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 ที่ถูกรันอยู่บน SDN Controller พร้อมกับการใช้งาน Group table หรือ Proxy ARP ทั้งในรูปแบบ Single Controller และ Multi Controllers

1.2 การตั้งค่าการจำลอง (Simulation setup)

เครื่องมือและอุปกร์ที่ใช้ในการทดลองมีดังนี้

คอมพิวเตอร์วางตักรุ่น Lenovo IdeaPad 5 14IIL05 รันด้วยระบบปฏิบัติการ Arch Linux 6.7.4-arch1-1 เป็นอุปกรณ์ที่ใช้สำหรับการรันซอฟต์แวร์ทดสอบทั้งหมด

GNOME Terminal Version 3.50.1 เป็นซอฟต์แวร์อยู่บนคอมพิวเตอร์วางตักที่จะใช้ในการ ssh เข้าไปใน server เพื่อทำการสั่งการทำงานต่างๆ

Oracle VM VirtualBox เวอร์ชั่น 7.0.14 ใช้สำหรับการจำลอง Server (เครื่องแม่ข่าย) ขึ้นมาเป็นอีกเครื่องหนึ่งบนคอมพิวเตอร์วางตักที่กล่าวไปข้างต้น ซึ่งจะทำให้เสมือนว่ามีเครื่องแม่ข่ายเพิ่มขึ้นมาอีกเครื่องหนึ่ง

ระบบปฏิบัติการ Ubuntu server 22.04.3 คือระบบปฏิบัติการที่ถูกรันอยู่บนเครื่องแม่ข่ายที่จำลองอยู่บน Oracle VM VirtualBox

Python version 3.9.18 และ 2.7.18 คือรันไทม์สำหรับรันสคริปต์ไฟล์ที่เขียนด้วยภาษา Python ซึ่งจะใช้ในการรัน RYU รวมไปถึงการจำลอง Topology ของ mininet

RYU version 4.34 คือ ระบบเครือข่ายที่กำหนดโดยซอฟต์แวร์ ที่ได้กล่าวไปข้างต้น ซึ่งเป็นซอฟต์แวร์หลักตัวหนึ่ง ที่จะถูกใช้ในการรันเชื่อมต่อกับ Controllers (ตัวควบคุม) บน mininet

mininet version 2.3.1b4 คือซอฟต์แวร์ที่จะทำการจำลอง Topology ที่ใช้ในการทดสอบขึ้นมาภายในเครื่องจำลอง Ubuntu server ที่รันอยู่บน Oracle VM VirtualBox

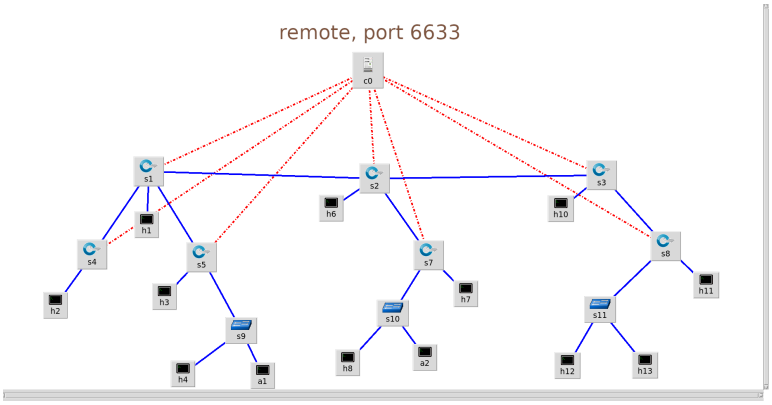
wireshark 3.6.2 คือซอฟต์แวร์ที่จะช่วยในการดักจับข้อมูลที่วิ่งผ่านระบบเครือข่าย ซึ่งจะเป็นซอฟต์แวร์ที่ใช้ในการวัดประสิทธิภาพในด้านเวลาและการป้องกันการโจมตี

vim version 8.2.2121 คือซอฟต์แวร์ที่ใช้ในการแก้ไขไฟล์ข้อความหรือไฟล์ข้อความต่างๆ บนระบบปฏิบัติการ Ubuntu server

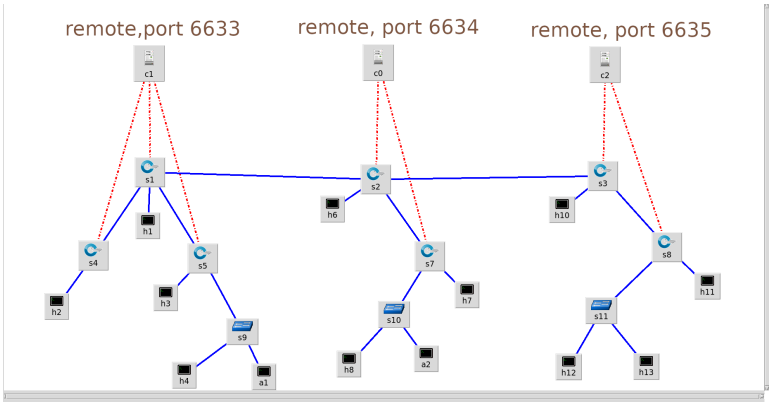
Vscode 1.86.1 เป็นซอฟต์แวร์อีกตัวหนึ่งที่ใช้ในการแก้ไขไฟล์ข้อความซึ่งมีส่วนเสริมที่ช่วยในการเขียน ซึ่งในการทดลองนี้จะนำมาใช้ในการแก้ไขไฟล์สคริปต์ของ Python ในกรณีที่ต้องการความแม่นยำหรือแก้ปัญหาที่ซับซ้อน

1.3 รูปแบบในการทดลอง

ในการจะทำการทดลองนั้นจำเป็นต้องมี python script 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 Controllers

โดยรูปภาพที่ 1 คือ Topology แบบมี 1 ตัวควบคุม โดยจะทำการกำหนดให้ตัวควบคุมเป็นแบบ remote และรูปภาพที่ 2 คือ Topology แบบมีหลายตัวควบคุม โดยจะทำการกำหนดให้ตัวควบคุมเป็นแบบ remote ทั้งหมด และกำหนด port เป็น 6633 6634 และ 6635 นอกจากนั้น เมื่อสร้าง script python ขึ้นมาแล้ว หรือดาวน์โหลดไฟล์จากลิงก์ที่ให้ไว้ จะทำการตั้งชื่อไฟล์เป็นดังนี้ Topology ที่มี 1 ตัวควบคุม ดังรูปภาพที่ 1 จะทำการตั้งชื่อเป็น single-topo.py และ Topology ที่มีหลายตัวควบคุม ดังรูปภาพที่ 2 จะทำการตั้งชื่อเป็น multi-topo.py

ในการทดลองครั้งนี้ผู้ทำการทดลองได้ออกแบบการทดลองให้มีความเข้มงวดและรัดกุมมากที่สุด เพื่อให้ผลลัพธ์ที่ได้มีความคลาดเคลื่อนน้อยที่สุด โดยผู้ทำการทดลองจะออกแบบการทดลองโดยให้มี 2 รูปแบบ 4 เงื่อนไข ดังนี้

เงื่อนไขที่ 1. (A1) group table แบบ 1 ตัวควบคุม

โดยจะทำการรัน ryu-manager ด้วย script ไฟล์ที่ชื่อ ex7_group_tables.py และทำการรันไฟล์ชื่อ single-topo.py ด้วย python เพื่อจำลอง Topology แบบ 1 ตัวควบคุม

เงื่อนไขที่ 2. (A2) group table แบบหลายตัวควบคุม

โดยจะทำการรัน ryu-manager ด้วย script ไฟล์ที่ชื่อ ex7_group_tables.py และทำการรันไฟล์ชื่อ multi-topo.py ด้วย python เพื่อจำลอง Topology แบบหลายตัวควบคุม

เงื่อนไขที่ 3. (A3) proxy arp แบบ 1 ตัวควบคุม

โดยจะทำการรัน ryu-manager ด้วย script ไฟล์ที่ชื่อ ex8_arp_proxy.py และทำการรันไฟล์ชื่อ single-topo.py ด้วย python เพื่อจำลอง Topology แบบ 1 ตัวควบคุม

เงื่อนไขที่ 4. (A4) proxy arp แบบหลายตัวควบคุม

โดยจะทำการรัน ryu-manager ด้วย script ไฟล์ที่ชื่อ ex8_arp_proxy.py และทำการรันไฟล์ชื่อ multi-topo.py ด้วย

python เพื่อจำลอง Topology แบบหลายตัวควบคุม

รูปแบบที่ 1. คือ ทำการทดสอบโดยจับแพ็กเก็ตทีละจำนวน 100,000 250,000 500,000 750,000 และ 1,000,000 แพ็กเก็ต จากนั้นนำมาวิเคราะห์เปรียบเทียบหาประสิทธิภาพด้านเวลาและการตรวจจับการโจมตี

รูปแบบที่ 2. คือ ทำการทดสอบโดยจำแพ็กเก็ตจำนวน 1,000,000 และ 2,000,000 แพ็กเก็ต จากนั้นนำมาวิเคราะห์เปรียบเทียบหาประสิทธิภาพด้านเวลาและการตรวจจับการโจมตี

สาเหตุที่มีการทดลองสองรูปแบบนั้นก็เพื่อให้สามารถตรวจสอบได้ว่าผลลัพธ์ที่ได้จากการทดสอบมีความถูกต้องน่าเชื่อถือ เป็นต้นว่าหากผลการทดสองจากทั้งสองรูปแบบสอดคล้องกัน ก็สามารถเชื่อได้ว่าผลการทดลองนั้นถูกต้อง นอกจากนั้นผู้ทำการทดลองยังได้กำหนดให้ลำดับวิธีของทั้ง 2 รูปแบบการทดลองมีความต่างกันในบางขั้นตอน เพื่อเพิ่มความเชื่อมั่นและความถูกต้องมากขึ้นไปอีก

1.4 ขั้นตอนลำดับวิธีทดลอง

ในการทดลอง ขั้นตอนลำดับจะต่างกันในส่วนของรูปแบบการทดลองเท่านั้น โดยที่เงื่อนไขต่างๆ จะยังคงมีลำดับการทดลองที่เหมือนกัน กล่าวคือในเงื่อนไขการทดลองนั้นจะต่างกันก็เพียงแต่ไฟล์ที่ทำการรันเท่านั้น โดยที่ลำดับขั้นตอนจะยังคงเหมือนเดิม ในแต่ละเงื่อนไขการทดลองจะทำการรันทดสอบ 3 รอบ และเลือกค่ากลางมาทำการวิเคราะห์ กล่าวคือ เลือกค่าที่ไม่น้อยที่สุดและไม่มากที่สุดจากค่าที่ได้จากการทดสอบ

รูปแบบการทดลองที่ 1 บน 1 ตัวควบคุม

ในขั้นต้นผู้ทำการทดลองต้องทำการเปิด gnome terminal ขึ้นมาและเปิดหน้าต่างย่อยเป็นจำนวน 3 หน้าต่าง ทุกหน้าต่างจะต้องทำการเชื่อมต่อไปยังเครื่องแม่ข่ายด้วย ssh ให้เรียบร้อย โดย หน้าต่างที่ 1 จะทำการรัน ryu-manager หน้าต่างที่ 2 จะทำการรัน mininet และหน้าต่างที่ 3 จะทำการรัน wireshark โดยมีลำดับขั้นตอนการทดลองดังนี้

1. หน้าต่างที่ 1 ทำการรันคำสั่ง $ ryu-manager learn-sdn-with-ryu/ryu-exercises/<python script>
โดย <python script> จะแทนด้วย ex7_group_tables.py หรือ ex8_arp_proxy.py ตามเงื่อนไขการทดลอง

2. หน่าต่างที่ 2 ทำการรันคำสั่ง $ sudo python3 single_topo.py
เพื่อทำการจำลอง topology แบบ 1 ตัวควบคุม ขึ้นมา

3. หน้าต่างที่ 3 ทำการรันคำสั่ง $ sudo wireshark โดยทำการจับแพ็กเก็ตที่ s4-eth1 (ที่เชื่อมกับ h2) และ s11-eth2 (ที่เชื่อมกับ h12) เป็นจำนวนที่เคยกล่าวไปข้างต้น

4. หลังจากที่ wireshark ทำการเริ่มจับแพ็กเก็ต หน้าต่างที่ 2 จะทำการรันคำสั่ง $ mininet> h2 hping3 h12 -S –flood -V โดยทันที เพื่อทำการให้ h2 โจมตีไปที่ h12

5. หลังจากทำการจับแพ็กเก็ตครบตามจำนวนที่กำหนด หน้าต่างที่ 2 ทำการหยุดการโจมตี จากนั้นทำการเก็บค่าสถิติและทำการปิดโปรแกรม wireshark หน้าต่างที่ 2 ทำการออกจาก mininet และรันคำสั่ง $ sudo mn -c หากมีการทดสอบรอบต่อไปต้องเริ่มจากขั้นตอนที่ 1 ใหม่เท่านั้น

รูปแบบการทดลองที่ 1 บนหลายตัวควบคุม

ในขั้นต้นผู้ทำการทดลองต้องทำการเปิด gnome terminal ขึ้นมาและเปิดหน้าต่างย่อยเป็นจำนวน 5 หน้าต่าง ทุกหน้าต่างจะต้องทำการเชื่อมต่อไปยังเครื่องแม่ข่ายด้วย ssh ให้เรียบร้อย โดย หน้าต่างที่ 1 2 และ 3 จะทำการรัน ryu-manager หน้าต่างที่ 4 จะทำการรัน mininet และหน้าต่างที่ 5 จะทำการรัน wireshark โดยมีลำดับขั้นตอนการทดลองดังนี้

1. หน้าต่างที่ 1 ทำการรันคำสั่ง
$ ryu-manager learn-sdn-with-ryu/ryu-exercises/<python script> –ofp-tcp-listen-port 6633

หน้าต่างที่ 2 ทำการรันคำสั่ง

$ ryu-manager learn-sdn-with-ryu/ryu-exercises/<python script> –ofp-tcp-listen-port 6634

หน้าต่างที่ 3 ทำการรันคำสั่ง

$ ryu-manager learn-sdn-with-ryu/ryu-exercises/<python script> –ofp-tcp-listen-port 6635

โดย <python script> จะแทนด้วย ex7_group_tables.py หรือ ex8_arp_proxy.py ตามเงื่อนไขการทดลอง

2. หน่าต่างที่ 4 ทำการรันคำสั่ง $ sudo python3 multi_topo.py

เพื่อทำการจำลอง topology แบบหลายตัวควบคุม ขึ้นมา

3. หน้าต่างที่ 5 ทำการรันคำสั่ง $ sudo wireshark โดยทำการจับแพ็กเก็ตที่ s4-eth1 (ที่เชื่อมกับ h2) และ s11-eth2 (ที่เชื่อมกับ h12) เป็นจำนวนที่เคยกล่าวไปข้างต้น

4. หลังจากที่ wireshark ทำการเริ่มจับแพ็กเก็ต หน้าต่างที่ 4 จะทำการรันคำสั่ง $ mininet> h2 hping3 h12 -S –flood -V โดยทันที เพื่อทำการให้ h2 โจมตีไปที่ h12

5. หลังจากทำการจับแพ็กเก็ตครบตามจำนวนที่กำหนด หน้าต่างที่ 4 ทำการหยุดการโจมตี จากนั้นทำการเก็บค่าสถิติ และทำการปิดโปรแกรม wireshark หน้าต่างที่ 2 ทำการออกจาก mininet และรันคำสั่ง $ sudo mn -c หากมีการทดสอบรอบต่อไปต้องเริ่มจากขั้นตอนที่ 1 ใหม่เท่านั้น

รูปแบบการทดลองที่ 2 บน 1 ตัวควบคุม

ในขั้นต้นผู้ทำการทดลองต้องทำการเปิด gnome terminal ขึ้นมาและเปิดหน้าต่างย่อยเป็นจำนวน 3 หน้าต่าง ทุกหน้าต่างจะต้องทำการเชื่อมต่อไปยังเครื่องแม่ข่ายด้วย ssh ให้เรียบร้อย โดย หน้าต่างที่ 1 จะทำการรัน ryu-manager หน้าต่างที่ 2 จะทำการรัน mininet และหน้าต่างที่ 3 จะทำการรัน wireshark โดยมีลำดับขั้นตอนการทดลองดังนี้

1. หน้าต่างที่ 1 ทำการรันคำสั่ง $ ryu-manager learn-sdn-with-ryu/ryu-exercises/<python script>

โดย <python script> จะแทนด้วย ex7_group_tables.py หรือ ex8_arp_proxy.py ตามเงื่อนไขการทดลอง

2. หน่าต่างที่ 2 ทำการรันคำสั่ง $ sudo python3 single_topo.py

เพื่อทำการจำลอง topology แบบ 1 ตัวควบคุม ขึ้นมา

3. หน้าต่างที่ 2 จะทำการรันคำสั่ง $ mininet> h2 hping3 h12 -S –flood -V เพื่อทำการให้ h2 โจมตีไปที่ h12

4. หน้าต่างที่ 3 ทำการรันคำสั่ง $ sudo wireshark จากนั้นจะทำการเริ่มจับแพ็กเก็ตหลังจากเริ่มการโจมตีแล้ว 30 วินาที โดยทำการจับแพ็กเก็ตที่ s4-eth1 (ที่เชื่อมกับ h2) และ s11-eth2 (ที่เชื่อมกับ h12) เป็นจำนวนที่เคยกล่าวไปข้างต้น

5. หลังจากทำการจับแพ็กเก็ตครบตามจำนวนที่กำหนด หน้าต่างที่ 2 ทำการหยุดการโจมตี จากนั้นทำการเก็บค่าสถิติ และทำการปิดโปรแกรม wireshark หน้าต่างที่ 2 ทำการออกจาก mininet และรันคำสั่ง $ sudo mn -c หากมีการทดสอบรอบต่อไปต้องเริ่มจากขั้นตอนที่ 1 ใหม่เท่านั้น

รูปแบบการทดลองที่ 2 บนหลายตัวควบคุม

ในขั้นต้นผู้ทำการทดลองต้องทำการเปิด gnome terminal ขึ้นมาและเปิดหน้าต่างย่อยเป็นจำนวน 5 หน้าต่าง ทุกหน้าต่างจะต้องทำการเชื่อมต่อไปยังเครื่องแม่ข่ายด้วย ssh ให้เรียบร้อย โดย หน้าต่างที่ 1 2 และ 3 จะทำการรัน ryu-manager หน้าต่างที่ 4 จะทำการรัน mininet และหน้าต่างที่ 5 จะทำการรัน wireshark โดยมีลำดับขั้นตอนการทดลองดังนี้

1. หน้าต่างที่ 1 ทำการรันคำสั่ง

$ ryu-manager learn-sdn-with-ryu/ryu-exercises/<python script> –ofp-tcp-listen-port 6633

หน้าต่างที่ 2 ทำการรันคำสั่ง

$ ryu-manager learn-sdn-with-ryu/ryu-exercises/<python script> –ofp-tcp-listen-port 6634

หน้าต่างที่ 3 ทำการรันคำสั่ง

$ ryu-manager learn-sdn-with-ryu/ryu-exercises/<python script> –ofp-tcp-listen-port 6635

โดย <python script> จะแทนด้วย ex7_group_tables.py หรือ ex8_arp_proxy.py ตามเงื่อนไขการทดลอง

2. หน่าต่างที่ 4 ทำการรันคำสั่ง $ sudo python3 multi_topo.py เพื่อทำการจำลอง topology แบบหลายตัวควบคุม ขึ้นมา

4. หน้าต่างที่ 4 จะทำการรันคำสั่ง $ mininet> h2 hping3 h12 -S –flood -V เพื่อทำการให้ h2 โจมตีไปที่ h12

3. หน้าต่างที่ 5 ทำการรันคำสั่ง $ sudo wireshark จากนั้นจะทำการเริ่มจับแพ็กเก็ตหลังจากเริ่มการโจมตีแล้ว 30 วินาที โดยทำการจับแพ็กเก็ตที่ s4-eth1 (ที่เชื่อมกับ h2) และ s11-eth2 (ที่เชื่อมกับ h12) เป็นจำนวนที่เคยกล่าวไปข้างต้น

5. หลังจากทำการจับแพ็กเก็ตครบตามจำนวนที่กำหนด หน้าต่างที่ 4 ทำการหยุดการโจมตี จากนั้นทำการเก็บค่าสถิติและทำการปิดโปรแกรม wireshark หน้าต่างที่ 2 ทำการออกจาก mininet และรันคำสั่ง $ sudo mn -c หากมีการทดสอบรอบต่อไปต้องเริ่มจากขั้นตอนที่ 1 ใหม่เท่านั้น

2 ผลลัพธ์และการอภิปราย (Results and discussion)

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

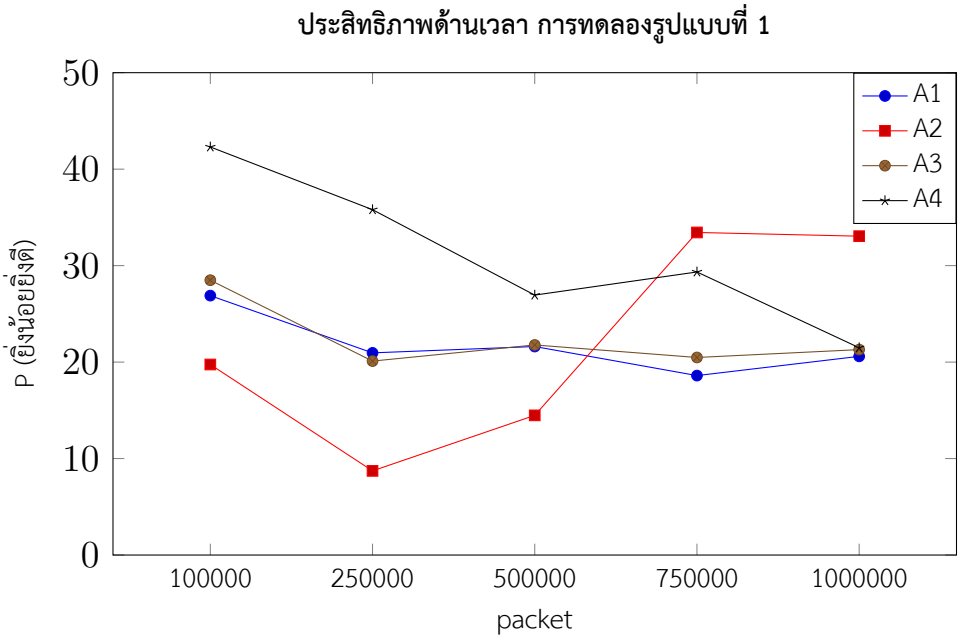
2.1 ผลลัพธ์การทดลอง

ผลลัพธ์ด้านเวลา

ในการทดสอบทั้ง 2 รูปแบบมีการจับแพ็กเก็ตหลายระดับ จึงอาจจะทำให้แพ็กเก็จจำนวนมากกว่าจะใช้เวลามากกว่า จึงได้ใช้สูตรคำนวนเพื่อให้สามารถเทียบวัดได้ โดยใช้สูตร

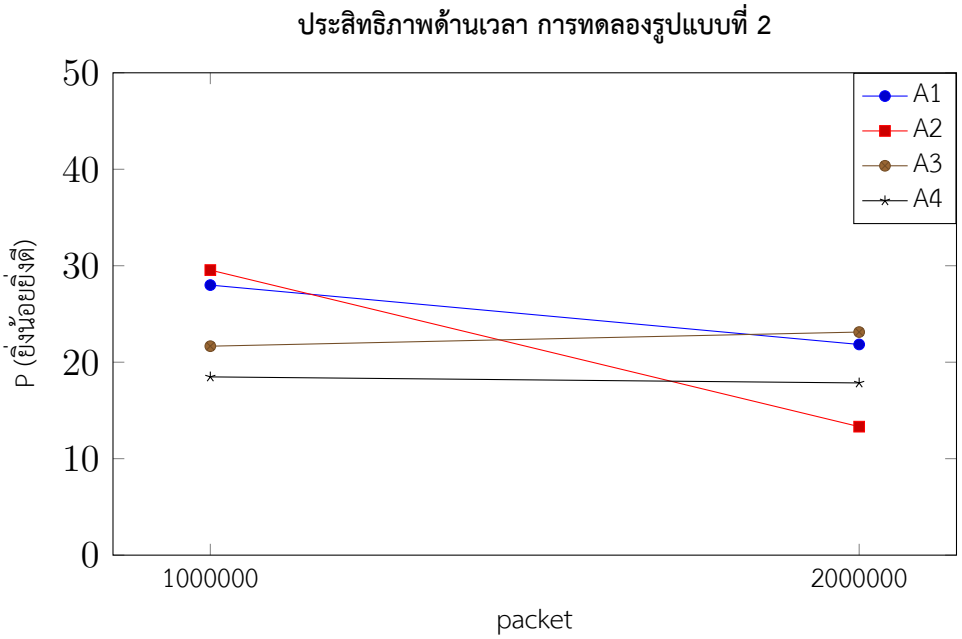
$$P = t \times \frac{m}{d}$$

โดยที่ P คือค่าคะแนน t คือเวลาที่ได้จากการวัด m คือจำนวนแพ็กเก็ตมากที่สุดที่ทำการวัด d คือจำนวนแพ็กเก็ตที่วัดในรอบนั้น



รูปภาพที่ 3: แผนภูมิแสดงผลลัพธ์ประสิทธิภาพด้านเวลาของการทดลองรูปแบบที่ 1

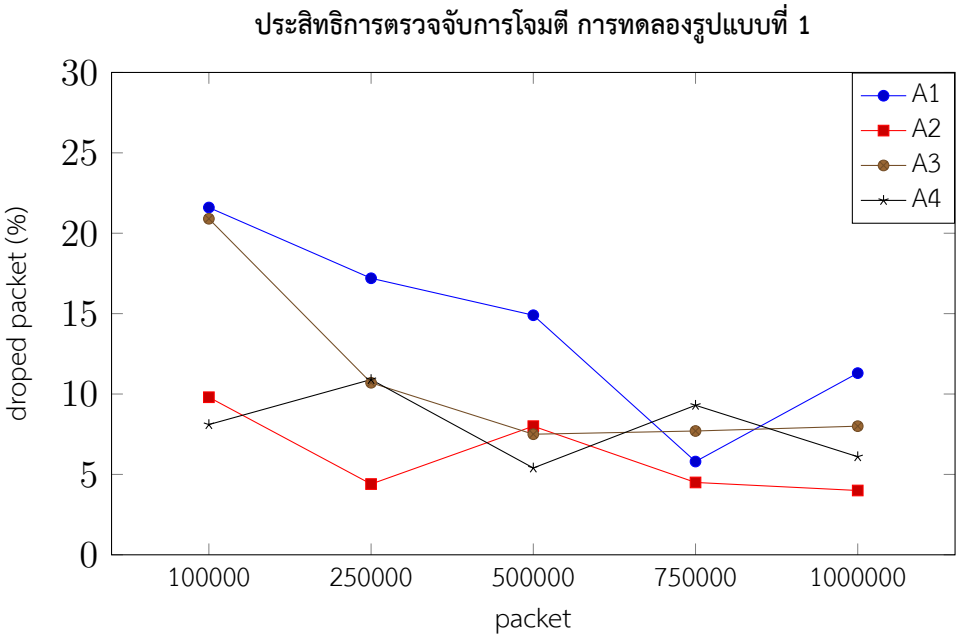
จากรูปภาพที่ 3 (A1) มีประสิทธิภาพไม่ต่างกันอย่างมีนัยสำคัญเมื่อแพ็กเก็ตเพิ่มขึ้น แต่มีแนวโน้มดีขึ้นเมื่อแพ็กเก็ตเพิ่มขึ้น (A2) มีแนวโน้มมีประสิทธิภาพน้อยลงเมื่อแพ็กเก็ตเพิ่มขึ้น แต่ไม่สม่ำเสมอ (A3) ประสิทธิภาพดีขึ้นเล็กน้อยอย่างไม่มีนัยสำคัญเมื่อแพ็กเก็ตเพิ่มขึ้น (A4) มีประสิทธิภาพดีขึ้นเมื่อแพ็กเก็ตเพิ่มขึ้น และมีแนวโน้มประสิทธิภาพจะดีขึ้นอีก เมื่อแพ็กเก็ตเพิ่มขึ้น



รูปภาพที่ 4: แผนภูมิแสดงผลลัพธ์ประสิทธิภาพด้านเวลาของการทดลองรูปแบบที่ 2

จากรูปภาพที่ 4 (A1) มีประสิทธิภาพดีขึ้นเมื่อแพ็กเก็ตเพิ่มขึ้น (A2) มีประสิทธิภาพดีขึ้นอย่างเห็นได้ชัดเมื่อแพ็กเก็ตเพิ่มขึ้น (A3) ประสิทธิภาพเพิ่มขึ้นเล็กน้อย แต่ไม่มีนัยสำคัญ (A4) ไม่มีการเปลี่ยนแปลงอย่างเห็นได้ชัด อาจกล่าวได้ว่าประสิทธิภาพเท่าเดิม

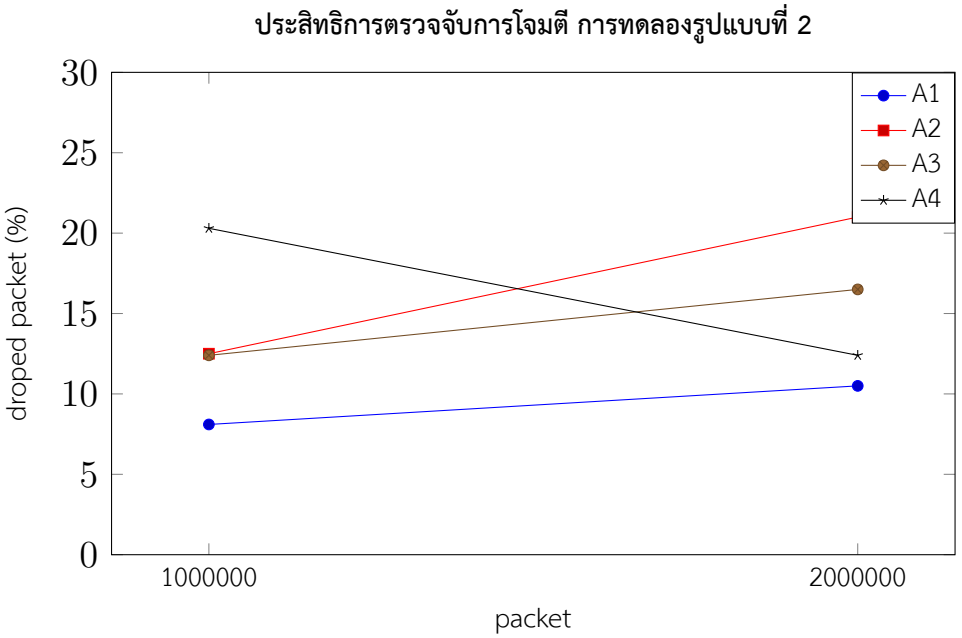
ผลลัพธ์ประสิทธิภาพด้านการตรวจจับการโจมตี



รูปภาพที่ 5: แผนภูมิแสดงผลลัพธ์ประสิทธิภาพด้านการตรวจจับการโจมตีของการทดลองรูปแบบที่ 1

จากรูปที่ 5 จะเห็นว่า (A1) มีประสิทธิภาพการตรวจจับที่น้อยลงเมื่อแพ็กเก็ตเพิ่มขึ้น (A2) มีประสิทธิภาพไม่คงที่ใน

แต่ละจำนวนแพ็กเก็ต และไม่มีความต่างอย่างมีนัยสำคัญ (A3) มีประสิทธิภาพน้อยลงเมื่อแพ็กเก็ตเพิ่มขึ้น แต่เมื่อถึงจุดหนึ่ง ประสิทธิภาพจะไม่ลดลงอีก (A4) มีประสิทธิภาพไม่คงที่ และไม่มีแนวโน้มจะลดลงหรือเพิ่มขึ้น



รูปภาพที่ 6: แผนภูมิแสดงผลลัพธ์ประสิทธิภาพด้านการตรวจจับการโจมตีของการทดลองรูปแบบที่ 2

จากรูปภาพที่ 6 (A1) มีประสิทธิภาพเพิ่มขึ้นเมื่อแพ็กเก็ตเพิ่มขึ้น และมีแนวโน้มประสิทธิภาพเพิ่มขึ้น (A2) ประสิทธิภาพเพิ่มขึ้นอย่างเห็นได้ชัด และมีแนวโน้มอย่างสูงว่าประสิทธิภาพจะเพิ่มขึ้นอีกหากแพ็กเก็ตเพิ่มขึ้น (A3) มีประสิทธิภาพเพิ่มขึ้นเมื่อแพ็กเก็ตเพิ่มขึ้น มีแนวโน้มเล็กน้อยว่าประสิทธิภาพจะเพิ่มขึ้นอีก (A4) มีประสิทธิภาพลดลงเมื่อแพ็กเก็ตเพิ่มขึ้น ซึ่งสวนทางกับเงื่อนไขอื่นอย่างเห็นได้ชัด

2.2 สรุปผลการทดลอง

จากที่ได้กล่าวไปว่า การที่ต้องทดลอง 2 รูปแบบ ก็เพื่อที่จะเปรียบเทียบและเพิ่มความถูกต้อง เป็นต้นว่า หากทั้งสองรูปแบบสอดคล้องกัน ก็เป็นที่น่าเชื่อว่าผลการทดลองนั้นออกมาถูกต้อง แต่หากทั้งสองรูปแบบออกมาขัดแย้งกัน ก็เป็นไปได้ว่าการทดลองนี้ไม่ถูกต้อง หรือมีข้อผิดพลาด ดังนั้น ในการที่จะสามารถบอกได้ชัดเจนเกี่ยวกับผลทดลองนั้น จำเป็นต้องดูทั้ง 2 รูปแบบประกอบกัน

โดยในประสิทธิภาพด้านเวลานั้น จะเห็นได้อย่างชัดเจนว่า (A2) นั้นผลออกมาขัดแย้งกันอย่างชัดเจน จึงกล่าวได้ว่าผลลัพธ์การทดสอบของ (A2) นั้นมีความน่าเชื่อถือต่ำ หรืออาจจะมีข้อผิดพลาดในการทดลอง จึงไม่อาจจะนำ (A2) ไปใช้ในการวิเคราะห์อื่นๆ ได้ ส่วนในผลลัพธ์อื่นๆ นั้น จะเห็นว่าทั้ง 2 รูปแบบไม่ได้ขัดแย้งกันอย่างชัดเจน ผลลัพธ์จึงมีความน่าเชื่อถือ โดย (A4) นั้นมีประสิทธิภาพที่ดีขึ้นเมื่อแพ็กเก็ตมีจำนวนเพิ่มขึ้น ส่วนที่เหลือนั้นไม่ได้มีความต่างอย่างชัดเจน สามารถกล่าวได้ว่า (A4) ให้ประสิทธิภาพด้านเวลาดีที่สุด

ในประสิทธิภาพด้านการตรวจจับการทดลองนั้น จะเห็นได้อย่างชัดเจนเช่นเดียวกับประสิทธิภาพด้านเวลา ว่า (A2) นั้นมีผลลัพธ์ที่ขัดแย้งกันจากทั้ง 2 รูปแบบ ดังนั้น (A2) จึงมีความน่าเชื่อถือต่ำ ในส่วน (A1) นั้น แม้ดูมีแนวโน้มว่าจะขัดแย้งกัน แต่เมื่อพิจารณาผลลัพธ์ในรูปแบบที่ 2 แล้ว ค่ามีการเพิ่มขึ้นเล็กน้อยเพียงเท่านั้น ทำให้ยังไม่ได้มีความขัดแย้งอย่างเห็นได้ชัด ส่วนอื่นๆ นั้น ถือว่าค่อนข้างมีความสอดคล้องกัน โดยที่ไม่ได้มีข้อใดมีประสิทธิภาพที่ดีกว่าอย่างเห็นได้ชัด หากดูตามกราฟแล้ว พอจะพูดได้ว่า (A3) ให้ประสิทธิภาพโดยรวมที่ดีที่สุด

ประสิทธิภาพด้านการตรวจจับการโจมตีนั้น แม้จะไม่ได้มีความขัดแย้งกันอย่างชัดเจน แต่ผลลัพธ์ก็ไม่ได้เป็นไปในทิศทางเดียวกันมากนัก ดังนั้นจึงต้องมีความระมัดระวังมากเป็นพิเศษในการนำผลลัพธ์ด้านการตรวจจับแพ็กเก็ตไป

วิเคราะห์ โดยหากต้องการนำข้อมูลใดๆ ไปวิเคราะห์หรือประกอบการตัดสินใจ แนะนำให้พิจารณาเลือกรูปแบบที่ 2 เป็นอันดับแรก เพราะมีช่วงจำนวนการจับแพ็กเก็ตมากกว่า

2.3 ข้อเสนอแนะ

ในการทดลองนี้ แม้ผู้ทำการทดลองจะออกแบบการทดลองให้มีความรัดกุม และเข้มงวดมากที่สุด แต่ก็หลีกเลี่ยงไม่ได้ที่จะมีข้อผิดพลาด หรือผลลัพธ์ที่คลาดเคลื่อน แม้แต่ผลลัพธ์ที่ไม่สอดคล้องเองก็ตาม อย่างไรก็ดี หากต้องการนำผลการทดลองไปใช้ ควรทำการทดสอบและวัดผลด้วยตัวเอง ตามวิธีที่เขียนไว้ หรืออาจจะวิธีการอื่นที่ดีกว่า ซึ่งอาจจะเพิ่มความรัดกุมขึ้นอีก หรือเพิ่มความแม่นยำของผลลัพธ์ โดยอิงตามทฤษฎีมากขึ้น และควบคุมตัวแปรได้อย่างเรียบร้อย