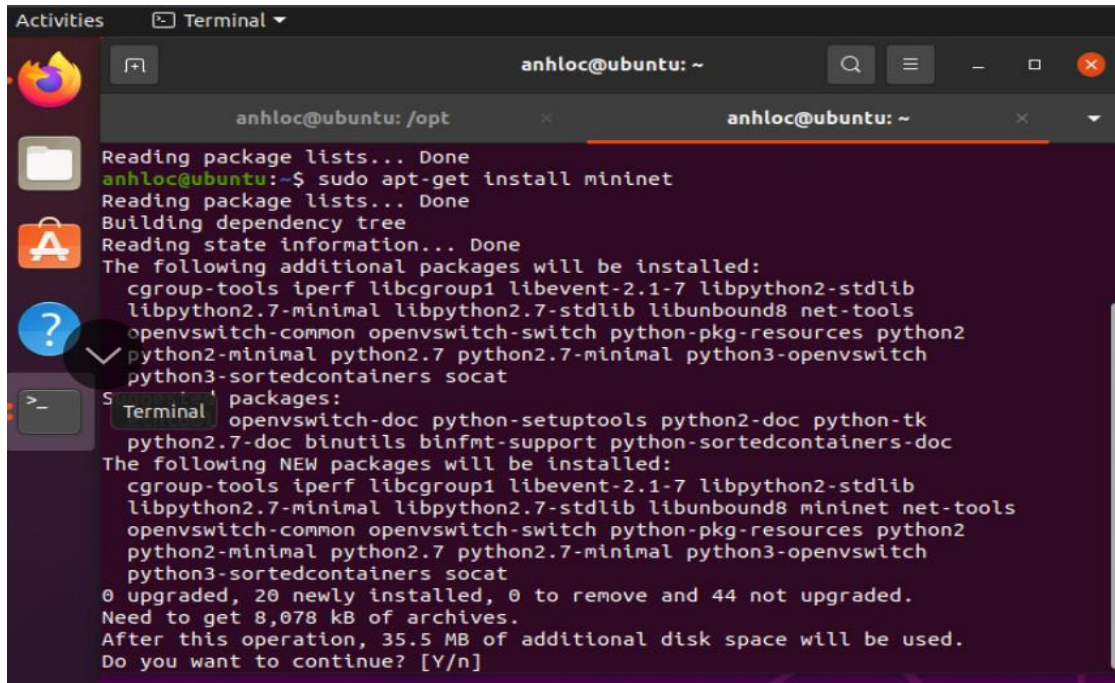
Username	Password
onos	rocks
karaf	karaf

Hình 3. 8 Tài khoản mật khẩu mặc định



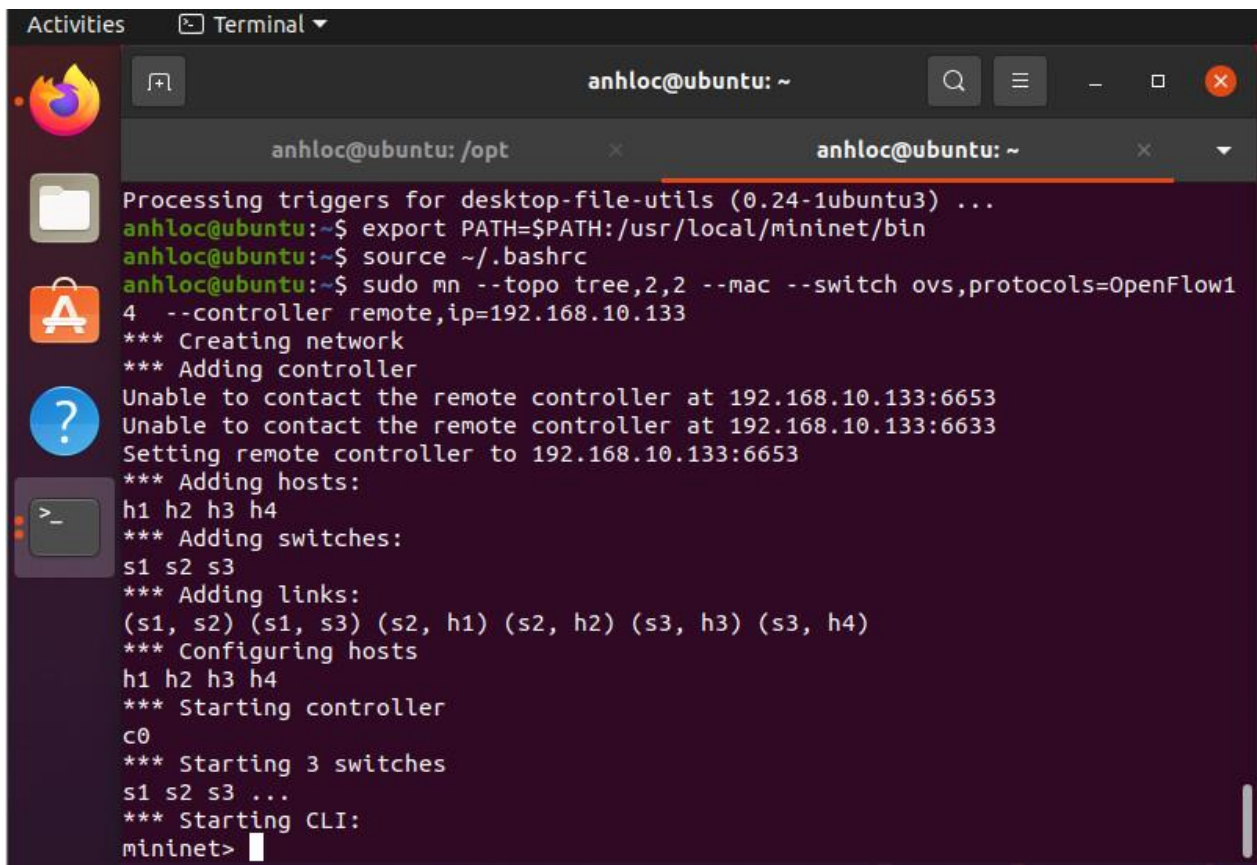
Hình 3. 9 Giao diện ONOS

Từ đây, thông thường bạn có thể xem tổng quan về cấu trúc liên kết mạng hiện được kết nối với (các) bộ điều khiển của bạn, nhưng vì chúng ta chưa thiết lập bất kỳ mạng nào nên chúng ta không thấy bất kỳ thiết bị nào.



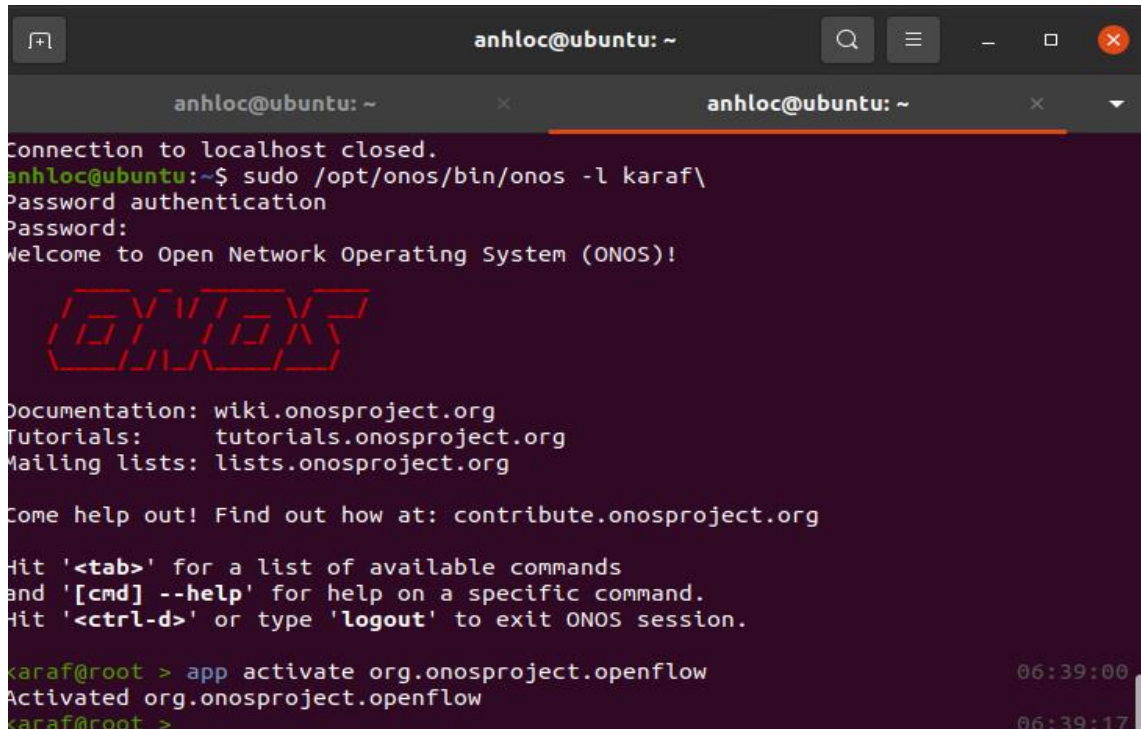
```
anhloc@ubuntu: /opt
anhloc@ubuntu: ~
Reading package lists... Done
anhloc@ubuntu:~$ sudo apt-get install mininet
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following additional packages will be installed:
  cgroup-tools iperf libcgroup1 libevent-2.1-7 libpython2-stdlib
  libpython2.7-minimal libpython2.7-stdlib libunbound8 net-tools
  openvswitch-common openvswitch-switch python-pkg-resources python2
  python2-minimal python2.7 python2.7-minimal python3-openvswitch
  python3-sortedcontainers socat
Suggested packages:
  python-doc python-setuptools python2-doc python-tk
  python2.7-doc binutils binfmt-support python-sortedcontainers-doc
The following NEW packages will be installed:
  cgroup-tools iperf libcgroup1 libevent-2.1-7 libpython2-stdlib
  libpython2.7-minimal libpython2.7-stdlib libunbound8 mininet net-tools
  openvswitch-common openvswitch-switch python-pkg-resources python2
  python2-minimal python2.7 python2.7-minimal python3-openvswitch
  python3-sortedcontainers socat
0 upgraded, 20 newly installed, 0 to remove and 44 not upgraded.
Need to get 8,078 kB of archives.
After this operation, 35.5 MB of additional disk space will be used.
Do you want to continue? [Y/n]
```

Hình 3. 10 Tải và cài đặt mininet



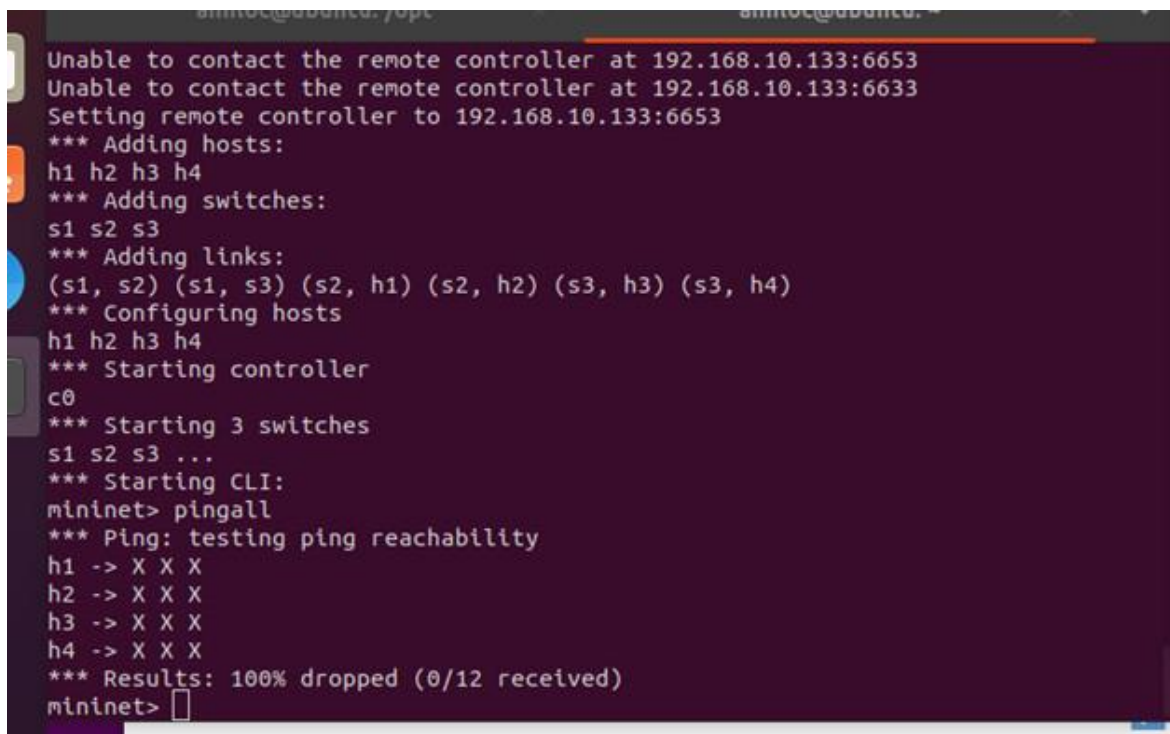
```
anhloc@ubuntu: ~
Processing triggers for desktop-file-utils (0.24-1ubuntu3) ...
anhloc@ubuntu:~$ export PATH=$PATH:/usr/local/mininet/bin
anhloc@ubuntu:~$ source ~/.bashrc
anhloc@ubuntu:~$ sudo mn --topo tree,2,2 --mac --switch ovs,protocols=OpenFlow1
4 --controller remote,ip=192.168.10.133
*** Creating network
*** Adding controller
Unable to contact the remote controller at 192.168.10.133:6653
Unable to contact the remote controller at 192.168.10.133:6653
Setting remote controller to 192.168.10.133:6653
*** Adding hosts:
h1 h2 h3 h4
*** Adding switches:
s1 s2 s3
*** Adding links:
(s1, s2) (s1, s3) (s2, h1) (s2, h2) (s3, h3) (s3, h4)
*** Configuring hosts
h1 h2 h3 h4
*** Starting controller
c0
*** Starting 3 switches
s1 s2 s3 ...
*** Starting CLI:
mininet>
```

Hình 3. 11 Tạo topo mạng 2x2 với các switch



```
anhloc@ubuntu: ~  
Connection to localhost closed.  
anhloc@ubuntu:~$ sudo /opt/onos/bin/onos -l karaf\  
Password authentication  
Password:  
Welcome to Open Network Operating System (ONOS)!  
  
ONOS  
  
Documentation: wiki.onosproject.org  
Tutorials:     tutorials.onosproject.org  
Mailing lists: lists.onosproject.org  
  
Come help out! Find out how at: contribute.onosproject.org  
  
Hit '<tab>' for a list of available commands  
and '[cmd] --help' for help on a specific command.  
Hit '<ctrl-d>' or type 'logout' to exit ONOS session.  
  
karaf@root > app activate org.onosproject.openflow  
Activated org.onosproject.openflow  
karaf@root >
```

Hình 3. 12 Cài đặt giao thức Openflow



```
Unable to contact the remote controller at 192.168.10.133:6653  
Unable to contact the remote controller at 192.168.10.133:6633  
Setting remote controller to 192.168.10.133:6653  
*** Adding hosts:  
h1 h2 h3 h4  
*** Adding switches:  
s1 s2 s3  
*** Adding links:  
(s1, s2) (s1, s3) (s2, h1) (s2, h2) (s3, h3) (s3, h4)  
*** Configuring hosts  
h1 h2 h3 h4  
*** Starting controller  
c0  
*** Starting 3 switches  
s1 s2 s3 ...  
*** Starting CLI:  
mininet> pingall  
*** Ping: testing ping reachability  
h1 -> X X X  
h2 -> X X X  
h3 -> X X X  
h4 -> X X X  
*** Results: 100% dropped (0/12 received)  
mininet>
```

Hình 3. 13 Ping để kiểm tra kết nối

```
anhloc@ubuntu:/opt$ sudo /opt/onos/bin/onos -l karaf
Password authentication
Password:
Welcome to Open Network Operating System (ONOS)!

  ONOS

Documentation: wiki.onosproject.org
Tutorials:    tutorials.onosproject.org
Mailing lists: lists.onosproject.org

Come help out! Find out how at: contribute.onosproject.org

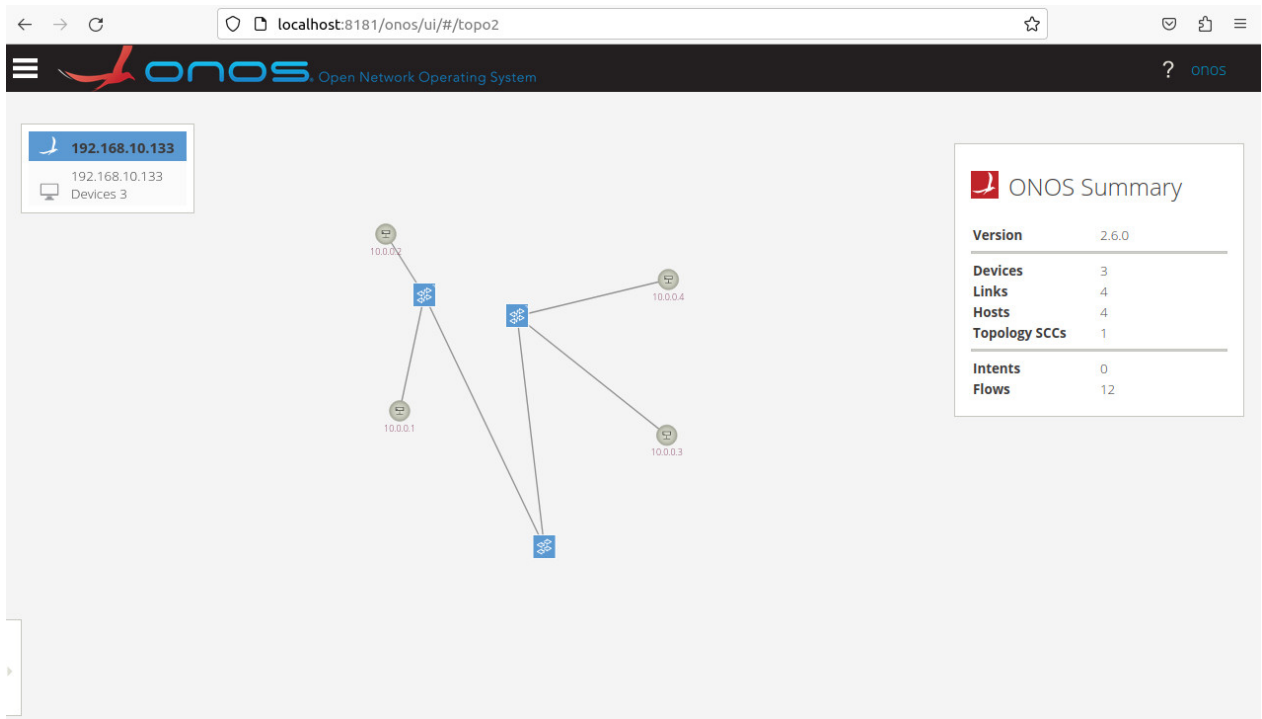
Hit '<tab>' for a list of available commands
and '[cmd] --help' for help on a specific command.
Hit '<ctrl-d>' or type 'logout' to exit ONOS session.

karaf@root > app activate org.onosproject.openflow                                04:51:12
Activated org.onosproject.openflow
karaf@root > app activate org.onosproject.fwd                                    04:51:51
Activated org.onosproject.fwd
karaf@root >                                                                    04:53:57
```

Hình 3. 14 Cài đặt ứng dụng Forwarding

```
anhloc@ubuntu: /opt      x      anhloc@ubuntu: ~      x      v
*** Adding links:
(s1, s2) (s1, s3) (s2, h1) (s2, h2) (s3, h3) (s3, h4)
*** Configuring hosts
h1 h2 h3 h4
* Ubuntu Software Controller
C0
*** Starting 3 switches
s1 s2 s3 ...
*** Starting CLI:
mininet> pingall
*** Ping: testing ping reachability
h1 -> X X X
h2 -> X X X
h3 -> X X X
h4 -> X X X
*** Results: 100% dropped (0/12 received)
mininet> pingall
*** Ping: testing ping reachability
h1 -> h2 h3 h4
h2 -> h1 h3 h4
h3 -> h1 h2 h4
h4 -> h1 h2 h3
*** Results: 0% dropped (12/12 received)
mininet>
```

Hình 3. 15 Sử dụng ping để kiểm tra kết nối

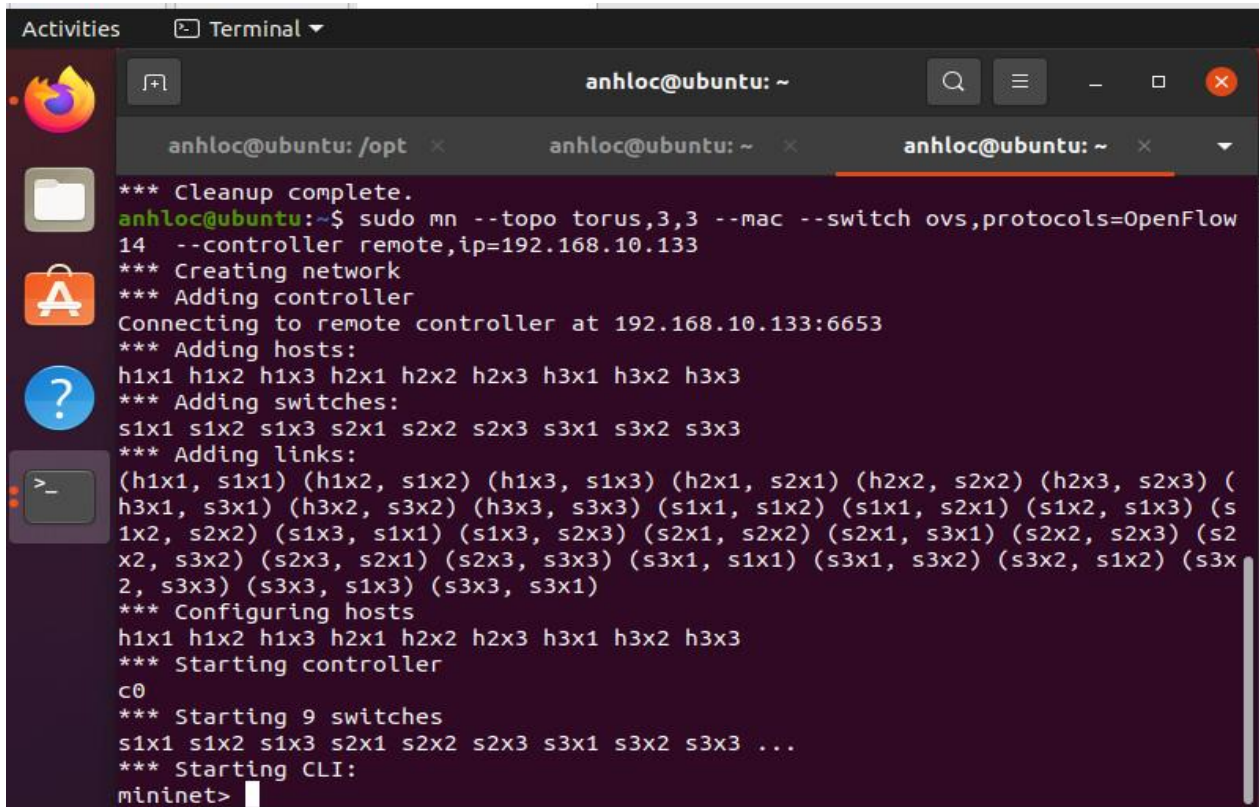


Hình 3. 16 Bốn máy chủ do mininet tạo ra trên GUI

The screenshot shows a terminal window with the following commands and output:

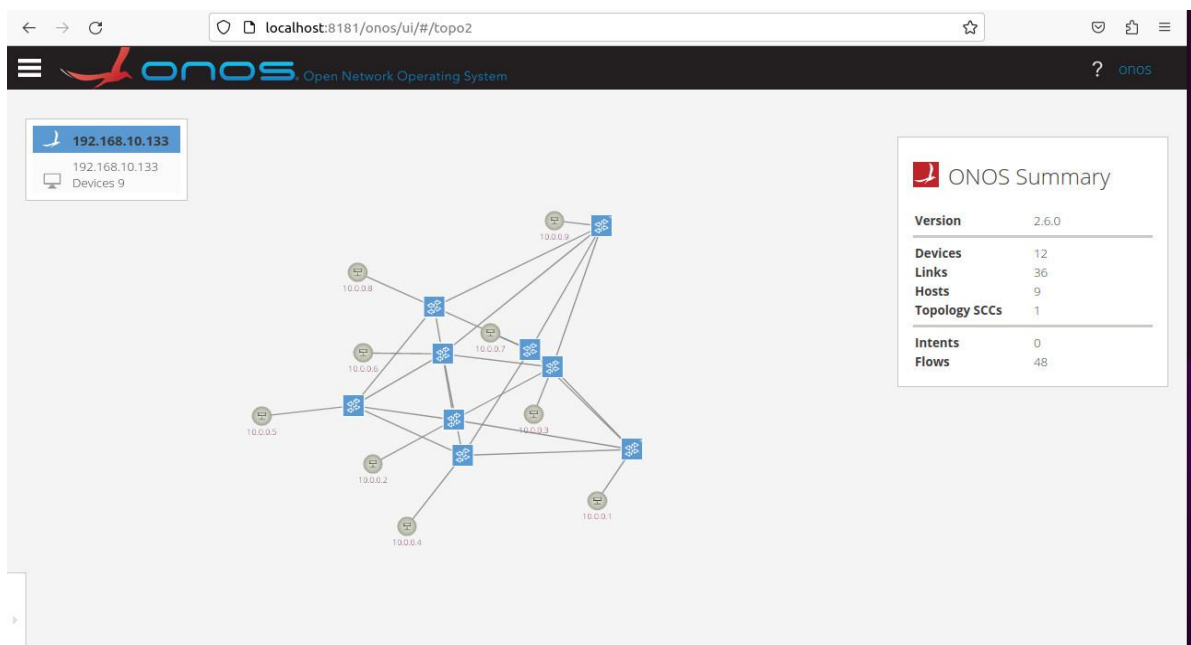
```
anhloc@ubuntu:~$ sudo mn -c
[sudo] password for anhloc:
1Sorry, try again.
[sudo] password for anhloc:
$ UbuntuSoftware h.
[sudo] password for anhloc:
*** Removing excess controllers/ofprotocols/ofdatapaths/pings/noxes
killall controller ofprotocol ofdatapath ping nox_core lt-nox_core ovs-openflow
d ovs-controller udpbwtest mnexec ivs 2> /dev/null
killall -9 controller ofprotocol ofdatapath ping nox_core lt-nox_core ovs-openf
lowd ovs-controller udpbwtest mnexec ivs 2> /dev/null
pkill -9 -f "sudo mnexec"
*** Removing junk from /tmp
rm -f /tmp/vconn* /tmp/vlogs* /tmp/*.out /tmp/*.log
*** Removing old X11 tunnels
*** Removing excess kernel datapaths
ps ax | egrep -o 'dp[0-9]+' | sed 's/dp/nl:/'
*** Removing OVS datapaths
ovs-vsctl --timeout=1 list-br
ovs-vsctl --if-exists del-br s1 -- --if-exists del-br s2 -- --if-exists del-br
s3
ovs-vsctl --timeout=1 list-br
*** Removing all links of the pattern foo-ethx
ip link show | egrep -o '([_-[:alnum:]]+-eth[[:digit:]]+)'
```

Hình 3. 17 Xóa các topo mạng đã tạo trước đó

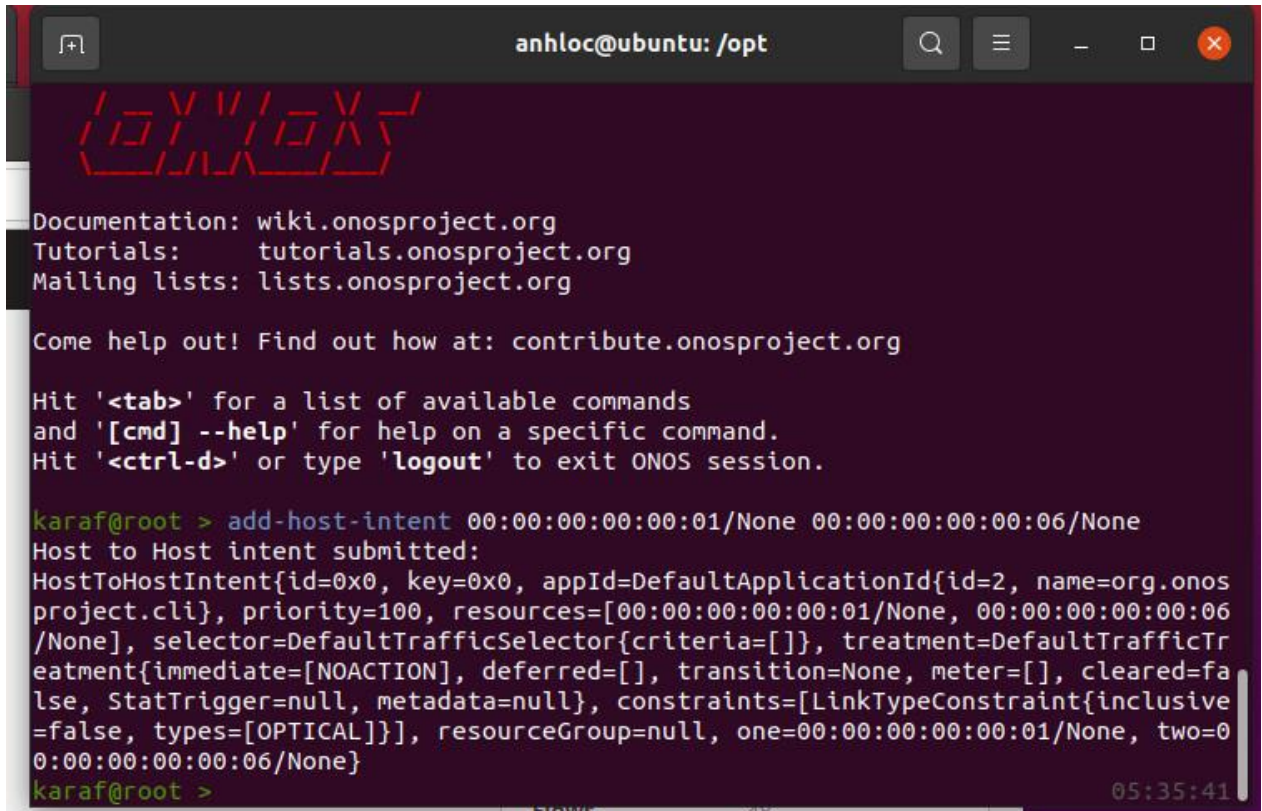


```
anhloc@ubuntu: ~  
anhloc@ubuntu: /opt x anhloc@ubuntu: ~ x anhloc@ubuntu: ~ x  
*** Cleanup complete.  
anhloc@ubuntu:~$ sudo mn --topo torus,3,3 --mac --switch ovs,protocols=OpenFlow  
14 --controller remote,ip=192.168.10.133  
*** Creating network  
*** Adding controller  
Connecting to remote controller at 192.168.10.133:6653  
*** Adding hosts:  
h1x1 h1x2 h1x3 h2x1 h2x2 h2x3 h3x1 h3x2 h3x3  
*** Adding switches:  
s1x1 s1x2 s1x3 s2x1 s2x2 s2x3 s3x1 s3x2 s3x3  
*** Adding links:  
(h1x1, s1x1) (h1x2, s1x2) (h1x3, s1x3) (h2x1, s2x1) (h2x2, s2x2) (h2x3, s2x3) (  
h3x1, s3x1) (h3x2, s3x2) (h3x3, s3x3) (s1x1, s1x2) (s1x1, s1x3) (s1x2, s1x3) (s  
1x2, s2x2) (s1x3, s1x1) (s1x3, s2x3) (s2x1, s2x2) (s2x1, s3x1) (s2x2, s2x3) (s2  
x2, s3x2) (s2x3, s2x1) (s2x3, s3x3) (s3x1, s1x1) (s3x1, s3x2) (s3x2, s1x2) (s3x  
2, s3x3) (s3x3, s1x3) (s3x3, s3x1)  
*** Configuring hosts  
h1x1 h1x2 h1x3 h2x1 h2x2 h2x3 h3x1 h3x2 h3x3  
*** Starting controller  
c0  
*** Starting 9 switches  
s1x1 s1x2 s1x3 s2x1 s2x2 s2x3 s3x1 s3x2 s3x3 ...  
*** Starting CLI:  
mininet>
```

Hình 3. 18 Tạo một topo mạng mới



Hình 3. 19 Kết quả cấu trúc mạng



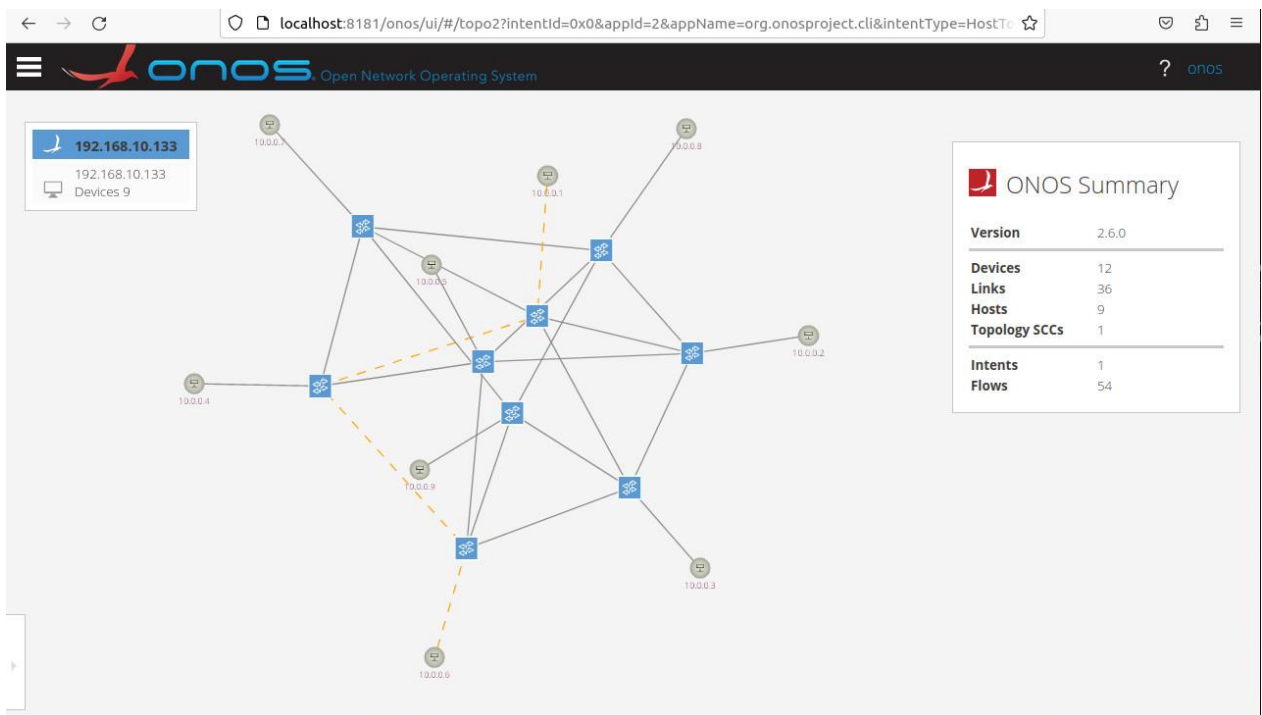
```
anhloc@ubuntu: /opt
Documentation: wiki.onosproject.org
Tutorials:    tutorials.onosproject.org
Mailing lists: lists.onosproject.org

Come help out! Find out how at: contribute.onosproject.org

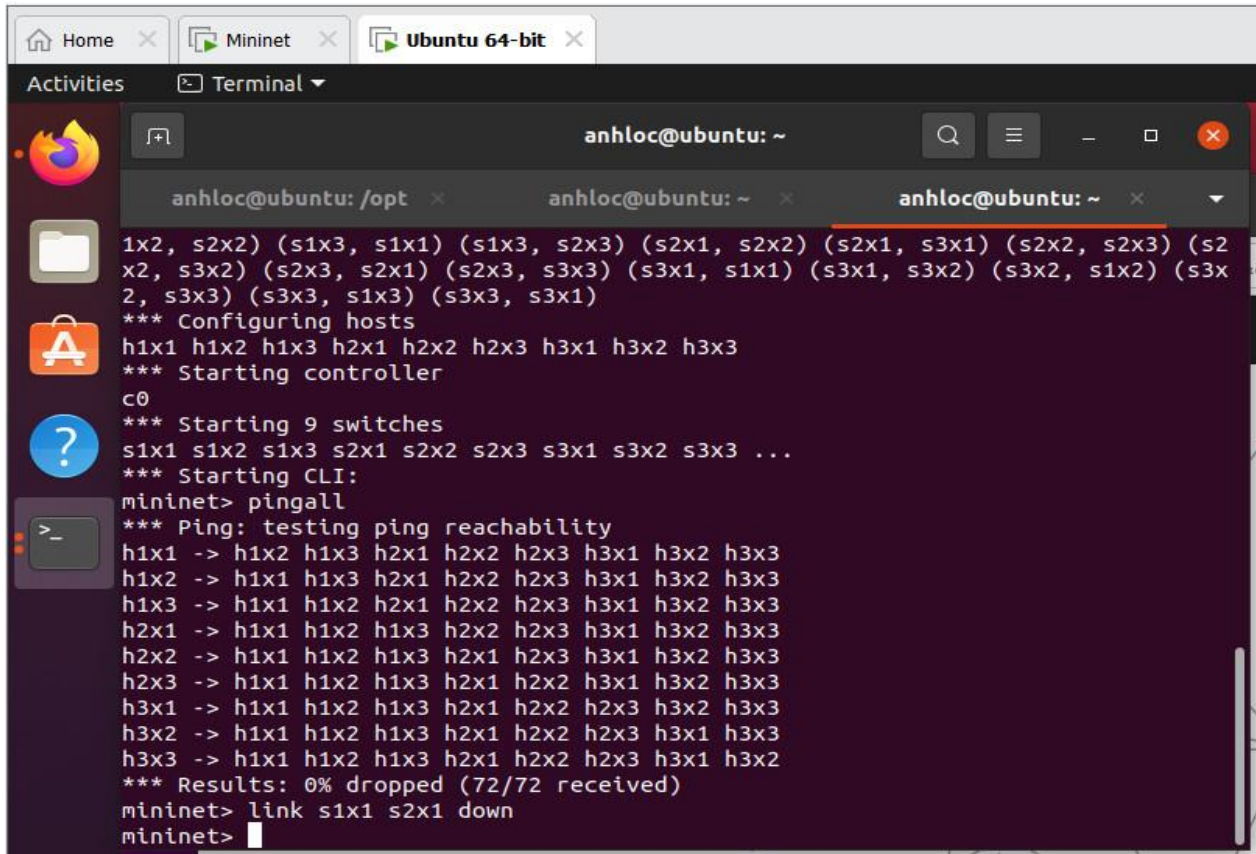
Hit '<tab>' for a list of available commands
and '[cmd] --help' for help on a specific command.
Hit '<ctrl-d>' or type 'logout' to exit ONOS session.

karaf@root> add-host-intent 00:00:00:00:00:01/None 00:00:00:00:00:06/None
Host to Host intent submitted:
HostToHostIntent{id=0x0, key=0x0, appId=DefaultApplicationId{id=2, name=org.onosproject.cli}, priority=100, resources=[00:00:00:00:00:01/None, 00:00:00:00:00:06/None], selector=DefaultTrafficSelector{criteria=[]}, treatment=DefaultTrafficTreatment{immediate=[NOACTION], deferred=[], transition=None, meter=[], cleared=false, StatTrigger=null, metadata=null}, constraints=[LinkTypeConstraint{inclusive=false, types=[OPTICAL]}], resourceGroup=null, one=00:00:00:00:00:01/None, two=00:00:00:00:00:06/None}
karaf@root>
```

Hình 3. 20 Tạo kết nối giữa host 1 và host 6

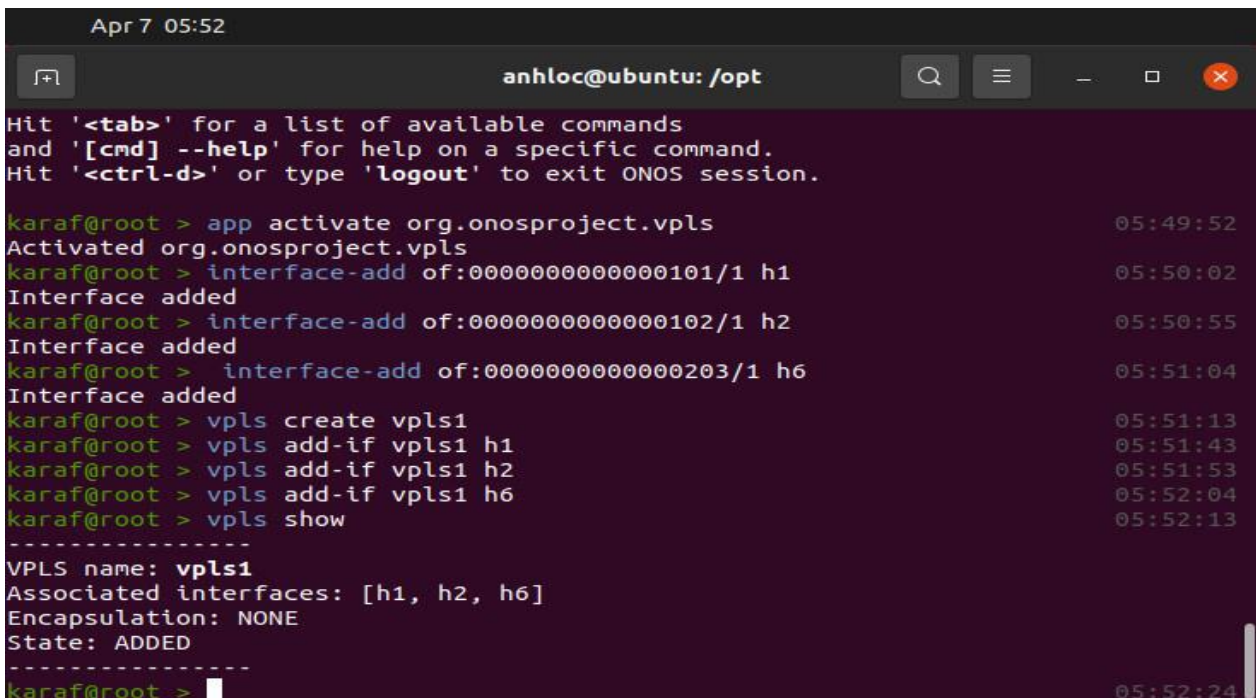


Hình 3. 21 Kết quả hiển thị bằng đường gạch đứt màu cam



```
anhloc@ubuntu: /opt x anhloc@ubuntu: ~ x anhloc@ubuntu: ~ x
1x2, s2x2) (s1x3, s1x1) (s1x3, s2x3) (s2x1, s2x2) (s2x1, s3x1) (s2x2, s2x3) (s2
x2, s3x2) (s2x3, s2x1) (s2x3, s3x3) (s3x1, s1x1) (s3x1, s3x2) (s3x2, s1x2) (s3x
2, s3x3) (s3x3, s1x3) (s3x3, s3x1)
*** Configuring hosts
h1x1 h1x2 h1x3 h2x1 h2x2 h2x3 h3x1 h3x2 h3x3
*** Starting controller
c0
*** Starting 9 switches
s1x1 s1x2 s1x3 s2x1 s2x2 s2x3 s3x1 s3x2 s3x3 ...
*** Starting CLI:
mininet> pingall
*** Ping: testing ping reachability
h1x1 -> h1x2 h1x3 h2x1 h2x2 h2x3 h3x1 h3x2 h3x3
h1x2 -> h1x1 h1x3 h2x1 h2x2 h2x3 h3x1 h3x2 h3x3
h1x3 -> h1x1 h1x2 h2x1 h2x2 h2x3 h3x1 h3x2 h3x3
h2x1 -> h1x1 h1x2 h1x3 h2x2 h2x3 h3x1 h3x2 h3x3
h2x2 -> h1x1 h1x2 h1x3 h2x1 h2x2 h2x3 h3x1 h3x2 h3x3
h2x3 -> h1x1 h1x2 h1x3 h2x1 h2x2 h3x1 h3x2 h3x3
h3x1 -> h1x1 h1x2 h1x3 h2x1 h2x2 h2x3 h3x2 h3x3
h3x2 -> h1x1 h1x2 h1x3 h2x1 h2x2 h2x3 h3x1 h3x3
h3x3 -> h1x1 h1x2 h1x3 h2x1 h2x2 h2x3 h3x1 h3x2
*** Results: 0% dropped (72/72 received)
mininet> link s1x1 s2x1 down
mininet>
```

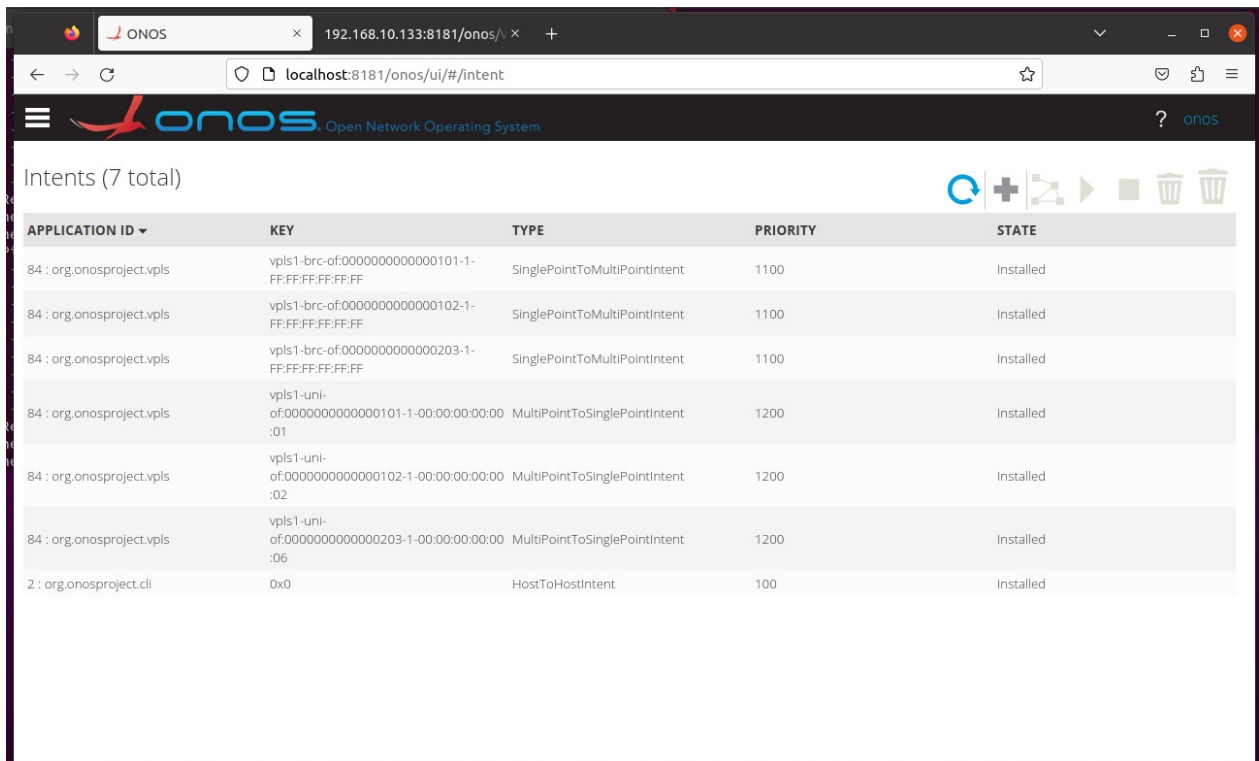
Hình 3. 22 Ngắt kết nối giữa switch 1 và switch 2



```
Apr 7 05:52
anhloc@ubuntu: /opt
Hit '<tab>' for a list of available commands
and '[cmd] --help' for help on a specific command.
Hit '<ctrl-d>' or type 'logout' to exit ONOS session.

karaf@root > app activate org.onosproject.vpls 05:49:52
Activated org.onosproject.vpls
karaf@root > interface-add of:00000000000000101/1 h1 05:50:02
Interface added
karaf@root > interface-add of:00000000000000102/1 h2 05:50:55
Interface added
karaf@root > interface-add of:00000000000000203/1 h6 05:51:04
Interface added
karaf@root > vpls create vpls1 05:51:13
karaf@root > vpls add-if vpls1 h1 05:51:43
karaf@root > vpls add-if vpls1 h2 05:51:53
karaf@root > vpls add-if vpls1 h6 05:52:04
karaf@root > vpls show 05:52:13
-----
VPLS name: vpls1
Associated interfaces: [h1, h2, h6]
Encapsulation: NONE
State: ADDED
-----
karaf@root >
```

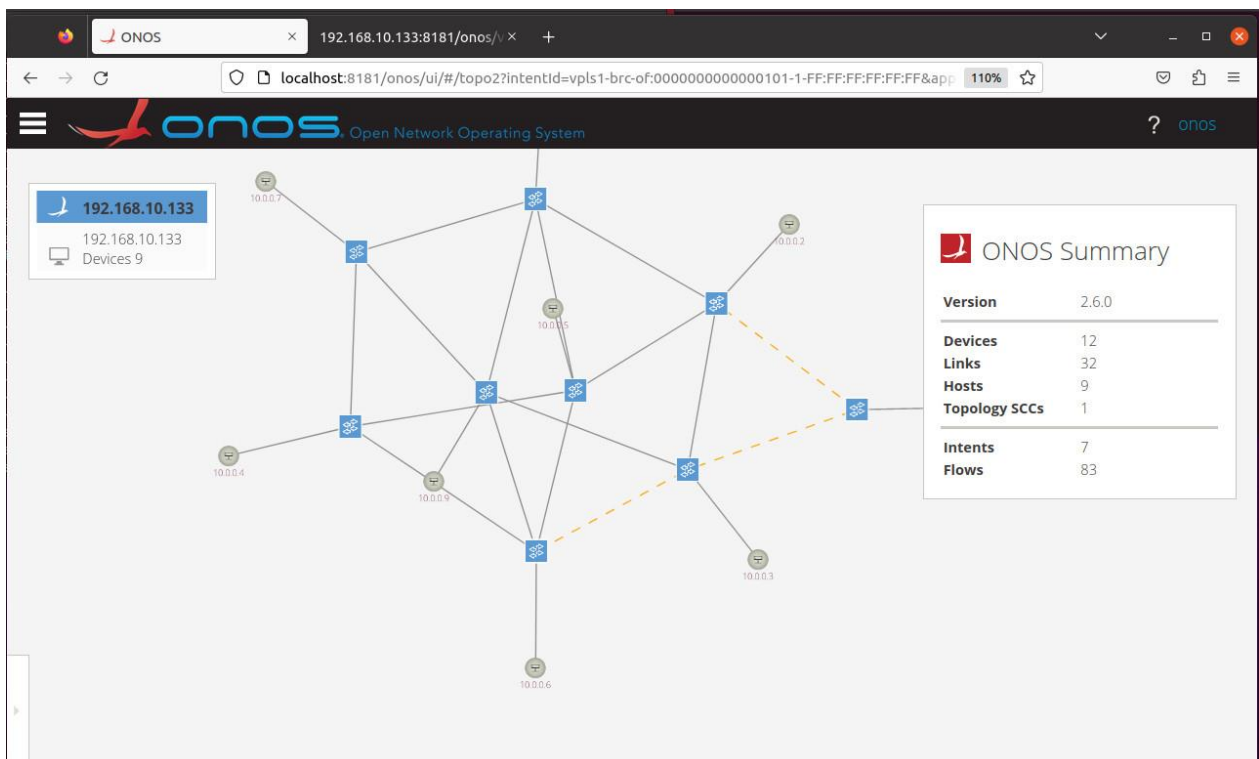
Hình 3. 23 Kích hoạt các ứng dụng cần thiết và tạo VPLS



The screenshot shows the ONOS web interface with the 'Intents' tab selected. It displays a table of 7 installed intents. The table has columns for Application ID, Key, Type, Priority, and State. The intents are categorized by application ID (84 and 2) and type (SinglePointToMultiPointIntent and MultiPointToSinglePointIntent).

APPLICATION ID	KEY	TYPE	PRIORITY	STATE
84 : org.onosproject.vpls	vpls1-brc-of:0000000000000101-1-FF:FF:FF:FF:FF:FF	SinglePointToMultiPointIntent	1100	Installed
84 : org.onosproject.vpls	vpls1-brc-of:0000000000000102-1-FF:FF:FF:FF:FF:FF	SinglePointToMultiPointIntent	1100	Installed
84 : org.onosproject.vpls	vpls1-brc-of:0000000000000203-1-FF:FF:FF:FF:FF:FF	SinglePointToMultiPointIntent	1100	Installed
84 : org.onosproject.vpls	vpls1-uni-of:0000000000000101-1-00:00:00:00:00:00:01	MultiPointToSinglePointIntent	1200	Installed
84 : org.onosproject.vpls	vpls1-uni-of:0000000000000102-1-00:00:00:00:00:00:02	MultiPointToSinglePointIntent	1200	Installed
84 : org.onosproject.vpls	vpls1-uni-of:0000000000000203-1-00:00:00:00:00:00:06	MultiPointToSinglePointIntent	1200	Installed
2 : org.onosproject.cli	0x0	HostToHostIntent	100	Installed

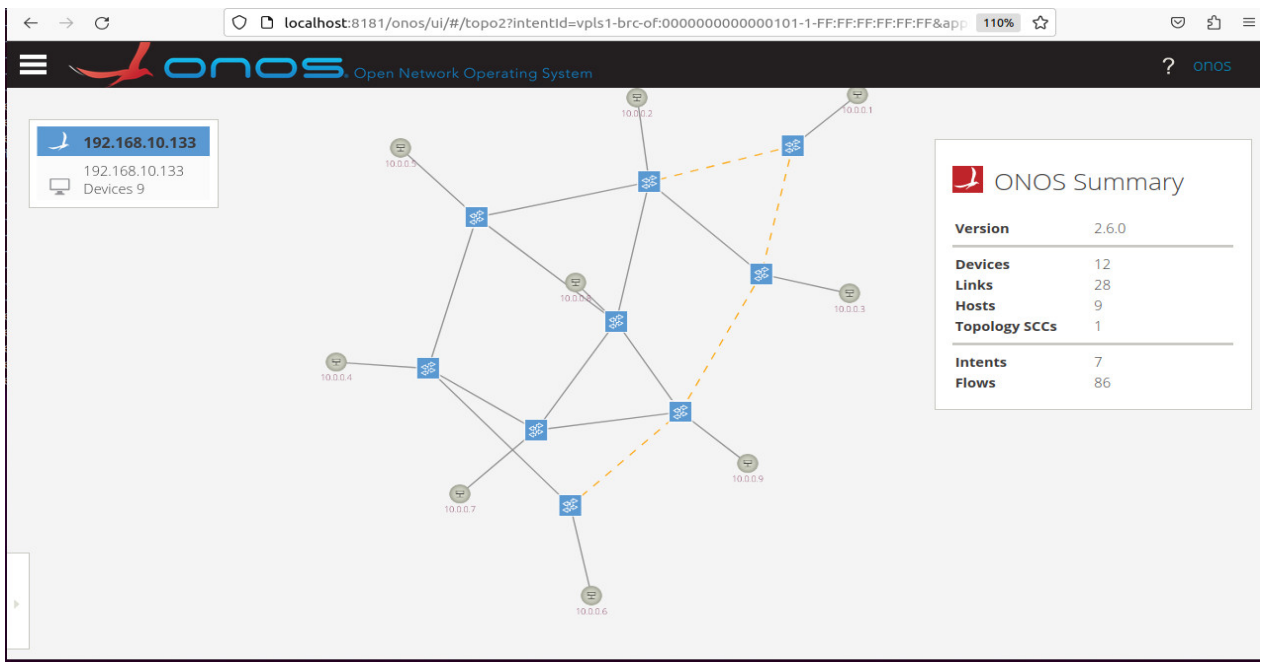
Hình 3. 24 Các ý định mới do VPLS tạo ra



Hình 3. 25 Giao diện thiết lập từ chế độ xem cấu trúc liên kết

```
anhloc@ubuntu: ~  
anhloc@ubuntu: /opt x anhloc@ubuntu: ~ x anhloc@ubuntu: ~ x  
h2x1 -> h1x1 h1x2 h1x3 h2x2 h2x3 h3x1 h3x2 h3x3  
h2x2 -> h1x1 h1x2 h1x3 h2x1 h2x3 h3x1 h3x2 h3x3  
h2x3 -> h1x1 h1x2 h1x3 h2x1 h2x2 h3x1 h3x2 h3x3  
h3x1 -> h1x1 h1x2 h1x3 h2x1 h2x2 h2x3 h3x2 h3x3  
h3x2 -> h1x1 h1x2 h1x3 h2x1 h2x2 h2x3 h3x1 h3x3  
h3x3 -> h1x1 h1x2 h1x3 h2x1 h2x2 h2x3 h3x1 h3x2  
*** Results: 0% dropped (72/72 received)  
mininet> link s1x1 s2x1 down  
mininet> pingall  
*** Ping: testing ping reachability  
h1x1 -> h1x2 h1x3 h2x1 h2x2 h2x3 h3x1 h3x2 h3x3  
h1x2 -> h1x1 h1x3 h2x1 h2x2 h2x3 h3x1 h3x2 h3x3  
h1x3 -> h1x1 h1x2 h2x1 h2x2 h2x3 h3x1 h3x2 h3x3  
h2x1 -> h1x1 h1x2 h1x3 h2x2 h2x3 h3x1 h3x2 h3x3  
h2x2 -> h1x1 h1x2 h1x3 h2x1 h2x3 h3x1 h3x2 h3x3  
h2x3 -> h1x1 h1x2 h1x3 h2x1 h2x2 h3x1 h3x2 h3x3  
h3x1 -> h1x1 h1x2 h1x3 h2x1 h2x2 h2x3 h3x2 h3x3  
h3x2 -> h1x1 h1x2 h1x3 h2x1 h2x2 h2x3 h3x1 h3x3  
h3x3 -> h1x1 h1x2 h1x3 h2x1 h2x2 h2x3 h3x1 h3x2  
*** Results: 0% dropped (72/72 received)  
mininet> link s1x1 s3x1 down  
mininet> link s1x3 s2x3 down  
mininet> link s2x2 s2x3 down  
mininet> |
```

Hình 3. 26 Phá vỡ một số liên kết



Hình 3. 27 Giao diện GUI sau phá vỡ nhưng không có gì thay đổi

3.2.2 Xây dựng thuật toán Native Bayer

Thông thường để xây dựng thuật toán Native Bayer cần thực hiện các bước sau:

- Đọc dữ liệu từ máy ảnh số: Thuật toán native Bayer bắt đầu bằng cách đọc dữ liệu từ máy ảnh số, bao gồm các giá trị pixel từ cảm biến màu Bayer.
- Xác định các pixel xanh lá cây và đỏ: Với mạng lưới Bayer, các pixel xanh lá cây và đỏ xen kẽ trên hàng. Bạn có thể xác định các pixel xanh lá cây và đỏ bằng cách lấy giá trị của các pixel chẵn hoặc lẻ trong hàng tương ứng.
- Xác định các pixel xanh dương và trắng đen: Tương tự như bước 2, các pixel xanh dương và trắng đen cũng xen kẽ trên hàng. Bạn có thể xác định các pixel xanh dương và trắng đen bằng cách lấy giá trị của các pixel chẵn hoặc lẻ trong hàng tương ứng.
- Tính toán giá trị màu: Sau khi xác định các pixel của các màu cơ bản, bạn có thể tính toán giá trị màu cho các pixel trung gian bằng cách sử dụng các giá trị pixel xung quanh. Ví dụ, để tính toán giá trị màu cho một pixel xanh lá cây trung gian, bạn có thể lấy giá trị trung bình của các pixel xung quanh của nó, bao gồm các pixel xanh lá cây, đỏ và xanh dương.
- Nâng cao chất lượng ảnh: Sau khi tính toán giá trị màu cho tất cả các pixel trung gian, bạn có thể áp dụng các kỹ thuật xử lý ảnh để nâng cao chất lượng ảnh, chẳng hạn như giảm nhiễu, tăng cường độ tương phản.

Một số đoạn code demo đơn giản về thuật toán native bayer:

```
# Import thư viện
from sklearn.datasets import load_iris
from sklearn.naive_bayes import GaussianNB
from sklearn.model_selection import train_test_split

# Load dữ liệu
iris = load_iris()

# Chia dữ liệu thành tập huấn luyện và tập kiểm tra
X_train, X_test, y_train, y_test = train_test_split(iris.data, iris.target, test_size=0.3)

# Tạo mô hình Naive Bayes
gnb = GaussianNB()
gnb.fit(X_train, y_train)

# Đánh giá hiệu suất của mô hình bằng cách sử dụng tập kiểm tra
accuracy = gnb.score(X_test, y_test)
print("Độ chính xác của mô hình là:", accuracy)
```

Hình 3. 28 xây dựng một mô hình phân loại Naive Bayes dựa trên dữ liệu iris

Đoạn code trên sử dụng thư viện Scikit-learn để xây dựng một mô hình phân loại Naive Bayes dựa trên dữ liệu iris. Mô hình này được huấn luyện và đánh giá độ chính xác bằng cách sử dụng tập dữ liệu iris, bao gồm các thông tin về đặc tính của loài hoa iris và loại của loài hoa đó.

```
import nltk
from collections import defaultdict
import string

data = [("Xin chào, tôi làm việc cho công ty ABC.", "Không spam"),
        ("Xin chào, bạn có muốn tăng cường kỹ năng lập trình của mình không?", "Không spam"),
        ("Chúc mừng! Bạn đã trúng thưởng.", "Spam"),
        ("Đăng ký ngay để nhận ưu đãi đặc biệt.", "Spam"),
        ("Chúc mừng bạn đã trúng thưởng lớn", "Spam"),
        ("Đăng ký ngay để nhận giảm giá và ưu đãi đặc biệt", "Spam"),
        ("Bạn đã trúng thưởng và giảm giá sản phẩm", "Spam"),
        ("Xin chào, tôi là thành viên của công ty", "Không spam"),
        ("Trân trọng xin chào bạn, tôi là người quen của bạn", "Không spam"),
        ("Bạn đã nhận được giảm giá bên tôi", "Spam"),
        ("Giảm giá đây chúc mừng bạn nhận được giảm giá 50%", "Spam"),
        ("Giả giá khủng mừng sinh nhật", "Spam"),
        ("Xin chào, tôi là đồng nghiệp của bạn", "Không spam")
        ]

def train(data):
    # Khởi tạo bộ đếm cho từng từ trong từng nhãn
    count = defaultdict(lambda: defaultdict(int))
    # Khởi tạo bộ đếm cho mỗi nhãn
    label_count = defaultdict(int)
    # Tách các từ trong từng email và đếm số lần xuất hiện của từng từ trong từng nhãn
    for email, label in data:
        for word in nltk.word_tokenize(email.translate(str.maketrans(' ', '', string.punctuation))):
            count[word.lower()][label.lower()] += 1
            label_count[label.lower()] += 1
    # Tính toán xác suất của từng từ trong từng nhãn
    probabilities = defaultdict(lambda: defaultdict(float))
    for word, label_count in count.items():
        total = sum(label_count.values())
        for label, count in label_count.items():
            probabilities[word][label] = count / total
    # Tính toán xác suất của mỗi nhãn
    total = sum(label_count.values())
    label_probabilities = {label: count / total for label, count in label_count.items()}
    return probabilities, label_probabilities
```

Hình 3. 29 Code sử dụng thuật toán native bayer để phân loại email 1


```
return probabilities, label_probabilities

def predict(email, probabilities, label_probabilities):
    email_probabilities = defaultdict(float)
    # Tính toán xác suất của email cho từng nhãn
    for label in label_probabilities:
        email_probabilities[label] = label_probabilities[label]
        for word in nltk.word_tokenize(email.translate(str.maketrans('', '', string.punctuation))):
            email_probabilities[label] *= probabilities[word.lower()][label]
    # Chuẩn hóa xác suất để tổng bằng 1
    total = sum(email_probabilities.values())
    if total == 0:
        num_labels = len(label_probabilities)
        for label in label_probabilities:
            email_probabilities[label] = 1/num_labels
    else:
        for label in email_probabilities:
            email_probabilities[label] /= total
    return email_probabilities

def classify(email, probabilities, label_probabilities):
    # Tính toán xác suất của email cho từng nhãn
    email_probabilities = predict(email, probabilities, label_probabilities)
    # Chọn nhãn có xác suất cao nhất
    return max(email_probabilities, key=email_probabilities.get)

# Huấn luyện mô hình
probabilities, label_probabilities = train(data)

# Dự đoán nhãn cho email mới
new_email = "Xin chào ,tôi là đồng nghiệp của bạn"
predicted_label = classify(new_email, probabilities, label_probabilities)

print(f"Nhãn của email '{new_email}' được dự đoán là: {predicted_label}")
```

Hình 3. 30 Code sử dụng thuật toán native bayer để phân loại email 2

Đoạn code trên thực hiện xây dựng một mô hình phân loại email là thư rác (spam) hay không là thư rác (không spam) bằng thuật toán Naive Bayes.

CHƯƠNG 4: KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN

4.1 Kết luận

Trong báo cáo này, chúng tôi đã giới thiệu và ứng dụng mô hình mạng SDN và thuật toán Naive Bayes vào mô hình mạng doanh nghiệp. Kết quả thực nghiệm đã cho thấy rằng mô hình SDN cung cấp các ưu điểm như khả năng điều khiển mạng linh hoạt, cấu hình mạng tự động, tối ưu hóa đường truyền, giảm chi phí và cải thiện tính sẵn sàng của mạng.

Đồng thời, kết quả thực nghiệm cũng cho thấy thuật toán Naive Bayes là một phương pháp học máy hiệu quả để xử lý dữ liệu trong mô hình mạng SDN. Thuật toán Naive Bayes có khả năng dự đoán và phân loại dữ liệu với độ chính xác cao và có thể được sử dụng để phát hiện các mối đe dọa an ninh mạng.

Tuy nhiên, mô hình mạng SDN còn một số hạn chế như khả năng quản lý bảo mật yếu, khó khăn trong việc triển khai, cấu hình và bảo trì mạng. Đối với thuật toán Naive Bayes, nó cũng có một số hạn chế như không xử lý được dữ liệu phức tạp, phụ thuộc vào giả định về sự độc lập của các thuộc tính và độ chính xác của phân loại có thể bị ảnh hưởng bởi sự không cân bằng trong dữ liệu.

Tóm lại, việc ứng dụng mô hình mạng SDN và thuật toán Naive Bayes vào mô hình mạng doanh nghiệp có thể mang lại nhiều lợi ích và giải quyết được một số vấn đề trong quản lý mạng. Tuy nhiên, cần có sự cân nhắc và đánh giá kỹ lưỡng về các ưu nhược điểm của mô hình trước khi triển khai và áp dụng trong thực tế.

4.2 Đánh giá mô hình mạng SDN

Sau khi cài đặt và thử nghiệm thì em có một số đánh giá cơ bản về mô hình mạng SDN như sau:

Về mặt ưu điểm thì mô hình mạng SDN có những ưu điểm sau:

Tính linh hoạt và dễ dàng quản lý: Mô hình mạng SDN có tính linh hoạt cao trong việc cấu hình, giám sát, quản lý và mở rộng mạng, đặc biệt là khi phải đáp ứng các yêu cầu mới hoặc tăng tải.

Tính độc lập về phần cứng: SDN tách biệt các phần mềm điều khiển và phần cứng trong mạng, giúp quản lý hệ thống dễ dàng hơn và cho phép sử dụng các thiết bị mạng khác nhau từ nhiều nhà sản xuất khác nhau.

Tính độc lập về nền tảng: Mô hình SDN giúp tách biệt phần cứng mạng với hệ điều hành và ứng dụng, cho phép cài đặt các ứng dụng và dịch vụ mạng một cách độc lập với phần cứng và hệ điều hành.

Tăng cường tính bảo mật: Mô hình mạng SDN cung cấp khả năng giám sát và phát hiện các mối đe dọa mạng nhanh hơn, từ đó đáp ứng các yêu cầu bảo mật và giúp tăng cường tính bảo mật cho mạng.

Tuy nhiên, mô hình mạng SDN cũng có một số hạn chế và nhược điểm, bao gồm:

Phụ thuộc vào mạng internet: SDN yêu cầu kết nối internet tốt để hoạt động tốt, nếu có sự cố về kết nối internet thì mô hình mạng SDN sẽ gặp khó khăn trong việc truyền tải và xử lý dữ liệu.

Chi phí đầu tư ban đầu cao: Với mô hình mạng SDN, việc triển khai yêu cầu phải có các thiết bị đặc biệt để xử lý, ngoài ra cần phải có những kỹ thuật viên có kinh nghiệm để quản lý và vận hành.

Tăng thời gian đáp ứng: Do phần mềm điều khiển SDN đóng vai trò quản lý, nên các yêu cầu của mạng sẽ phải được xử lý qua nhiều phần mềm. Điều này dẫn đến thời gian đáp ứng chậm hơn so với các mô hình mạng truyền thống.

4.3 Kiến thức đạt được

Trong quá trình thực hiện đề tài "Ứng dụng SDN và thuật toán Naive Bayes vào mô hình mạng doanh nghiệp", chúng em đã đạt được những kiến thức sau:

- Hiểu về kiến trúc mạng SDN và cách thức hoạt động của nó.
- Tìm hiểu về thuật toán Naive Bayes và ứng dụng của nó trong phân loại dữ liệu.
- Hiểu về quá trình xây dựng mô hình mạng SDN và các công cụ hỗ trợ trong quá trình này.
- Đánh giá hiệu quả của mô hình mạng SDN với sự hỗ trợ của thuật toán Naive Bayes và so sánh với các mô hình mạng khác.

Từ những kiến thức trên, chúng ta có thể áp dụng và phát triển các mô hình mạng SDN trong thực tế để giải quyết các vấn đề liên quan đến quản lý mạng và bảo mật thông tin trong doanh nghiệp.

4.4 Hướng phát triển

Dựa trên những kết quả đạt được và kiến thức thu được trong quá trình thực hiện đề tài "Ứng dụng SDN và thuật toán Naive Bayes vào mô hình mạng doanh nghiệp", chúng em có đề xuất một số hướng phát triển như sau:

- Nghiên cứu và áp dụng các thuật toán machine learning khác như Random Forest, Decision Tree,... để tăng độ chính xác của mô hình phân loại dữ liệu.
- Tìm hiểu và ứng dụng các công nghệ mới như kỹ thuật blockchain để cải thiện tính bảo mật của mô hình mạng SDN.
- Nghiên cứu và phát triển các công cụ hỗ trợ quản lý và giám sát mạng SDN để tăng tính linh hoạt và dễ dàng quản lý hệ thống.
- Nghiên cứu và áp dụng các kiến thức mới như Internet of Things (IoT), Big Data để cải thiện tính hiệu quả và khả năng ứng dụng của mô hình mạng SDN.
- Phát triển các sản phẩm và dịch vụ liên quan đến mô hình mạng SDN và thuật toán Naive Bayes để cung cấp cho các doanh nghiệp và tổ chức có nhu cầu sử dụng.
- Tiếp tục nghiên cứu và phát triển các kỹ thuật mới để giải quyết các vấn đề về bảo mật và quản lý mạng trong doanh nghiệp.

DANH MỤC TÀI LIỆU THAM KHẢO

- [1] Software-Defined Networking with OpenFlow, Second Edition by Siamak Azodolmolky, Yonatan Koren, and Shiva Shankar, Morgan Kaufmann Publishers.
- [2] Practical SDN and OpenFlow Fundamentals by Flavio Bonomi, Raj Jain, and Srinivasan Keshav, O'Reilly Media.
- [3] Machine Learning: A Probabilistic Perspective by Kevin P. Murphy, MIT Press.
- [4] Pattern Recognition and Machine Learning by Christopher M. Bishop, Springer.
- [5] Data Mining: Practical Machine Learning Tools and Techniques by Ian H. Witten, Eibe Frank, and Mark A. Hall, Morgan Kaufmann Publishers.
- [6] Software-Defined Networking: Design and Deployment by Thomas D. Nadeau, Ken Gray, and H. Jonathan Chao. Nhà xuất bản: Wiley.
- [7] Naive Bayes Classification Algorithm by Priti Srinivas Sajja and Rajendra Akerkar. Nhà xuất bản: Springer.
- [8] Software-Defined Networking with OpenFlow, 2nd Edition by Siamak Azodolmolky, Yiming Jiang, and David Tipper. Nhà xuất bản: Packt Publishing.
- [9] Data Mining and Machine Learning in Cybersecurity by Sumeet Dua and Xian Du. Nhà xuất bản: Auerbach Publications.
- [10] Network Function Virtualization by Rafael Fernández-Pascual, Enrique Hernandez-Orallo, and Arturo Mayoral López de Lerma. Nhà xuất bản: Wiley.