ระบบจัดการเครือข่ายและควบคุมการจราจรบนเครือข่าย โดยใช้สถาปัตยกรรมแบบเอสดีเอ็น

NETWORK MANAGEMENT AND TRAFFIC CONTROL SYSTEM USING SDN ARCHITECTURE

โดย

กฤษฎา อิงอาน

KRISSADA INGARN

สถาพร แดงน้อย

SATHAPON DANGNOI

อาจารย์ที่ปรึกษา

ผู้ช่วยศาสตราจารย์ ดร.สุเมธ ประภาวัต

ปริญญานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรวิทยาศาสตรบัณฑิต

สาขาวิชาเทคโนโลยีสารสนเทศ

คณะเทคโนโลยีสารสนเทศ

สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

ภาคเรียนที่ 2 ปีการศึกษา 2560

NETWORK MANAGEMENT AND TRAFFIC CONTROL SYSTEM USING SDN ARCHITECTURE

KRISSADA INGARN

SATHAPON DANGNOI

A PROJECT SUBMITTED IN PARTIAL FULFILLMENT

OF THE REQUIREMENT FOR THE DEGREE OF

BACHELOR OF SCIENCE PROGRAM IN INFORMATION TECHNOLOGY

FACULTY OF INFORMATION TECHNOLOGY

KING MONGKUT'S INSTITUTE OF TECHNOLOGY LADKRABANG

2/2017

ใบรับรองปริญญานิพนธ์ประจำปีการศึกษา 2560

คณะเทคโนโลยีสารสนเทศ

สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

เรื่อง ระบบจัดการเครือข่ายและควบคุมการจราจรบนเครือข่าย โดยใช้สถาปัตยกรรมแบบเอสดีเอ็น

NETWORK MANAGEMENT AND TRAFFIC CONTROL SYSTEM USING SDN ARCHITECTURE

ผู้จัดทำ

1. **นายกฤษฎา อิงอาน รหัสนักศึกษา 57070002**
2. **นายสถาพร แดงน้อย รหัสนักศึกษา 57070120**

..... อาจารย์ที่ปรึกษา

(ผู้ช่วยศาสตราจารย์ ดร.สุเมธ ประภาวัต)

หัวข้อโครงการ	ระบบจัดการเครือข่ายและควบคุมการจราจรบนเครือข่าย โดยใช้สถาปัตยกรรมแบบเอสดีเอ็น	
นักศึกษา	นายกฤษฎา อิงอาน	รหัสนักศึกษา 57070002
	นายสถาพร แดงน้อย	รหัสนักศึกษา 57070120
ปริญญา	วิทยาศาสตรบัณฑิต	
สาขาวิชา	เทคโนโลยีสารสนเทศ	
ปีการศึกษา	2560	
อาจารย์ที่ปรึกษา	ผู้ช่วยศาสตราจารย์ ดร.สุเมธ ประภาวัต	

บทคัดย่อ

ในยุคที่เทคโนโลยีเครือข่ายมีการพัฒนาไปอย่างรวดเร็ว เพื่อรองรับปริมาณการใช้งานเครือข่ายที่เพิ่มสูงขึ้นอย่างต่อเนื่อง ซึ่งได้เกิดแนวคิดในการบริหารจัดการระบบเครือข่ายที่ชื่อว่า Software Define Network (SDN) ที่เป็นการนำซอฟต์แวร์เข้ามาควบคุมการทำงานของระบบเครือข่ายให้เป็นไปตามความต้องการ เพื่อช่วยเพิ่มความยืดหยุ่นและความสามารถในการควบคุมการทำงานของระบบเครือข่ายให้ดียิ่งขึ้น แต่อุปกรณ์เครือข่ายต้องสามารถรองรับการทำงานของ SDN ด้วย ซึ่งต้องมีการลงทุน ทางผู้จัดทำจึงได้พัฒนาระบบบริหารจัดการเครือข่ายที่มีการทำงานอ้างอิงตามแนวคิดของ SDN แต่เป็นการประยุกต์ให้สามารถทำงานบนอุปกรณ์เครือข่ายเดิมที่ไม่รองรับกับการทำงานนั้นให้ทำงานเสมือนกับใช้งานอุปกรณ์เครือข่ายที่รองรับการทำงานของแนวคิดดังกล่าว อีกทั้งยังมีการสร้างช่องทางการเชื่อมต่อสำหรับการเรียกใช้งานความสามารถของระบบเพื่อให้โปรแกรมประยุกต์สำหรับการจัดการเครือข่ายสามารถนำไปใช้ในการทำงานได้ และมีการพัฒนาโปรแกรมประยุกต์ต้นแบบสำหรับการจัดการจราจรภายในเครือข่ายที่นำหลักการของวิศวกรรมจราจรมาปรับใช้ในการควบคุมเส้นทางการจราจรเพื่อลดความคับคั่งบนเส้นทางหนึ่ง ๆ ที่อาจเกิดขึ้น

กิตติกรรมประกาศ

ปริญญานิพนธ์ฉบับนี้สำเร็จลุล่วงไปได้ด้วยดี ด้วยความกรุณาของอาจารย์ที่ปรึกษาโครงงานผู้ช่วยศาสตราจารย์ ดร.สุเมธ ประภาวัต ที่คอยให้ความรู้ ให้คำแนะนำในการทำงานให้คำปรึกษาเมื่อเกิดปัญหา และให้กำลังใจตลอดระยะเวลาการทำโครงงาน ขอขอบพระคุณอาจารย์เป็นอย่างสูง

ขอขอบพระคุณอาจารย์ในคณะเทคโนโลยีสารสนเทศ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบังทุกท่านที่คอยให้ความรู้ที่สามารถนำไปประยุกต์ใช้กับการทำโครงการได้

ขอขอบคุณ นายธนานพ ทองถาวร และนายวรวัชร ณรงคะชวนะ ที่คอยให้คำปรึกษาและคำแนะนำตลอดระยะเวลาในการทำโครงงาน

ขอขอบคุณครอบครัว และเพื่อนพี่น้องชาวไอทีลาดกระบังทุกคนที่คอยช่วยเหลือ ให้การสนับสนุนและให้กำลังใจเสมอมา

กฤษฎา อิงอาน

สถาพร แดงน้อย

สารบัญ (ต่อ)

หน้า

สารบัญรูป

รูปที่ **หน้า**

สารบัญรูป (ต่อ)

รูปที่ **หน้า**

บทที่ 1
บทนำ

1.1 ความเป็นมาและความสำคัญของปัญหา

อันเนื่องมาจากการขยายตัวของระบบเครือข่ายที่เพิ่มสูงขึ้น ทำให้ในปัจจุบันระบบเครือข่ายมีขนาดที่ใหญ่และมีความซับซ้อนมากยิ่งขึ้น ซึ่งเมื่อเป็นเช่นนั้นก็เป็นไปได้ยากที่ผู้ดูแลระบบจะสามารถบริหารจัดการระบบเครือข่ายให้สามารถทำงานได้อย่างเต็มประสิทธิภาพ เพราะว่าเมื่อมีปริมาณของข้อมูลที่รับส่งภายในเครือข่ายเป็นจำนวนมากและใช้กระบวนการหาเส้นทางเป็นแบบอัตโนมัติ ซึ่งจะเลือกเส้นทางในการรับส่งข้อมูลจากเส้นทางที่ดีที่สุดแล้วนั้น ก็มีความเป็นไปได้ที่อาจจะเกิดการคับคั่งของข้อมูลในเส้นทางหนึ่งๆที่มีการใช้งานเป็นจำนวนมาก ซึ่งเมื่อเป็นเช่นนั้นก็อาจจะส่งผลตามมาคือ การรับส่งข้อมูลอาจเกิดความล่าช้าหรือข้อมูลอาจจะสูญหายในกรณีที่อุปกรณ์เครือข่ายไม่สามารถรองรับปริมาณของข้อมูลที่มากเกินไปได้ ผู้จัดทำจึงได้มีแนวคิดที่จะสร้างระบบขึ้นมาเพื่อใช้ในการบริหารจัดการในส่วนนี้ โดยระบบที่พัฒนาขึ้นนั้นจะทำงานควบคู่ไปกับการทำงานตามปกติของระบบเครือข่ายที่ใช้การหาเส้นทางแบบอัตโนมัติและจะคอยทำการตรวจสอบหาเส้นทางการรับส่งข้อมูลภายในระบบเครือข่ายที่มีความคับคั่งแล้วทำการย้ายเส้นทางการส่งของข้อมูลที่ใช้เส้นทางนั้นอยู่บางส่วนไปใช้เส้นทางอื่น เพื่อช่วยลดความคับคั่งของข้อมูลในเส้นทางนั้น ๆ และทำให้การรับส่งข้อมูลภายในระบบเครือข่ายมีประสิทธิภาพมากยิ่งขึ้นอีกด้วย โดยระบบที่พัฒนาขึ้นนั้นจะอ้างอิงมาจากแนวคิดของเอสดีเอ็น ซึ่งจะเน้นไปที่การพัฒนาเพื่อให้อุปกรณ์เครือข่ายที่ไม่รองรับการทำงานของเทคโนโลยีเอสดีเอ็นให้มีความสามารถในการทำงานที่เทียบเคียงกับอุปกรณ์ที่รองรับ เพื่อช่วยเพิ่มประสิทธิภาพการทำงานของระบบให้เพิ่มสูงขึ้น โดยที่ไม่จำเป็นต้องลงทุนซื้ออุปกรณ์เครือข่ายใหม่

1.2 ความมุ่งหมายและวัตถุประสงค์

1. เพื่อศึกษาแนวทางการพัฒนาระบบจัดการเครือข่ายโดยการประยุกต์หลักการแบบเอสดีเอ็น
2. เพื่อศึกษาแนวทางการควบคุมการจราจรบนเครือข่ายเพื่อการกระจายภาระงานบนเครือข่าย
3. เพื่อพัฒนาระบบแบบจำลองสำหรับการจัดการเครือข่ายและการควบคุมการจราจรบนเครือข่าย

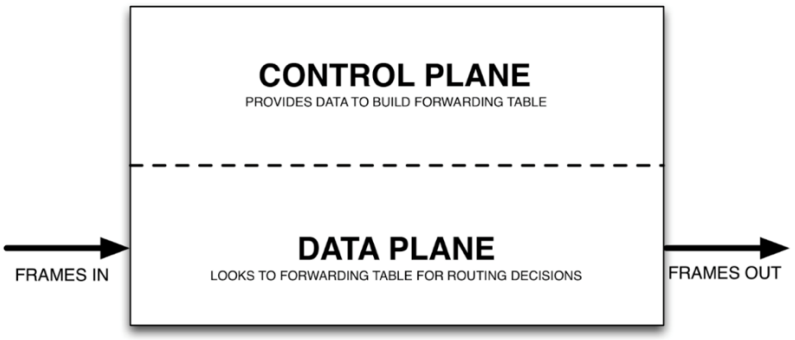
บทที่ 2
ทฤษฎีและงานวิจัยที่เกี่ยวข้อง

ในบทนี้จะกล่าวถึงทฤษฎีและเทคโนโลยีที่มีความเกี่ยวข้องกับโครงการ ซึ่งผู้พัฒนาได้ศึกษาเพื่อนำมาใช้เป็นแนวทางในการพัฒนาและดำเนินโครงการ โดยมีรายละเอียดดังนี้

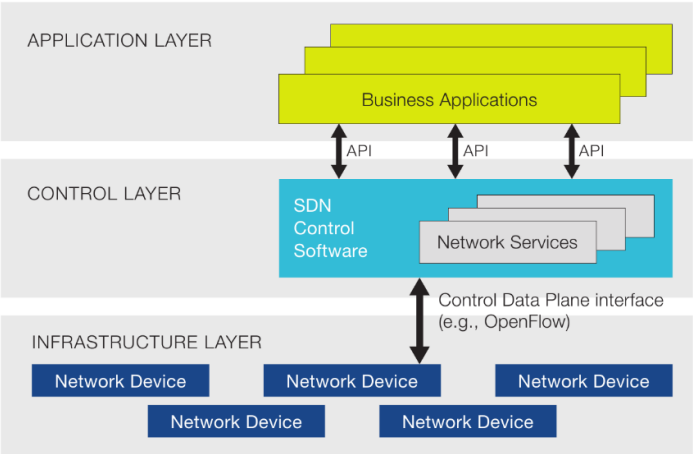
2.1 Software Define Network (SDN)

SDN [1-3] หรือเอสดีเอ็นเป็นแนวคิดในการบริหารจัดการระบบเครือข่าย โดยใช้ซอฟต์แวร์เข้ามาควบคุมการทำงานของระบบเครือข่ายและรวมศูนย์การควบคุมทั้งหมดมาไว้ที่ซอฟต์แวร์เพียงที่เดียว เพื่อทำให้ง่ายต่อการบริหารจัดการระบบเครือข่าย รวมไปถึงลดความซับซ้อนของระบบเครือข่ายที่เกิดจากการประมวลผลการทำงานของอุปกรณ์เครือข่ายแต่ละตัวเพื่อทำงานร่วมกันลง ซึ่งแนวคิดนี้มีความต้องการที่จะเปลี่ยนลักษณะการทำงานของอุปกรณ์เครือข่ายให้แตกต่างจากเดิม โดยที่ปกติในอุปกรณ์เครือข่ายแต่ละตัวจะมีลักษณะการทำงานแบ่งเป็น 2 ส่วน คือ

1. Control Plane ที่เป็นส่วนของการประมวลผลสร้างตารางเส้นทางของระบบเครือข่าย เช่น การประมวลผลเพื่อหาเส้นทางแบบอัตโนมัติ ซึ่งจะต้องมีการแลกเปลี่ยนข้อมูลระหว่างอุปกรณ์เครือข่ายและนำข้อมูลดังกล่าวมาประมวลผลเพื่อสร้างเป็นตารางเส้นทางของระบบเครือข่าย
2. Data Plane หรือ Forwarding Plane ทำหน้าที่ในการรับและส่งต่อข้อมูลออกจากอุปกรณ์เครือข่าย ซึ่งการส่งต่อข้อมูลจะเปรียบเทียบจากตารางเส้นทางของระบบเครือข่ายเพื่อหาทิศทางในการส่งข้อมูลไปยังปลายทางและส่งต่อข้อมูลออกทางอินเตอร์เฟสของอุปกรณ์เครือข่ายตามทิศทางนั้น ๆ



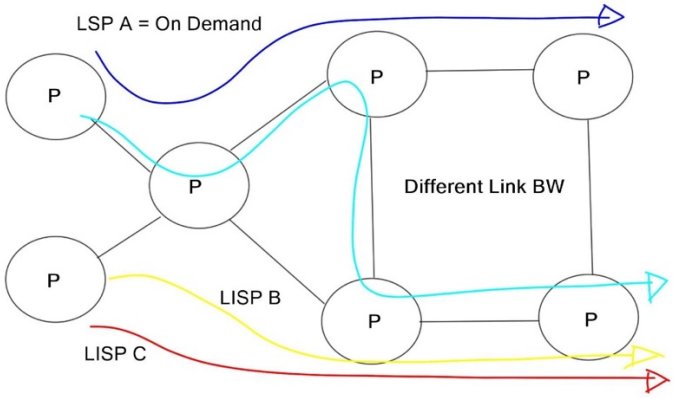
รูปที่ 2.1 แสดงลักษณะโครงสร้างการทำงานของอุปกรณ์เครือข่าย



รูปที่ 2.3 แสดงสถาปัตยกรรมของ SDN

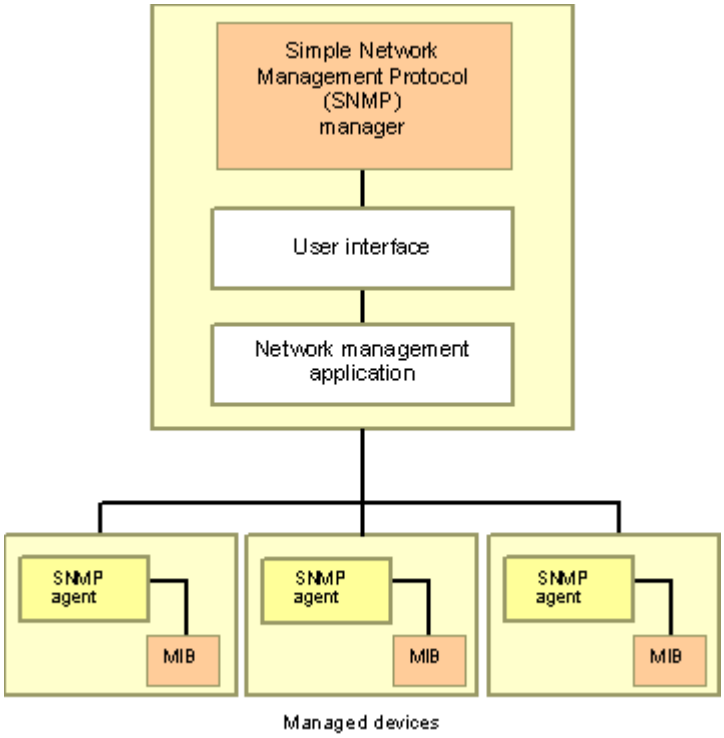
2.2 Traffic Engineering

Traffic Engineering [4-6] คือ แนวคิดในการบริหารจัดการการรับส่งข้อมูลภายในระบบเครือข่ายให้มีประสิทธิภาพการทำงานที่สูงขึ้นและสอดคล้องกับความต้องการในการใช้งานระบบเครือข่ายมากยิ่งขึ้น โดยการนำข้อมูลปริมาณการใช้งานของระบบเครือข่ายทั้งในอดีตและปัจจุบันมาวิเคราะห์เพื่อคาดการณ์ปริมาณการใช้งานที่จะเกิดขึ้นในอนาคตและหาแนวทางในการบริหารจัดการเส้นทางในการรับส่งข้อมูลของข้อมูลจำนวนนั้นเพื่อให้การรับส่งข้อมูลภายในระบบเครือข่ายมีประสิทธิภาพมากตรงตามที่ต้องการ ซึ่งในการทำวิศวกรรมจราจรนั้นอาจจะแตกต่างกันไปตามความต้องการใช้งานของระบบเครือข่ายนั้น ๆ เช่น ต้องการการรับส่งข้อมูลที่มีความล่าช้าน้อย ความเร็วในการรับส่งที่สูง เป็นต้น ซึ่งขึ้นอยู่กับความต้องการใช้งานและประเภทของข้อมูลที่มีการรับส่งภายในระบบเครือข่ายด้วย แต่ในส่วนของซอฟต์แวร์ที่พัฒนาขึ้นนั้น ผู้พัฒนามีจุดประสงค์เพื่อลดความคับคั่งของข้อมูลที่อาจจะเกิดขึ้นในเส้นทางหนึ่ง ๆ ซึ่งอาจจะส่งผลทำให้การรับส่งข้อมูลภายในระบบเครือข่ายมีความล่าช้าและอาจทำให้เกิดการสูญหายของข้อมูล โดยการกระจายการรับส่งข้อมูลออกไปยังเส้นทางต่าง ๆ ทำให้ความคับคั่งในเส้นทางหนึ่ง ๆ ลดลง



รูปที่ 2.4 แสดงการกระจายเส้นทางการส่งของข้อมูลในระบบเครือข่าย

Management Information Base (MIB) ซึ่งจะจัดเก็บข้อมูลต่าง ๆของอุปกรณ์เครือข่ายเอาไว้ และในส่วนนี้เองเครื่องที่ใช้ในการบริหารจัดการนั้นสามารถส่งคำสั่งเพื่อร้องขอข้อมูลหรือเปลี่ยนแปลงค่าการทำงานบางอย่างของอุปกรณ์เครือข่ายได้ ซึ่งในการตรวจสอบข้อมูลและสถานะการทำงานสามารถทำได้ 2 รูปแบบคือ อุปกรณ์ที่มีหน้าที่บริหารจัดการส่งคำสั่งเพื่อร้องขอข้อมูลจากอุปกรณ์เครือข่ายเครื่องอื่น ๆ เมื่ออุปกรณ์เครือข่ายได้รับคำร้องขอก็จะส่งข้อมูลที่ต้องการกลับมาให้ หรือกำหนดเงื่อนไขให้อุปกรณ์เครือข่ายส่งข้อมูลมาให้เครื่องที่มีหน้าที่บริหารจัดการเมื่อมีการกระทำตรงตามเงื่อนไขที่กำหนด



รูปที่ 2.6 แสดงโครงสร้างของ SNMP

ภายในฐานข้อมูล Management Information Base นั้นจะจัดเก็บข้อมูลในรูปของวัตถุ (Object) ที่มีหมายเลขกำกับแต่ละวัตถุ เรียกว่า Object ID (OID) ซึ่งการจัดเก็บจะมีลักษณะเป็นลำดับชั้นคล้ายต้นไม้ เพื่อให้ง่ายต่อการใช้งานและการจัดการ

```
R1#show cdp neighbors detail
-------------------------
Device ID: R2
Entry address(es):
  IP address: 10.1.12.2
Platform: Cisco 3640,  Capabilities: Router Switch IGMP
Interface: Serial0/0,  Port ID (outgoing port): Serial0/0
Holdtime : 172 sec

Version :
Cisco IOS Software, 3600 Software (C3640-JK9S-M), Version 12.4(13a), RELEASE SOFTWARE (fc1)
Technical Support: http://www.cisco.com/techsupport
Copyright (c) 1986-2007 by Cisco Systems, Inc.
Compiled Tue 06-Mar-07 20:25 by prod_rel_team

advertisement version: 2
VTP Management Domain: ''

-------------------------
Device ID: R3
Entry address(es):
  IP address: 10.1.13.3
Platform: Cisco 3640,  Capabilities: Router Switch IGMP
Interface: FastEthernet1/0,  Port ID (outgoing port): FastEthernet0/0
Holdtime : 126 sec

Version :
Cisco IOS Software, 3600 Software (C3640-JK9S-M), Version 12.4(13a), RELEASE SOFTWARE (fc1)
Technical Support: http://www.cisco.com/techsupport
Copyright (c) 1986-2007 by Cisco Systems, Inc.
Compiled Tue 06-Mar-07 20:25 by prod_rel_team

advertisement version: 2
VTP Management Domain: ''
Duplex: full
```

รูปที่ 2.8 แสดงตัวอย่างข้อมูลที่ได้รับจากโพรโทคอล CDP

2.4 การควบคุมการทำงานของอุปกรณ์เครือข่าย

2.4.1 IP-based Routing

เป็นรูปแบบในการรับส่งข้อมูลภายในระบบเครือข่ายที่ใช้หมายเลขแอดเดรสของปลายทางในการกำหนดทิศทางการรับส่งข้อมูล ซึ่งมีการใช้งานกันอย่างแพร่หลายในปัจจุบัน รวมไปถึงการนำไปใช้ควบคู่กับกระบวนการหาเส้นทางแบบอัตโนมัติที่มีกระบวนการในการแลกเปลี่ยนข้อมูลซึ่งกันและกันเพื่อสร้างเป็นตารางเส้นทางสำหรับการรับส่งข้อมูลภายในระบบเครือข่าย เช่น โพรโทคอล RIP [13], OSPF [14] เป็นต้น โดยการเลือกเส้นทางที่ใช้ในการรับส่งข้อมูลจะเลือกเส้นทางที่ดีที่สุดจากการเปรียบเทียบค่า Cost ของแต่ละเส้นทางและเลือกเส้นทางที่มีค่า Cost ดีที่สุด ซึ่งค่า Cost อาจจะแตกต่างกันออกไปตามแต่ละโพรโทคอล เช่น จำนวนของอุปกรณ์เครือข่ายที่ข้อมูลต้องผ่านจนกว่าจะถึงปลายทาง, Bandwidth ของเส้นทาง, Delay ของการรับส่งข้อมูล เป็นต้น

2.4.2 Policy-based Routing

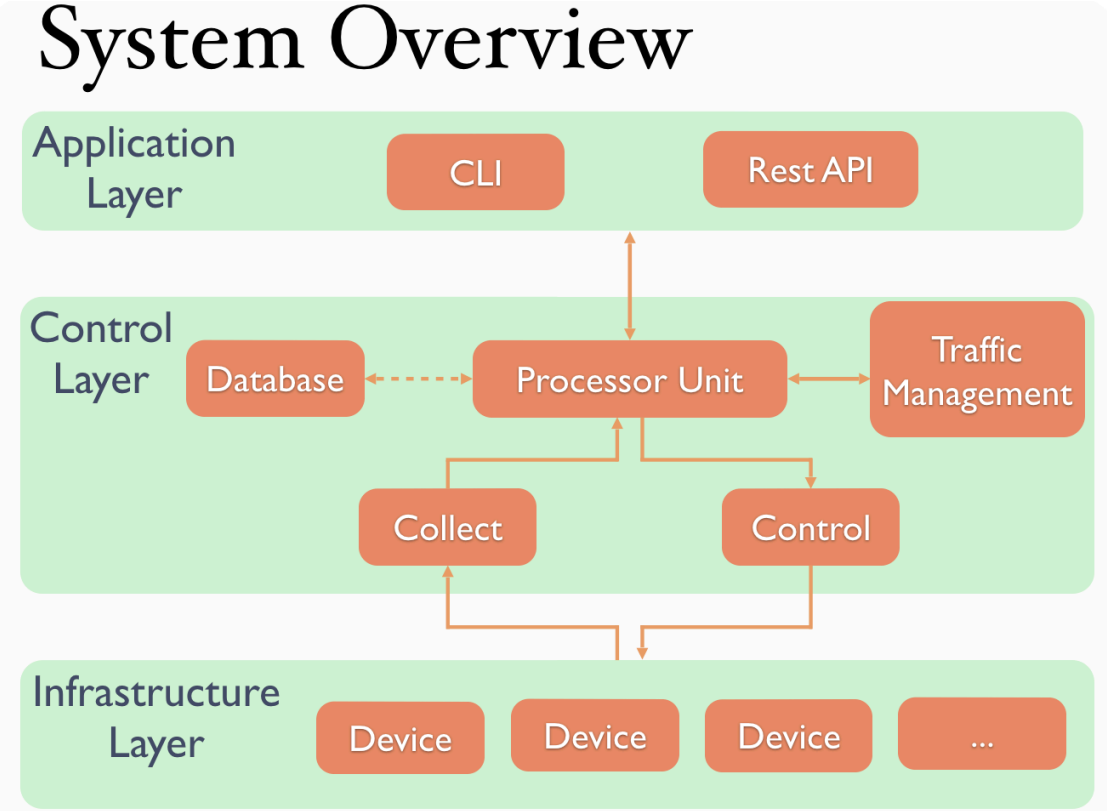
Policy-based Routing [15-17] เป็นรูปแบบการกำหนดเส้นทางการรับส่งข้อมูลภายในระบบเครือข่าย โดยใช้นโยบาย (Policy) ในการกำหนดทิศทางการรับส่งของข้อมูลแทนการใช้หมายเลขแอดเดรสปลายทางเป็นตัวกำหนด ซึ่งนโยบายจะประกอบไปด้วย 2 ส่วน คือ ลักษณะหรือคุณสมบัติของข้อมูล เช่น หมายเลขแอดเดรสต้นทาง หมายเลขแอดเดรสปลายทาง หมายเลขพอร์ต เป็นต้น และการดำเนินการต่อข้อมูลที่มีลักษณะตรงตามที่ระบุไว้ข้างต้น เช่น ส่งข้อมูลออกอินเตอร์เฟสใดของอุปกรณ์เครือข่าย ซึ่งในการทำงานเมื่ออุปกรณ์เครือข่ายได้รับข้อมูลเข้ามาจะตรวจสอบกับนโยบายที่มีการกำหนดไว้ ถ้าข้อมูลที่เข้ามามีลักษณะตรงตามที่ระบุก็จะดำเนินการกับข้อมูลตามที่ระบุในนโยบาย ซึ่งซอฟต์แวร์ที่พัฒนาใช้ Policy-based Routing เฉพาะของบริษัท Cisco ที่ชื่อว่า route map

บทที่ 3

แนวคิดและการดำเนินงาน

ในบทนี้จะกล่าวถึงสถาปัตยกรรมและการออกแบบของระบบที่ผู้พัฒนาได้นำเสนอมาข้างต้น

3.1 สถาปัตยกรรมของระบบ

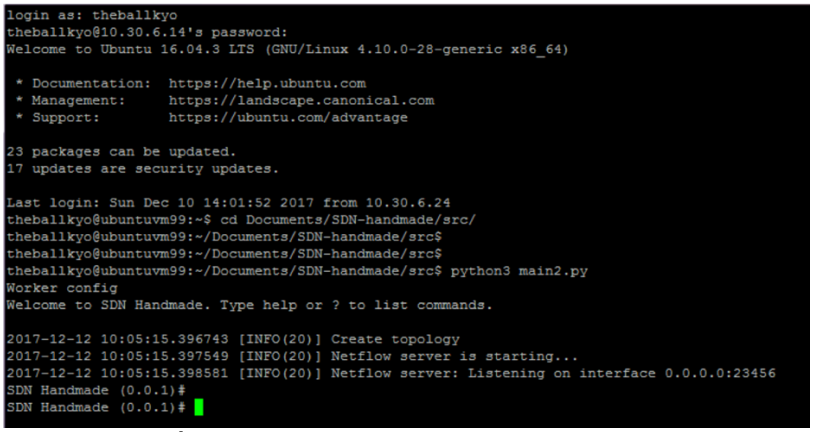


รูปที่ 3.1 แสดงแผนผังภาพรวมของระบบ

สถาปัตยกรรมของระบบแบ่งออกเป็น 3 ระดับชั้นตามลักษณะการทำงานและอ้างอิงตามสถาปัตยกรรมของเอสดีเอ็น ซึ่งระบบที่พัฒนาขึ้นจะเน้นไปที่การพัฒนาในชั้น Control Layer เพื่อให้รองรับกับการพัฒนาโปรแกรมประยุกต์สำหรับการบริหารจัดการระบบเครือข่ายโดยที่ไม่ต้องพัฒนาให้โปรแกรมประยุกต์ไปติดต่อสั่งการอุปกรณ์เครือข่ายเองและมีการพัฒนาโปรแกรมประยุกต์สำหรับการบริหารจัดการการจราจรภายในระบบเครือข่าย เพื่อเป็นโปรแกรมประยุกต์ต้นแบบสำหรับการบริหารจัดการระบบเครือข่าย ซึ่งในแต่ละระดับชั้นของสถาปัตยกรรมระบบมีรายละเอียดดังนี้

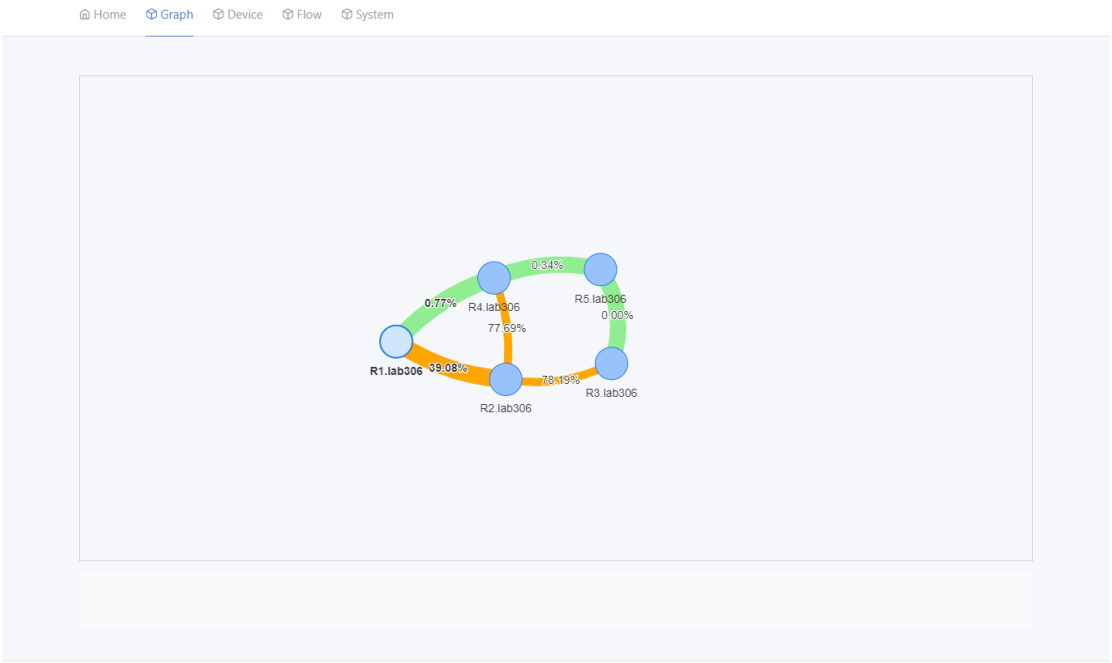
- **Application Layer**: เป็นส่วนของโปรแกรมประยุกต์ที่ทำหน้าที่ในการบริหารจัดการระบบเครือข่ายและเป็นส่วนติดต่อกับผู้ใช้งาน โดยกระบวนการทำงานจะทำการติดต่อกับ

คำสั่ง คือ พิมพ์คำสั่ง help เพื่อเรียกดูคำสั่งที่สามารถใช้งานได้ ซึ่งเป็นความสามารถในลักษณะเดียวกับอุปกรณ์เครือข่ายของ Cisco

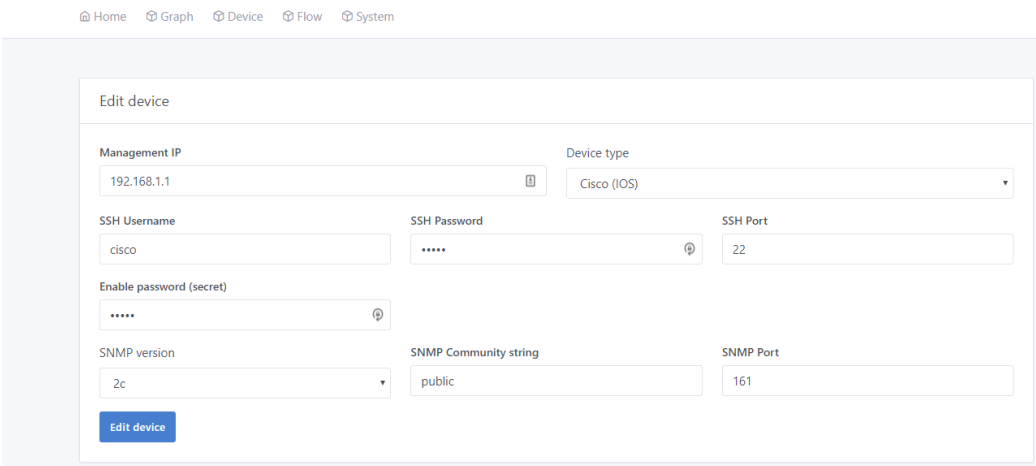


รูปที่ 3.2 แสดงตัวอย่าง Command Line Interface

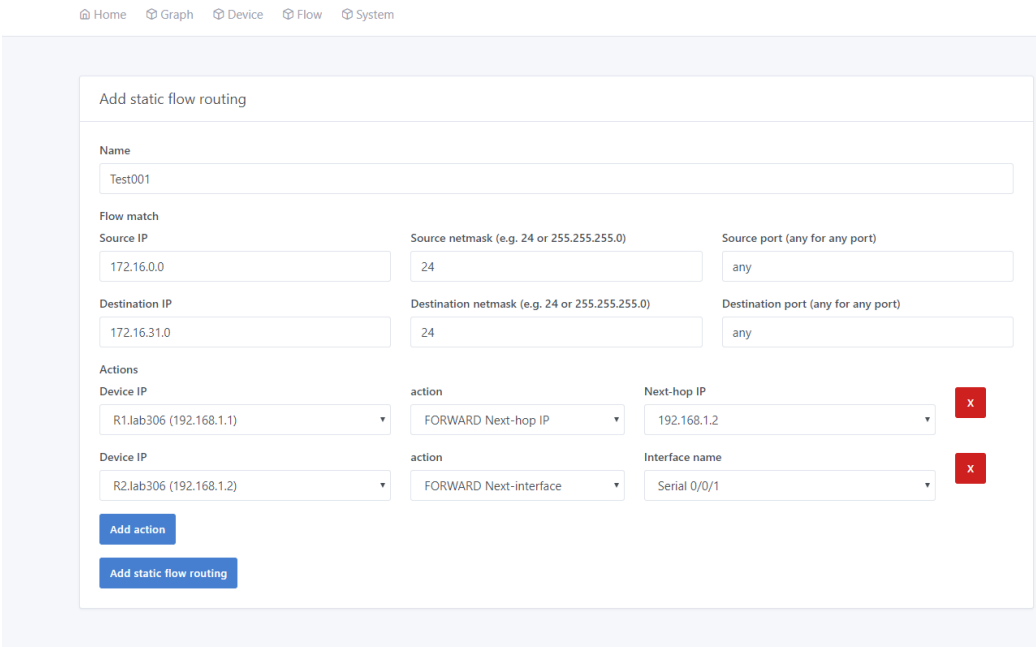
2. Graphical User Interface (GUI) เป็นการใช้งานระบบในรูปแบบของเว็บไซต์ โดยผู้ใช้สามารถใช้งานระบบหรือเรียกดูข้อมูลการทำงานของระบบเครือข่ายผ่านเว็บไซต์ได้เลย อีกทั้งยังสามารถเรียกดูแผนภาพการเชื่อมต่อของระบบเครือข่ายที่แสดงปริมาณการใช้งานของเส้นทางต่าง ๆ ภายในระบบเครือข่ายในรูปแบบของกราฟฟิคแบบเรียลไทม์ได้อีกด้วย



รูปที่ 3.3 แสดงหน้าจอแผนภาพการเชื่อมต่อของระบบเครือข่ายผ่านเว็บไซต์



รูปที่ 3.6 แสดงหน้าจอการแก้ไขข้อมูลอุปกรณ์เครือข่ายผ่านเว็บไซต์



รูปที่ 3.7 แสดงหน้าจอบังคับเปลี่ยนเส้นทางของอุปกรณ์เครือข่ายแบบกำหนดเองผ่านเว็บไซต์

8. หมายเลขพอร์ตของโพรโทคอล SNMP ซึ่งค่าเริ่มต้นคือ 161

ก่อนการใช้งานระบบผู้ใช้งานจำเป็นจะต้องมีการตั้งค่าอุปกรณ์เครือข่ายเบื้องต้น เพื่อให้สามารถทำงานร่วมกับระบบได้ ดังนี้

- เปิดการใช้งานการเข้าถึงจากระยะไกลผ่านโพรโทคอล Secure Shall (SSH)
- เปิดการใช้งานโพรโทคอล SNMP
- เปิดการใช้งาน NetFlow ตั้งค่าให้ส่งออกข้อมูลไปที่ระบบและกำหนดการส่งออกโฟลว์ Active เป็นเวลา 1 นาที โฟลว์ Inactive เป็นเวลา 20 วินาที และกำหนดค่า refresh-rate เป็น 1 โดยกำหนด version ของการส่งออกข้อมูล คือ 9
- กำหนด policy ที่อินเตอร์เฟสที่มีการเชื่อมต่อและใช้งานให้มีการใช้งาน policy ที่ชื่อ SDN-handmade โดยเข้าไปที่หน้าตั้งค่าอินเตอร์เฟสและพิมพ์คำสั่ง “ip policy route-map SDN-handmade”

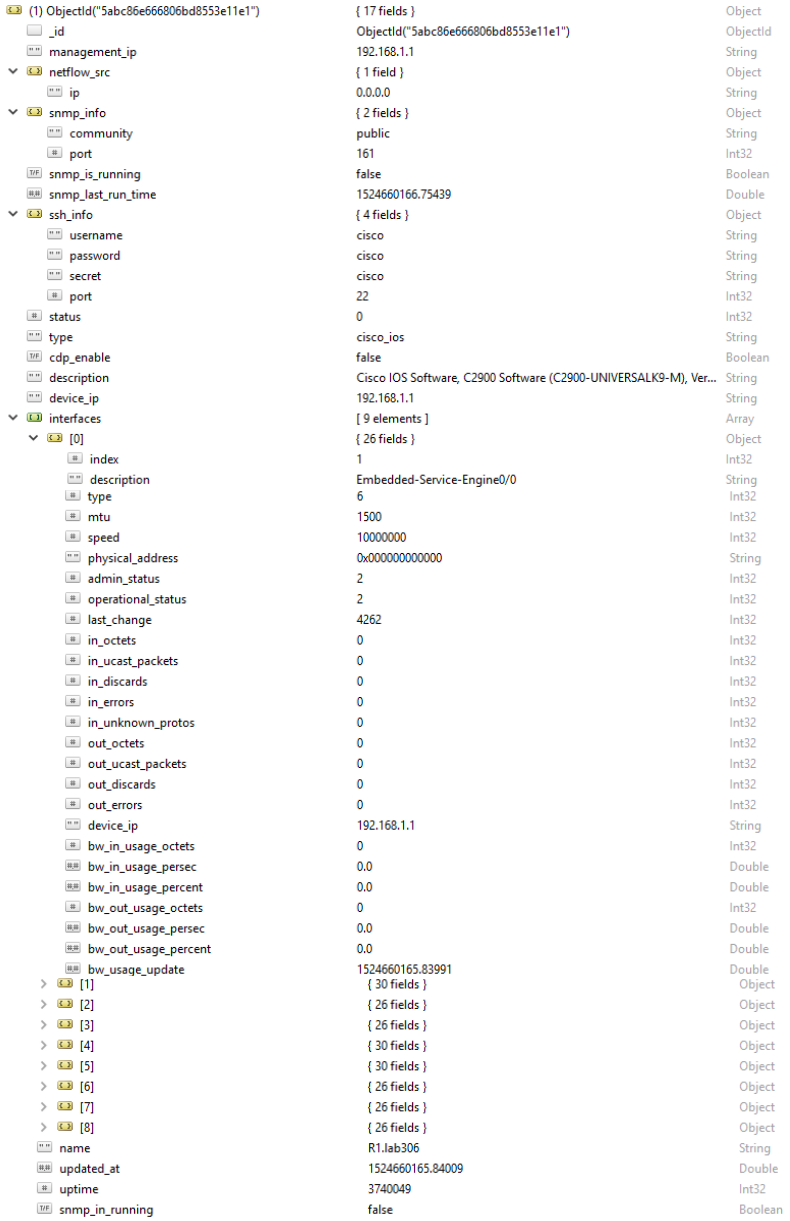
3.2.2 การตรวจสอบและเก็บสถานะของระบบเครือข่าย

ระบบมีกระบวนการในการจัดเก็บข้อมูลของระบบเครือข่าย ซึ่งข้อมูลที่จัดเก็บเข้าสู่ระบบจะอ้างอิงตามการใช้งานของระบบจัดการจราจรเครือข่ายที่มีการใช้งานกับอุปกรณ์เครือข่ายของ Cisco เท่านั้น แต่ในกรณีที่ต้องการจัดเก็บข้อมูลอื่น ๆ หรือจัดเก็บจากอุปกรณ์เครือข่ายประเภทอื่นก็สามารถที่จะกำหนดเพิ่มเติมเข้าสู่ระบบได้ โดยกระบวนการในการจัดเก็บข้อมูลมีอยู่ 2 ลักษณะ ได้แก่

1. ระบบเป็นผู้ร้องขอข้อมูลจากอุปกรณ์เครือข่าย ซึ่งระบบที่พัฒนาขึ้นในปัจจุบันใช้โพรโทคอล SNMP ในการร้องขอข้อมูลจากอุปกรณ์เครือข่าย โดยข้อมูลที่ระบบร้องขอนั้นจะประกอบไปด้วย

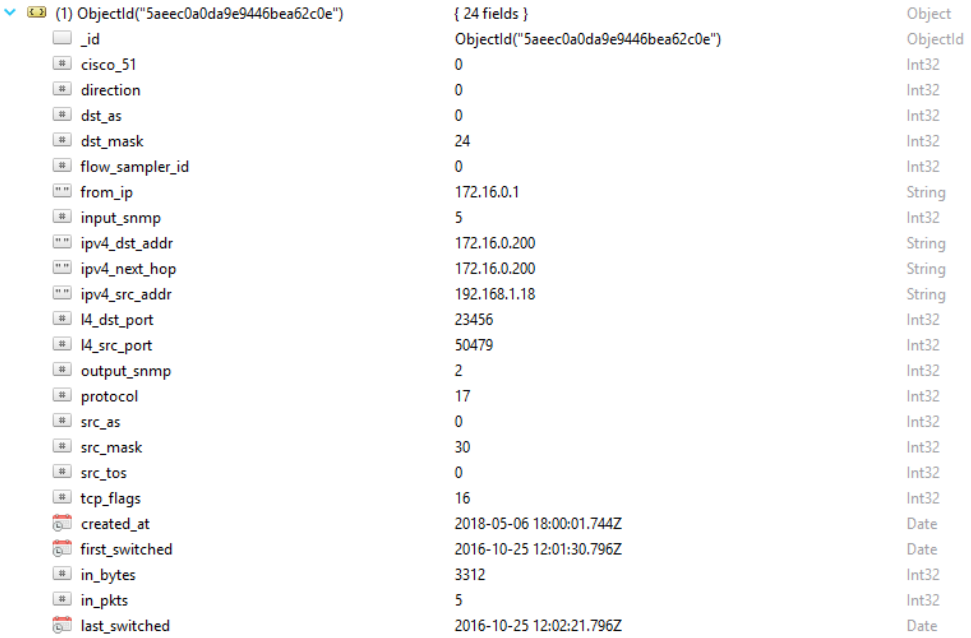
- ข้อมูลพื้นฐานของอุปกรณ์เครือข่าย เช่น ชื่ออุปกรณ์เครือข่าย หมายเลขแอดเดรส เป็นต้น ซึ่งจะจัดเก็บข้อมูลในตาราง Device Table
- ตารางเส้นทางของอุปกรณ์เครือข่าย เพื่อนำมาใช้ในการตรวจสอบเส้นทางการรับส่งข้อมูลของอุปกรณ์เครือข่ายในปัจจุบัน ซึ่งข้อมูลดังกล่าวจะถูกจัดเก็บในตาราง Device Routing Table
- ข้อมูลอินเตอร์เฟสของอุปกรณ์เครือข่าย ซึ่งบอกถึงข้อมูลต่าง ๆ ของอินเตอร์เฟส เช่น ชื่ออินเตอร์เฟส หมายเลขแอดเดรสของอินเตอร์เฟส ปริมาณการใช้งานของอินเตอร์เฟส เป็นต้น โดยที่ข้อมูลในส่วนนี้นั้น จะถูกนำมาใช้ในกระบวนการตรวจสอบความคับคั่งของเส้นทางภายในระบบเครือข่าย ซึ่งจะถูกจัดเก็บในตาราง Device Table

- Device Table ซึ่งจะจัดเก็บข้อมูลเกี่ยวกับอุปกรณ์เครือข่าย เช่น หมายเลขแอดเดรสของอุปกรณ์ ชื่อ ข้อมูลการเข้าถึงจากระยะไกล ข้อมูลอินเตอร์เฟส เป็นต้น โดยที่ข้อมูลภายในตารางจะมาจากการที่ผู้ใช้งานเพิ่มข้อมูลอุปกรณ์เครือข่ายเข้าสู่ระบบและระบบได้รับจากการร้องขอข้อมูลจากอุปกรณ์เครือข่ายด้วยโพรโทคอล SNMP



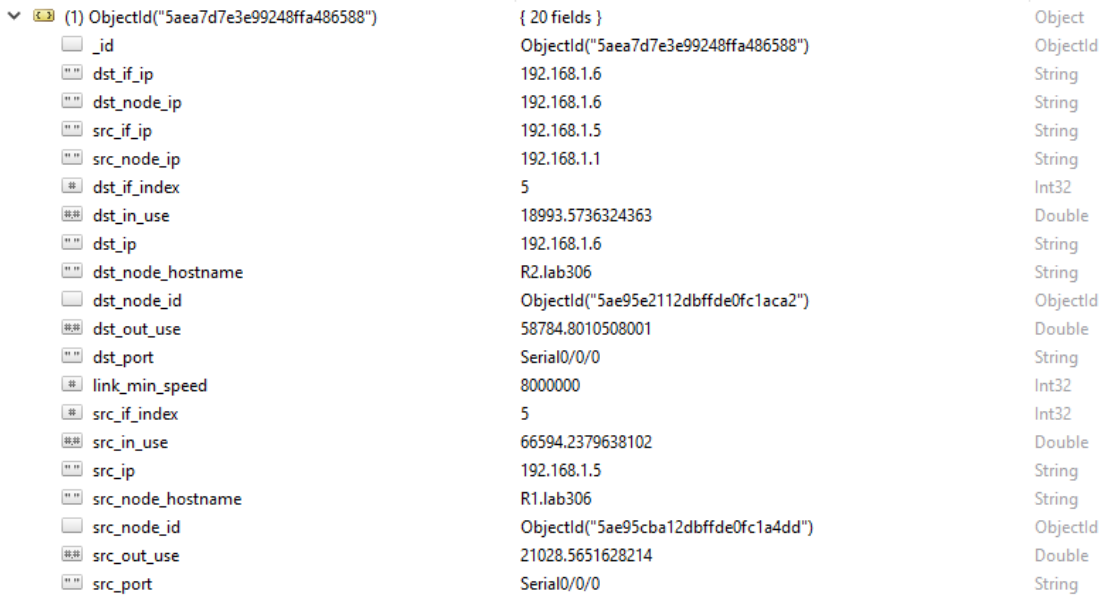
รูปที่ 3.10 แสดงตัวอย่างข้อมูลภายในตาราง Device Table

- Flow Stat Table เป็นตารางที่จัดเก็บข้อมูลที่ได้รับจาก NetFlow ซึ่งเป็นข้อมูลรายละเอียดเกี่ยวกับโฟลว์ข้อมูลที่มีการรับส่งผ่านอุปกรณ์เครือข่ายแต่ละเครื่อง เพื่อนำมาใช้ในกระบวนการเลือกโฟลว์ที่จะบังคับเปลี่ยนเส้นทางการรับส่ง โดยข้อมูลจะถูกอัพเดทจากการที่ระบบได้รับข้อมูลของ NetFlow ที่ส่งมายังระบบ



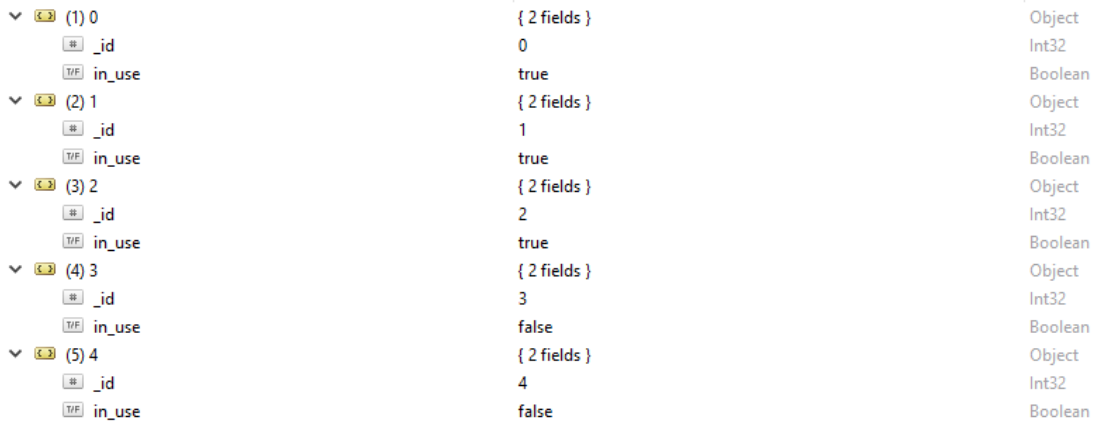
รูปที่ 3.13 แสดงตัวอย่างข้อมูลภายในตาราง Flow Stat Table

- Link Utilization Table: จัดเก็บข้อมูลปริมาณการใช้งานของเส้นทางภายในระบบเครือข่าย เพื่อนำมาใช้การตรวจสอบหาเส้นทางที่มีการใช้งานคับคั่งและเลือกเส้นทางใหม่สำหรับโฟลว์ข้อมูลที่ต้องการบังคับเส้นทางการรับส่ง ซึ่งข้อมูลจะอัพเดทจากการที่ระบบได้รับข้อมูลจากโพรโทคอล SNMP



รูปที่ 3.15 แสดงตัวอย่างข้อมูลภายในตาราง Link Utilization Table

- Used Flow ID List Table: จัดเก็บข้อมูลของหมายเลขโฟลว์ข้อมูลที่มีการใช้งานไปแล้วภายในระบบ ซึ่งจะอัพเดทจากกระบวนการทำงานของระบบ

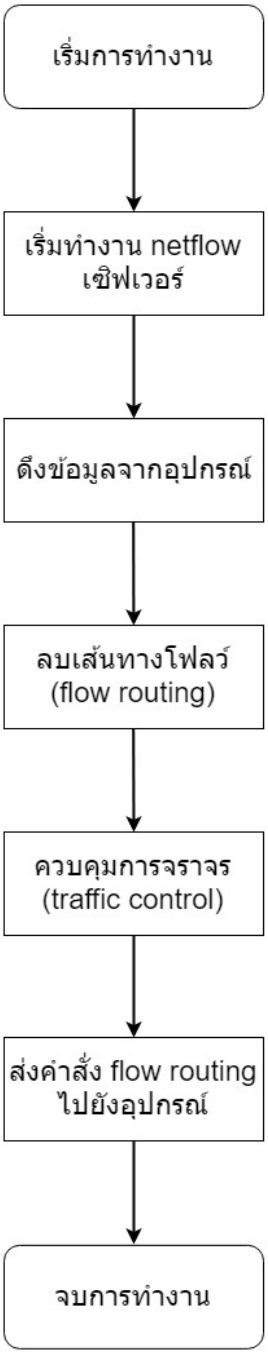


รูปที่ 3.16 แสดงตัวอย่างข้อมูลภายในตาราง Used Flow ID List Table

- ค้นหาเส้นทางที่เป็นไปได้ทั้งหมด จากอุปกรณ์เครือข่ายเครื่องที่เกิดความคับคั่งไปยังปลายทาง โดยเส้นทางดังกล่าวจะไม่นับรวมเส้นทางการรับส่งข้อมูลเดิมของโฟลว์ข้อมูลและเส้นทางที่มีการย้อนกลับไปในทิศทางที่โฟลว์ข้อมูลใช้ในการรับส่งข้อมูล เพื่อป้องกันการวนซ้ำ
- ค้นหาเส้นทางที่มีขนาดแบนด์วิดท์คงเหลือในปัจจุบันเพียงพอต่อการใช้งานของโฟลว์ข้อมูล
- ถ้าพบเส้นทางที่มีขนาดแบนด์วิดท์คงเหลือเพียงต่อการใช้งานของโฟลว์มากกว่า 1 เส้นทาง ระบบจะเลือกเส้นทางแรกที่ระบบค้นพบ
- ถ้าไม่สามารถหาเส้นทางการรับส่งข้อมูลที่มีขนาดแบนด์วิดท์คงเหลือเพียงพอต่อการใช้งานของโฟลว์ข้อมูลได้ ระบบจะตรวจสอบเส้นทางการรับส่งข้อมูลของโฟลว์และเริ่มกระบวนการค้นหาเส้นทางใหม่ โดยค้นหาเส้นทางจากอุปกรณ์เครือข่ายก่อนหน้าที่โฟลว์ข้อมูลใช้ในการรับส่งข้อมูลไปยังปลายทาง
- ถ้าไม่พบเส้นทางที่มีขนาดแบนด์วิดท์คงเหลือเพียงพอต่อการใช้งานของโฟลว์ข้อมูล ระบบจะทำกระบวนการค้นหาเส้นทางโดยย้อนหลังกลับไปหาเส้นทางจากอุปกรณ์เครือข่ายก่อนหน้าที่โฟลว์ข้อมูลใช้ในการรับส่งข้อมูล
- ระบบจะทำกระบวนการค้นหาเส้นทางโดยย้อนกลับไปหาเส้นทางจากอุปกรณ์เครือข่ายก่อนหน้า จนกว่าระบบจะสามารถค้นหาเส้นทางที่มีขนาดแบนด์วิดท์เพียงพอต่อการรับส่งข้อมูลของโฟลว์ได้
- ถ้าระบบค้นหาเส้นทางจนไปถึงอุปกรณ์เครือข่ายแรกที่โฟลว์ข้อมูลใช้ในการรับส่งข้อมูลแล้วไม่พบเส้นทาง ระบบจะเริ่มกระบวนการเลือกโฟลว์ข้อมูลที่จะบังคับเปลี่ยนเส้นทางใหม่โดยเลือกโฟลว์ข้อมูลที่มีขนาดใหญ่รองลงมาแทนและทำกระบวนการค้นหาเส้นทางใหม่อีกครั้ง
- กระบวนการเลือกโฟลว์และค้นหาเส้นทางใหม่จะดำเนินการวนซ้ำไปเรื่อย ๆ จนกว่าจะค้นพบโฟลว์และเส้นทางใหม่สำหรับการบังคับเปลี่ยนเส้นทาง
- โดยการกระบวนการเลือกโฟลว์และเส้นทางสำหรับการบังคับเปลี่ยนเส้นทางนี้ผู้ใช้งานสามารถกำหนดเองได้ โดยการกำหนดลักษณะของโฟลว์ข้อมูลที่ต้องการจะบังคับเปลี่ยนเส้นทาง ได้แก่ หมายเลขแอดเดรสต้นทาง หมายเลขแอดเดรสปลายทาง หมายเลขพอร์ตต้นทาง หมายเลขพอร์ตปลายทาง และอีกส่วนคือ การกำหนดทิศทางการส่งออกโฟลว์ข้อมูลของอุปกรณ์เครือข่าย โดยกำหนดอุปกรณ์เครือข่ายและทิศทางการส่งออกโฟลว์ข้อมูล เช่น อุปกรณ์เครือข่าย R1 ส่งออกโฟลว์ข้อมูลทางอินเตอร์เฟส Serial0/0/0, อุปกรณ์เครือข่าย R2 ส่งออกโฟลว์ข้อมูลทางอินเตอร์เฟส Serial0/0/1 เป็นต้น

3.2.6 กระบวนการทำงานของระบบ

1. กระบวนการทำงานโดยรวมของระบบ

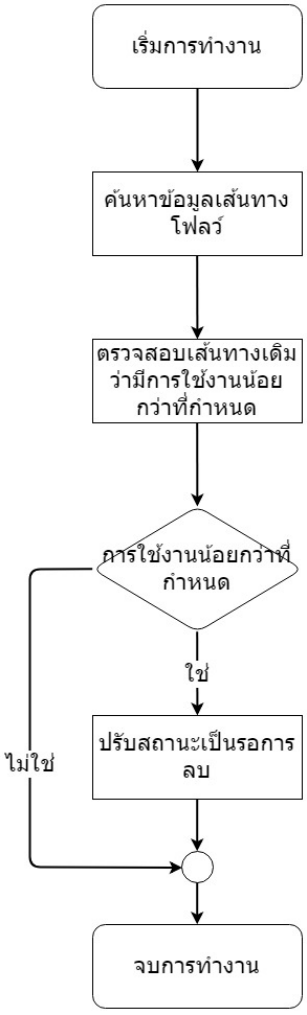


รูปที่ 3.18 แสดงกระบวนการทำงานโดยรวมของระบบ

การทำงานของระบบจะเริ่มจากการเปิดการทำงานในส่วนของ NetFlow Server ภายในระบบเพื่อรอรับข้อมูล NetFlow ที่อุปกรณ์จะส่งมาให้กับระบบ หลังจากนั้นจะทำการดึงข้อมูลต่าง ๆ จากอุปกรณ์เครือข่ายโดยสร้างคำสั่งในการร้องขอข้อมูลจากโพรโทคอล SNMPแล้วส่งไปยังอุปกรณ์ หลังจากนั้นระบบจะทำการตรวจสอบโฟลว์การบังคับเปลี่ยนเส้นทางที่ได้สร้างไว้ก่อน

กรณีดังกล่าวระบบจะทำการตรวจสอบโฟลว์ของการบังคับเปลี่ยนเส้นทางทั้งหมดที่ระบบสร้างไว้จากตาราง Flow Routing Table และตรวจสอบการคงอยู่ของโฟลว์ข้อมูลดังกล่าวจากตาราง Flow Stat Table ซึ่งจัดเก็บข้อมูลของโฟลว์ที่มีการใช้งานอยู่ทั้งหมดภายในเครือข่ายในปัจจุบัน โดยที่ถ้าไม่พบโฟลว์ข้อมูลดังกล่าวจากตารางแสดงว่า โฟลว์ข้อมูลดังกล่าวสิ้นสุดการใช้งานไปแล้ว ซึ่งระบบจะทำการเปลี่ยนสถานะของโฟลว์การบังคับเปลี่ยนเส้นทางดังกล่าวเป็น รอการลบ

กรณีที่ 2 ย้ายกลับไปใช้งานเส้นทางเดิม

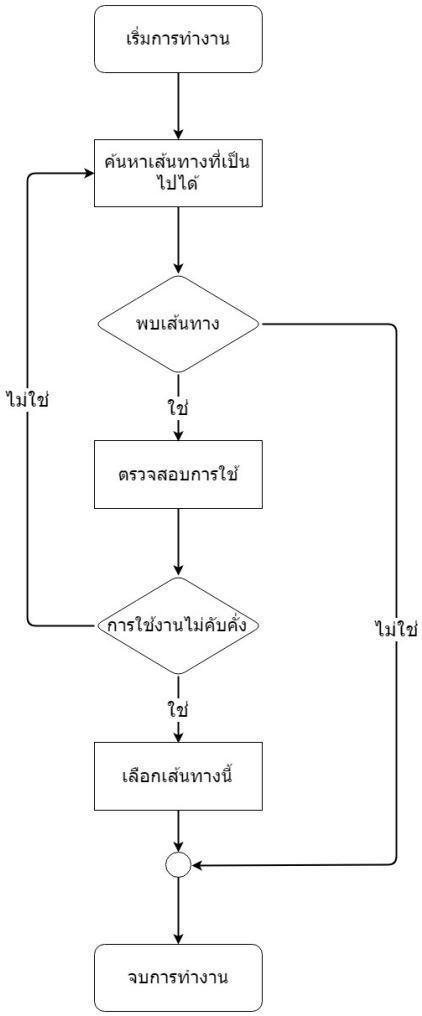


รูปที่ 3.20 แสดงกระบวนการลบโฟลว์การบังคับเปลี่ยนเส้นทางกรณีที่ 2

ระบบจะทำการตรวจสอบโฟลว์การบังคับเปลี่ยนเส้นทางทั้งหมดจากตาราง Flow Routing Table และทำการตรวจสอบปริมาณการใช้งานของเส้นทางเดิมที่โฟลว์ใช้งานก่อนการบังคับเปลี่ยนเส้นทาง ถ้าปริมาณการใช้งานของเส้นทางรวมกับปริมาณการใช้งานของโฟลว์แล้วต่ำกว่าเกณฑ์

เลือกเส้นทางที่สามารถรองรับการใช้งานของโฟลว์ได้ เมื่อเลือกโฟลว์และเส้นทางได้แล้วระบบจะทำการบันทึกค่าดังกล่าวลงใน Flow Routing Table

4. กระบวนการเลือกเส้นทางใหม่



รูปที่ 3.22 แสดงกระบวนการเลือกเส้นทางใหม่

การเลือกเส้นทางใหม่ จะเริ่มจากการค้นหาเส้นทางที่เป็นไปได้ โดยค้นหาจากการสร้างแผนภาพการเชื่อมต่อของทั้งระบบเครือข่ายและหาเส้นทางที่เป็นไปได้จากจุดเริ่มต้นไปยังปลายทาง ซึ่งเส้นทางที่ค้นหาได้จะไม่มีการเรียงลำดับของเส้นทาง หลังจากนั้นระบบจะทำการตรวจสอบว่าสามารถย้ายโฟลว์ไปใช้เส้นทางนั้นได้หรือไม่ จากการตรวจสอบปริมาณการใช้งานของลิงค์ที่มีขนาดเล็กที่สุดของเส้นทางเมื่อรวมกับปริมาณการใช้งานของโฟลว์ข้อมูลที่จะย้ายไปแล้วเกินกว่าเกณฑ์ความคับคั่งที่กำหนดไว้หรือไม่ ถ้าไม่เกินระบบก็จะเลือกเส้นทางดังกล่าว แต่ถ้าเกินระบบก็จะทำการค้นหาเส้นทางใหม่

ดำเนินการใด ๆ แต่ถ้าระบบพบว่าโฟลว์ข้อมูลอยู่ในสถานะรอการรับ ระบบจะดำเนินการในการสร้างคำสั่งเพื่อยกเลิกการบังคับเปลี่ยนเส้นทางของโฟลว์และส่งไปยังอุปกรณ์ หลังจากนั้นจะทำการลบข้อมูลของโฟลว์การบังคับเปลี่ยนเส้นทางดังกล่าวออกจาก Flow Routing Table

3.3.2 แผนภาพ Use-case Description

ตารางที่ 3.1 แสดงคำอธิบายของยูสเคส เพิ่มอุปกรณ์เครือข่าย

Use Case Name:	เพิ่มอุปกรณ์เครือข่าย	
Actor:	ผู้ดูแลระบบ	
Brief Description:	ผู้ใช้ทำการป้อนคำสั่งเพื่อเพิ่มอุปกรณ์เครือข่ายเข้าสู่ระบบ แล้วระบบจะทำการติดต่อขอข้อมูลจากอุปกรณ์เครือข่ายเอง	
Flow of Event:	Actor	System
	1. ผู้ใช้ป้อน ไอพีแอดเดรสของอุปกรณ์เครือข่าย ชื่อผู้ใช้ รหัสผ่าน พอร์ต Secure shell, คอมมิวนิตี้สตริง และ พอร์ต SNMP 2. ผู้ใช้กดปุ่มเอ็นเทอร์	3.ระบบรับค่าที่ผู้ใช้ป้อนแล้วติดต่อขอข้อมูลกับอุปกรณ์เครือข่าย
Pre-Conditions:	เข้าสู่โหมดการตั้งค่าโดยการพิมพ์ config ในหน้าจอ	
Post-Conditions:	ข้อมูลของอุปกรณ์เครือข่ายถูกเพิ่มเข้าสู่ระบบ	

ตารางที่ 3.3 แสดงคำอธิบายของยูสเคส ควบคุมเส้นทาง

Use Case Name:	ควบคุมเส้นทาง	
Actor:	ผู้ดูแลระบบ	
Brief Description:	ผู้ใช้ทำการป้อนคำสั่งเพื่อควบคุมเส้นทางของแต่ละโฟลว์ในเครือข่าย และระบบจะส่งคำสั่งไปแก้ไขเส้นทางยังอุปกรณ์เครือข่ายแต่ละตัวตามที่ผู้ใช้งานตั้งค่าไว้	
Flow of Event:	Actor	System
	1. ผู้ใช้ป้อนข้อมูลเส้นทางที่ต้องการจะควบคุม และใส่ข้อมูลการเปลี่ยนเส้นทางของแต่ละอุปกรณ์เครือข่าย 2.ผู้ใช้ส่งคำสั่ง “apply” เพื่อยืนยันการแก้ไขเส้นทาง	3.ระบบรับค่าที่ผู้ใช้ป้อน และสั่งการแก้ไขเส้นทางไปยังอุปกรณ์เครือข่ายต่างๆ
Pre-Conditions:	เข้าสู่โหมดการตั้งค่าโดยการพิมพ์ config ในหน้าจอ	
Post-Conditions:	เส้นทางถูกเปลี่ยนไปตามที่ผู้ใช้กำหนด	

ตารางที่ 3.5 แสดงคำอธิบายของยูสเคส ดูข้อมูลอินเตอร์เฟสของอุปกรณ์เครือข่าย

Use Case Name:	ดูข้อมูลอินเตอร์เฟสของอุปกรณ์เครือข่าย	
Actor:	ผู้ดูแลระบบ	
Brief Description:	ผู้ใช้ทำการป้อนคำสั่งเพื่อดูข้อมูลอินเตอร์เฟสของอุปกรณ์เครือข่ายในระบบ	
Flow of Event:	Actor	System
	1. ผู้ใช้ป้อน ไอพีแอดเดรสของอุปกรณ์เครือข่าย 2.ผู้ใช้กดปุ่ม เพื่อยืนยันคำสั่ง	3.ระบบรับค่าที่ผู้ใช้ป้อน และแสดงข้อมูลอินเตอร์ของอุปกรณ์เครือข่ายตามที่ผู้ใช้ระบุ
Pre-Conditions:	เพิ่มข้อมูลอุปกรณ์เครือข่ายเข้าสู่ระบบ	
Post-Conditions:	ระบบแสดงข้อมูลอินเตอร์เฟสของอุปกรณ์เครือข่าย	

บทที่ 4

การทดลอง และการประเมินผล

4.1 การทดลองการใช้งานผ่าน Command Line Interface

4.1.1 การเพิ่มอุปกรณ์

การทดลองในส่วนนี้จะเป็นการเพิ่มอุปกรณ์เครือข่ายเข้าสู่ระบบ เพื่อให้ระบบรู้จักอุปกรณ์เครือข่ายที่มีอยู่ในระบบเครือข่ายนั้น ๆ โดยมีขั้นตอนในการเพิ่มอุปกรณ์เครือข่ายเข้าสู่ระบบ ดังนี้

1. ทำการเข้าสู่โหมด config โดยพิมพ์ว่า “config”
2. พิมพ์คำสั่ง “add device”
3. ใส่ประเภทอุปกรณ์เครือข่ายที่ต้องการเพิ่มเข้าสู่ระบบ ซึ่งปัจจุบันระบบรองรับเพียง cisco_ios ซึ่งเป็นอุปกรณ์เครือข่ายของบริษัทซิสโก้ที่รันระบบปฏิบัติการที่มีชื่อว่า IOS
4. ใส่ชื่อผู้ใช้สำหรับการเข้าถึงอุปกรณ์ระยะไกลผ่านโพรโทคอล Secure shell
5. ใส่รหัสผ่านสำหรับการเข้าถึงอุปกรณ์ระยะไกลผ่านโพรโทคอล Secure shell
6. ใส่รหัสการเปิดใช้งานของอุปกรณ์เครือข่าย
7. ใส่หมายเลขพอร์ตของโพรโทคอล Secure shell ซึ่งค่าเริ่มต้นคือ 22
8. ใส่เวอร์ชั่นของโพรโทคอล SNMP ซึ่งปัจจุบันรองรับเพียงเวอร์ชั่น 2c
9. ใส่ค่า SNMP Community string ซึ่งค่าเริ่มต้นคือ public
10. ใส่หมายเลขพอร์ตของโพรโทคอล SNMP ซึ่งค่าเริ่มต้นคือ 161

```
Welcome to SDN Handmade. Type help to list commands.

SDN Handmade (0.0.1 Beta)# config
Enter to config mode
SDN Handmade (0.0.1 Beta)(config)# add device
Add device to topology
If you want to cancel this task. Please enter `exit`
Device type: cisco_ios
Device management ip(v4): 1.1.1.1
SSH username: cisco
SSH password: test
SSH secret (enable password): 1234
SSH Port [22]:
SNMP Version [2c]:
SNMP Community string [public]:
SNMP Port [161]:
Added device to topology
```

รูปที่ 4.1 หน้าจอแสดงการเพิ่มอุปกรณ์ผ่านทาง Command Line Interface

4.1.5 การแสดงข้อมูลเส้นทางของอุปกรณ์เครือข่าย

ทดลองโดยการพิมพ์คำสั่ง “show device <หมายเลขแอดเดรสของอุปกรณ์> route”

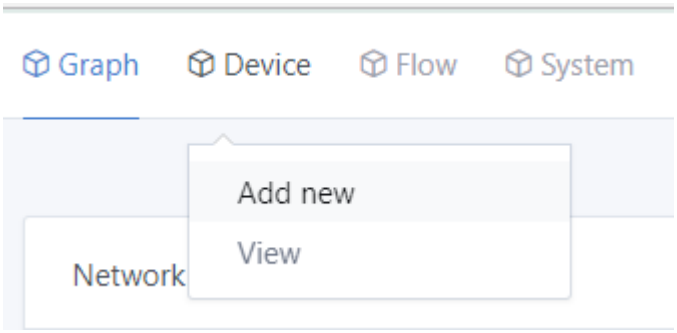
```
SDN Handmade (0.0.1 Beta)# show device 192.168.1.1 route
Protocol        Destination                 Next-hop
[local        ] 172.16.0.0/24           -> 0.0.0.0
[local        ] 172.16.0.1/32           -> 0.0.0.0
[rip          ] 172.16.10.0/24          -> 192.168.1.2
[local        ] 172.16.20.0/24          -> 0.0.0.0
[local        ] 172.16.20.1/32          -> 0.0.0.0
[rip          ] 172.16.30.0/24          -> 192.168.1.6
[rip          ] 172.16.31.0/24          -> 192.168.1.6
[local        ] 192.168.1.0/30          -> 0.0.0.0
[local        ] 192.168.1.1/32          -> 0.0.0.0
[local        ] 192.168.1.4/30          -> 0.0.0.0
[local        ] 192.168.1.5/32          -> 0.0.0.0
[rip          ] 192.168.1.8/30          -> 192.168.1.2
[rip          ] 192.168.1.8/30          -> 192.168.1.6
[rip          ] 192.168.1.12/30         -> 192.168.1.2
[rip          ] 192.168.1.16/30         -> 192.168.1.6
[rip          ] 192.168.1.20/30         -> 192.168.1.2
[rip          ] 192.168.1.20/30         -> 192.168.1.6
SDN Handmade (0.0.1 Beta)#
```

รูปที่ 4.5 หน้าจอแสดงข้อมูลเส้นทางของอุปกรณ์

4.2 การทดลองการใช้งานระบบผ่าน Web UI

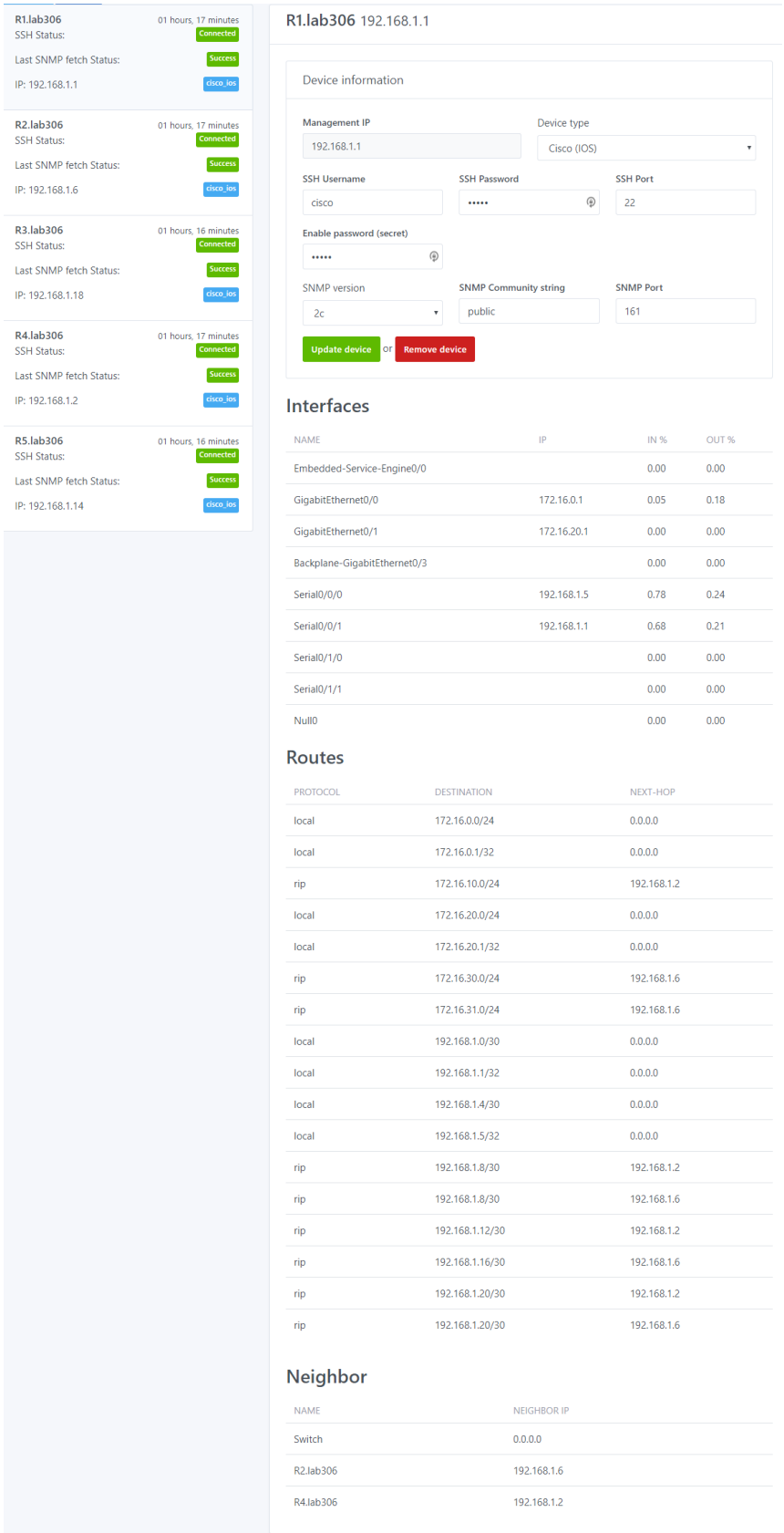
4.2.1 การเพิ่มอุปกรณ์

เลือก Device > Add new



รูปที่ 4.6 เมนูการเพิ่มอุปกรณ์

หลังจากนั้นให้ใส่ข้อมูลของอุปกรณ์ แล้วกด Add device



รูปที่ 4.9 หน้าจอแสดงข้อมูลอุปกรณ์

4.2.4 แสดงข้อมูลการใช้งานของโฟลว์

คลิกที่เมนู Flow > Flow stat

Flow stat

#	FROM	SOURCE	DESTINATION	IN_PKTS	IN_BYTES	FIRST SWITCHED	LAST SWITCHED
1	172.16.0.1	192.168.1.2:61140	172.16.0.200:23456	7	9456	20/05/2018 07:56:09	20/05/2018 07:57:09
2	172.16.0.1	192.168.1.6:61140	172.16.0.200:23456	6	8064	20/05/2018 07:56:31	20/05/2018 07:57:22
3	192.168.1.2	0.0.0.0:68	255.255.255.255:67	10	6040	20/05/2018 07:56:10	20/05/2018 07:57:05
4	172.16.0.1	0.0.0.0:68	255.255.255.255:67	8	4832	20/05/2018 07:56:13	20/05/2018 07:57:10
5	172.16.0.1	192.168.1.14:55595	172.16.0.200:23456	5	3504	20/05/2018 07:57:00	20/05/2018 07:57:54
6	192.168.1.2	192.168.1.14:55595	172.16.0.200:23456	5	3504	20/05/2018 07:57:00	20/05/2018 07:57:54
7	172.16.0.1	192.168.1.18:51687	172.16.0.200:23456	5	3312	20/05/2018 07:56:08	20/05/2018 07:57:02
8	192.168.1.6	192.168.1.18:51687	172.16.0.200:23456	5	3312	20/05/2018 07:56:08	20/05/2018 07:57:02
9	192.168.1.2	172.16.0.200:54625	192.168.1.14:161	1	101	20/05/2018 07:57:30	20/05/2018 07:57:30

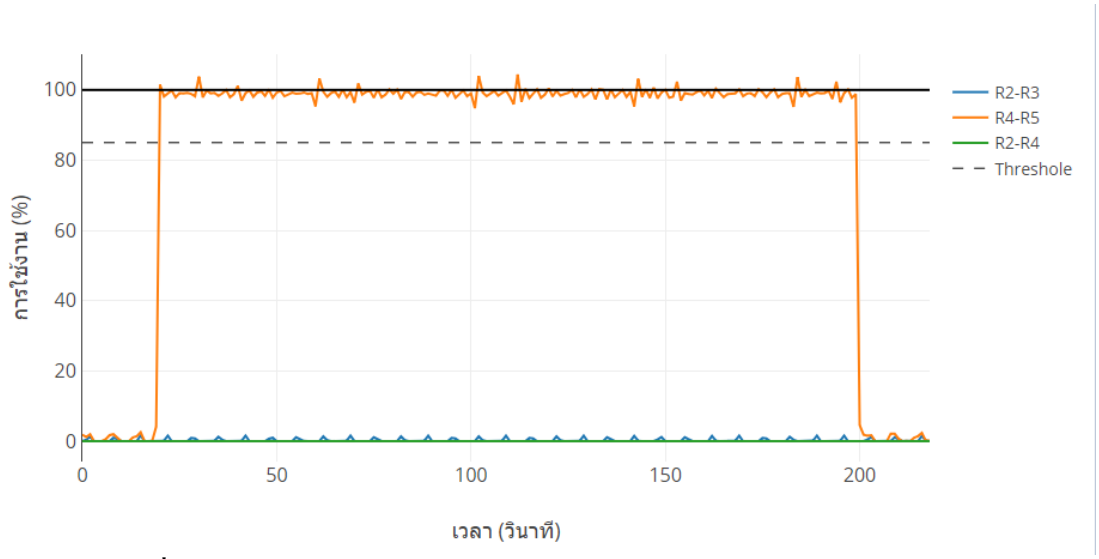
รูปที่ 4.12 หน้าแสดงข้อมูลการใช้งานของโฟลว์

4.3 การทดลองระบบเปลี่ยนเส้นทางแบบอัตโนมัติ

โดยการจำลองระบบเครือข่ายและสร้างสถานการณ์การส่งข้อมูลภายในระบบเครือข่ายในปริมาณมาก ๆ จนทำให้เกิดความคับคั่งขึ้นภายในเส้นทางหนึ่ง ๆ จากนั้นเปิดใช้งานระบบเพื่อทดสอบกระบวนการทำงานและผลจากการทำงานของระบบ โดยจะมีการตรวจสอบค่าปริมาณการใช้งานของลิงค์ต่าง ๆ เพื่อเปรียบเทียบความแตกต่างระหว่างก่อนและหลังการใช้งานระบบ โดยการทดสอบจะแบ่งเป็น 2 รอบดังนี้

4.3.1 การทดลองรูปแบบที่ 1

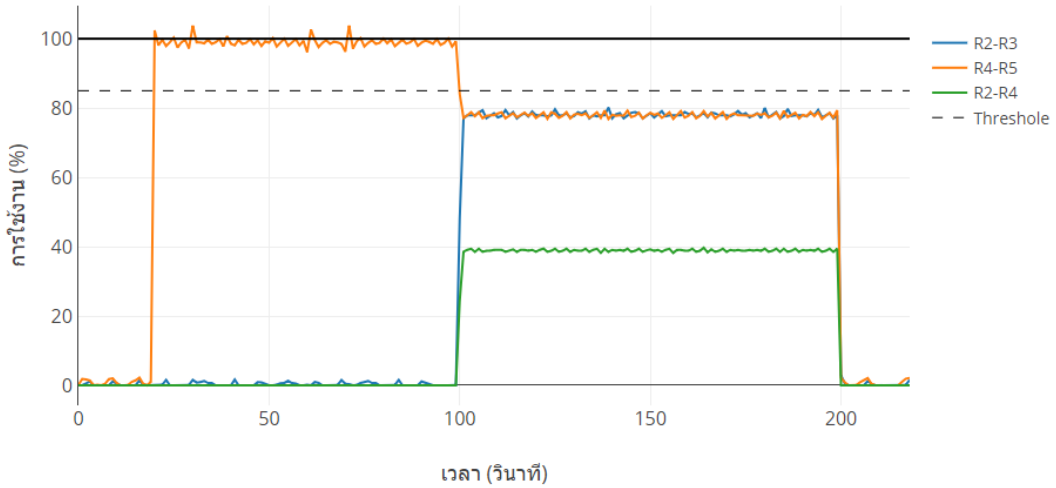
ในการทดลองนี้ จะใช้ PC ทั้งหมด 4 PC โดยมี PC1, PC2, PC3 และ PC6 และอุปกรณ์เครือข่ายจำนวน 5 ตัวโดยมี R1, R2, R3, R4 และ R5 ซึ่งระบบถูกติดตั้งอยู่ที่ Controller ที่เชื่อมต่ออยู่กับ R1 โดยลักษณะการเชื่อมต่อจะเป็นดังรูปที่ 4.13



รูปที่ 4.15 กราฟแสดงการใช้งานของลิงค์ระหว่างเราส์เตอร์ก่อนใช้งานระบบ

จะเห็นว่ามีการใช้เส้นทางที่ไปยัง R6 ผ่านเส้นทาง R1, R4, R5, R3 สำหรับ PC1 และ PC2 ส่วน PC 3 จะใช้เส้นทาง R4, R5 และ R5 จึงทำให้มีการใช้งานที่สูงในลิงค์ระหว่าง R4 และ R5

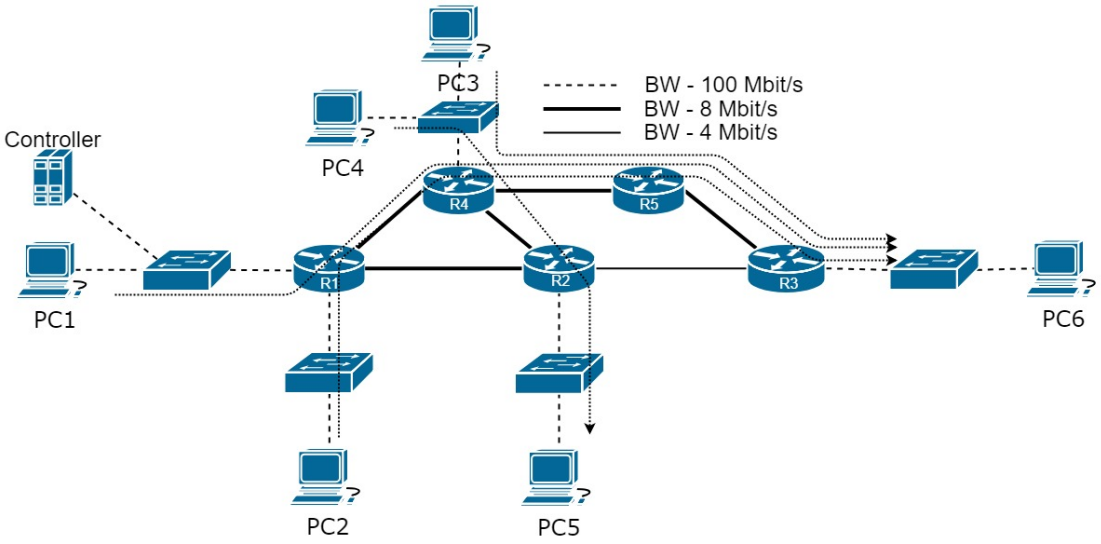
หลังจากนั้นทำการเปิดใช้งานโปรแกรมและทำการทดลองแบบเดิมอีกรอบ โดยผลที่ได้หลังจากการเปิดใช้งานโปรแกรมได้ดังนี้



รูปที่ 4.16 กราฟแสดงการใช้งานของลิงค์ระหว่างเราส์เตอร์หลังใช้งานระบบ

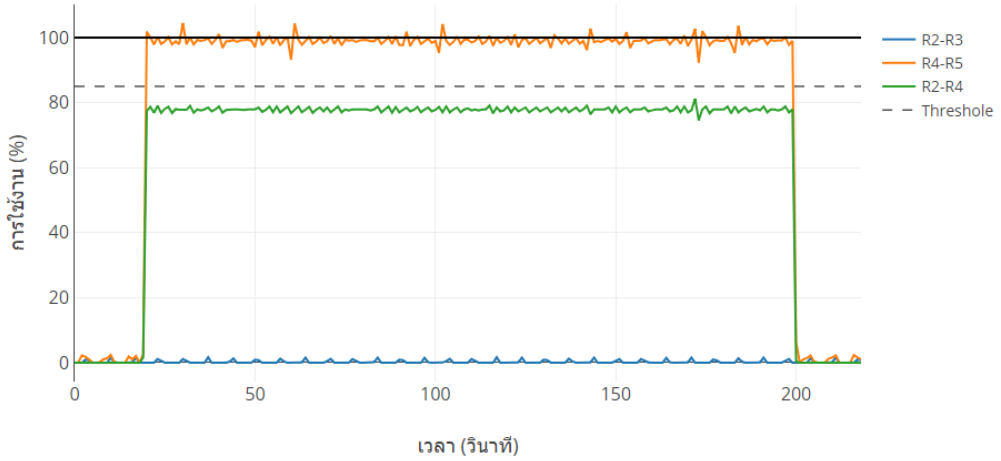
จากกราฟจะเห็นว่าช่วงเวลาวินาทีที่ 100 ระบบมีการย้ายเส้นทางจากบางส่วนให้ไปใช้งานเส้นทาง R4, R2 และ R3 เพื่อลดความคับคั่งของเส้นทาง R4, R5 และ R3 ส่งผลใช้เปอร์เซ็นต์การใช้งานมีการลดลง โดยเส้นทางจากการที่ระบบได้ทำการย้ายระหว่างเกิดความคับคั่งเป็นดังนี้

ในการทดลองจะมีการให้ PC1, PC2 และ PC3 ทำการส่งข้อมูลไปยัง PC6 พร้อมกันเป็นเวลา 180 วินาที โดยมีการกำหนดปริมาณการส่งข้อมูลไว้ที่เครื่องละ 3 Mbit/s และให้ PC4 ส่งข้อมูลไปยัง PC5 โดยมีการกำหนดปริมาณการส่งข้อมูลไว้ที่ 6 Mbit/s โดยเส้นทางการส่งข้อมูลก่อนใช้งานโปรแกรมจะเป็นได้ดังรูปที่ 4.19



รูปที่ 4.19 เส้นทางการส่งข้อมูลของการทดลองที่ 1

โดยมีการวัดผลการใช้งานลิงค์ที่เชื่อมต่อกันระหว่างเราสเตอร์ ที่เป็น bottleneck link ของเส้นทางที่ใช้ส่งข้อมูลไปยัง PC6 ด้วยได้ดังนี้



รูปที่ 4.20 กราฟแสดงการใช้งานของลิงค์ระหว่างเราส์เตอร์ก่อนใช้งานระบบ

จะเห็นว่ามีการใช้เส้นทางที่ไปยัง R6 ผ่านเส้นทาง R1, R4, R5, R3 สำหรับ PC1 และ PC2 ส่วน PC 3 จะใช้เส้นทาง R4, R5 และ R5 จึงทำให้มีการใช้งานที่สูงในลิงค์ระหว่าง R4 และ R5 และ

จากรูปเส้นทางจะเห็นว่าระบบได้มีการย้ายเส้นทางจาก PC1 ไปยัง PC6 จากเดิมที่ใช้งานเส้นทาง R1, R4, R5 และ R3 ไปใช้เส้นทาง R1, R2 และ R3 โดยที่การคับคั่งของเส้นทางได้เกิดขึ้นที่ลิงค์ระหว่าง R4 และ R5 โดยที่เลือกเส้นทางนี้เนื่องจากเส้นทาง R4, R2 และ R3 มีการใช้งานอยู่ที่ค่อนข้างสูง ที่ลิงค์ระหว่าง R2 และ R4 ไม่เพียงพอต่อการที่จะย้ายไปได้

5.4 ข้อเสนอแนะและแนวทางในการพัฒนาระบบในอนาคต

ในอนาคตจะมีการปรับปรุงระบบให้สามารถสั่งงานผ่านทางเว็บเบราวเซอร์แบบวิชวลได้สมบูรณ์แบบมากกว่านี้ ซึ่งจะทำให้การใช้งานระบบมีความง่ายมากขึ้น สามารถวิเคราะห์ปริมาณการใช้งานของโฟลว์ที่เป็นโฟลว์เล็กๆได้ เพื่อสามารถเลือกโฟลว์ที่ดีกว่าเดิมในการย้ายได้

บรรณานุกรม (ต่อ)

[11] Cisco. (2016). **Cisco Discovery Protocol Version 2**. https://www.cisco.com/c/en/us/td/docs/ios-xml/ios/cdp/configuration/15-mt/cdp-15-mt-book/nm-cdp-discover.html.

[12] Cisco. (2008). **Cisco Discovery Protocol**. https://www.cisco.com/c/en/us/td/docs/routers/access/3200/software/wireless/3200WirelessConfigGuide/CiscoDiscovProto.pdf.

[13] Cisco. **Routing Information Protocol**. https://www.cisco.com/c/en/us/td/docs/switches/lan/catalyst2960x/software/15-2_5_e/configuration_guide/b_1525e_consolidated_2960x_cg/b_1525e_consolidated_2960x_cg_chapter_01011110.pdf.

[14] Cisco. **OSPF Design Guide**. https://www.cisco.com/c/en/us/support/docs/ip/open-shortest-path-first-ospf/7039-1.html#t9.

[15] Cisco. (2014). **Configuring Policy-Based Routing**. https://www.cisco.com/c/en/us/td/docs/ios/12_2/qos/configuration/guide/fqos_c/qcfpbr.pdf.

[16] Cisco. (2005). **Understanding Policy Routing**. https://www.cisco.com/c/en/us/support/docs/ip/border-gateway-protocol-bgp/10116-36.html.

[17] Fabio Semperboni. (2013). **PBR: Route a packet based on source IP address**. http://www.ciscozine.com/pbr-route-a-packet-based-on-source-ip-address/.

[18] Cisco. **Route Maps**. https://www.cisco.com/c/en/us/td/docs/security/asa/asa94/config-guides/cli/general/asa-94-general-config/routes-maps.pdf.

[19] www.ssh.com. **SSH (SECURE SHELL)**. https://www.ssh.com/ssh/

[20] Mohammad Al-Fares, Barath Raghavan, Sivasankar Radhakrishnan, Nelson Huang and Amin Vahdat. **Hedera: Dynamic Flow Scheduling for Data Center Network**. https://www1.icsi.berkeley.edu/~barath/papers/hedera-nsdi10.pdf.

ภาคผนวก ก
วิธีการใช้งานระบบ

4. เข้าไปที่โฟลเดอร์ SDN-handmade/frontend และทำการแก้ไขไฟล์ nuxt.config.js โดยแก้ไขในส่วน baseURL ให้เป็น url ของเครื่องเซิฟเวอร์ที่รันไฟล์ main_web.py ในข้อที่ 3

```
axios: {
  baseURL: 'http://localhost:5001/api/v1/'
  // baseURL: 'http://10.30.6.49:5001/api/v1/'
  // proxyHeaders: false
},
```

รูปที่ ก.4 แสดงการตั้งค่า baseURL

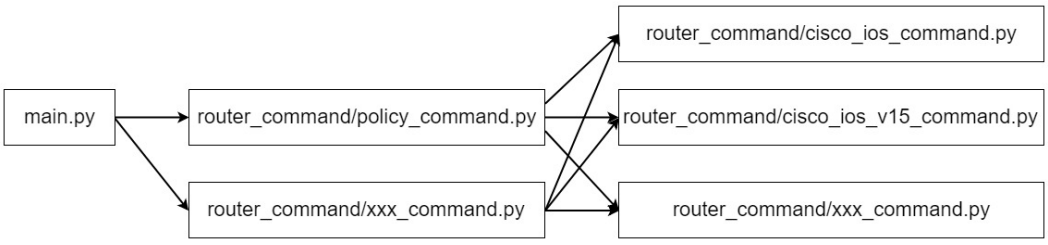
5. ติดตั้งไลบารี่ที่จำเป็นต่อการทำงานของระบบ โดยพิมพ์คำสั่ง “npm install”
6. รันระบบส่วนของ Frontend โดยพิมพ์คำสั่ง “npm run dev”

```
[root@sdn frontend]# npm run dev

> frontend@1.0.0 dev /root/SDN-handmade/frontend
> nuxt

… debug nuxt › axios › baseURL: http://192.168.153.210/api/v1/
… debug nuxt › axios › browserBaseURL: http://192.168.153.210/api/v1/
  nuxt:build App root: /root/SDN-handmade/frontend +0ms
  nuxt:build Generating /root/SDN-handmade/frontend/.nuxt files... +1ms
  nuxt:build Generating files... +8ms
  nuxt:build Generating routes... +9ms
  nuxt:build Building files... +32ms
  nuxt:build Adding webpack middleware... +719ms
  ████████████ 40% building modules
```

รูปที่ ก.5 แสดงการรันระบบส่วนของ Frontend

2. ขั้นตอนการเพิ่มอุปกรณ์ชนิดใหม่เข้าสู่ระบบ

โครงสร้างของไฟล์ที่เกี่ยวข้องกับการเพิ่มอุปกรณ์ชนิดใหม่เข้าสู่ระบบ มีลักษณะดังนี้



รูปที่ ก.6 แสดงโครงสร้างของไฟล์สำหรับการเพิ่มอุปกรณ์ชนิดใหม่เข้าสู่ระบบ

```
def generate_config_command(device_type, flow, flow_id, flow_name, action):
    if device_type == 'cisco_ios':
        return cisco_ios_cmd.generate_cmd(flow, flow_id, flow_name, action)
    elif device_type == 'cisco_ios_v15':
        return cisco_ios_15_cmd.generate_cmd(flow, flow_id, flow_name, action)
```

รูปที่ ก.8 แสดงข้อมูลในฟังก์ชัน generate_config_command

- generate_remove_command เป็นฟังก์ชันที่มีลักษณะคล้ายกับฟังก์ชันก่อนหน้า แต่แตกต่างตรงที่ฟังก์นี้ใช้ตรวจสอบประเภทของอุปกรณ์เครือข่าย เพื่อสร้างคำสั่งสำหรับการยกเลิกการควบคุมการทำงานของอุปกรณ์

```
def generate_remove_command(device_type, flow):
    if device_type == 'cisco_ios':
        return cisco_ios_cmd.generate_remove_command(flow)
    elif device_type == 'cisco_ios_v15':
        return cisco_ios_15_cmd.generate_remove_command(flow)
```

รูปที่ ก.9 แสดงข้อมูลในฟังก์ชัน generate_remove_command

- get_netmiko_type เป็นฟังก์ชันสำหรับตรวจสอบประเภทอุปกรณ์เพื่อให้เรียกฟังก์ชันที่สร้างขึ้นมา เมื่อเพิ่มอุปกรณ์เครือข่ายก็เพิ่มเงื่อนไขในการตรวจสอบประเภทของอุปกรณ์

```
def get_netmiko_type(device_type):
    if device_type == 'cisco_ios':
        return cisco_ios_cmd.get_netmiko_type()
    elif device_type == 'cisco_ios_v15':
        return cisco_ios_15_cmd.get_netmiko_type()
```

รูปที่ ก.10 แสดงข้อมูลในฟังก์ชัน get_netmiko_type

วิธีการใช้งาน REST API

การเรียกใช้ api นั้นสามารถทำได้โดยการสร้าง http request และใส่ url เป็นของเซิฟเวอร์ api ซึ่งจะมีลักษณะดังนี้ http://<server_ip>:5001/api/v1<api> โดยรูปแบบการ request ที่จะอธิบายด้านล่างจะอยู่ในรูป <http_method> <url> เช่น GET /device คือให้ใช้ http method GET ในการเรียกข้อมูลที่ url http:<server_ip>:5001/api/v1/device และใน url ข้อความที่มีตัวอักษร : นำหน้า เช่น :device_id ให้แทนเป็นข้อมูลของค่านั้น ๆ แทน และในส่วนของการ response จะเป็น http_status พร้อมทั้งรูปแบบข้อมูลตัวอย่าง

ตารางที่ ข.1 แสดงรายชื่อของ API ทั้งหมด

	URL	คำอธิบาย
1	GET /device	ขอข้อมูลอุปกรณ์ทั้งหมด
2	POST /device	เพิ่มอุปกรณ์เข้าสู่ระบบ
3	PATCH /device/:device_id	แก้ไขข้อมูลอุปกรณ์
4	DELETE /device?device_id=:device_id	ลบข้อมูลอุปกรณ์
5	GET /device/:device_id	ขอข้อมูลอุปกรณ์ตามไอดีของอุปกรณ์
6	GET /device/:device_id/neighbor	ขอข้อมูลอุปกรณ์ที่เชื่อมต่อกัน
7	GET /link	ขอข้อมูลลิงค์ทั้งหมด
8	GET /link/:link_id	ขอข้อมูลลิงค์ตามลิงค์ไอดี
9	GET /path/:from_device_ip,:to_device_ip	ข้อมูลเส้นทางจากอุปกรณ์เครื่องหนึ่งไปยังอุปกรณ์อีกเครื่องหนึ่ง
10	GET /flow	ขอข้อมูลโฟลว์ทั้งหมด
11	GET /flow/routing	ขอข้อมูลเส้นทางของโฟลว์
12	POST /flow/routing	สร้างเส้นทางของโฟลว์
13	PATCH /flow/routing	แก้ไขข้อมูลเส้นทางของโฟลว์
14	DELETE /flow/routing?flow_id=:flow_id	ลบเส้นทางของโฟลว์
15	GET /flow/routing/:flow_id	ขอเส้นทางของโฟลว์
16	GET /routes/device_id	ขอข้อมูลตารางเส้นทางของอุปกรณ์

1 การขอข้อมูลอุปกรณ์ทั้งหมด

GET /device

Response: Status 200 OK

ตารางที่ ข.2 (ต่อ) ตัวอย่างข้อมูลที่คืนค่าข้อมูลอุปกรณ์ทั้งหมด


```
    {
     "index":1,
     "description":"Embedded-Service-Engine0/0",
     "type":6,
     "mtu":1500,
     "speed":10000000,
     "physical_address":"0x000000000000",
     "admin_status":2,
     "operational_status":2,
     "last_change":4275,
     "in_octets":0,
     "in_ucast_packets":0,
     "in_discards":0,
     "in_errors":0,
     "in_unknown_protos":0,
     "out_octets":0,
     "out_ucast_packets":0,
     "out_discards":0,
     "out_errors":0,
     "device_ip":"192.168.1.1",
     "bw_in_usage_octets":0,
     "bw_in_usage_persec":0.0,
     "bw_in_usage_percent":0.0,
     "bw_out_usage_octets":0,
     "bw_out_usage_persec":0.0,
     "bw_out_usage_percent":0.0,
     "bw_usage_update":1526992492.0793455
    },
   ],
   "interfaces_update_time":1526992491.794834,
```

Response: Status 201 Created

ตารางที่ ข.4 ตัวอย่างข้อมูลที่คืนค่าของการเพิ่มอุปกรณ์


```
{
 "success":true,
 "message":{
  "management_ip":"192.168.1.40",
  "type":"cisco_ios",
  "ssh_info":{
   "username":"cisco",
   "password":"cisco_pass",
   "port":22,
   "secret":"enable_pass"
  },
  "snmp_info":{
   "version":"2c",
   "community":"public",
   "port":161
  }
 }
}
```

4 การลบข้อมูลอุปกรณ์

DELETE /device?device_id=:device_id

Response: Status 204 No Content

5 การขอข้อมูลอุปกรณ์ตามไอดีของอุปกรณ์

GET /device/:device_id

Response: Status 200 OK

ตารางที่ ข.7 ตัวอย่างการคืนค่าข้อมูลของการขอข้อมูลอุปกรณ์ตามไอดีของอุปกรณ์


```
{
 "device":{
  "_id":{
   "$oid":"5af97cbf02b9c80969284ca6"
  },
  "management_ip":"192.168.1.1",
  "device_ip":"192.168.1.1",
  "snmp_info":{
   "version":"2c",
   "community":"public",
   "port":161
  },
  "snmp_is_running":false,
  "snmp_last_run_time":1526992492.2371502,
  "ssh_info":{
   "username":"cisco",
   "password":"cisco",
   "port":22,
   "secret":"cisco"
  },
  "status":1,
  "type":"cisco_ios",
  "is_ssh_connect":true,
```

ตารางที่ ข.7 (ต่อ) ตัวอย่างการคืนค่าข้อมูลของการขอข้อมูลอุปกรณ์ตามไอดีของอุปกรณ์


```
      "bw_out_usage_persec":0.0,
      "bw_out_usage_percent":0.0,
      "bw_usage_update":1526992492.0793455
    },
  ],
  "interfaces_update_time":1526992491.794834,
  "name":"R1.lab306",
  "updated_at":1526992492.0795121,
  "uptime":4448648,
  "is_snmp_connect":true
 },
 "success":true
}
```


6 การขอข้อมูลอุปกรณ์ที่เชื่อมต่อกัน

GET /device/:device_id/neighbor

Response: Status 200 OK

ตารางที่ ข.8 ตัวอย่างข้อมูลที่คืนค่าของการขอข้อมูลอุปกรณ์ที่เชื่อมต่อกัน


```
{
 "neighbor":[
  {
   "ip_addr":"0.0.0.0",
   "version":"Cisco Internetwork Operating System Software \nIOS (tm) C2950 Software (C2950-I6Q4L2-M), Version 12.1(22)EA8a, RELEASE SOFTWARE (fc1)\nCopyright (c) 1986-2006 by cisco Systems, Inc.\nCompiled Fri 28-Jul-06 15:16 by weiliu",
   "name":"Switch",
   "port":"FastEthernet0/24",
   "local_ifindex":2,
   "device_ip":"192.168.1.1"
  },
```

7 การขอข้อมูลลิงค์ทั้งหมด

GET /link

Response: Status 200 OK

ตารางที่ ข.9 ตัวอย่างข้อมูลที่คืนค่าของการขอข้อมูลลิงค์ทั้งหมด


```
{
 "links":[
   {
    "_id":{
     "$oid":"5af97e6c02b9c80969285511"
    },
    "dst_if_ip":"192.168.1.2",
    "dst_node_ip":"192.168.1.2",
    "src_if_ip":"192.168.1.1",
    "src_node_ip":"192.168.1.1",
    "dst_if_index":6,
    "dst_in_use":18751.97607585901,
    "dst_ip":"192.168.1.2",
    "dst_node_hostname":"R4.lab306",
    "dst_node_id":{
     "$oid":"5af97e4302b9c809692853c0"
    },
 ],
 "status":"ok"
}
```

ตารางที่ ข.10 (ต่อ) ตัวอย่างข้อมูลที่คืนค่าของการขอข้อมูลลิงค์ตามไอดี


```
   "src_port":"Serial0/0/1"
  },
  "status":"ok"
}
```


9 การข้อมูลเส้นทางจากอุปกรณ์เครื่องหนึ่งไปยังอุปกรณ์อีกเครื่องหนึ่ง

GET /path/:from_device_ip,:to_device_ip

Response: Status 200 OK

ตารางที่ ข.11 ตัวอย่างข้อมูลที่คืนค่าของการขอข้อมูลเส้นจากอุปกรณ์หนึ่งไปยังอีกอุปกรณ์หนึ่ง


```
{
  "paths":[
    [
      "192.168.1.1",
      "192.168.1.6"
    ],
  ],
  "status":"ok"
}
```


10 การขอข้อมูลโฟลว์ทั้งหมด

GET /flow

Response: Status 200 OK

ตารางที่ ข.12 ตัวอย่างข้อมูลที่คืนค่าของการขอข้อมูลโฟลว์ทั้งหมด


```
{
  "flows":[
    {
      "_id":{
        "$oid":"5b037c08eb6bb6611acaf91f"
      },
      "cisco_51":0,
```

ตารางที่ ข.12 (ต่อ) ตัวอย่างข้อมูลที่คืนค่าของการขอข้อมูลโฟลว์ทั้งหมด


```
  "status":"ok"
}
```


11 การขอข้อมูลเส้นทางของโฟลว์

GET /flow/routing

Response: Status 200 OK

ตารางที่ ข.13 ตัวอย่างข้อมูลที่คืนค่าของการขอข้อมูลเส้นทางของโฟลว์


```
{
 "flows":[
  {
   "_id":{
    "$oid":"5b0635c669324570da9f63ab"
   },
   "new_flow":{
    "name":"test-flow-routing",
    "src_ip":"10.0.0.0",
    "src_port":"any",
    "src_wildcard":"0.0.0.255",
    "dst_ip":"172.16.99.1",
    "dst_port":"5201",
    "dst_wildcard":"0.0.0.0",
    "actions":[
     {
      "device_id":{
       "$oid":"5af97cbf02b9c80969284ca6"
      },
      "action":2,
      "data":"192.168.1.6"
     },
     {
```

ตารางที่ ข.13 (ต่อ) ตัวอย่างข้อมูลที่คืนค่าของการขอข้อมูลเส้นทางของโฟลว์


```
 ],
 "status":"ok"
}
```


12 การสร้างเส้นทางของโฟลว์

POST /flow/routing

ข้อมูลที่ต้องส่งไป

ตารางที่ ข.14 ตัวอย่างข้อมูลที่ใช้ในการส่งเพื่อสร้างเส้นทางของโฟลว์


```
{
 "name":"test-flow-routing",
 "src_ip":"10.0.0.0",
 "src_subnet":"0.0.0.255",
 "src_port":"any",
 "dst_ip":"172.16.99.1",
 "dst_subnet":"0.0.0.0",
 "dst_port":"5201",
 "actions":[
  {
   "device_id":"5af97cbf02b9c80969284ca6",
   "action":2,
   "data":"192.168.1.6"
  },
  {
   "device_id":"5af97e4302b9c809692853c0",
   "action":3,
   "data":"GigabitEthernet0/1"
  },  {
   "device_id":"5af97e7202b9c8096928555e",
   "action":"1",
   "data":""
```

Response: Status 200 OK

ตารางที่ ข.17 ตัวอย่างข้อมูลที่คืนค่าของการแก้ไขข้อมูลเส้นทางของโฟลว์


```
{
 "status":true,
 "message":"Update flow routing!"
}
```


14 การลบเส้นทางของโฟลว์

DELETE /flow/routing?flow_id=:flow_id

Response: Status 204 No Content

15 การขอเส้นทางของโฟลว์

GET /flow/routing/:flow_id

ตารางที่ ข.18 ตัวอย่างข้อมูลที่คืนค่าของการขอข้อมูลเส้นทางของโฟลว์


```
{
 "flow":{
  "_id":{
   "$oid":"5b0635c669324570da9f63ab"
  },
  "new_flow":{
   "name":"test-flow-routing",
   "src_ip":"10.0.0.0",
   "src_port":"any",
   "src_wildcard":"0.0.0.255",
   "dst_ip":"172.16.99.1",
   "dst_port":"5201",
   "dst_wildcard":"0.0.0.0",
   "actions":[
    {
     "device_id":{
      "$oid":"5af97cbf02b9c80969284ca6"
```

ตารางที่ ข.19 (ต่อ) ตัวอย่างข้อมูลที่คืนค่าของการขอข้อมูลตารางเส้นทางของอุปกรณ์


```
        "$oid":"5b040e6c69324522cf090cfe"
      },
      "dst":"172.16.0.0",
      "mask":"255.255.255.0",
      "tos":0,
      "next_hop":"0.0.0.0",
      "if_index":2,
      "type":3,
      "proto":2,
      "age":44447,
      "info":"0.0",
      "next_hop_AS":0,
      "metric1":0,
      "metric2":-1,
      "metric3":-1,
      "metric4":-1,
      "metric5":-1,
      "status":1,
      "start_ip":2886729729,
      "end_ip":2886729982,
      "device_id":{
        "$oid":"5af97cbf02b9c80969284ca6"
      },
      "updated_at":1526992492.2342358
    },
  ],
  "status":"ok"
}
```