การทดลองหาประสิทธิภาพด้านเวลาและการป้องกันการโจมตีของ RYU SDN Framework โดยใช้ Group table และ Proxy arp ในรูปแบบ Single Controller และ Multi Controllers

ณวสันต์ วิศิษฏ์ศิงขร

แขนงวิชาเทคโนโลยีเครือข่ายและความมั่นคงทางไซเบอร์
สาขาวิชาเทคโนโลยีสารสนเทศและการสื่อสาร
คณะวิทยาศาสตร์ มหาวิทยาลัยอุบลราชธานี

สารบัญ

1 ระเบียบวิธี

1.1 วัตถุประสงค์

Software-defined networking (ระบบเครือข่ายที่กำหนดโดยซอฟต์แวร์) นั้นมีการใช้งานและการตั้งค่าที่หลากหลาย ไม่ว่าจะเพื่อให้เหมาะกับ Network system (ระบบเครือข่าย) ขององค์กรหรือหน่วยงานของตน ยังต้องช่วยเพิ่มประสิทธิภาพของระบบเครือข่ายอีกด้วย นอกจากนั้น ระบบเครือข่ายที่กำหนดโดยซอฟต์แวร์ ยังถูกนำมาใช้เพื่อป้องกันการโจมตีผ่านระบบเครือข่าย ไม่ว่าจะเป็น Dos หรือ DDos อย่างไรก็ตามระบบเครือข่ายที่กำหนดโดยซอฟต์แวร์นั้นก็คือ Software (ซอฟต์แวร์) รูปแบบหนึ่ง ดังนั้นก็จะมีซอฟต์แวร์หลายตัวที่ถูกพัฒนาขึ้นเป็น “ระบบเครือข่ายที่กำหนดโดยซอฟต์แวร์” เช่น RYU OpenDaylight NOX POX ฯลฯ ดังนั้น วัตถุประสงค์ของการทดลองครั้งนี้ก็เพื่อหาประสิทธิภาพทั้งในด้านเวลาและการตรวจจับการโจมตีของ RYU SDN framework (RYU) ซึ่งเป็นระบบเครือข่ายที่กำหนดโดยซอฟต์แวร์ประเภทหนึ่ง ที่ถูกรันอยู่ใน SDN Controller พร้อมกับการใช้งาน Group table หรือ Proxy ARP ทั้งในรูปแบบ Single Controller และ Multi Controller

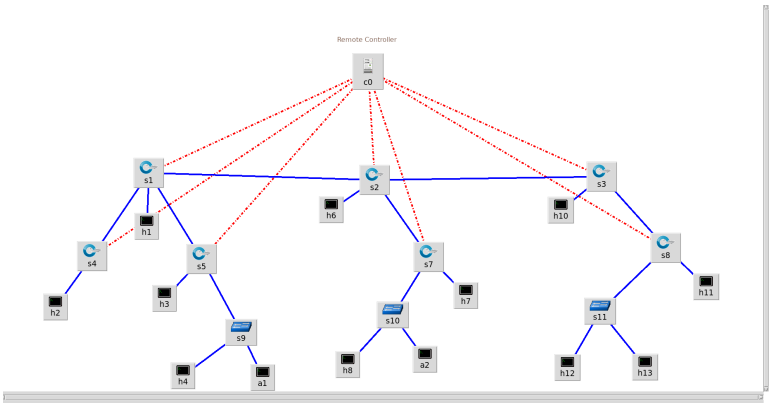
1.2 การตั้งค่าการจำลอง (Simulation setup)

เครื่องมือและอุปกร์ที่ใช้ในการทดลองมีดังนี้

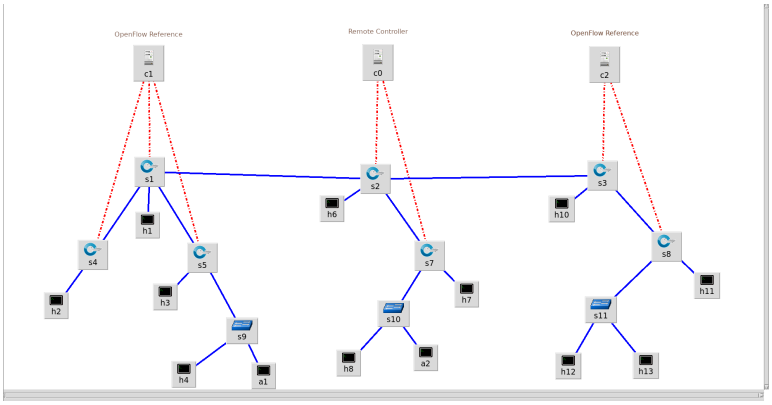
- Notebook Lenovo IdeaPad 5 14IIL05 รันด้วยระบบปฏิบัติการ Arch Linux 6.7.4-arch1-1 เป็นอุปกรณ์ที่ใช้สำหรับการรันซอฟต์แวร์ทดสอบทั้งหมด
- GNOME Terminal Version 3.50.1 เป็นซอฟต์แวร์อยู่บน Notebook ที่จะใช้ในการ ssh เข้าไปใน server
- Oracle VM VirtualBox เวอร์ชั่น 7.0.14 ใช้สำหรับการจำลองเครื่องเซิร์ฟเวอร์ขึ้นมาเป็นอีกเครื่องหนึ่งบน Notebook ที่กล่าวไปข้างต้น ซึ่งจะทำให้เหมือนว่ามีเครื่องเซิร์ฟเวอร์แยกเป็นอีกเครื่องหนึ่งจริงๆ
- ระบบปฏิบัติการ Ubuntu server 22.04.3 คือระบบปฏิบัติการที่ถูกรันอยู่บนเครื่อง Oracle VM VirtualBox
- Python version 3.9.18 และ 2.7.18 คือ runtime สำหรับรัน script ไฟล์ที่เขียนด้วยภาษา Python ซึ่งจะใช้ในการรัน RYU รวมไปถึงการจำลอง Topology ของ mininet
- RYU version 4.34 คือ ซอฟต์แวร์ ที่ได้กล่าวไปข้างต้น ซึ่งเป็นซอฟต์แวร์หลักตัวหนึ่ง ที่จะถูกใช้ในการทดลองครั้งนี้
- mininet version 2.3.1b4 คือซอฟต์แวร์ที่จะทำการจำลอง Topology ขึ้นมาภายในเครื่องจำลอง Ubuntu server ที่รันอยู่บน Oracle VM VirtualBox
- wireshark 3.6.2 คือซอฟต์แวร์ที่จะช่วยในการดักจับข้อมูลที่วิ่งผ่านระบบเครือข่าย ซึ่งจะเป็นซอฟต์แวร์ที่ใช้ในการวัดประสิทธิภาพในด้านเวลาและการป้องกันการโจมตี
- vim version 8.2.2121 คือซอฟต์แวร์ที่ใช้ในการแก้ไขไฟล์ข้อความหรือไฟล์ script ต่างๆ บนระบบปฏิบัติการ Ubuntu server
- Vscode 1.86.1 เป็นซอฟต์แวร์อีกตัวหนึ่งที่ใช้ในการแก้ไขไฟล์ข้อความซึ่งมี Extensions ช่วยในการเขียน ซึ่งในการทดลองนี้จะนำมาใช้ในการแก้ไขไฟล์ script ของ Python ในกรณีที่ต้องการความแม่นยำหรือแก้ปัญหาที่ซับซ้อน

1.3 รูปแบบในการทดลอง

ก่อนที่จะทำการทดลองนั้นต้องทำการเตรียม python scripts ที่ใช้ในการสร้าง topology จำลองบน mininet ซึ่งต้องสร้างทั้งแบบ Single Controller และ Multi Controllers โดยสามารถใช้ mininet gui สร้างได้ ตามภาพด้านล่างนี้ หรือสามารถดาวน์โหลดไฟล์ scripts ได้ที่ http://projectcs.sci.ubu.ac.th/nawasan/sdn-topo-mininet



รูปภาพที่ 1: Topology แบบ Single Controller



รูปภาพที่ 2: Topology แบบ Multi Controller

ในการทดลองครั้งนี้มี 4 รูปแบบ ซึ่งจะกำหนดให้มีความต่างเฉพาะที่กำหนด และขั้นตอนวิธียังคงเหมือนกัน โดยสิ่งที่ต้องเตรียมคือ Terminal ให้ทำการเปิดขึ้นมา 3 หน้าต่าง และทำการ ssh เข้าไปที่เซิร์ฟเวอร์จำลองให้เรียบร้อย โดยหน้าต่างที่ 1 จะใช้สำหรับการรัน RYU ทั้งแบบ group table และ proxy arp หน้าต่างที่ 2 จะใช้สำหรับการรัน topology จำลอง ทั้งแบบ Single Controller และ Multi Controllers และให้ h2 โจมตีไปยัง h12 หน้าต่างที่ 3 จะใช้สำหรับการรัน wireshark สำหรับตรวจจับแพ็กเกจ โดยจะมีขั้นตอนดังนี้

1. หน้าต่างที่ 1 ทำการรันคำสั่ง $ ryu-manager learn-sdn-with-ryu/ryu-exercises/<python script>
   โดย python script จะแทนด้วย ex7_group_tables.py และ ex8_arp_proxy.py
   ซึ่งได้มาจาก https://github.com/knetsolutions/learn-sdn-with-ryu.git

2. หน้าต่างที่ 2 ทำการรันคำสั่ง $ sudo python3 <python script>
   โดย python script จะแทนด้วย single_topo.py และ multi_topo.py

3. หน้าต่างที่ 3 ทำการรันคำสั่ง $ sudo wireshark
   โดยจับที่ขา s4-eth1 (ที่เชื่อมกับ h2) และ s11-eth2 (ที่เชื่อมกับ h12) ปริมาณ 1,000,000 แพ็กเกจ

4. หน้าต่างที่ 2 ให้ h2 โจมตี h12 ด้วยคำสั่ง $ mininet> h2 hping3 h12 -S –flood -V

5. เมื่อทำการจับแพ็กเกจครบ 1,000,000 แพ็กเกจแล้ว ทำการเก็บข้อมูลวิเคราะห์จาก wireshark จากนั้นให้ทำการหยุดรันในทุกๆ หน้าต่าง หากจะทำการทดลองรอบถัดไปต้องรันขั้นตอนใหม่ตั้งแต่ขั้นตอนที่ 1.

ในการทดลองครั้งนี้จะมี 4 รูปแบบ โดยทั้ง 4 รูปแบบจะทำการรัน 3 รอบ คือ

1. (A1) group table แบบ Single Controller
   หน้าต่างที่ 1 จะทำการรัน ryu ด้วย script ไฟล์ที่ชื่อ ex7_group_tables.py และ หน้าต่างที่ 2 ทำการรัน single-topo.py

2. (A2) group table แบบ Multi Controllers
   หน้าต่างที่ 1 จะทำการรัน ryu ด้วย script ไฟล์ที่ชื่อ ex7_group_tables.py และ หน้าต่างที่ 2 ทำการรัน multi-topo.py

3. (A3) proxy arp แบบ Single Controller
   หน้าต่างที่ 1 จะทำการรัน ryu ด้วย script ไฟล์ที่ชื่อ ex8_arp_proxy.py และ หน้าต่างที่ 2 ทำการรัน single-topo.py

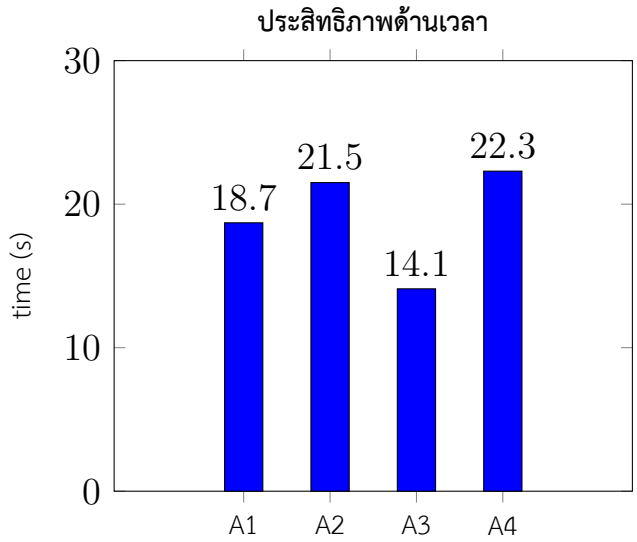
4. (A4) proxy arp แบบ Multi Controllers
   หน้าต่างที่ 1 จะทำการรัน ryu ด้วย script ไฟล์ที่ชื่อ ex8_arp_proxy.py และ หน้าต่างที่ 2 ทำการรัน multi-topo.py

2 ผลลัพธ์และการอภิปราย (Results and discussion)

เหตุผลที่ต้องทำการรัน 3 รอบ ในแต่ละรูปแบบนั้น เนื่องมาจากในขั้นตอนการเตรียมเครื่องมือและซอฟต์แวร์นั้นได้มีการรันทดสอบ และพบว่าค่ามีความแกว่งในบางครั้ง โดยในการรันครั้งแรกดรอปแพ็กเกจได้ 8% ในการรันครั้งที่ 2 อาจจะเพิ่มเป็น 30% หรือ 35% ดังนั้นจึงได้มีการออกแบบขั้นตอนการทดลองที่เหมือนกันมากที่สุดเพื่อลดปัจจัยที่อาจจะกระทบและทำให้ผลลัพธ์เปลี่ยนไปอย่างมีนัยสำคัญ นอกจากนั้นจึงได้มีการรันทดสอบ 3 รอบของแต่ละรูปแบบเพื่อลดโอกาสที่ค่าจะออกมาคลาดเคลื่อนหรือผิดไปจากที่ควรจะเป็น และในการรันทดสอบนั้นมีการจับแพ็กเกจด้วย wireshark ที่ s4-eth1 และ s11-eth2 ซึ่งทำให้มี 2 ค่าที่เกิดขึ้นจากทั้ง 2 ขาของการจับแพ็กเกจ ทางผู้ทดลองต้องการทำให้เป็นค่าเพียงหนึ่งค่า จึงจะทำการรวมค่าจาก s4-eth1 และ s11-eth2 ให้เป็นค่าหนึ่งโดยการบวก และในการรันทดสอบ 3 ครั้งนั้น ผู้ทำการทดลองพิจารณาแล้วว่าจะเลือกค่าอัตราการดรอปแพ็กเกจที่น้อยที่สุดมาใช้ในการวิเคราะห์ ด้วยเหตุที่ว่าค่าที่มีการแกว่งนั้นมักจะเป็นค่าที่มีค่ามากหรือมากที่สุด

2.1 ผลลัพธ์ประสิทธิภาพด้านเวลา

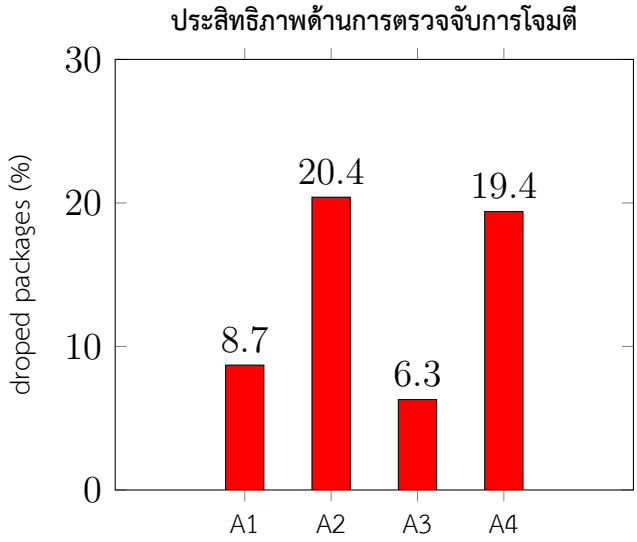
ผลลัพธ์การรันทดสอบการโจมตีในด้านประสิทธิภาพเวลา มีผลลัพธ์เป็นดังนี้ Group table แบบ Single Controller (A1) มีค่า time span เท่ากับ 18.7 Group table แบบ Multi Controllers (A2) มีค่า time span เท่ากับ 21.5 Proxy Arp แบบ Single Controller (A3) มีค่า time span เท่ากับ 14.1 Proxy Arp แบบ Multi Controllers (A4) มีค่า time span เท่ากับ 22.3



รูปภาพที่ 3: แผนภูมิแสดงผลลัพธ์ประสิทธิภาพด้านเวลา

2.2 ผลลัพธ์ประสิทธิภาพด้านการตรวจจับการโจมตี

ผลลัพธ์การรันทดสอบการโจมตีในด้านประสิทธิภาพการตรวจจับการโจมตี มีผลลัพธ์เป็นดังนี้ Group table แบบ Single Controller (A1) มีค่า droped packages เท่ากับ 8.7% Group table แบบ Multi Controllers (A2) มีค่า droped packages เท่ากับ 20.4% Proxy Arp แบบ Single Controller (A3) มีค่า droped packages เท่ากับ 6.3% Proxy Arp แบบ Multi Controllers (A4) มีค่า droped packages เท่ากับ 19.4%



รูปภาพที่ 4: แผนภูมิแสดงผลลัพธ์ประสิทธิภาพด้านการตรวจจับการโจมตี

2.3 สรุปผลการทดลอง

จากผลลัพธ์ที่ได้รายงานข้างต้น จะสังเกตได้ว่าในประสิทธิภาพด้านเวลานั้นไม่ได้มีความแตกต่างกันมาก แต่ก็พอจะกล่าวได้ว่าการใช้ Proxy Arp นั้นให้ประสิทธิภาพด้านความเร็วที่มากกว่า group table รวมไปถึงการใช้ Multi Controller ก็ให้ประสิทธิภาพด้านความเร็วที่มากกว่า Single Controllers และหากพิจารณาระหว่าง group table และ proxy arp

บน Single Controller จะเห็นได้ว่า Proxy arp ให้ประสิทธิภาพด้านความเร็วอย่างเห็นได้ชัด แต่หากพิจารณาบน Muti Controllers ก็จะพบว่าไม่ได้มีความต่างเท่าใดนัก อาจกล่าวได้ว่าให้ประสิทธิภาพที่เท่าๆ กัน

ในประสิทธิภาพด้านการตรวจจับการโจมตีนั้น จะสังเกตได้ว่า ทั้ง Group table และ Proxy arp ไม่ได้มีความต่างอย่างมีนัยสำคัญ แต่ว่าสิ่งที่ต่างอย่างเห็นได้ชัดเจนคือการรันแบบ Single Controller และ Multi Controllers โดยการรันแบบ Multi Controllers จะสามารถตรวจจับการโจมตีและทำการ drop packages ได้มากกว่าการรันแบบ Single Controller ดังนั้นหากพิจารณาต้องการการตรวจจับและป้องกันการโจมตี การรันด้วย Multi Controllers ก็ถือเป็นตัวเลือกที่น่าสนใจ

2.4 ข้อเสนอแนะ

ในการทดลองครั้งนี้ถูกรันบนเครื่องเซิร์ฟเวอร์จำลอง และสร้าง Topology จำลองขึ้น นอกจากนั้นการออกแบบขั้นตอนการทดลองยังคงไม่รัดกุมและชัดเจนมากเท่าที่ควร ซึ่งอาจจะทำให้ผลลัพธ์คลาดเคลื่อนจากความเป็นจริง อาจกล่าวได้ว่าการทดลองนี้เป็นเพียงการทดลองเบื้องต้นเพียงเท่านั้น ดังนั้นผู้อ่านจึงต้องมีความระมัดระวัง และรอบคอบในการนำไปใช้