ระบบตรวจจับการบุกรุกด้วยการเรียนรู้ของเครื่องบนสถาปัตยกรรมเอสดีเอ็น

พาทิศ แก้วจินดา และ สุกฤษฎิ์ อั้นอยู่

คณะเทคโนโลยีสารสนเทศ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง กรุงเทพฯ

Email: 62070138@it.kmitl.ac.th, 62070202@it.kmitl.ac.th

บทคัดย่อ

สถาปัตยกรรมแบบถูกกำหนดโดยซอฟต์แวร์หรือสถาปัตยกรรมแบบเอสดีเอ็นในปัจจุบันเป็นที่นิยมใช้กันมากยิ่งขึ้น ซึ่งการใช้งานเครือข่ายแบบเอสดีเอ็นนั้นยังมีโอกาสการถูกโจมตีในรูปแบบต่าง ๆ โดยผู้จัดทำได้ศึกษาและพัฒนาระบบสำหรับตรวจสอบการโจมตีบนเครือข่ายรูปแบบสถาปัตยกรรมเอสดีเอ็นสำหรับการเฝ้าระวังการถูกโจมตีในรูปแบบ DoS เพื่อตรวจสอบการถูกโจมตีบนเครือข่าย

คำสำคัญ - สถาปัตยกรรมเอสดีเอ็น; ระบบตรวจจับการโจมตี; DoS Attack; Machine Learning

1. บทนำ

1.1. ที่มาและความสำคัญ

ในปัจจุบันการใช้ SDN นิยมและแพร่หลายมากขึ้นในการทำระบบเครือข่าย ทำให้มีรูปแบบการโจมตีบนเครือข่ายเพิ่มมากขึ้น จึงทำให้มีการพัฒนาเครื่องมือที่สามารถตรวจสอบการโจมตีบนเครือข่ายในรูปแบบของ SDN ขึ้นมาโดยอาศัยข้อมูลจาก controller ในการนำข้อมูลที่ได้ไปใช้กับระบบตรวจจับการโจมตี โดยจะเน้นไปในการถูกโจมตีในรูปแบบของ DoS เพื่อสามารถเฝ้าระวังและป้องกันการถูกโจมตีในรูปแบบ DoS ได้

1.2 วัตถุประสงค์

- เพื่อพัฒนาระบบตรวจจับการโจมตีที่ออกแบบตามสถาปัตยกรรมเอสดีเอ็น

- เพื่อพัฒนาและเปรียบเทียบประสิทธิภาพของเครื่องมือที่ใช้ตรวจจับการถูกโจมตีแบบ DoS

1.3. ขอบเขตการพัฒนาโครงงาน

ศึกษาการทำงานของ SDN และพัฒนาระบบการตรวจสอบการโจมตีบนเครือข่ายในรูปแบบ SDN โดยจะเน้นไปที่การถูกโจมตีในรูปแบบของ DoS ด้วยการใช้วิธีการตรวจสอบในรูปแบบต่างๆเพื่อหารูปแบบที่มีประสิทธิภาพสูงที่สุดและนำมาพัฒนาเป็นเครื่องมือสำหรับตรวจจับการถูกโจมตีแบบ DoS ได้

1.4. ขั้นตอนการดำเนินงาน

1) ศึกษารูปแบบการโจมตีบนเครือข่ายและวงจรการโจมตีต่างๆ

2) ศึกษาระบบตรวจจับการโจมตีที่มีอยู่ในปัจจุบัน

3) ศึกษาเทคโนโลยี Software-defined network

4) ศึกษาวิธีการจำลอง SDN ด้วยเครื่องมือต่าง ๆ

5) ศึกษาเครื่องมือที่ใช้ในการพัฒนาระบบตรวจจับการโจมตีเครือข่าย

6) ศึกษาและพัฒนาเครื่องมือที่ใช้ในการจำลองการโจมตี

7) พัฒนาระบบตรวจจับการโจมตีบนเครือข่าย

8) ทดสอบและประเมินเครื่องมือที่พัฒนาขึ้น

1.5. ประโยชน์ที่คาดว่าจะได้รับ

1) นักศึกษาได้รับความรู้และประสบการณ์ในการตรวจสอบการถูกโจมตีบนเครือข่าย

2) ได้เครื่องมือในการตรวจสอบและระบุการโจมตีบนเครือข่าย

2. ทฤษฎีและงานวิจัยที่เกี่ยวข้อง

จากการดำเนินงานในโครงงาน 1 ได้มีการพัฒนาระบบตรวจจับการโจมตีโดยการใช้งาน EWMA และ Entropy แล้ว ในโครงงาน 2 นั้นจึงได้มีความต้องการในการพัฒนาระบบตรวจจับการโจมตีโดยการใช้งาน Machine Learning หรือ Deep Learning ในการพัฒนาซึ่งเครื่องมือทั้ง 2 ประเภทที่ได้กล่าวมานั้นมีความซับซ้อนและหลากหลายในการพัฒนาอย่างมาก

2.1. ระบบเครือข่ายที่กำหนดโดยซอฟต์แวร์ (Software-defined Network)

ระบบเครือข่ายที่กำหนดโดยซอฟต์แวร์หรือ SDN นั้นเป็นระบบเครือข่ายมีการควบคุมแบบรวมศูนย์ กล่าวคือเป็นเครือข่ายที่มีอุปกรณ์เครือข่ายอยู่หลายตัวแต่ถูกควบคุมด้วยอุปกรณ์เพียงตัวเดียว โดยโครงสร้างของ SDN นั้นมี 3 ชั้น คือ

1) Application Layer เป็นชั้นที่มีแอปพลิเคชันต่าง ๆ ที่ใช้เกี่ยวข้องกับ Controller
2) Control Layer เป็นชั้นที่มีอุปกรณ์ควบคุมหรือ Controller อยู่ โดย Control Plane ทำหน้าที่เป็นเหมือนสมองของระบบคือเป็นอุปกรณ์ในการควบคุม switch
3) Data Layer เป็นชั้นที่มี switch ทำหน้าที่ในการรับ-ส่งข้อมูลระหว่าง end-user Device ที่อยู่ในเครือข่ายเดียวกัน ซึ่งการรับ-ส่งดังกล่าวจะมีการควบคุมมาจากในส่วนของ Control Plane

เนื่องจาก controller เป็นอุปกรณ์ในการควบคุมและจัดการอุปกรณ์ต่าง ๆ เพียงตัวเดียว จึงทำให้ง่ายต่อการจัดการอุปกรณ์เครือข่ายอื่น ๆ ได้ง่ายเช่นการเปลี่ยนแปลง

ข้อกำหนดในแต่ละอุปกรณ์หรือการตรวจตราอุปกรณ์เครือข่ายต่าง ๆ อีกทั้งยังสามารถส่งต่อข้อมูลของอุปกรณ์เครือข่ายไปยังโปรแกรมหรือแอปพลิเคชันที่ต้องการใช้ข้อมูลจากอุปกรณ์เครือข่ายได้โดยง่ายอีกด้วยเพราะสามารถดึงข้อมูลของอุปกรณ์เครือข่ายหลาย ๆ ตัวจากตัว controller ได้เลย

2.1.1. Mininet

Mininet คือเครื่องมือในการสร้างระบบ SDN พัฒนาด้วยภาษา python โดย Mininet สามารถจำลอง switch หรือ application ที่ทำงานบน SDN ได้ ซึ่งจุดประสงค์ของ Mininet คือสร้าง virtual environment เพื่อศึกษาและทดลองบน SDN

ใน Mininet จะมีโหนดอยู่ทั้งหมด 3 ประเภทคือ host, switch และ controller ซึ่ง host และ switch สามารถติดต่อกันได้โดยมี controller ที่สร้างขึ้นมาเป็นตัวควบคุมและจัดการเครือข่าย

2.1.2. Ryu

Ryu เป็น framework ที่ใช้ในการสร้าง controller ที่ใช้ใน SDN ซึ่ง Ryu พัฒนาด้วยภาษา python เช่นเดียวกับ Mininet จึงทำให้การพัฒนา controller นั้นสะดวกขึ้น

2.1.3 Google Colaboratory หรือ Google Colab

Google Colab ชื่อเต็มคือ Google Colaboratory เป็นบริการ Software as a Service (Saas) โฮสต์โปรแกรม Jupyter Notebook บน Cloud จาก Google ดังนั้นจึงสามารถใช้ Google Colab สร้าง Notebook เขียนโปรแกรมภาษา Python ได้ฟรี และยังสามารถใช้ GPU, TPU ได้ฟรีเช่นกันแต่อาจมีการถูกจำกัดการใช้งาน

2.2. Cyber Attack

Cyber Attack คือการโจมตีทางไซเบอร์โดยผลลัพธ์ของการโจมตีจะขึ้นอยู่กับประเภทของการโจมตีเช่นการใช้ ransomware เพื่อใช้ในการเรียกค่าไถ่จากเหยื่อที่โดนโจมตี หรือการใช้ DoS หรือ DDoS ในการโจมตีเหยื่อเพื่อทำให้เหยื่อไม่สามารถทำงานหรือให้บริการได้อย่างปกติหรือการใช้ spyware เพื่อขโมยข้อมูลของเหยื่อเป็นต้น

โดยในรายงานนี้จะเน้นการโจมตีที่ส่งผลให้เหยื่อไม่สามารถทำงานหรือให้บริการได้อย่างปกติและการโจมตีที่ผู้โจมตีสามารถรู้จุดอ่อนของเหยื่อได้

2.2.1. Denial-of-Service Attack: DoS

DoS เป็นการโจมตีทางไซเบอร์ประเภทหนึ่งที่ทำให้อุปกรณ์ในเครือข่ายนั้นไม่สามารถใช้งานได้ โดยการโจมตีจะเป็นการป่วน traffic ของอุปกรณ์เป้าหมายจนไม่สามารถทำงานได้หรือส่งคำสั่งบางอย่างที่ทำให้อุปกรณ์เป้าหมายล่มไป

การโจมตีประเภทนี้จะไม่ส่งผลถึงข้อมูลภายในอุปกรณ์เป้าหมาย กล่าวคือการโจมตีนี้จะไม่ได้มีการขโมยหรือดัดแปลงหรือทำลายข้อมูลในอุปกรณ์เป้าหมาย แต่การโจมตีประเภทนี้มีจุดประสงค์คือการรบกวนการทำงานขององค์กรเป้าหมายไม่ให้สามารถทำงานได้ตามปกติ ซึ่งเป้าการโจมตีของ DoS มักจะเป็น web server ของบริษัททางการเงิน, บริษัทที่เกี่ยวข้องกับสื่อและองค์กรของรัฐเป็นต้น

2.2.2. Distributed Denial-of-Service: DDoS

DDoS นั้นจะมีพฤติกรรมการโจมตีคล้ายคลึงกับ DoS เพียงแต่ DDoS นั้นคือการใช้อุปกรณ์หลายตัวในการโจมตีต่างจาก DoS ที่ใช้อุปกรณ์เพียงเครื่องเดียวในการทำการโจมตี ซึ่งการโจมตีแบบ DDoS นั้นจะตรวจจับได้ยากกว่า DoS เนื่องจาก DoS เป็นการโจมตีที่มาจากอุปกรณ์เดียวส่งมาที่เครื่องของเหยื่อจึงสามารถรู้ถึงการโจมตีและอุปกรณ์ที่โจมตีได้และสามารถบล็อคอุปกรณ์ได้อย่างถูกต้อง แต่ DDoS นั้นมีอุปกรณ์หลายตัวติดต่อเข้ามาหาเหยื่อหลายตัว จึงทำให้แยกแยะการเชื่อมต่อที่มาจากอุปกรณ์ต้นทางแต่ละเครื่องได้ยากว่าการเชื่อมต่อนั้น ๆ เป็นการเชื่อมต่อปกติหรือเป็นการเชื่อมต่อที่ตั้งใจจะเข้ามาโจมตีอุปกรณ์

2.3. วิธีการตรวจจับการโจมตีด้วย Threshold และ Entropy

2.3.1. Exponentially Weighted Moving Average: EWMA

Exponentially Weighted Moving Average หรือ EWMA เป็นค่าวัดในเชิงปริมาณหรือสถิติประเภทหนึ่งที่เกี่ยวข้องกับอนุกรมเวลา สำหรับใช้งาน EWMA ผู้ใช้งานมีหน้าที่ในการกำหนด α ซึ่งใช้เป็นน้ำหนักในการเลือกใช้ข้อมูลเก่าหรือใหม่ หากกำหนด α ไว้สูง EWMA ก็จะมีค่าที่เปลี่ยนแปลงใกล้เคียงกับค่าใหม่ หากกำหนด α ไว้ต่ำ EWMA ก็จะมีค่าที่อ้างอิงจากค่าเก่า

EMWA หาได้จากสมการดังนี้

$$EWMA_t = \alpha * r_t + (1 - \alpha) * EWMA_{t-1} \quad (1)$$

โดย α คือค่าน้ำหนักที่กำหนดโดยผู้ใช้งาน

r_t คือค่าที่เข้ามาในช่วงเวลาหนึ่ง

และเนื่องจาก EWMA เป็น recursive function จึงหมายความว่าค่า EWMA ของปัจจุบันต้องใช้ค่า EWMA ในอดีตมาคำนวณด้วย โดยสามารถสร้างเป็นสมการดังนี้ได้

$$EWMA_t = \alpha * r_t + (1 - \alpha) * (\alpha * r_{t-1} + (1 - \alpha) * EWMA_{t-2}) \quad (2)$$

2.4. การเลือก Model สำหรับ Machine Learning

สำหรับการพัฒนา Machine Learning ในโครงงานนี้ ทางผู้พัฒนาได้เลือกใช้การพัฒนาแบบ Supervised Learning เพราะว่าสามารถเลือกวิธีการประมวลผลของข้อมูลสำหรับ Machine Learning ได้ ซึ่ง Model ที่ทางผู้พัฒนาสนใจมีทั้งหมด 3 ตัวคือ

2.4.1. XGBoost (Extreme Gradient Boosting)

XGBoost (Extreme Gradient Boosting) เป็นอัลกอริทึมแบบ gradient boosting เป็นรุ่นที่ปรับแต่งมาจาก Gradient boosting อีกที เป็น model ที่นำเอา Decision Tree มา train ต่อๆกันหลายๆ tree โดย decision tree จะเรียนรู้จาก error ของ tree ก่อนหน้า ทำให้ความแม่นยำของในการทำ prediction จะ แม่นยำมากขึ้นเรื่อยๆ เมื่อมีการเรียนรู้ของ tree ต่อเนื่องกันจนมีความลึกมากพอ และ model จะหยุดเรียนรู้เมื่อไม่เหลือ pattern ของ error จาก tree ก่อนหน้าให้เรียนรู้แล้ว

XGBoost ได้ความนิยมสูงเนื่องจากความสามารถในการให้ความแม่นยำสูงและการจัดการกับชุดข้อมูลที่ซับซ้อนได้อย่างมีประสิทธิภาพ

โดยคุณสมบัติและองค์ประกอบสำคัญของ XGBoost มีดังนี้

- Gradient Boosting : เป็นการนำ Decision Tree มาพัฒนาให้กลายเป็นโมเดลที่สามารถทำนายได้แม่นยำมากยิ่งขึ้น
- การทำ Regularization : XGBoost นำเอาเทคนิคการปรับค่าแบบ regularization เข้ามาเพื่อป้องกันการเกิด overfitting
- ยังมีการใช้ Tree-Based Models เป็นหลัก
- Gradient Optimization : XGBoost ใช้อัลกอริทึมการปรับค่าแบบ gradient ที่กำหนดค่าน้ำหนักที่เหมาะสมสำหรับแต่ละต้นตัดสินใจอย่างมีประสิทธิภาพ ช่วยเร่งความเร็วในการฝึกและเพิ่มความแม่นยำ
- สามารถจัดการกับ Missing Values ได้ : XGBoost สามารถจัดการกับ Missing Values ใน dataset ได้ โดยการเรียนรู้ทิศทางการของข้อมูลที่ผิดพลาดได้ใน Tree สามารถตัดสินใจส่งค่านั้นไปทาง ซ้าย หรือขวาของโหนดได้ในระหว่างการทำงานอยู่ภายใน Tree
- การประมวลผลแบบขนาน: ช่วยให้การฝึกและการทำนายสามารถทำงานพร้อมกันได้ โดยใช้ประโยชน์จากหน่วยประมวลผลหลายคอร์ที่มีใน CPU และ - Framework การคำนวณแบบกระจาย เพื่อเพิ่มความเร็วในการฝึกและทำนาย ซึ่งทำให้เหมาะสำหรับชุดข้อมูลขนาดใหญ่
- Feature Importance : สามารถวิเคราะห์หา Feature ที่สำคัญและมีผลกับโมเดลได้

2.4.2. Decision Tree

Decision Tree เป็นอัลกอริทึมที่ใช้ในงานที่เกี่ยวข้องกับการจัดการกับข้อมูลแบบตัดสินใจหรือการ

จัดหมวดหมู่ (classification) และการทำนาย (prediction) โดย Decision Tree จะสร้างโครงสร้างที่มีลักษณะเป็นต้นไม้ โดยแต่ละโหนดในต้นไม้จะแทน features ของข้อมูล และการแบ่งตัดสินใจขึ้นอยู่กับค่าของ feature

Decision Tree เป็น Rule-Based Model ที่จะสร้างเงื่อนไข ขึ้นมาจากข้อมูลในตัวแปร เพื่อที่จะแบ่งข้อมูลออกเป็นกลุ่มใหม่ที่สามารถอธิบาย Target ได้ดีที่สุด โดยการสร้างเงื่อนไข If-else ในแต่ละตัวแปร จะถูกกำหนดด้วย Objective Function ซึ่ง Model Decision Tree มี Objective Function อยู่หลายตัวตามประเภทของ Decision Tree นั้น ๆ

2.4.4 SHAP

SHAP (SHapley Additive exPlanations) เป็น Framework สำหรับอธิบายผลลัพธ์จาก Model ของ Machine Learning ช่วยให้ผู้ใช้ที่มีความค่อนข้างไม่มั่นใจในตัวของ Model บางครั้งอาจจะทำให้ Model นั้นยังไม่มีประสิทธิภาพมากพอ ดังนั้น SHAP สามารถดู Model แล้ววิเคราะห์ตัว Model ออกมาให้ผู้ใช้เห็นข้อมูลของ Model ได้มากยิ่งขึ้น สามารถเข้าใจการทำนายจากโมเดลได้ และสามารถอธิบาย Features ต่างๆภายในโมเดล รวมถึงการแสดงผลออกมาในรูปแบบของกราฟ

3. การดำเนินงาน

3.1. ส่วนประกอบระบบ

ระบบงานจะเป็นการใช้งานโครงสร้าง SDN ซึ่งจะมีทั้งหมด 3 ชั้นดังที่กล่าวไปในบทที่ 2 ซึ่งในการดำเนินงานของโครงงานนี้ได้มีการใช้งานโครงสร้างต่าง ๆ ของ SDN ดังนี้

1) Data Layer ทำหน้าที่ในการสร้างข้อมูล Traffic เพื่อส่งให้ Control Layer
2) Control Layer ทำหน้าที่สั่งการ Switch และ ส่งข้อมูลของ Traffic ไปให้ IDS System ในชั้น Application Layer
3) Application Layer หรือ IDS System ทำหน้าที่ในตรวจจับการโจมตีที่ได้มาจาก Control Layer

ซึ่งตัว IDS System นั้นมี Model ที่ใช้ในการตรวจจับการโจมตีอยู่ ซึ่งในโครงงานนี้ตัว Model ที่ใช้ในการตรวจจับการโจมตีจะมี 2 ตัวดังนี้

3.1.1. EWMA Model

EWMA Model เป็น Model ตรวจจับที่ใช้สมการ EWMA มาประยุกต์ในการตรวจจับการโจมตี โดยเป็นการนำข้อมูลที่ได้จาก Controller มากรองเอาข้อมูลที่ต้องการและนำไปคำนวณหาการโจมตีโดยใช้สมการ EWMA มาเกี่ยวข้อง

3.1.2. ML Model

ML Model เป็น Model ตรวจจับที่ใช้สมการ EWMA มาประยุกต์ในการตรวจจับการโจมตี โดยเป็นการนำข้อมูลที่ได้จาก Controller มาทำเป็นชุดข้อมูลเพื่อใช้ในการ Train และ Test ตัว Machine Learning เพื่อใช้ในการตรวจจับการโจมตี

3.2. การพัฒนา Model

การพัฒนาตัว IDS ของ โครงงานซึ่งในโครงงานนี้เลือกใช้ IDS ที่มีวิธีการตรวจจับหลายรูปแบบ ไม่ว่าจะเป็นรูปแบบของการตรวจหาโดยใช้ EWMA หรือ การวัดจากการหาค่า Entropy นั้นสามารถใช้วิธีที่นอกเหนือจากนี้ได้นั่นคือการใช้ Machine Learning เข้ามาช่วยตรวจจับความผิดปกติในเครือข่ายได้

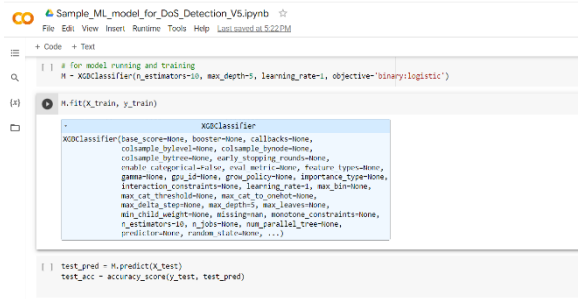
สำหรับสัดส่วนในการแบ่งชุดข้อมูลที่ใช้ในการ Train และ Test ของ Model จะแบ่งโดยใช้อัตราส่วน 70:30 ชื่นในแต่ละส่วนก็จะมีการแบ่งข้อมูลที่เป็นการโจมตีและข้อมูลที่ไม่ใช่การโจมตีในสัดส่วนที่อยู่ในชุดข้อมูลหลัก กล่าวคือหากชุดข้อมูลหลักมีสัดส่วนระหว่างข้อมูลที่เป็นการโจมตีและข้อมูลที่ไม่เป็นการโจมตีที่ 55:45 ชุดข้อมูลที่ใช้ในการ Train หรือ

Test ก็ต้องมีสัดส่วนข้อมูลที่เป็นการโจมตีและข้อมูลที่ไม่เป็นการโจมตีอยู่ที่ 55:45 เช่นกัน

สำหรับเครื่องมือที่ใช้ในการพัฒนาโมเดลคือ XGBoost ที่ได้กล่าวไปในบทก่อนหน้านั้น โดยทางผู้พัฒนาได้เตรียมใช้ตัว XGBClassifier ในการเป็น Algorithm สำหรับ XGBoost ในการตรวจจับการโจมตี และได้กำหนด Features ที่ใช้ในการตรวจจับการโจมตีทั้งหมด 8 ตัว

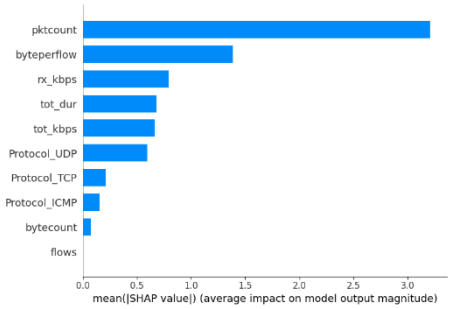
1) Packet_count คือปริมาณ Packet ใน Flow
2) Byte_count คือปริมาณ Byte ใน Flow
3) Tot_dur คือระยะเวลาที่ Flow เกิดขึ้นมา
4) Rx_kbps คือปริมาณการรับข้อมูลของ Port ในรูปแบบ kb ต่อวินาที
5) Tot_kbps คือปริมาณการรับและส่งข้อมูลของ Port ในรูปแบบ kb ต่อวินาที
6) Flows คือปริมาณ Flow ที่เกิดขึ้นในระบบ
7) Byteperflow คือปริมาณข้อมูลต่อ Flow ในรูปแบบ Byte
8) Protocol โดยแยกเป็น ICMP, TCP และ UDP

เมื่อเลือก Feature ที่ต้องการใส่ใน Model แล้ว จึงนำข้อมูลที่มี Feature ดังกล่าวใส่เข้าไปใน Model เพื่อทำการ Train และ Test เพื่อหาผลลัพธ์ ซึ่งผลลัพธ์จะอยู่ในบทถัดไป



รูปที่ 3.1. ตัวอย่างการสร้าง XGBClassifier

หลังจากทำการ Train และ Test เรียบร้อยแล้ว จึงมีการใช้ SHAP เพื่อดูว่า Feature ใดที่ส่งผลต่อผลลัพธ์ของ Model มากที่สุด ซึ่งมีผลตามรูปที่ 3.2.



รูปที่ 3.2. ผลลัพธ์จากการใช้ SHAP

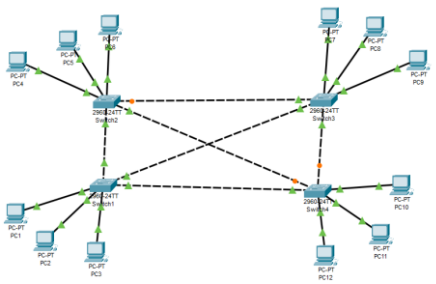
จากรูปจะเห็นได้ว่า pktcount ส่งผลต่อผลลัพธ์ของ Model มากที่สุดและรองลงมาคือ byteperflow และลดลงเรื่อย ๆ จนเห็นได้ว่า flow แทบไม่มีผลต่อการคำนวณเลย ข้อสังเกตคือ Protocol UDP มีผลต่อการคำนวณผลลัพธ์ของ Model มากกว่า Protocol อื่น ๆ ซึ่งทางผู้พัฒนาคิดว่าเกิดจากชุดข้อมูลมีข้อมูล UDP น้อยกว่า Protocol อื่น จึงอาจส่งผลให้ UDP มีความสำคัญต่อผลลัพธ์ของ Model มากกว่า Protocol อื่น

3.3. การทดลอง Model ที่ได้พัฒนาขึ้น

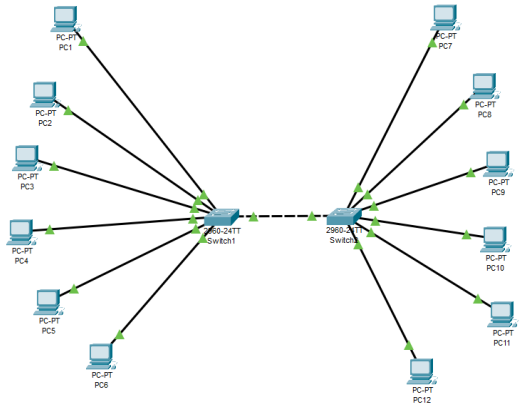
หลังจากที่ได้อธิบายการสร้าง Machine Learning Model แล้ว ทางผู้พัฒนาจึงต้องการจะทดลอง IDS System ที่มีโดยการสร้างชุดข้อมูลหนึ่งเพื่อมาทดลอง IDS System เพื่อวัดประสิทธิภาพในการตรวจจับการโจมตี

โดย topology ที่ใช้ในการทดลองมีทั้งหมด 3 ตัว ดังรูปด้านล่าง

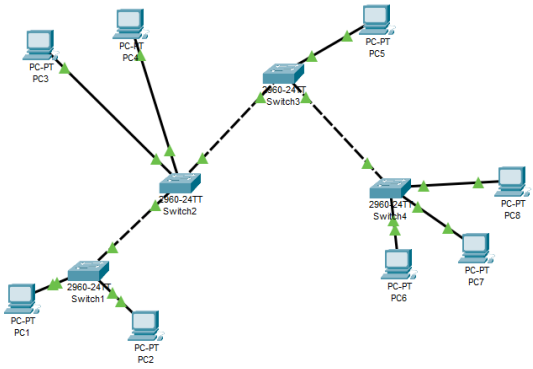
โดย Topology ที่ใช้ในการทดลองจะเป็นดังรูป 3.3.



รูปที่ 3.3. รูป Topology ของ Scenario 1



รูปที่ 3.4. รูป Topology ของ Scenario 2



รูปที่ 3.5. รูป Topology ของ Scenario 3

ซึ่งในชุดข้อมูลนี้มีการกำหนดให้มีเครื่องใน Topology ทำการส่ง Traffic ที่เป็นการโจมตี 3 เครื่อง และส่ง Traffic แบบปกติ 3 เครื่อง โดยใช้เวลาในการสร้างข้อมูลประมาณ 7 ชั่วโมง

4. ผลการทดลอง

จากบทที่แล้วที่ได้มีการพัฒนา Model โดยใช้ XGBoost เพื่อใช้ในการตรวจจับการโจมตีที่เกิดขึ้นในเครือข่ายนั้น ในบทนี้จะกล่าวถึงผลการทดลองที่ได้กล่าวไปในบทก่อนหน้าดังนี้

4. ผลการทดลอง

4.1. ผลลัทธ์จากการพัฒนา Model

ในการพัฒนา Model นั้น ทางผู้พัฒนาได้ใช้ Evaluation Metric ทั้งหมด 4 ประเภทคือ

1) Accuracy

ในการคำนวณ Accuracy สามารถหาได้จากสมการ $Accuracy = \frac{TP+TN}{TP+TN+FP+FN}$

2) Precision

ในการคำนวณ Precision สามารถหาได้จากสมการ $Precision = \frac{TP}{TP+FP}$

3) Recall

ในการคำนวณ Recall สามารถหาได้จากสมการ $Recall = \frac{TP}{TP+FN}$

4) F1-Score

ในการคำนวณ F1-Score สามารถหาได้จากสมการ $F1 - Score = \frac{2\,TP}{2TP+FP+FN}$

โดย

TP คือ True Positive หรือ การโจมตีเกิดขึ้นจริง

TN คือ True Negative หรือ ไม่มีการโจมตีเกิดขึ้นจริง

FP คือ False Positive หรือ มีการโจมตีเกิดขึ้นไม่จริง

FN คือ False Negative หรือ ไม่มีการโจมตีเกิดขึ้นไม่จริง

โดยผลลัพธ์จากการทดลองทั้ง 2 โมเดลจะมีดังตารางด้านล่าง

ตารางที่ 4.1. Evaluation Metric ของ EWMA Model

Model	Accuracy	Precision	Recall	F1-Score
EWMA Model	56%	0.9	0.39	0.55

ตารางที่ 4.2. ตาราง Evaluation Metric ของ ML Model

Scenario	Accuracy	Precision		Recall		F1-Score	
		0	1	0	1	0	1
Scenario 1	56%	41%	90%	91%	39%	57%	55%
Scenario 2	63%	52%	75%	69%	60%	59%	67%
Scenario 3	68%	57%	83%	81%	60%	67%	70%
Overall	62%	50%	83%	80%	53%	61%	64%

4.2. สรุปผลการทดลอง

จากผลลัพธ์ดังกล่าวจะเห็นได้ว่า ML Model มี Accuracy มากกว่าการใช้งาน EWMA Model อย่างมาก เนื่องจาก EWMA ใช้เพียงปริมาณ Packet ในการเป็น Feature ตรวจจับการโจมตี จึงทำให้มีข้อผิดพลาดมากกว่าตัว ML Model ที่มีการใช้ Feature ในการตรวจจับมากกว่านั่นเอง

4.3. ข้อดีและข้อเสียของแต่ละ Model

จากผลลัพธ์ในหัวข้อ 4.2. นั้นแสดงให้เห็นว่า ML Model มีประสิทธิภาพดีกว่า EWMA Model แต่ว่าทั้ง 2 Model นั้นก็จะมีข้อดีและข้อเสียแตกต่างกันออกไป

โดยสำหรับข้อดีและข้อเสียของ EWMA Model มีข้อดีคือImplement ง่ายเพราะใช้ Feature เพียงตัวเดียวในการตรวจจับการโจมตีและเหมาะกับการตรวจจับการโจมตีแบบ Spike Attack แต่สำหรับข้อเสียของ EWMA Model คือมีความแม่นยำต่ำ

สำหรับข้อดีและข้อเสียของ ML Model มีข้อดีคือความแม่นยำสูงแต่ข้อเสียคือความแม่นยำของการตรวจจับการโจมตีขึ้นอยู่กับชุดข้อมูลที่ใช้ในการ Train

การพัฒนาระบบการ
วิเคราะห์และแจ้งเตือนปัญหาเครือข่าย

ธัชพนธ์ ศรีอากาศไกรแสง[1] และ ภัทรพล เงารัตนพันธิกุล[2]

[1]คณะเทคโนโลยีสารสนเทศ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

[2]คณะเทคโนโลยีสารสนเทศ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

Emails: 62070090@kmitl.ac.th, 62070147@kmitl.ac.th

บทคัดย่อ

เนื่องจากในปัจจุบันผู้ใช้งานระบบเครือข่ายมักพบเจอปัญหาระหว่างการใช้งานระบบเครือข่าย โดยปัญหาอาจจะเกิดที่ระบบเครือข่ายเองหรือเกิดที่อุปกรณ์นั้นมีความเสียหาย ซึ่งอาจจะทำให้ส่งผลกระทบต่อด้านต่างๆได้ จึงสนใจที่จะทำการพัฒนาระบบในการวิเคราะห์ข้อมูลของเครือข่ายเพื่อนำมาปรับปรุงให้ดีขึ้นและแจ้งเตือนเมื่อเครือข่ายเกิดปัญหาขึ้นได้ โดยงานวิจัยนี้ได้ทำการออกแบบและพัฒนาระบบที่สามารถนำข้อมูล Packet ที่ดักจับมาจาก Wireshark ในระบบเครือข่ายมาวิเคราะห์ปัญหาและจัดเก็บเป็นสถิติเพื่อที่จะนำไปใช้ในอนาคต ซึ่งตัวระบบสามารถตรวจจับและคัดแยก Protocol ต่างๆ ภายใน Packet โดยนำข้อมูลมาตรวจสอบความล่าช้าและความสูญหายแล้วทำการแจ้งเตือนได้ โดยการแจ้งเตือนจะใช้หลักการกำหนดค่าดีเลย์ไว้เมื่อเกิดค่าดีเลย์ที่สูงกว่ากำหนดจะทำการแจ้งเตือน หลังพัฒนาได้ผลสรุปของระบบสามารถรับข้อมูลจาก Wireshark, คัดแยก Protocol, วิเคราะห์ข้อมูล,แจ้งเตือนปัญหาและเก็บบันทึกสถิติได้อย่างถูกต้อง

จากการพัฒนาและทดสอบด้านความถูกต้องตัวระบบสามารถคัดแยก Protocol จับคู่เก็บค่าต่างๆได้ครบถ้วนโดยมีการแจ้งเตือนเมื่อเกิดปัญหาอย่างถูกต้อง ในด้านประสิทธิภาพของโปรแกรมเมื่อมีจำนวน Packet มากจะใช้เวลามากขึ้น ซึ่งการนำไปใช้งานจริงควรใช้สเปคคอมที่สูงและเริ่มใช้จากเครือข่ายเล็กๆ เพื่อทดสอบประสิทธิภาพในการวิเคราะห์และระยะเวลาที่ใช้ในการประมวลผลของโปรแกรม

คำสำคัญ – ระบบเครือข่าย; วิเคราะห์ข้อมูล; พัฒนา

1. บทนำ

ที่ผ่านมา ผู้ที่ใช้งานระบบเครือข่ายต่างๆนั้นมักจะพบเจอกับปัญหาระหว่างการใช้งานระบบเครือข่าย เช่น การส่งข้อมูลล่าช้า การส่งข้อมูลไปไม่ถึงปลายทาง ระบบเครือข่ายล่ม และอื่นๆอีกมากมาย โดยปัญหาอาจเกิดขึ้นจากตัวเครือข่ายเอง หรืออาจเกิดจากตัวอุปกรณ์ที่มีความเสียหาย จึงทำให้ผู้ที่ใช้งานระบบเครือข่ายนั้นเกิดความไม่พึงพอใจ และส่งผลกระทบต่อการทำงานในด้านต่างๆได้

ดังนั้น ปัญหาต่างๆที่เกิดขึ้นระหว่างการใช้งานระบบเครือข่าย จึงเป็นปัญหาที่ควรได้รับการแก้ไข เพื่อไม่ให้มีการเกิดขึ้น หรือเกิดขึ้นน้อยที่สุด รวมทั้งควรมีแนวทางในการวางแผนการป้องกันปัญหาให้เกิดผล ทั้งในระยะสั้นและระยะยาว ซึ่งการป้องกันและแก้ไขปัญหาต่างๆนั้น จะช่วยให้ผู้ที่ใช้งานมีความไว้วางใจใน ระบบเครือข่าย อีกทั้งยังช่วยพัฒนาให้ระบบเครือข่ายนั้นมีประสิทธิภาพมากขึ้นอีกด้วย

2. แนวคิดและทฤษฎีระบบการวิเคราะห์และแจ้งเตือนปัญหาเครือข่าย

2.1 แนวคิดในการพัฒนาระบบการวิเคราะห์และแจ้งเตือนปัญหาเครือข่าย

ในการนำ Traffic มาใช้ในกระบวนการวิเคราะห์ จะมีทั้งวิธีที่สามารถดักจับ Traffic แบบ real-time และอีกวิธีคือการดักจับตัว Traffic มาเป็นไฟล์แล้วค่อยนำมาประมวลผลภายในโปรแกรม โดยการทำงานแบบ real-time นั้นจะทำการดักจับ Traffic และนำมาประมวลผลเลยอย่างต่อเนื่อง จะมีการวิเคราะห์และ

สรุปผลการวิเคราะห์เป็นช่วงๆ โดยสามารถกำหนดเวลาในการวิเคราะห์และบันทึกผลได้ ในขณะที่โปรแกรมกำลังทำงานจะมีการเก็บค่าต่างๆไว้เพื่อนำมาทำการวิเคราะห์ อีกการทำงานคือแบบดักจับ Traffic นั้นจะดักจับแล้วเก็บมาเป็นไฟล์แล้วจึงนำมาประมวลผลซึ่งแตกต่างจากวิธีแรก โดยสามารถเลือกการทำงานได้ภายในโปรแกรม ผลลัพธ์ที่ได้ออกมาจะเป็นข้อมูลสรุปค่าดีเลย์และค่าต่างๆออกมาเป็นไฟล์ excel และ pdf หลังจากนั้นจะทำการส่งผลสรุปไปยังอีเมลล์ของผู้ดูแล เมื่อเกิดปัญหาก็จะสามารถส่งแจ้งเตือนไปยังไลน์และทางอีเมลล์ได้ รวมถึงมีกาแจ้งเตือนผ่านทางหน้าจอเพื่อให้ได้รับรู้ถึงปัญหาได้อย่างรวดเร็ว

2.2 Wireshark

Wireshark เป็นอุปกรณ์ซึ่งเป็น free และ open-source ใช้สำหรับแก้ปัญหาระบบเครือข่าย ซอฟต์แวร์ การพัฒนาระบบสัญญาณการสื่อสารและการศึกษา Wireshark คือ program สำหรับวิเคราะห์ packet ใน network สามารถติดตั้งได้หลาย platform ทั้ง Linux, Unix หรือ Window โดยอาศัย pcap ในการจับ packet บน interface ของเครื่อง และมี TShark เป็น command line version สำหรับวิเคราะห์บน Linux และ Unix สำหรับ window มี graphic user interface (GUI) ในการใช้งาน

2.3 Python

Python เป็นภาษาการเขียนโปรแกรมที่ใช้อย่างแพร่หลายในเว็บแอปพลิเคชัน การพัฒนาซอฟต์แวร์ วิทยาศาสตร์ข้อมูล และแมชชีนเลิร์นนิง (ML) นักพัฒนาใช้ Python เนื่องจากมีประสิทธิภาพ เรียนรู้ง่าย และสามารถทำงานบนแพลตฟอร์มต่างๆ ได้มากมาย ทั้งนี้ซอฟต์แวร์ Python สามารถดาวน์โหลดได้ฟรี ผสานการทำงานร่วมกับระบบทุกประเภท และเพิ่มความเร็วในการพัฒนา และเราได้ใช้ Python ในการพัฒนาตัวระบบของเรา และ Python ยังมี Library ต่างๆให้เรียกใช้มากมาย โดยก่อนการใช้งาน Library บางตัวอาจจะต้องมีการติดตั้งก่อนจึงจะสามารถใช้งานได้

2.4 Pyshark

Python wrapper สำหรับ tshark อนุญาตให้แยกวิเคราะห์แพ็กเก็ต python โดยใช้ Wireshark dissectors มีโมดูลการแยกวิเคราะห์แพ็กเก็ตค่อนข้างน้อย โมดูลนี้แตกต่างกันเพราะไม่ได้แยกวิเคราะห์แพ็กเก็ตใด ๆ มันใช้ความสามารถของ tshark (Wireshark command-line utility) เพื่อส่งออก XML เพื่อใช้การแยกวิเคราะห์ แพ็คเกจนี้อนุญาตให้แยกวิเคราะห์จากไฟล์การดักจับหรือการดักจับแบบสด โดยใช้ตัวแยกสัญญาณ Wireshark ทั้งหมดที่ติดตั้งไว้ ทดสอบบน windows/Linux การใช้งาน Pyshark มี object ที่ใช้ capture เช่น file, live, remote โดยอ่านจากแหล่งที่มาจากไฟล์นั้นๆจึงสามารถใช้เป็นตัววนซ้ำเพื่อรับ packet ได้โดย packet ที่ดักจับได้สามารถใช้ตัวกรองต่างๆเพื่อนำมาคัดกรอง protocol ที่ต้องการจะหาได้

2.5 HTTP request-response

HTTP หรือ Hypertext Transfer Protocol คือโพรโทคอลสื่อสารผ่าน internet ใช้ในการรับและส่งข้อมูล ในการแลกเปลี่ยนข้อมูลระหว่าง Client และ Server โดยการส่งจาก Client ไป Server จะเรียกว่า http request ส่วนข้อมูลที่ Server ตอบกลับมาที่ Client จะเรียกว่า http Response

2.6 DNS

มีหน้าที่แปลงชื่อ Domain เป็น IP Address เพื่อให้ง่ายต่อการจดจำ โดยหากไม่มี DNS ผู้ใช้ก็จะต้องจำ IP Address ของเว็บไซต์ ซึ่ง DNS ทำให้ผู้ใช้สามารถเข้าสู่เว็บไซต์ได้สะดวกมากยิ่งขึ้น

2.7 ICMP

เป็นโปรโตคอลเพื่อใช้วินิจฉัยปัญหาเครือข่าย ส่วนใหญ่กำหนดว่าข้อมูลนั้นจะไปถึงปลายทางที่กำหนดไว้ทันหรือไม่ โดยส่วนมากโปรโตคอล ICMP จะใช้กับอุปกรณ์เครือข่าย เช่น router ICMP เป็นสิ่งสำคัญในการทดสอบและรายงานข้อผิดพลาด โดยวัตถุประสงค์หลักของ ICMP คือการรายงานข้อผิดพลาด เมื่ออุปกรณ์สองเครื่องเชื่อมต่อผ่านอินเทอร์เน็ต ICMP จะสร้างข้อผิดพลาดในกรณีที่ข้อมูลไม่ได้ไปในปลายทางที่ต้องการ

2.8 TCP

ใช้ในการเชื่อมต่ออุปกรณ์เครือข่ายบนอินเทอร์เน็ต ทำหน้าที่ควบคุมการรับส่งข้อมูลระหว่างผู้ส่งกับผู้รับ เพื่อใช้แลกเปลี่ยนข้อมูลระหว่างกัน โดยมีการ

ตรวจสอบให้แน่ชัดว่าทุกแพ็คเก็ตที่จัดส่งไปยังปลายทางนั้นเป็นไปตามลำดับที่ถูกต้องตามที่ต้นทางส่งออกมา มีการสร้างการเชื่อมต่อทั้งสองฝั่ง ดังนั้นข้อมูลที่มีการแลกเปลี่ยนนั้นมีความน่าเชื่อถือสูง

3. การออกแบบและพัฒนาระบบการวิเคราะห์และแจ้งเตือนปัญหาเครือข่าย

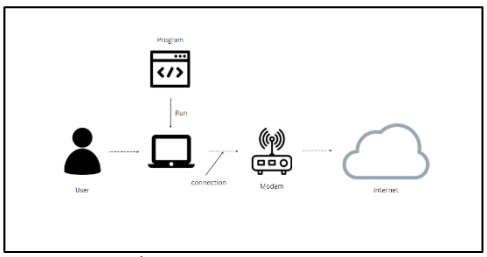
3.1 ความต้องการระบบ

การพัฒนาระบบเริ่มจากสาเหตุปัญหาของระบบเครือข่ายมีปัญหา ซึ่งส่งผลกระทบต่อผู้ใช้งาน ทำให้เกิดปัญหาในการใช้งานระบบเครือข่ายไม่สามารถตอบสนองความต้องการได้อย่างครบถ้วน ดังนั้น จึงเกิดความต้องการที่จะทำการตรวจสอบระบบเครือข่ายเพื่อที่จะสามารถช่วยในการแก้ไขปัญหาได้รวดเร็วและสามารถวิเคราะห์ปัญหาได้

1. ระบบสามารถเรียกดู ใช้ค่าต่างๆ ของ packet ที่ทำการรวบรวมมาวิเคราะห์ เช่นค่าดีเลย์ ขนาดของ packet เป็นต้น
2. สามารถตรวจสอบค่าดีเลย์ และแจ้งเตือนเมื่อเกิดปัญหาได้
3. สามารถทำกราฟบันทึกสถิติของข้อมูลได้
4. สามารถเก็บข้อมูลลงไฟล์ Excel, PDF เพื่อสามารถนำข้อมูลมาวิเคราะห์ภายหลังได้

3.2 ภาพรวมระบบ

ทำการมองภาพรวมของระบบว่าควรที่จะอยู่ในส่วนไหนของเครือข่ายภายในองค์กร และได้ทำ Architecture diagram ออกมาดังรูป

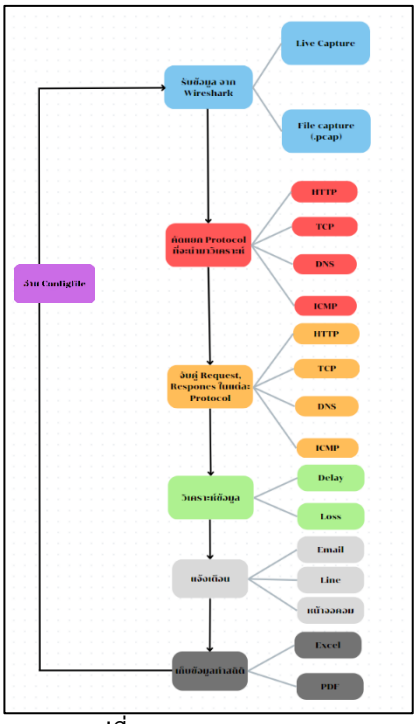


รูปที่ 1 Architecture diagram

โดยจาก Diagram การทำงานจะแบ่งออกเป็น 2 ส่วน คือ ส่วนที่ผู้ใช้งานทำการรัน Wireshark เพื่อรวบรวม Traffic ที่ต้องการวิเคราะห์มี 2 วิธี คือ Live Capture, File Capture และอีกส่วนคือ ทำการรันโปรแกรมเพื่อให้ระบบนำข้อมูลที่ได้มากรองข้อมูลที่ใช้แล้วนำข้อมูลที่ได้ไปวิเคราะห์ และหากมีปัญหาก็จะทำการแจ้งเตือนให้ทราบผ่านทาง Email, Line และคอมพิวเตอร์ที่ทำการรันโปรแกรมอยู่ โดยเราจะมุ่งเน้นไปที่การพัฒนาตัวระบบในส่วนนี้

3.3 ขั้นตอนการทำงานของระบบการวิเคราะห์และแจ้งเตือนปัญหาเครือข่าย

หลังจากที่ได้ออกแบบ Architecture diagram แล้ว ก็จะทำการเขียนขั้นตอนการทำงานต่างๆ กำหนด Feature ของตัวระบบ และแต่ละส่วนจะใช้เครื่องมือหรืออัลกอริที่มอะไร โดยทำเป็น Block diagram เพื่อให้เห็นภาพและเข้าใจง่ายมากยิ่งขึ้น ดังรูป

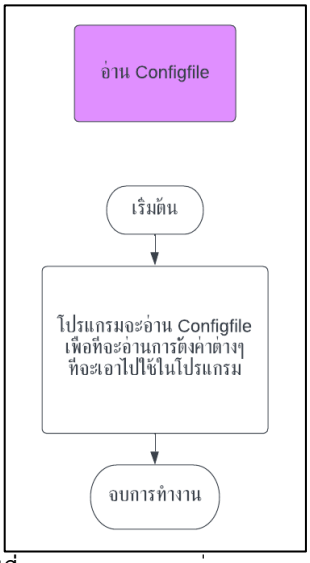


รูปที่ 2 Block diagram

จากรูปจะเห็นได้ว่าการทำงานของระบบ จะเป็นการทำงานแบบวนซ้ำไปเรื่อยๆ โดยส่วนแรกจะมีให้เลือก 2 แบบคือ รับข้อมูลแบบ Live capture และ รับข้อมูลแบบ File capture ซึ่งขั้นตอนการทำงานของตัวระบบเราจะเหมือนกันทั้ง 2 วิธี และจะแบ่งการทำงานออกเป็น 5 ส่วน คือ

1. การอ่าน Configfile

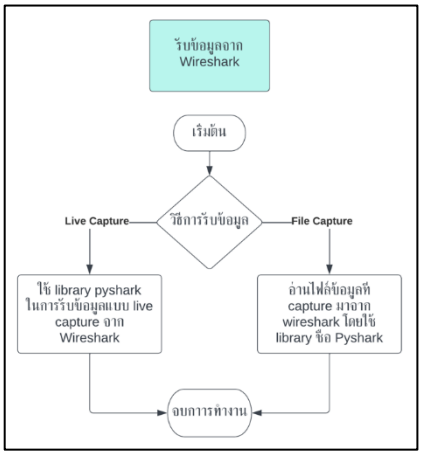
โปรแกรมจะอ่าน Configfile ก่อนเป็นอันดับแรก เพื่อที่จะดูการตั้งค่าต่างๆ ที่จะเอาไปใช้ในโปรแกรม เช่น การรับข้อมูล, Protocol ที่จะเอาไปวิเคราะห์, ชื่อ ที่อยู่ไฟล์ pcap, Networkinterface, Email ที่ใช้ในการส่งแจ้งเตือน



รูปที่ 3 Flowchart การอ่าน Configfile

2. รับข้อมูลจาก Wireshark

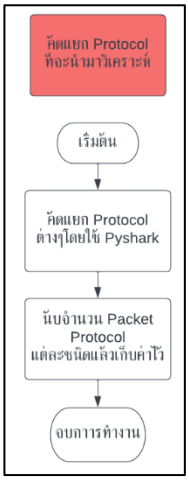
เลือกวิธีรับข้อมูลจาก Wireshark โดยจะใช้ Library ที่ชื่อว่า Pyshark ที่สามารถรับข้อมูลจาก Wireshark มาแบบ Real-time ได้ และสามารถอ่านไฟล์นามสกุล .pcap ที่มาจาก Wireshark ได้



รูปที่ 4 Flowchart การรับข้อมูลจาก Wireshark

3. คัดแยก Protocol ที่จะนำมาวิเคราะห์

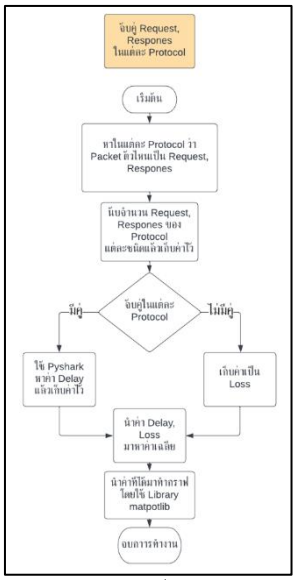
หลังจากที่อ่านไฟล์ได้แล้ว ก็จะทำการคัดPacket โดยทำการเรียกใช้ Pyshark ในการ Filter Protocolต่างๆ และหาว่าตัวไหนที่เป็น Request หรือ Response โดยทำการ เรียกใช้ Pyshark ในการตรวจสอบ แล้วนับจำนวนเก็บค่าเอาไว้



รูปที่ 5 Flowchart คัดแยก Protocol

4. จับคู่ Request และ Response

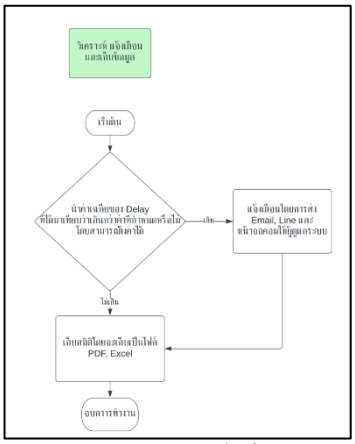
ทำการแยก Request/Response ของแต่ละ Protocol จากนั้นก็จะมาทำการจับคู่ Request Response ของแต่ละ Protocol เพื่อหาค่า 2 ค่า คือ Delay และ ค่า Loss โดยจะทำการเขียนอัลกอริทึ่มขึ้นมาเอง เมื่อจับคู่ได้แล้ว ก็จะเทียบค่าเวลาเพื่อหา Delay และถ้าตัวไหนไม่มีคู่ก็จะนับว่าเป็นค่า Loss จากนั้นทำการหาค่าเฉลี่ย และทำกราฟแสดงค่าที่เก็บเอาไว้ในแต่ช่วงเวลา แล้วเก็บข้อมูลเอาไว้



รูปที่ 6 Flowchart การจับคู่ Request-Response

5. วิเคราะห์ แจ้งเตือน และเก็บข้อมูล

นำค่าที่ได้มาวิเคราะห์ปัญหา โดยจะทำการเขียนอัลกอริที่มเทียบค่า Delay และ loss ที่ได้ว่าเกินกว่าค่าที่กำหนดหรือไม่ ถ้าไม่มีปัญหาก็จะทำกราฟเก็บสถิติเอาไว้ ถ้าเกินกว่าค่าที่กำหนดก็จะทำการแจ้งเตือนไปยังผู้ดูแลระบบโดยจะแจ้งเตือนที่หน้าจอคอม, Line Notify, Email และจะมีการ บันทึกข้อมูลทั้งหมดไว้ใน Excel และ PDF เพื่อให้สะดวกต่อการนำข้อมูลมาตรวจสอบและนำไปวิเคราะห์ต่างๆ เมื่อเสร็จทุกขั้นตอนระบบจะทำการรันไฟล์ใหม่และทำตามขั้นตอนต่อไป



รูปที่ 7 Flowchart การวิเคราะห์ แจ้งเตือน เก็บข้อมูล

4. การทดลองและวิเคราะห์

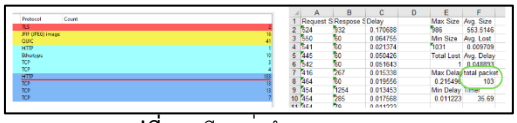
4.1 วิธีการทดลอง

ในระหว่างศึกษาเกี่ยวกับกระบวนการทำงานของโปรแกรม ทำการทดสอบผ่านระบบเครือข่ายภายในบ้าน โดยลักษณะการทดลองจะเป็นการดักจับ packet ภายในเครือข่ายบันทึกเป็นไฟล์หรือทำการดักจับแบบ Realtime ผ่านระบบ หลังจากนั้นทำการทดลองโดยใช้โค้ดในการหาผลลัพธ์ที่ต้องการ ซึ่งจะมีการทดลองด้านความถูกต้อง ด้านประสิทธิภาพ

1. ทดลองด้านความถูกต้องของโปรแกรม

1.1 ความถูกต้องในการคัดกรอง Protocol

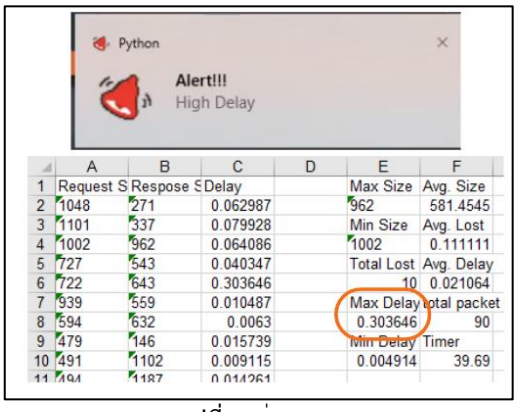
เพื่อที่จะทดสอบว่าโปรแกรมนั้น สามารถคัดกรองได้ถูกต้องหรือไม่ โดยจะทดลองด้วยการ Run โปรแกรมที่ทำเสร็จแล้ว จากนั้น จะนำค่าจำนวน Packet Protocol ที่ได้จากโปรแกรม เทียบกับค่าจำนวน Packet ใน Wireshark และทำการเทียบในทุกๆไฟล์ และเมื่อนำค่าจำนวน Packet Protocol ที่ได้จากโปรแกรม เทียบกับค่าจำนวน Packet ใน Wireshark และทำการเทียบในทุกๆไฟล์แล้ว ได้ผลว่า มีจำนวน Packet ที่เท่ากันในทุกๆไฟล์ จึงได้ข้อสรุปว่า โปรแกรมเรานั้น สามารถคัดแยก Protocol ได้อย่างถูกต้อง ในทุกๆ Protocol



รูปที่ 8 เทียบค่าจำนวน Packet

1.2 ความถูกต้องในการวิเคราะห์และแจ้งเตือนเมื่อพบปัญหาของโปรแกรม

เพื่อที่จะทดสอบว่าโปรแกรมนั้น สามารถวิเคราะห์และแจ้งเตือนเมื่อพบปัญหาได้หรือไม่ ทดสอบโดยการตรวจสอบค่า Delay ในไฟล์ที่ทำการแจ้งเตือนว่าพบปัญหา และตรวจสอบว่าค่าในไฟล์นั้นเกินกว่าที่กำหนดจริงหรือไม่ และเมื่อตรวจสอบค่า Delay ในไฟล์แล้ว ได้ผลว่า ค่า Delay ในไฟล์นั้น เกินกว่าค่าที่กำหนดไว้จริง จึงได้ข้อสรุปว่า โปรแกรมเราสามารถวิเคราะห์และแจ้งเตือนเมื่อเกิดปัญหาได้ถูกต้อง

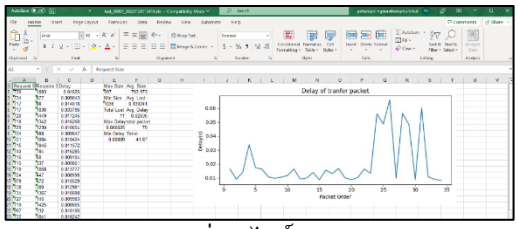


รูปที่ 9 ค่า Delay

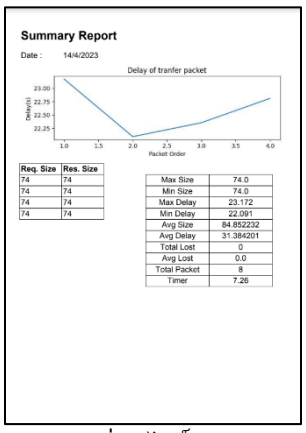
1.3 ความถูกต้องในการนำข้อมูลที่วิเคราะห์มาทำกราฟและเก็บทำสถิติ

เพื่อที่จะทดสอบว่าโปรแกรมนั้น สามารถทำกราฟและทำสถิติได้ถูกต้องหรือไม่ ทดสอบโดย Run โปรแกรมที่ทำเสร็จแล้ว และตรวจสอบข้อมูลภายในไฟล์ Excel และไฟล์ PDF ว่าครบถ้วนหรือไม่ และเมื่อตรวจสอบข้อมูลภายในไฟล์ Excel และ PDF แล้ว ได้ผลว่า โปรแกรมสามารถสร้างไฟล์ Excel และ PDF ได้และสามารถนำข้อมูลที่วิเคราะห์แล้วมาบันทึกลงทั้งใน Excel และ PDF ได้ และสามารถสร้างกราฟและนำมาใส่ในไฟล์

ทั้ง 2 ได้อีกด้วย จึงสรุปได้ว่า โปรแกรมเราสามารถนำข้อมูลที่วิเคราะห์มาทำกราฟและเก็บทำสถิติได้ถูกต้อง



รูปที่ 10 ไฟล์ Excel

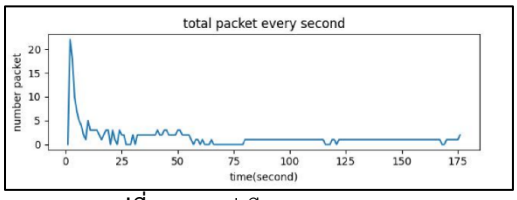


รูปที่ 11 ไฟล์ PDF

2. ทดลองด้านประสิทธิภาพของโปรแกรม

2.1 จำนวน Packet

ทำการทดลอง โดยทำการส่ง packet ไปใน traffic เรื่อยๆและรันโปรแกรมเพื่อให้ระบบประมวลผล packet ที่เข้ามายัง traffic ซึ่งในการทำงานแบบ live เมื่อเวลาเดินไปเรื่อยแล้ว packet เข้ามาทีหลังจะไม่ได้ทำการประมวลผลทันทีเนื่องจากโปรแกรมนั้นยังประมวลผล packet ก่อนหน้าไม่เสร็จ และได้แสดงผลการทดลองออกมาเป็นกราฟดังรูป



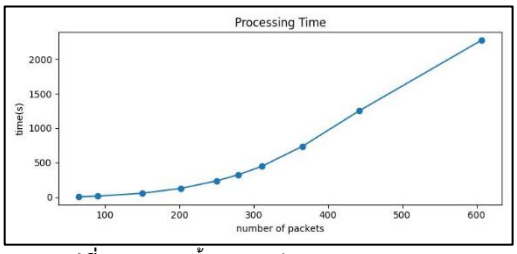
รูปที่ 12 กราฟเทียบ Packet/เวลา

และการทดสอบระยะเวลาในการทำงานของโปรแกรมต่อ packet ที่มี ซึ่งจำนวน packet มีผลต่อระยะการทำงานของโปรแกรมอย่างมาก จากที่เห็นเมื่อโปรแกรมทำงานโดยที่มี 65 packet จะใช้เวลาเพียงแค่ 5 วินาที แต่เมื่อมี 607 packet ซึ่งมากกว่าเดิม 10 เท่า กับใช้เวลาถึง 2278 วินาทีซึ่งมากกว่าเดิมหลายร้อยเท่า

number of packets	time(s)
65	5
90	13
150	55
202	125
250	234

number of packets	time(s)
279	322
311	445
366	738
442	1250
607	2278

รูปที่ 13 ข้อมูลของ Packet/เวลา

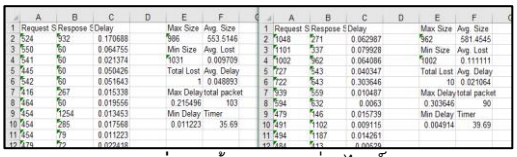


รูปที่ 14 กราฟข้อมูลของจำนวน Packet/เวลา

สรุปได้ว่า ประสิทธิภาพของโปรแกรมขณะที่ระบบทำงานแบบ live capture จะมีความเร็วในการรับ packet มากในช่วงแรก แต่เมื่อเวลาผ่านไปนั้นมีอัตราการประมวลผลที่ช้าลงอย่างเห็นได้ชัด จึงเห็นได้ว่าตัวโปรแกรมนั้นยังมีประสิทธิภาพไม่ดีในการทำงานที่มี packet เป็นจำนวนมาก และการทำงานในรูปแบบรับไฟล์แล้วนำมาประมวลผลก็จะใช้เวลาในการทำงานนานมากขึ้นเมื่อมี packet เป็นจำนวนมาก เนื่องจากโปรแกรมมีประสิทธิภาพที่ไม่ดีนั้น อาจจะผลมาจากตัวโปรแกรม ที่โค้ดนั้นมีการทำงานซ้ำของฟังก์ชันเดิมมากเกินไป เพราะตัวโปรแกรมต้องมีการรันโค้ดรับตัว packet ซ้ำๆและ ประมวลผล packet ซ้ำ อีกทั้งเมื่อมีปัญหาต่อเนื่องติดกันก็จะทำการแจ้งเตือนซ้ำๆ จึงอาจเป็นสาเหตุนึงที่ทำให้ตัวโปรแกรมยังมีประสิทธิภาพที่ไม่ดี

2.2 ขนาดของแต่ละ Packet

ไม่ใช่แค่จำนวน Packet ที่ส่งผลต่อความเร็วในการวิเคราะห์ แต่ยังมีอีกสิ่งหนึ่งที่ส่งผลก็คือ ขนาดของ packet



รูปที่ 12 ข้อมูลของแต่ละไฟล์

จากรูปจะเห็นได้ว่า ถึงแม้ไฟล์รูปทางซ้ายจะมีจำนวน Packet อยู่ที่ 103 Packet มากกว่าไฟล์รูปทางขวาที่มี 90 Packet แต่ขนาดของไฟล์โดยเฉลี่ยแล้วไฟล์รูปทางซ้ายมีขนาดที่น้อยกว่า ก็สามารถทำให้การวิเคราะห์ข้อมูลในไฟล์นั้นทำได้เร็วกว่านั้นเอง ซึ่งการทำงานแบบ live capture ก็มีการทำงานเหมือนกัน ต่างกันแค่วิธีรับข้อมูล จึงสรุปได้เหมือนกัน

2.3 สเปคของเครื่องคอมพิวเตอร์ที่ Run Program

และอย่างสุดท้ายที่ส่งผลต่อประสิทธิภาพในการทำงานและความเร็วในการวิเคราะห์ ก็คือสเปคของเครื่องคอมพิวเตอร์ ซึ่งเราได้ทำการทดสอบโดยให้ Run Program ที่คอม 3 เครื่อง ที่สเปคต่างกัน

Device specifications	Total Packet	264
	Timer	124.2
Device name	DESKTOP-PPJSFMG	
Processor	Intel(R) Core(TM) i3-4130 CPU @ 3.40GHz 3.40 GHz	
Installed RAM	8.00 GB	

รูปที่ 13 ข้อมูลเวลาการประมวลผลเครื่องที่ 1

Device specifications	Total Packet	264
	Timer	326.21
Device name	DESKTOP-0F	
Processor	AMD Ryzen 5 3600 6-Core Processor GHz	3.59
Installed RAM	16.0 GB	

รูปที่ 14 ข้อมูลเวลาการประมวลผลเครื่องที่ 2

Device specifications	Total Packet	264
	Timer	342.57
Device name	DESKTOP-LSNKB68	
Processor	Intel(R) Core(TM) i5-10300H CPU @ 2.50GHz 2.50 GHz	
Installed RAM	8.00 GB (7.79 GB usable)	

รูปที่ 15 ข้อมูลเวลาการประมวลผลเครื่องที่ 3

จากรูปทั้ง 3 จะเห็นว่าสเปคคอมพิวเตอร์ที่ต่างกันนั้น ส่งผลให้เวลาในการประมวลผลต่างกันอย่างเห็นได้ชัด โดยเครื่องที่ประมวลผลได้เร็วที่สุดไปถึงช้าสุด คือ รูปที่ 1, รูปที่ 2 และ รูปที่ 3 ตามลำดับ และจากการทดลองและสังเกตุ สิ่งหลักๆที่ทำให้คอมแต่ละเครื่องประมวลผลไม่เท่ากันคือ การใช้งานหน่วยความจำแบบ SSD , CPU และ RAM ของแต่ละเครื่อง ซึ่งเครื่องที่ 1 ที่เร็วที่สุด จะเห็นว่า Ram และ CPU นั้นจะน้อยกว่าเครื่องที่ 2 แต่เครื่องที่ 1 ใช้ SSD ทำให้มีการประมวลผลที่เร็วกว่าเครื่องที่ 2 และเครื่องที่ 3 ก็ใช้ SSD แต่ CPU ของเครื่องที่ 3 นั้น ถึงจะมี 6-core เท่ากับเครื่องที่ 2 แต่ GHz น้อยกว่ามาก และ RAM ก็น้อยกว่าเครื่องที่ 2 ทำให้มีเวลาในการประมวลผลช้ากว่าไม่มาก

เราจึงสรุปได้ว่า การใช้งานหน่วยความจำแบบ SSD, CPU และ RAM นั้นมีผลต่อเวลาในการประมวลของโปรแกรม

4.2 สรุปผลการทดลอง

จากการทดลองทดสอบโปรแกรมของเราทั้งหมดแล้ว จึงสรุปออกเป็นหัวข้อหลักๆ 2 หัวข้อ คือ

1. ด้านความถูกต้องของโปรแกรม

โปรแกรมสามารถทำงานออกมาได้ถูกต้องและครบถ้วนในทุกขั้นตอนที่เราได้ทดลองมา ไม่มีข้อผิดพลาดในส่วนใดเลย เช่น การรับข้อมูล, การคัดแยก Protocol, การจับคู่ Request-Response, การนำข้อมูลมาวิเคราะห์, การแจ้งเตือน และการบันทึกข้อมูลทำสถิติ ความถูกต้องคิดเป็น 100%

2. ด้านประสิทธิภาพของโปรแกรม

ประสิทธิภาพของโปรแกรมนั้นสามารถทำงานได้ดีในกรณีที่ Packet มีจำนวนน้อย - ค่อนข้างเยอะ และยังทำงานได้ไม่เต็มประสิทธิภาพมากพอในกรณีที่มี Packet เป็นจำนวนเยอะมากๆ และเพื่อการใช้งานโปรแกรมได้เต็มประสิทธิภาพมากขึ้น เราจึงแนะนำสเปคคอมพิวเตอร์ขั้นต่ำในการใช้งานโปรแกรม คือ

2.1 ใช้งานหน่วยความจำแบบ SSD

2.2 CPU 6-Core 3.4 GHz ขึ้นไป

2.3 RAM 16 GB ขึ้นไป

5. สรุปผล

5.1 สรุปผลที่ได้จากการทำโครงงาน

การพัฒนาระบบวิเคราะห์และแจ้งเตือนปัญหาเครือข่ายของเรา ได้มีการทดสอบใช้งานจริงจากเครือข่ายที่บ้านของตัวเอง และการทำงานของระบบเรานั้นสามารถทำงานออกมาได้ตามที่เราวางแผนเอาไว้ เราจึงได้สรุปหัวข้อออกมาดังนี้

1. โปรแกรมสามารถอ่านไฟล์นามสกุล .pcap และสามารถรับข้อมูลแบบ Live Capture จาก Wireshark ได้

2. โปรแกรมสามารถคัดกรอง Protocol ต่างๆได้ ดังนี้ HTTP, .DNS, ICMP และ TCP

3. โปรแกรมสามารถที่จะจับคู่ Packet Request และ Packet Response ในแต่ละ Protocol ได้
3. โปรแกรมสามารถวิเคราะห์และแจ้งเตือนทาง หน้าจอคอมพิวเตอร์, Line Notify และ Email เมื่อพบปัญหาได้
4. โปรแกรมสามารถบันทึกข้อมูลสถิติลงไฟล์ Excel และไฟล์ PDF ได้

การทำงานทั้งหมดที่ได้ในเทอมนี้คิดเป็น 100% ของโครงงานทั้งหมด

โดยสรุปว่า ระบบวิเคราะห์และแจ้งเตือนปัญหาเครือข่ายของเรา สามารถทำงานได้ออกมาตามที่ได้ตั้งเป้าเอาไว้ ทั้งการคัดกรอง Protocol ที่สามารถคัดแยกได้อย่างแม่นยำ การวิเคราะห์ข้อมูลที่สามารถวิเคราะห์ข้อมูลได้อย่างถูกต้อง แล้วเมื่อพบเจอปัญหาระบบก็สามารถทำการแจ้งเตือนได้อย่างทันที และการนำข้อมูลที่วิเคราะห์ได้ไปเก็บเป็นไฟล์ทำสถิติ ก็สามารถทำงานตามที่เราได้ออกแบบเอาไว้ และในส่วนของการที่จะนำระบบไปใช้งานจริงนั้น ระบบเราสามารถที่จะเอาไปใช้งานจริงได้ แต่ประสิทธิภาพของระบบอาจจะยังมีการประมวลผลที่ไม่เร็วพอ ทำให้เครื่องคอมพิวเตอร์ที่จะใช้งานระบบนี้นั้น ต้องมีสเปคที่สูงและยังต้องมีการ Optimize Code เพิ่มเติม เพื่อทำให้ระบบของเรา สามารถทำงานได้อย่างมีประสิทธิภาพมากยิ่งขึ้น ทั้งนี้การทดสอบการทำงานต่างๆ เราได้ทำที่เครือข่ายที่บ้านของเราเท่านั้นและเราคาดหวังว่า โครงงาน การพัฒนาระบบการวิเคราะห์และแจ้งเตือนปัญหาเครือข่ายจะสามารถให้น้องรุ่นหลังได้ทำการศึกษาและพัฒนาต่อๆไป

5.2 ข้อเสนอแนะ

ระบบวิเคราะห์และแจ้งเตือนปัญหาเครือข่ายของเรา สามารถที่จะนำไปใช้งานได้จริงในสภาพแวดล้อมต่างๆ แต่เครื่องคอมพิวเตอร์ที่ใช้งานระบบนี้นั้น ต้องมีสเปคที่สูงกว่าสเปคขั้นต่ำที่เราได้แนะนำไว้ คือ

2.1 ใช้งานหน่วยความจำแบบ SSD
2.2 CPU 6-Core 3.4 GHz ขึ้นไป
2.3 RAM 16 GB ขึ้นไป

เนื่องจาก คอมพิวเตอร์ที่เราใช้ทดลอง ทดสอบระบบนั้น เพียงพอที่จะรับ Packet แค่ในปริมาณที่น้อย - ค่อนข้างมาก และในเครื่อข่ายโดยปกตินั้นมี traffic จำนวนมากทำให้ตัวระบบของเราสามารถนำไปใช้งานจริงได้แต่ประสิทธิภาพจะไม่มากพอที่จะสามารถทำงานในสถานการณ์ที่มี Packet จำนวนมากๆได้

เอกสารอ้างอิง

[1] Sarayut Nonsiri. “ภาษาโปรแกรม Python คืออะไร ?”[Online]. Available: https://www.9experttraining.com/articles/python-%E0%B8%84%E0%B8%B7%E0%B8%AD%E0%B8%AD%E0%B8%B0%E0%B9%84%E0%B8%A3 2022

[2] saixiii. “Wireshark” [Online]. Available: https://saixiii.com/wireshark-sniffer/ 2017

[3] KimiNewt. “Pyshark” [Online]. Available: http://kiminewt.github.io/pyshark/ 2022

[4] Tanabodin Kamol. “BASIC HTTP” [Online]. Available: https://medium.com/icreativesystems/basic-http-3a2b05e5aa19 2019

[5] John Bumgarner. “Pyshark Packet Analysis” [Online]. Available: https://github.com/johnbumgarner/pyshark_packet_analysis/blob/master/pyshark_packet_analysis.py 2020

[6] Muhammad Maisam Abbas. “Python Export to Excel” [Online]. Available: https://www.delftstack.com/howto/python/python-write-to-an-excel-spreadsheet/ 2022

[7] Sven. “Python-charts-in-excel” [Online]. Available: https://github.com/Sven-Bo/python-charts-in-excel/blob/master/PythonCharts_Jupyter_Notebook/PythonChartsInExcel.ipynb 2021

การพัฒนาระบบตรวจสอบความผิดปกติและป้องกันผู้บุกรุก โดยใช้ระบบเครือข่ายแบบที่ถูกกำหนดโดยซอฟต์แวร์

กวิน ลิมะวรารัตน์ และ วุฒิ จารุสุภัทร

อาจารย์ที่ปรึกษาผู้ช่วยศาสตราจารย์ อัครินทร์ คุณกิตติ

คณะเทคโนโลยีสารสนเทศ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

Emails: 62070008@it.kmitl.ac.th, 62070171@ it.kmitl.ac.th, akharin.kh@kmitl.ac.th

บทคัดย่อ

โครงงานนี้เราพัฒนาระบบตรวจสอบความผิดปกติและป้องกันการบุกรุกในระบบเครือข่ายโดยใช้ Software-Defined Network (SDN) ในการปิดกั้นการติดต่อสื่อสาร เพื่อเพิ่มความปลอดภัยภายในเครือข่าย โดยโปรแกรมสามารถรับความผิดปกติประมวลผล และส่งคำสั่งไป SDN Controller เพื่อปิดกั้นการติดต่อสื่อสารได้ พร้อมทั้งมีการส่งแจ้งเตือนไปยัง Email หรือ Line และได้มีการเพิ่มฟังก์ชัน การตรวจจับการส่งคำสั่งไปปิดกั้นการเข้าถึงซ้ำ มี config file ที่ใช้ตั้งค่าโปรแกรม ผลการทดสอบพบว่าระบบสามารถทำงานได้ถูกต้องและมีประสิทธิภาพ โดยระยะเวลาที่ใช้ในการป้องกันเฉลี่ยเท่ากับ 0.4864 วินาที ในการนำไปใช้งานจริง ควรเริ่มจากเครือข่ายเล็ก ๆ เพื่อทดสอบประสิทธิภาพของเครื่องที่ใช้ในการรันระบบ และเวลาที่ใช้ในการป้องกันอาจเพิ่มขึ้นถ้ามีการแยกส่วนการทำงาน

คำสำคัญ – *Software-Defined Network; Intrusion Prevention System; Software-Defined Network Controller*

1. บทนำ

1.1 ที่มาและความสำคัญ

เนื่องจากภัยคุกคามจากอินเทอร์เน็ตและเครือข่ายคอมพิวเตอร์ในปัจจุบันอยู่ในระดับสูง ซึ่งข้อมูลในระบบเครือข่าย หรือการที่ระบบสามารถทำงานได้อย่างปกตินั้นมีความสำคัญต่อเจ้าของระบบและผู้ใช้งานระบบจึงจำเป็นต้องมีการรักษาความปลอดภัยภายในระบบที่สูงตามไปด้วย โดยในการใช้งานระบบรักษาความปลอดภัยของเครือข่ายที่เป็นการจำกัดการเข้าถึงจากภายนอกเครือข่ายเข้าถึงภายในเครือข่ายหรือไฟร์วอลล์ (Firewall) เพียงอย่างเดียวอาจไม่เพียงพอต่อการป้องกันภัยคุกคามจากอินเทอร์เน็ต ผู้จัดทำจึงต้องการนำระบบตรวจสอบความผิดปกติและป้องกันผู้บุกรุก (Intrusion Prevention System) มาใช้งานร่วมกับระบบเครือข่ายแบบที่ถูกกำหนดโดยซอฟต์แวร์ (Software-Defined Network)

โดยระบบตรวจสอบความผิดปกติและป้องกันผู้บุกรุก (Intrusion Prevention System) คือ ซอฟต์แวร์หรือฮาร์ดแวร์ที่ได้รับการออกแบบมาเพื่อให้ตรวจสอบการบุกรุก จะทำงานคล้ายกับระบบตรวจสอบความผิดปกติและผู้บุกรุก (Intrusion Detection System) โดย Intrusion Prevention System จะวิเคราะห์ข้อมูลที่ถูกส่งผ่านภายในเครือข่ายว่ามีความผิดปกติหรือไม่

ประโยชน์ของการนำระบบตรวจสอบความผิดปกติและป้องกันผู้บุกรุก (Intrusion Prevention System) มาใช้งานร่วมกับระบบเครือข่ายแบบที่ถูกกำหนดโดยซอฟต์แวร์ (Software-Defined Network) คือ ระบบตรวจสอบความผิดปกติและป้องกันผู้บุกรุก (Intrusion Prevention System) จะทำการตรวจสอบการบุกรุกหรือความผิดปกติในเครือข่าย และส่วนของระบบเครือข่ายแบบที่ถูกกำหนดโดยซอฟต์แวร์ (Software-Defined Network) จะช่วยในการปิดกั้นการติดต่อสื่อสารหรือระงับความผิดปกติหรือระงับการบุกรุกนั้น ๆ

1.2. วัตถุประสงค์

1) เพื่อพัฒนาระบบตรวจสอบความผิดปกติและป้องกันผู้บุกรุก โดยใช้ระบบเครือข่ายแบบที่ถูกกำหนดโดย

ซอฟต์แวร์เพื่อป้องกันผู้บุกรุกและตรวจสอบความผิดปกติในระบบ

2) เพื่อให้สามารถบันทึกเหตุการณ์ความผิดปกติหรือผู้บุกรุกระบบเครือข่ายที่อาจเกิดขึ้น สามารถตรวจสอบย้อนหลังได้ และสามารถนำข้อมูลไปวิเคราะห์ภัยคุกคามที่อาจเกิดขึ้นภายหลังได้

3) เพื่อเป็นเครื่องมือในการวัดประสิทธิภาพในการป้องการภัยของระบบรักษาความปลอดภัยในเครือข่าย

4) เพื่อเป็นเครื่องมือสำหรับการสืบสวนหาบุคคลที่โจมตีหรือบุกรุกระบบเครือข่าย

2. การทบทวนวรรณที่เกี่ยวข้อง

2.1. Software Defined Network

Software Defined Network คือ สถาปัตยกรรมที่มีการแยกส่วนการทำงานเป็น 2 ส่วน คือส่วน Control plane ที่ใช้ในการตัดสินใจ และ ส่วนของ Data plane ที่ใช้ในการส่ง โดยจะมีซอฟต์แวร์ที่ทำหน้าที่ควบคุมการทำงานต่าง ๆ ของอุปกรณ์ในระบบเครือข่าย ทำให้ง่ายต่อการจัดการระบบเครือข่ายที่มีขนาดใหญ่

2.2. Intrusion Detection System

Intrusion Detection System คือ ระบบที่ใช้ในการตรวจจับการบุกรุกหรือพฤติกรรมบางอย่างที่มีความเสี่ยงต่อระบบ โดยเมื่อตรวจพบถึงสิ่งผิดปกติหรือผู้บุกรุกก็จะทำการแจ้งเตือนกับผู้ดูแลระบบให้รับทราบ

2.3. Intrusion Prevention System

Intrusion Prevention System คือ ระบบที่พัฒนาต่อมาจาก Intrusion Detection System ใช้ในการรักษาความปลอดภัยของเครือข่าย ที่จะเฝ้าสังเกตและตรวจสอบเครือข่าย เพื่อหากิจกรรมที่เป็นอันตรายต่อเครือข่าย และดำเนินการป้องกันความอันตรายเหลานั้น เช่น รายงาน, บล็อคการเข้าถึงเครือข่าย และอื่น ๆ เมื่อมีอันตรายเกิดขึ้น โดยอาจอยู่ในรูปแบบของฮาร์ดแวร์หรือซอฟต์แวร์ก็ได้

2.4. เทคโนโลยีและเครื่องมือที่ใช้ในการพัฒนา

1. Snort

Snort คือ ระบบตรวจสอบความผิดปกติและป้องกันผู้บุกรุกเครือข่ายแบบ open source เป็นระบบที่สามารถตรวจจับการบุกรุกเครือข่ายวิเคราะห์การรับส่งข้อมูลแบบเรียลไทม์ สามารถทำการบันทึก log ของ packet ที่มีการส่งผ่านในเครือข่ายได้ สามารถวิเคราะห์ protocol ที่ใช้ สามารถตรวจจับการโจมตีและ probe ต่าง

2. Mininet

Mininet คือ Network emulator ซึ่งจะทำการจำลองการสร้าง Switch, Host, Controller และ Link ที่ทำการเชื่อมโยงระหว่างอุปกรณ์ที่เราสร้างขึ้นมา โดย Host ที่ Mininet สร้างขึ้นมาจะเป็น Linux host ส่วน Switch ที่ Mininet สร้างขึ้นมาจะรองรับการใช้งาน OpenFlow ดังนั้นมันจึงสามารถรองรับการทำ Customize routing และ SDN ได้

3. Ryu

Ryu คือ framework ที่เป็นตัวควบคุมระบบเครือข่ายที่ถูกกำหนดโดยซอฟต์แวร์ (SDN Controller) ที่ออกแบบมาเพื่อเพิ่มความคล่องตัวของเครือข่ายโดยทำให้ง่ายต่อการจัดการและปรับวิธีจัดการการรับส่งข้อมูล โดยทั่วไปแล้วตัวควบคุมระบบเครือข่ายที่ถูกกำหนดโดยซอฟต์แวร์ (SDN Controller)

4. OpenFlow

OpenFlow เป็น protocol ที่ใช้เป็นมาตรฐานใน Software-Defined Networking โดย protocol นี้ใช้ในการสื่อสารกันระหว่างตัว SDN Controller กับเหล่าอุปกรณ์ภายในเครือข่าย อย่างเช่น switch, router เป็นต้น โดยจะสามารถกำหนดค่าในอุปกรณ์ได้ว่าแพ็คเก็ตที่เข้ามาจะถูกส่งไปที่ใด

5. REST API

REST API เป็นอินเทอร์เฟซ หรือเรียกว่าช่องทางที่ใช้ในการแลกเปลี่ยนสื่อสารข้อมูลกันของแอปพลิเคชันผ่านทางโปรโตคอล HTTP ซึ่งอยู่ในรูปแบบ request และ response โดยจะมี client ที่เป็นฝั่งที่ร้องขอข้อมูลหรือว่าทรัพยากรจากทางฝั่งของ server โดย server ก็จะส่งข้อมูลหรือทรัพยากรตามที่ฝั่ง client นั้นร้องขอ โดยมี HTTP method ที่สำคัญคือ GET, POST, PUT และ DELETE

6. Python

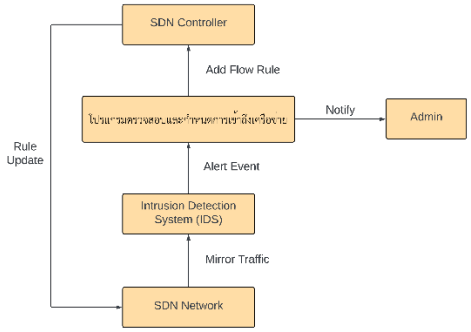
Python คือ ภาษาในการเขียนโปรแกรมระดับสูง ถูกใช้งานอย่างแพร่หลาย มีไวยากรณ์ที่เรียบง่ายและง่ายต่อการเรียนรู้ มีการทำงานแบบ Interpreter สามารถใช้ในการทำงานได้หลากหลายรูปแบบ

7. Line

Line คือ แอปพลิเคชันที่ใช้ในการติดต่อสื่อสารกับบุคคลอื่นโดยผ่ายเครือข่ายอินเทอร์เน็ต สามารถใช้ในการแชท หรือโทรหาผู้อื่นได้

3. การออกแบบระบบ

3.1. ภาพรวมของระบบ

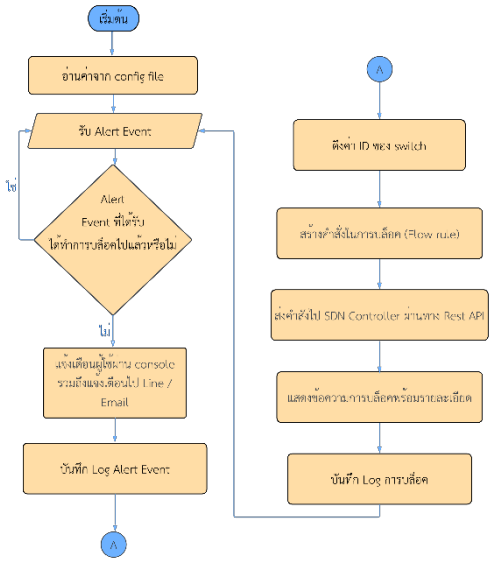


รูปที่ 1. แผนผังการทำงานของระบบ

ในระบบนั้นเป็นการทำงานร่วมกันของแต่ละระบบโดยแบ่งออกเป็น 3 ส่วน คือส่วน Intrusion Detection System ส่วนโปรแกรมตรวจสอบและกำหนดการเข้าถึงเครือข่าย และส่วนของ SDN Controller

SDN Network คือ ระบบเครือข่ายที่ตัวระบบตรวจสอบความผิดปกติและป้องกันผู้บุกรุกโดยใช้ระบบเครือข่ายแบบที่ถูกกำหนดโดยซอฟต์แวร์นั้นทำงานอยู่ โดยตัวของ SDN Network จะมีการ Mirror Traffic ไปยังตัวของ Intrusion Detection System (IDS) คือส่วนที่ใช้ในการตรวจจับสิ่งผิดปกติและผู้บุกรุกที่เกิดขึ้นภายในระบบ IDS โดยจะทำการวิเคราะห์ข้อมูลที่มีการแลกเปลี่ยนกันระหว่างระบบเครือข่ายโดยเมื่อ IDS ตรวจพบการโจมตีหรือความผิดปกติในระบบเครือข่ายจะทำการส่ง Alert Event ไปยังโปรแกรมตรวจสอบและกำหนดการเข้าถึงเครือข่าย โปรแกรมก็จะทำการประมวลผลและส่งคำสั่งในการป้องกันการโจมตีไปยังตัวของ SDN Controller พร้อมทั้งส่งการแจ้งเตือนไปยังผู้ดูแลระบบ โดย SDN Controller เมื่อได้รับคำสั่งก็จะทำการอัปเดตคำสั่งที่ได้รับมาไปยังอุปกรณ์ผ่านในระบบเครือข่ายเพื่อให้เกิดการป้องกันการโจมตีและความผิดปกติที่เกิดขึ้น ในโครงงานนี้เราได้ใช้ตัวโปรแกรม Snort ในการทำหน้าที่ IDS และ Ryu Controller ในการทำหน้าที่ SDN Controller

3.2. แผนผังอัลกอริทึมของโปรแกรมตรวจสอบและกำหนดการเข้าถึงเครือข่าย



รูปที่ 2. แผนผังอัลกอริทึมของโปรแกรมตรวจสอบและกำหนดการเข้าถึงเครือข่าย

ในแผนผังอัลกอริทึมของโปรแกรมตรวจสอบและกำหนดการเข้าถึงเครือข่ายจะอธิบายขั้นตอนการทำงานของโปรแกรม โดยแบ่งเป็น 4 ขั้นตอนหลัก ๆ ดังนี้

1) โปรแกรมจะรับ input คือข้อมูลการแจ้งเตือนถึงความผิดปกติในเครือข่าย หรือ Alert Message ซึ่งประกอบไปด้วย ข้อความแจ้งเตือนความผิดปกติ Source และ Destination IP Address, Protocol, Priority, วันที่และเวลา และอื่น ๆ

2) ตรวจสอบว่าเหตุการณ์ที่ได้รับแจ้งเตือนเข้ามาเป็นเหตุการณ์ซ้ำหรือไม่ โดยมีอัลกอริทึมตรวจสอบ

3) หากไม่จะแสดงข้อความแจ้งเตือนที่หน้า console ของโปรแกรมและเก็บบันทึก log การแจ้งเตือนความผิดปกติ และส่งการแจ้งเตือนไปยัง Line และ Email ตามที่ได้ระบุไว้ใน Config file

4) โปรแกรมจะสร้างคำสั่งและส่งคำสั่ง block traffic ผ่าน REST API โดยคำสั่งจะมีรายละเอียดที่ต้องใช้คือ id ของ switch, ระยะเวลาในการ block, priority, protocol ที่ block และ Source และ Destination IP Address ที่ต้องการ block ส่งไปยัง Ryu Controller เพื่อไปอัปเดต flow table ของ switch ในเครือข่ายเพื่อไม่ให้ผู้โจมตีสามารถโจมตีได้อีก

3.3. การอ่าน Config file

ในการตั้งค่าโปรแกรมจะมีไฟล์ config.ini ที่เป็นไฟล์ที่ใช้ในการ config ค่าข้อมูลต่าง ๆ ของโปรแกรม โดยเราสามารถตั้งค่าและแก้ไขข้อมูลผ่านทางไฟล์นี้ได้เลยไม่จำเป็นต้องแก้ไขผ่านทางโปรแกรม เป็นการเพิ่มความสะดวกให้กับผู้ใช้งานระบบ โดยมี 2 ส่วน คือ 1. ส่วน PROGRAMINFO ที่ใช้ตั้งค่าเกี่ยวกับรายละเอียดของโปรแกรม 2. NOTIFICATION ที่ใช้ในการตั้งค่าเกี่ยวกับการส่งแจ้งเตือนไปยังผู้ใช้งาน

3.4. การรับข้อมูล Alert Event

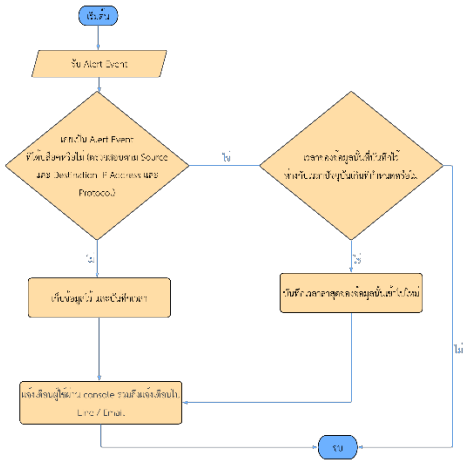
1) รับ Alert Event ผ่าน Unix Domain Socket เป็นการสร้างการเชื่อมต่อกับ socket เพื่อทำการรับข้อมูล โดยมีรูปแบบของการรับ Alert Event เป็น text-based ที่มีโครงสร้างเป็น JSON

2) รับ Alert Event ผ่านการอ่าน log file จะเป็นการไฟล์ทีละบรรทัดและเก็บค่าลงตัวแปรที่ได้มีการสร้างเอาไว้เพื่อนำไปใช้ โดยมีรูปแบบของการรับ Alert Event เป็น text-based ที่มีรูปแบบ plain text

3.5. การแจ้งเตือนเมื่อตรวจพบการโจมตีระบบและผู้บุกรุก

โดยเมื่อระบบได้ตรวจพบการโจมตีและผู้บุกรุกจะมีการแจ้งเตือนผ่านทางหน้าต่างโปรแกรมรวมไปถึงการแจ้งเตือนไปยัง Email และ แอปพลิเคชัน Line เพื่อให้เจ้าของระบบรับรู้ถึงการโจมตีระบบที่เกิดขึ้น ณ เวลานั้น

3.6 การตรวจสอบการบล็อคซ้ำ



รูปที่ 3. แสดงอัลกอริทึมการตรวจสอบการบล็อคซ้ำ

กลไกในการตรวจสอบว่าเป็นการบล็อคการเข้าถึงซ้ำหรือไม่ โดยหากทำการบล็อคแล้วจะมีการบันทึกเวลาไว้ และหากจะมีการบล็อคอีกครั้ง จะตรวจสอบเวลาที่บันทึกไว้ว่าเกินที่กำหนดหรือไม่ หากเกินเวลาที่กำหนด จะส่งคำสั่งไปบล็อคการเข้าถึงอีกครั้ง

3.7. การป้องกันเมื่อตรวจพบการโจมตีระบบและผู้บุกรุก

เมื่อเราได้รับ Alert Event จาก IDS ตัวโปรแกรมก็จะทำตรวจสอบการบล็อตซ้ำก่อน ถ้าไม่ใช่ก็จะทำการ mapping ข้อมูลต่าง ๆ ของผู้โจมตีระบบหรือผู้บุกรุกตาม Alert Event ที่ได้รับมา แล้วนำมาสร้างเป็นคำสั่ง flow rule ที่จะใช้ในการส่งไปยังตัว SDN Controller เพื่อทำการบล็อคการเข้าถึงของผู้โจมตีระบบและผู้บุกรุกผ่านทาง Rest API

3.8. การบันทึก log

ในการบันทึก log ของโปรแกรมตรวจสอบและกำหนดการเข้าถึงเครือข่ายจะเป็นการบันทึก log ลงไปในไฟล์ตาม path ที่กำหนดไว้ โดยมีการบันทึก 2 ส่วนด้วยกันคือ 1. บันทึก log ของเหตุการณ์ที่ตรวจพบ และ 2. บันทึก log การบล็อค

4. การทดสอบ

4.1. การเตรียมการทดสอบ

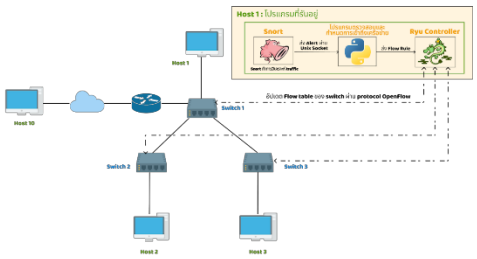
1) รายละเอียดเครื่อง Virtual Machine ที่ใช้ในการทดลอง

- Ubuntu 18.04.6 LTS, Processor 1 core, RAM 2 GB, Storage 50 GB

2) โปรแกรมที่ใช้ในการทดลอง

- Mininet, Snort, Ryu

3) รายละเอียด topology ที่ใช้ในการทดลอง



รูปที่ 4. Topology ที่ใช้ในการทดลอง

4.2. การทดสอบความถูกต้อง การทำงานของระบบ (Functional Testing)

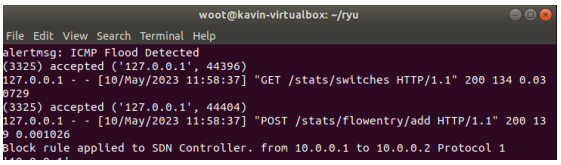
วิธีการทดสอบ ทดสอบการทำงานเป็นแต่ละส่วน 5 ส่วน คือ การรับข้อมูล Alert จาก Snort ส่งไปยังโปรแกรมตรวจสอบและกำหนดการเข้าถึงเครือข่าย ทั้งในรูปแบบ UNIX DOMAIN SOCKET และการอ่าน log file, การบล็อคการเข้าถึง, การตรวจจับบล็อคซ้ำ และการแจ้งเตือนเมื่อตรวจพบความผิดปกติ โดยจะทดสอบการโจมตีระบบโดยการ Generate Traffic ที่เป็น ICMP Flood และอีกส่วนจะทดสอบการทำงานโดยรวมของระบบ โดยจำลองการโจมตีระบบโดย Generate Traffic ในรูปแบบ ICMP Flood

1) การรับข้อมูล Alert Message จาก Snort ส่งไปยังโปรแกรมตรวจสอบและกำหนดการเข้าถึงเครือข่าย

เมื่อ Snort ตรวจจับความผิดปกติในระบบตาม Rules ที่ได้เขียนเพื่อกำหนดไว้ จะส่งข้อมูลไปยังโปรแกรมตรวจสอบและกำหนดการเข้าถึงเครือข่ายโดยการรับข้อมูลจาก snort ทำได้ 2 รูปแบบคือ

1. การรับข้อมูล Alert Message ผ่าน UNIX DOMAIN SOCKET

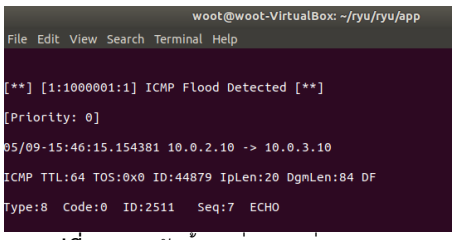
ผลการทดลอง สามารถรับข้อมูล Alert Message ผ่านทาง UNIX DOMAIN SOCKET ได้ถูกต้องตามที่ตรวจพบความผิดปกติ โดยสังเกตจาก Source IP address Destination IP address และ protocol ที่แสดงออกมาทางหน้าจอโปรแกรมมีความถูกต้องตามที่เราได้จำลองขึ้นมา



รูปที่ 5. การรับข้อมูลผ่าน UNIX DOMAIN SOCKET

2. การรับข้อมูล Alert Message ผ่านการอ่าน log ที่ได้บันทึกไว้

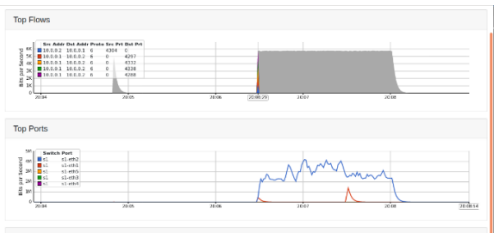
ผลการทดลอง สามารถรับข้อมูล Alert Message จากการอ่าน log file ได้ถูกต้องตามที่ตรวจพบความผิดปกติ



รูปที่ 6. การรับข้อมูลผ่านการอ่าน log file

2) การบล็อคการเข้าถึง

ผลการทดสอบ เมื่อดูจากรูปภาพกราฟ traffic ของระบบ และ การใช้คำสั่งเพื่อเรียกดู flow ของ switch จะเห็นได้ว่าตัวโปรแกรมตรวจสอบและกำหนดการเข้าถึงเครือข่ายนั้นได้มีการสร้าง flow rule ไปให้ SDN Controller เพื่อทำการบล็อค IP address ของผู้โจมตีได้อย่างถูกต้องตามระยะเวลาที่กำหนดไว้คือ 60 วินาที

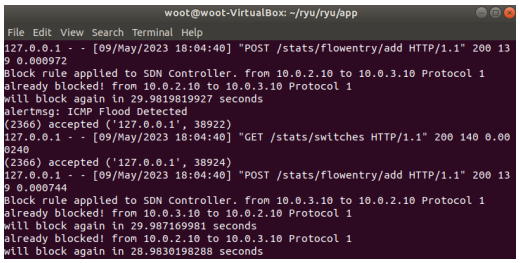


รูปที่ 7. หน้ากราฟ traffic ในระบบ

3) การตรวจจับการบล็อคซ้ำ

ผลการทดสอบ สามารถตรวจจับการบล็อคซ้ำได้ถูกต้องตาม Source และ Destination IP Address

และ Protocol และกำหนดเวลาตามที่ได้เขียนโปรแกรมไว้

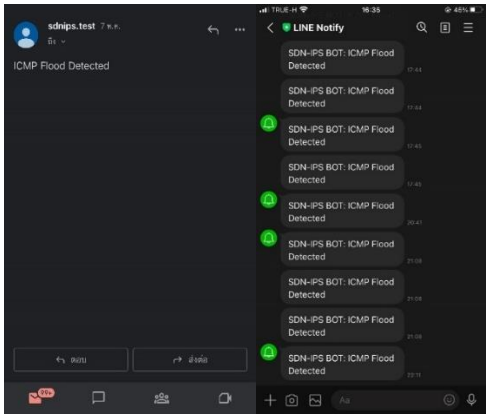


รูปที่ 8. การตรวจจับการบล็อคซ้ำ

4) การแจ้งเตือนเมื่อตรวจพบความผิดปกติ

หลังจากที่ตัวโปรแกรมตรวจสอบและกำหนดการเข้าถึงเครือข่าย ตรวจพบความผิดปกติที่เกิดขึ้นในระบบ ระบบต้องมีการส่งการแจ้งเตือนไปหาผู้ดูแลระบบผ่านทาง Email และ แอปพลิเคชัน Line ได้ทันทีหลังจากตรวจพบ

ผลการทดสอบพบว่าสามารถทำการแจ้งเตือนเมื่อตรวจพบความผิดปกติได้ถูกต้อง



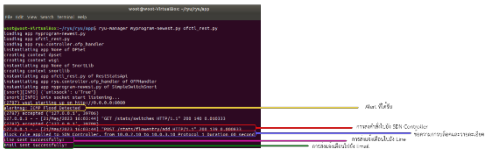
รูปที่ 9. การแจ้งเตือนผ่านทาง Email และ Line Notify

5) การทดสอบการทำงานโดยรวมทั้งระบบ โดยจำลองการโจมตีระบบโดย Generate Traffic ในรูปแบบ ICMP Flood

เป็นการจำลองเหตุการณ์ว่ามี host หนึ่งได้ทำการ DDoS Attack ไปยังอีก host โดย Generate Traffic ในรูปแบบ ICMP Flood ในเครือข่ายเกิดขึ้น โดยหลังจากที่การโจมตีเกิดขึ้น ระบบสามารถตรวจจับเหตุการณ์ DDoS Attack ที่เกิดขึ้นได้ถูกต้อง และ ได้ทำการบล็อค IP Address นั้นทันทีโดยไม่มีการส่งคำสั่งไปบล็อคซ้ำ ๆ หากยังไม่เลยระยะเวลาที่ตัวโปรแกรมนั้นกำหนดเอาไว้ พร้อมทั้งมีการส่งข้อความแจ้งเตือนไปยัง Email และแอปพลิเคชัน Line ที่กำหนดเอาไว้ใน Config file และบันทึกการทำงานของโปรแกรมลง log file ได้อย่างถูกต้อง

```
mininet> h2 hping3 h3 --icmp --faster
```

รูปที่ 10. คำสั่งที่ใช้ในการทดสอบการทำงานโดยรวมทั้งระบบ โดยจำลองการโจมตีระบบโดย Generate Traffic ในรูปแบบ ICMP



รูปที่ 11. การทำงานตามลำดับของโปรแกรมตรวจสอบและกำหนดการเข้าถึงเครือข่าย เมื่อมีความผิดปกติแบบ ICMP Flood

4.3. การทดสอบความเร็วในการทำงาน (Performance Testing)

1) ทดสอบความเร็วในการป้องกันการเข้าถึงเครือข่าย

ความเร็วในการป้องกันการเข้าถึงเครือข่าย จะคิดจากการนำเวลาในการทำงาน 3 ส่วน คือ เวลาในการทำงานของ Snort เวลาในการทำงานของโปรแกรมตรวจสอบและกำหนดการเข้าถึงเครือข่าย และเวลาในการทำงานของ Ryu Controller มารวมกันเป็นเวลาทั้งหมดที่ใช้ในการป้องกันการเข้าถึงเครือข่าย และมีรูปแบบในการทดสอบ 5 รูปแบบ คือหาเวลาที่ใช้ในการป้องกันการเข้าถึงเครือข่าย โดยมีการโจมตีเครือข่าย คือ Generate ICMP, TCP และ UDP Flood Traffic และการเข้าถึงเครือข่ายโดยไม่ได้รับอนุญาต คือ FTP และ HTTP Attemption โดยจะทดสอบรูปแบบละ 5 ครั้ง และนำเวลาที่ได้มาหาค่าเฉลี่ย

ผลการทดสอบเวลาที่ใช้ในการป้องกันการโจมตีเครือข่ายโดยการ Generate ICMP, TCP และ UDP Traffic

ตารางที่ 1. ตารางบันทึกระยะเวลาเฉลี่ยที่ใช้ในการป้องกันการเข้าถึงเครือข่ายจากการโจมตีโดยการ Generate ICMP, TCP และ UDP Traffic

โปรโตคอล	เวลาในการทำงานของ Snort (วินาที)	เวลาในการทำงานของโปรแกรมตรวจสอบและกำหนดการเข้าถึงเครือข่าย (วินาที)	เวลาในการทำงานของ Ryu Controller (วินาที)	เวลาทั้งหมดที่ใช้ในการป้องกัน (วินาที)
ICMP	0.3439	0.106	0.0322	0.4821
TCP	0.4244	0.1184	0.0206	0.5634
UDP	0.1366	0.0628	0.0226	0.222

ผลการทดสอบเวลาที่ใช้ในการป้องกันการการเข้าถึงเครือข่ายไม่ได้รับอนุญาตโดยการ FTP และ HTTP Attempt เฉลี่ย

ตารางที่ 2. ตารางบันทึกระยะเวลาเฉลี่ยที่ใช้ในการป้องกันการเข้าถึงเครือข่ายจากการเข้าถึงโดยไม่ได้รับอนุญาตแบบ FTP และ HTTP Attempt

โปรโตคอล	เวลาในการทำงานของ Snort (วินาที)	เวลาในการทำงานของโปรแกรมตรวจสอบและกำหนดการเข้าถึงเครือข่าย (วินาที)	เวลาในการทำงานของ Ryu Controller (วินาที)	เวลาทั้งหมดที่ใช้ในการป้องกัน (วินาที)
FTP	0.7015	0.026	0.016	0.7435
HTTP	0.3866	0.01976	0.0146	0.4210

จากผลการทดสอบความเร็วในการป้องกันการเข้าถึงเครือข่ายทั้งการโจมตีผ่านการ Generate ICMP, TCP และ UDP Flood Traffic และการเข้าถึงโดยไม่ได้รับอนุญาตแบบ FTP และ HTTP Attempt พบว่า เวลาเฉลี่ยที่ใช้ในการป้องกันการเข้าถึงเครือข่ายทั้ง 5 รูปแบบเฉลี่ยอยู่ที่ 0.4864 วินาที

2) ทดสอบความเร็วในการแจ้งเตือนไปยัง Email และ Line

ความเร็วที่ใช้ในการแจ้งเตือนไปยัง Email และ Line คิดจากการจับเวลานับตั้งแต่ได้รับ Alert จนส่งคำสั่งให้ไปแจ้งเตือน โดยจะมีการจับเวลา ทั้งเวลาที่ใช้ในการแจ้งเตือนไปยัง Email และ Line และจะทดสอบรูปแบบละ 5 ครั้ง และนำเวลาที่ได้มาหาค่าเฉลี่ย

ผลการทดสอบเวลาที่ใช้ในการส่งแจ้งเตือนไปยัง Email และ Line

ตารางที่ 3. ตารางบันทึกความเร็วในการแจ้งเตือนไปยัง Email และ Line

ครั้งที่	ระยะเวลาที่ใช้ในการแจ้งเตือนไปยัง Email (วินาที)	ระยะเวลาที่ใช้ในกาแจ้งเตือนไปยัง Line (วินาที)
1	5.3170	0.5001
2	3.8317	0.6409
3	3.7821	0.6818
4	3.6702	0.5658
5	4.7182	0.4864
เฉลี่ย	4.0518	0.575

จากผลการทดสอบความเร็วที่ใช้ในการส่งแจ้งเตือนไปยัง Email และ Line พบว่า พบว่า เวลาเฉลี่ยที่ใช้ในการส่งแจ้งเตือนไปยัง Email เท่ากับ 0.575 วินาที และเวลาเฉลี่ยที่ใช้ในการส่งแจ้งเตือนไปยัง Line อยู่ที่ 4.0518 วินาที

4.4. สรุปผลการทดสอบ

การทดสอบระบบในครั้งนี้ แบ่งการทดสอบออกเป็น 2 ส่วน คือ การทดสอบด้านความถูกในการทำงานของระบบ และการทดสอบความเร็วในการทำงาน ในการทดสอบด้านความถูกต้องในการทำงานของระบบในครั้งนี้สามารถตรวจจับความผิดปกติที่เกิดขึ้นใน

เครือข่าย บล็อคการเข้าถึง และทำฟังก์ชันต่าง ๆ ได้อย่างถูกต้อง 100% ตามเป้าหมายและตามลำดับที่ออกแบบไว้ ส่วนในด้านความเร็วในการทำงาน ความเร็วในการป้องกันการโจมตีเฉลี่ยจากทั้ง 5 รูปแบบอยู่ที่ 0.4864 วินาที โดยมีระยะเวลาที่ใช้ในการป้องกันน้อยที่สุดคือ 0.1503 วินาที ระยะเวลามากที่สุดที่ใช้ในการป้องกันคือ 0.9927 วินาที โดยจากการแบ่งส่วนเวลาในการทำงานของระบบเป็น 3 ส่วน เวลาในการทำงานของ Snort ที่ใช้ในการตรวจจับความผิดปกติภายในเครือข่ายนั้นใช้เวลามากที่สุด ส่วนการส่งแจ้งเตือนไปยัง Line นั้นสามารถทำได้รวดเร็วกว่าการส่งไปยัง Email พอสมควร โดยผู้จัดทำคาดว่าที่การส่ง Email นั้นช้าเป็นเพราะการส่ง Email ต้องมีการสร้าง SMTP connection ก่อนซึ่งกินระยะเวลาพอสมควรในจุดนั้น

5. วิคราะห์และสรุปผล

5.1. สรุปผลการดำเนินงาน

ระบบตรวจสอบความผิดปกติและป้องกันผู้บุกรุก โดยใช้ระบบเครือข่ายแบบที่ถูกกำหนดโดยซอฟต์แวร์ สามารถทำงานได้ตามที่ตั้งเป้าเอาไว้ โดยระยะเวลาในการป้องกันการโจมตีเฉลี่ยอยู่ที่ 0.4864 วินาที และในแต่ละฟังก์ชันสามารถทำงานได้ถูกต้อง ทั้งการตรวจจับความผิดปกติและบล็อคการเข้าถึง ที่สามารถบล็อคได้ถูกต้องและแม่นยำตามที่ได้เขียนโปรแกรม การส่งการแจ้งเตือนความผิดปกติไปยัง Email และ Line ได้อย่างถูกต้อง และอัลกอริทึมภายในโปรแกรมที่เขียนขึ้นเพื่อตรวจสอบการแจ้งเตือนหรือบล็อคการซ้ำ ก็สามารถทำงานได้อย่างถูกต้องตามที่ต้องการ อีกทั้งการเพิ่มการรับการแจ้งเตือนผ่าน Snort ด้วยการอ่าน log file ก็สามารถทำได้ โดยอาจสามารถนำไปปรับใช้กับการอ่าน log file จากโปรแกรมตัวสอบความผิดปกติอื่น ๆ ได้ และในการพัฒนาระบบนี้ขึ้นมา ยังคงเป็นการทำผ่าน Virtual Machine เท่านั้น

5.2 ปัญหาและอุปสรรคที่พบ

ในการพัฒนา Command Line Interface เพื่อจะให้ผู้ใช้สามารถพิมพ์คำสั่งเพื่อให้ผู้ใช้งานสามารถใช้ฟังก์ชันปลดบล็อคได้นั้นยังเกิดปัญหาเมื่อนำมาทำงานร่วมกับโปรแกรมที่พัฒนาขึ้น

5.3. ข้อเสนอแนะและแนวทางในการพัฒนาในอนาคต

- พัฒนา Command Line Interface หรือ Graphic Use Interface เพื่อให้ผู้ใช้งานมีความสะดวกมากขึ้น
- พัฒนาให้โปรแกรมที่พัฒนาขึ้นสามารถใช้ร่วมกันกับโปรแกรมตรวจสอบความผิดปกติในระบบเครือข่ายที่หลากหลาย และ SDN-Controller ที่มีความหลากหลายมากยิ่งขึ้น

เอกสารอ้างอิง

[1] Cisco and/or its affiliates. “Snort User Manual 2.9.16”. [Online]. Available : https://snort-org-site.s3.amazonaws.com/production/document_files/files/000/000/249/original/snort_manual.pdf. 2020.

[2] Mininet Project Contributors. “Mininet Walkthrough”. [Online]. Available : http://mininet.org/walkthrough/. 2022.

[3] Nippon Telegraph and Telephone Corporation Revision d6cda4f4. “Snort Integration”. [Online]. Available : https://ryu.readthedocs.io/en/latest/snort_integrate.html. 2014.

[4] SDxCentral, LLC. “What Is a Ryu Controller?”. [Online]. Available : https://www.sdxcentral.com/networking/sdn/definitions/what-the-definition-of-software-defined-networking-sdn/what-is-sdn-controller/openflow-controller/what-is-ryu-controller/. 2023.

[5] Joska de Langen. “Sending Emails With Python”. [Online]. Available : https://realpython.com/python-send-email/. 2023.

[6] LINE Corporation. “LINE Notify”. [Online]. Available : https://notify-bot.line.me/en/. 2023.

ระบบบริหารจัดการสภาพแวดล้อมส่วนตัวในการฝึกปฏิบัติในการเรียนการสอนทางด้านระบบคลาวด์

กฤตณัฏฐ์ ศิริพรนพคุณ[1] และ ธัชวุฒิ วิจิตรบรรจง[2]

คณะเทคโนโลยีสารสนเทศ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง กรุงเทพฯ

Emails: [1]62070002@kmitl.ac.th, [2]62070091@kmitl.ac.th

บทคัดย่อ

ปัจจุบันการลงทุนในเครือข่ายคอมพิวเตอร์ และคอมพิวเตอร์แม่ข่ายมีอัตราลดลงเนื่องจากค่าใช้จ่ายสูง และความเสี่ยงของอุปกรณ์ที่อาจไม่รองรับการใช้งานใหม่ ๆ ทำให้บริการ Cloud Computing เช่น Amazon Web Services (AWS) รับความนิยมมากขึ้น เนื่องจากสามารถให้บริการเครือข่ายคอมพิวเตอร์และคอมพิวเตอร์แม่ข่ายโดยไม่ต้องลงทุนมาก สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบังได้เห็นความสำคัญต่อแนวโน้มของทักษะทางด้านนี้จึงเปิดสอนหลักสูตร Cloud Computing โดยส่วนใหญ่ได้ใช้เนื้อหาจาก AWS Academy แต่เนื้อหาสำหรับการเรียน ไม่สามารถแก้ไขให้ตรงตามความต้อง และวัตถุประสงค์ในการเรียนได้ ผู้จัดทำจึงพัฒนาระบบสำหรับการเรียนการสอนที่รองรับการแก้ไขและปรับปรุงเนื้อหาให้ทันสมัย โดยระบบนี้ยังสามารถแยกสิ่งแวดล้อมการฝึกปฏิบัติของนักเรียนแต่ละคนได้เป็นอิสระต่อกัน สามารถช่วยในการจัดการค่าใช้จ่ายในการเรียนการสอนได้อย่างมีประสิทธิภาพ สามารถปิดการใช้งานเมื่อจบภาคเรียนการศึกษา และนำมาเปิดใช้ใหม่เมื่อต้องการ และสามารถทำงานได้บนสภาพแวดล้อมที่หลากหลายได้

คำสำคัญ – *Cloud Technology; Online Learning; Amazon Web Services;*

1. บทนำ

ในการจัดการเรียนการสอนทางการปฏิบัติด้าน Cloud Computing ปัจจุบันทางสถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบังมีการใช้ระบบในการฝึกปฏิบัติของ AWS Academy ทำสามารถใช้บริการสร้างสภาพแวดล้อมสำหรับการใช้งาน Amazon Web Service ในการฝึกปฏิบัติได้ แต่มีข้อจำกัดที่ผู้ฝึกสอนไม่สามารถปรับแต่งขอบเขต และเนื้อของการฝึกปฏิบัติในแต่ละหัวข้อให้สอดคล้องกับเนื้อหาในแต่ละรายวิชา หรือปรับปรุงเนื้อหาให้เป็นปัจจุบันได้ เนื่องจากเนื้อหาภายในระบบฝึกปฏิบัติของ AWS Academy เป็นเนื้อหาที่ถูกกำหนดโดย Amazon Web Service การปรับเปลี่ยนรูปแบบการฝึกปฏิบัตินั้นตัวผู้ใช้งานไม่มีสิทธิ์ที่จะปรับเปลี่ยนได้เอง ทำให้ไม่สามารถปรับแต่งรูปแบบการฝึกปฏิบัติให้มีความสอดคล้องกับการปรับปรุงระบบที่เกิดขึ้นอย่างรวดเร็วในการให้บริการทางด้าน Cloud Computing ในปัจจุบัน

ทางผู้จัดทำจึงได้พัฒนาระบบสำหรับการเรียนการสอนทางด้าน Cloud Computing บน Amazon Web Services ที่สามารถแยกสภาพแวดล้อมที่ใช้ในการฝึกปฏิบัติของนักเรียนแต่ละคนออกจากกันได้โดยสิ้นเชิง และนอกจากนี้ตัวระบบที่พัฒนาจะรองรับการแก้ไข และปรับปรุงเนื้อหาของอาจารย์ผู้สอน เพื่อให้สอดคล้องกับเทคโนโลยีที่มีการปรับเปลี่ยนตลอดเวลา

2. การทบทวนวรรณกรรมที่เกี่ยวข้อง

2.1 Cloud Computing

เป็นระบบคอมพิวเตอร์ หรือเครือข่ายของคอมพิวเตอร์ โดยจะเป็นระบบคอมพิวเตอร์สำหรับให้บริการสำหรับผู้ที่ต้องการเช่า และใช้งานได้ตามความต้องการของผู้ใช้ โดยที่ผู้ใช้ไม่จำเป็นต้องมีการติดตั้งอุปกรณ์ใด ๆ ในการใช้งานคอมพิวเตอร์ และสามารถจัดสรรทรัพยากรได้ตามความต้องการ หรือตามความต้องการใช้งาน

2.2 Amazon Web Services

Amazon Web Services หรือ AWS เป็นผู้ให้บริการแพลตฟอร์ม Cloud Computing ที่นำเสนอบริการมากกว่า 200 บริการจากศูนย์ข้อมูลต่าง ๆ ทั่วโลก โดยในโครงการนี้ได้มีการเลือกใช้งานเทคโนโลยีของ AWS ดังนี้

2.2.1. AWS Lambda

Lambda เป็นบริการในรูปแบบ Serverless ของ AWS ที่จะเรียกใช้โปรแกรมที่เขียนไว้เพื่อตอบสนองหรือทำงาน จากการกระตุ้นโดยกิจกรรมที่กำหนดไว้ โดยที่ผู้ใช้งานไม่ต้องจัดเตรียมระบบแม่ข่ายด้วยตนเอง โดยมีภาษาคอมพิวเตอร์ที่รองรับการทำงานหลากหลายภาษา

2.2.2. Amazon SQS

Amazon Simple Queue Service (SQS) เป็นบริการจัดส่งข้อความ ตัวกลางที่รับ-ส่งข้อมูลระหว่างบริการต่าง ๆ หรือโปรแกรมที่สร้างขึ้น ลดความซับซ้อนในการจัดการขั้นตอนการทำงาน และการส่งข้อมูลระหว่างการทำงานของโปรแกรมโดยไม่มีข้อจำกัดเรื่องขนาด และปริมาณข้อความที่ส่ง ยังรองรับการทำงานจัดลำดับข้อความ (Queue) ไปยังบริการอื่น ๆ ด้วย

2.2.3. Amazon SDK

AWS SDK เป็นชุดเครื่องมือคำสั่งบนภาษาคอมพิวเตอร์ ที่ใช้ในการพัฒนาโปรแกรมที่จะทำงานร่วมกับบริการต่าง ๆ บน AWS ได้สะดวก และง่าย โดยปกติจะมีชุดคำสั่งเพื่อเชื่อมต่อ และใช้งานฟังก์ชันของบริการต่าง ๆ โดยอัตโนมัติ

2.2.4. AWS Organizations

AWS Organizations ทำให้ผู้ดูแลระบบสามารถจัดการบัญชี AWS หลายๆบัญชีได้จากศูนย์กลาง เพื่อควบคุม และกำหนดสิทธิ์การเข้าถึง AWS Services ของแต่ละบัญชีได้ โดยสามารถสร้างบัญชี และกลุ่มของผู้ใช้รวมทั้งการปรับใช้ Policy ให้กับบัญชีเหล่านั้นได้

2.2.5. AWS CloudFormation

AWS Cloud Formation เป็นบริการของ AWS ที่อยู่ในรูปแบบ Infrastructure as Code ที่ช่วยสร้างโมเดล จัดเตรียม และจัดการทรัพยากรของ AWS ได้อย่างง่ายดาย ผ่านชุดแม่แบบที่สร้างขึ้นมา

2.3 Java Script

JavaScript เป็นภาษาคอมพิวเตอร์แบบ Scripting language ที่ออกแบบมาสำหรับใช้งานในการพัฒนาการโต้ตอบบนเว็บไซต์ ภายหลังมีการพัฒนาชุดโปรแกรม Node.js ที่มีการใช้ภาษา JavaScript ในการทำงานบนเครื่องคอมพิวเตอร์ภายนอกเว็บเบราว์เซอร์เหมือนภาษาโปรแกรมอื่น ๆ

2.3.1. Next.js framework

Next.js เป็นเฟรมเวิร์กใช้สำหรับการสร้างเว็บไซต์ หรือเว็บแอปพลิเคชันที่สามารถใช้งานได้ง่าย มีความยืดหยุ่น มีความรวดเร็ว และมีความปลอดภัยที่สูง

2.3.2. AdonisJS framework

AdonisJS เป็น เฟรมเวิร์กที่ช่วยให้สามารถสร้างเว็บไซต์ หรือเว็บแอปพลิเคชัน มีจุดเด่นคือมีเครื่องมือที่ช่วยในการพัฒนาระบบเว็บไซต์ครบวงจร

2.4 Database

ฐานข้อมูล คือ กลุ่มข้อมูลขนาดใหญ่ที่ถูกเก็บรวบรวมไว้ที่ใดที่หนึ่ง โดยเป็นข้อมูลที่มีความสัมพันธ์กัน ซึ่งถูกจัดเก็บอย่างเป็นระบบ โดยมีซอฟต์แวร์เข้ามาควบคุมกระบวนการใช้งาน การทำงาน หรือการประมวลผล

2.4.1. MariaDB

MariaDB เป็นระบบฐานข้อมูลเชิงสัมพันธ์ (Relational database) แบบ Open Source ที่ได้รับความนิยมสูง เนื่องจากสามารถใช้ชุดโปรแกรมในการเชื่อมต่อร่วมกับ MySQL ได้ จึงมีการรองรับการใช้งานในภาษาโปรแกรมส่วนใหญ่ มีการใช้ภาษา SQL ในการเข้าถึงข้อมูล

2.5 Python3

Python เป็นภาษาคอมพิวเตอร์ที่ใช้อย่างแพร่หลายในเว็บแอปพลิเคชัน การพัฒนาซอฟต์แวร์ วิทยาศาสตร์ข้อมูล เนื่องจากเป็นภาษาที่เรียนรู้ง่าย และสามารถทำงานบนแพลตฟอร์มต่างๆ ได้หลากหลาย

2.6 Serverless

Serverless เป็นรูปแบบการให้บริการบนระบบ Cloud โดยจะเป็นการให้บริการทรัพยากรในการทำงานต่าง ๆ โดยที่ผู้ใช้ไม่จำเป็นต้องจัดการทรัพยากรนั้นด้วย

ตนเองเลย ทำให้ผู้ใช้สามารถใช้งานระบบ Cloud ได้ง่ายขึ้น และไม่ต้องจัดการทรัพยากรที่ใช้งานอยู่ด้วยตนเอง

2.7 Golang

ภาษา Go (หรือเรียกว่า Golang) ถูกสร้างโดย Google มีจุดเด่นในหลายๆด้านเช่น สามารถทำงานได้เร็ว ออกแบบมาเพื่อให้รองรับการทำ Concurrent และ Multithreading ได้ง่าย เหมาะสำหรับงานที่ต้องการรองรับ Request เป็นจำนวนมาก มี Standard Library ที่ครอบคลุมการใช้งานใน Application ยุคใหม่ เป็นต้น

2.7.1. AWS Nuke

AWS Nuke เป็นโปรแกรมที่เขียนด้วยภาษา Go โดยใช้งาน AWS SDK ในการลบทรัพยากรที่เรากำหนดไว้โดยอัตโนมัติ โดยมีการรองรับบริการและประเภททรัพยากรของ AWS หลายประเภท

2.7.2. Cloud Nuke

Cloud Nuke เป็นเครื่องมือที่คล้ายกันกับ AWS Nuke แต่ต่างกันที่ Cloud Nuke นั้นไม่จำกัดการรองรับเพียงแค่ AWS เท่านั้นแต่ยังรองรับผู้ให้บริการคลาวด์อื่นๆ ถูกออกแบบมาให้มีความซับซ้อนน้อย สามารถแก้ไขปรับปรุงชุดคำสั่งได้ง่าย

2.8 Message Broker

Message Broker เป็นซอฟต์แวร์ที่ช่วยให้แอปพลิเคชัน ระบบ และบริการให้สามารถสื่อสาร และแลกเปลี่ยนข้อมูลระหว่างกันได้โดยไม่ต้องเชื่อมต่อโดยตรง ทำหน้าที่เป็นตัวกลางในการจัดการข้อความ และทำให้ข้อมูลสามารถส่งถึงปลายทางได้อย่างมีประสิทธิภาพและมีความน่าเชื่อถือ

2.8.1 RabbitMQ

เป็น Message Broker ที่ได้รับการออกแบบมาเพื่อรองรับการสื่อสารข้อมูลระหว่างระบบต่าง ๆ ภายในองค์กร ซึ่งมีความยืดหยุ่นสูง และเป็นมาตรฐาน ระบบของ RabbitMQ ทำงานผ่าน AMQP

2.9 MinIO

เป็นชุดโปรแกรม Open Source สำหรับเป็น Object Storage ในการจัดเก็บไฟล์สำหรับแอปพลิเคชันต่าง ๆ ได้ โดยมี API สำหรับการเชื่อมต่อที่สามารถใช้ร่วมกับ AWS S3 ได้ ทำให้มีความง่ายในการเปลี่ยนจากการใช้งาน AWS S3 มาใช้งาน MinIO แทนได้โดยไม่ต้องเปลี่ยนแปลงโค้ด

2.10 เทคโนโลยี Container

คอนเทนเนอร์เป็นเทคโนโลยีที่ช่วยในการแยกการใช้งานและการพัฒนาโปรแกรมให้เป็นอิสระ มีประสิทธิภาพ คอนเทนเนอร์ทำให้นักพัฒนาสามารถสร้างและทดสอบโปรแกรมในสภาพแวดล้อมที่เหมือนกันในแต่ละอุปกรณ์ได้

2.10.1. Docker

Docker คือเครื่องมือที่เป็น Open Source ที่ใช้สำหรับช่วยในการสร้าง พัฒนา จัดเก็บ และการปรับใช้งานภายใน Container มีความยืดหยุ่นและเป็นมาตรฐานสำหรับการดำเนินงานของแอปพลิเคชัน ช่วยลดขั้นตอนในการทำงาน และสามารถทำงานทางด้านจัดการสภาพแวดล้อมของ Container ได้ง่ายยิ่งขึ้น

2.10.2. Docker File

เป็นไฟล์เอกสารสำหรับบอกขั้นตอน และวิธีในการสร้าง Docker Image ขึ้นมา

2.10.3. Docker Image

เป็นแบบแผนสำหรับในการสร้าง Container ที่จะประกอบไปด้วยไฟล์ต่าง ๆ ที่ติดตั้งพร้อมกับคำสั่ง และข้อมูลที่กำหนดจาก Docker file สามารถนำไปใช้งานได้ทันทีผ่านชุดคำสั่งสำหรับการสร้าง หรือสามารถนำไปใช้เป็นแบบแผนในการสร้าง Docker Image อื่น ๆ ได้

2.10.4. Docker Compose

เป็นเครื่องมือที่ช่วยในการกำกับการสร้าง และการทำงานของ Docker Container ให้มีความสะดวก และง่ายยิ่งขึ้น ซึ่งจะใช้คำสั่งเป็นไฟล์ YAML สำหรับเป็นแม่แบบ สามารถกำหนดการทำงานของ Container ในด้านต่าง ๆ ได้

2.11 Web Server

เป็นชุดโปรแกรมที่ทำงานบนเครื่องแม่ข่าย ทำหน้าที่รับคำสั่ง และตอบกลับฝั่งผู้ใช้บริการจากเว็บ

เบราว์เซอร์ ผ่านโปรโตคอล HTTP หรือ HTTPS และส่งกลับผลลัพธ์ที่ผู้ใช้ต้องการต่าง ๆ

3. วิธีการดำเนินการวิจัย

3.1 การรวบรวมความต้องการ

ดำเนินการเก็บรวบรวมความต้องการการใช้ระบบจากผู้ที่ใช้ระบบจากทั้งอาจารย์ผู้สอน และนักศึกษาในรายวิชาทางด้านระบบ Cloud Computing และมีการสังเกตการใช้งานระบบการเรียนการสอนที่ใช้อยู่เดิม จากนั้นนำมาสรุปความต้องการของระบบดังนี้

1) สามารถให้บริการสร้างสภาพแวดล้อมสำหรับการใช้งาน AWS (Amazon Web Service) ที่แต่ละผู้ใช้งานจะมี Environment เป็นของตัวเอง

2) สามารถปรับแต่งขอบเขตของสิทธิ์ในการใช้งานAWS Environment ให้สอดคล้องกับเนื้อหาของการฝึกในแต่ละหัวข้อได้อย่างอิสระตามความต้องการของผู้สอน

3) สามารถจัดการ และติดตามการใช้งบประมาณในการจัดการการสร้างสภาพแวดล้อมได้อย่างละเอียด รวมไปถึงการแจ้งเตือนที่รวดเร็วถ้ามีการใช้งานที่สูงเกินปกติ

4) สามารถทำงานได้บนสภาพแวดล้อมที่หลากหลาย สามารถปิดการทำงานเมื่อปิดภาคเรียนการศึกษาได้ และสามารถนำกลับมาใช้งานได้ใหม่ในภาคเรียนการศึกษาถัดไป

3.2 การออกแบบโครงสร้างระบบ

ดำเนินการออกแบบระบบโดยการแบ่งระบบออกเป็นส่วนประกอบต่าง ๆ ที่สำคัญดังรูปที่ 1 โดยมีรายละเอียดแต่ละส่วนดังนี้

3.2.1. ระบบเว็บไซต์

เป็นส่วนโปรแกรมติดต่อผู้ใช้เพื่อแสดงผลข้อมูล และใช้เป็นระบบสำหรับผู้ใช้เพื่อติดต่อไปยังระบบส่วนอื่น ๆ

3.2.2. ระบบแกนกลาง

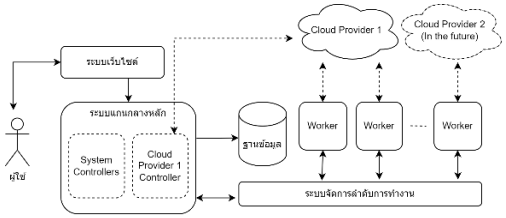
เป็นส่วนควบคุมการทำงานของระบบส่วนอื่นทั้งหมด โดยมีการออกแบบส่วนงานการติดต่อไปยัง Cloud Provider แยกออกจากส่วนงานอื่น เพื่อที่จะสามารถเพิ่ม Cloud Provider อื่น ๆ ได้ง่ายขึ้น ในอนาคต

3.2.3. ระบบจัดการลำดับการทำงาน

เป็นส่วนระบบสำหรับกระจายการทำงานไปยังส่วนการทำงาน (Worker) เพื่อทำงานตามที่กำหนด

3.2.4. ระบบหน่วยการทำงาน (Worker)

เป็นส่วนระบบสำหรับทำงานตามที่ได้รับมอบหมายจากระบบจัดการลำดับการทำงาน ตามฟังก์ชัน และเงื่อนไขที่กำหนด



รูปที่ 1. แผนภาพส่วนประกอบของระบบ

3.3 วิเคราะห์ระบบและวางแผน

ดำเนินการศึกษา ออกแบบระบบสำหรับการทำงานในส่วนต่าง ๆ และแนวทางในการจัดการระบบ

3.3.1. การจัดการบัญชี AWS สำหรับใช้งานในการเรียนการสอน

ออกแบบและศึกษาวิธี และแนวทางในการจัดการบัญชี AWS หลายบัญชีสำหรับนำมาใช้ในการเรียนการสอนให้มีประสิทธิภาพ

3.3.2. ระบบเว็บไซต์ และระบบแกนกลาง

3.3.3. ฐานข้อมูล

ศึกษาความเหมาะสมของชนิดฐานข้อมูลที่จะนำมาใช้งาน

3.3.4. ระบบจัดลำดับการทำงาน

ศึกษาความเหมาะสมของระบบจัดลำดับการทำงานต่าง ๆ ถึงความเหมาะสมในการนำมาใช้งาน

3.3.5. Worker

ศึกษาแนวทางในการออกแบบ พัฒนาชุดโปรแกรม Worker สำหรับทำงานในหน้าที่ต่าง ๆ ให้มีประสิทธิภาพมากที่สุด

1) สำหรับสร้างบัญชี AWS

2) สำหรับสร้างสภาพแวดล้อม

3) สำหรับทำลายสภาพแวดล้อม

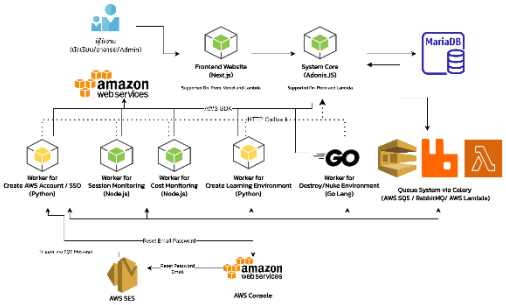
4) สำหรับตรวจสอบ และแจ้งค่าใช้จ่าย

5) สำหรับตรวจสอบเวลาการทำงาน

3.4 วิเคราะห์ค่าใช้จ่าย

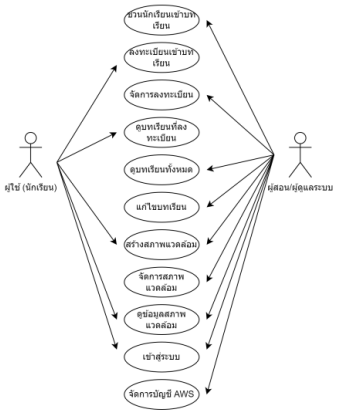
ศึกษาค่าใช้จ่ายสำหรับการดำเนินงาน และค่าใช้จ่ายในการทำงานของระบบ บนสถานการณ์ต่าง ๆ เพื่อเลือกเทคโนโลยีที่นำมาใช้

3.5 สรุปแผน และออกแบบระบบ



รูปที่ 2. แผนภาพบริการ และการทำงานของระบบ

3.5.1. ระบบเว็บไซต์

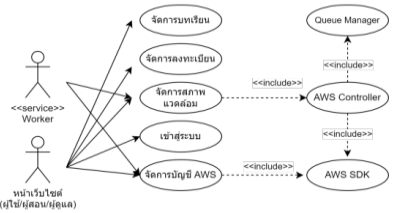


รูปที่ 3. Use Case Diagram ของเว็บไซต์

พัฒนาด้วยชุดเครื่องมือ Next.js สำหรับติดต่อกับผู้ใช้โดยตรง สามารถทำงานได้บนสภาพแวดล้อมที่หลากหลาย ทั้งบนเครื่องแม่ข่ายส่วนตัว และการทำงานบนสถาปัตยกรรม Server less โดยมี Use Case Diagram ดังรูปที่ 3

3.5.2. ระบบแกนกลาง

เป็นส่วนจัดการการทำงานในระบบทั้งหมด พัฒนาด้วยชุดเครื่องมือ Adonis.js สามารถทำงานได้บนสภาพ แวดล้อมที่หลากหลาย ทั้งบนเครื่องแม่ข่ายส่วนตัว และการทำงานบนสถาปัตยกรรม Serverless โดยมี Use Case Diagram ดังรูปที่ 4



รูปที่ 4. Use Case Diagram ของระบบ แกนกลาง

โดยมีการแบ่งระบบของผู้ใช้ภายในเป็น 3 ระดับดังนี้

1) ผู้สอน (Instructor) จะได้รับสิทธิ์ในการสร้าง และจัดการคอร์สที่ตนเป็นเจ้าของ

2) ผู้เรียน (Student) จะได้รับสิทธิ์ในการดูคอร์ส และเนื้อหาเฉพาะที่ตนลงทะเบียนไว้

3) ผู้ช่วยสอน (TA) จะได้รับสิทธิ์ในการดูคอร์สที่ตนได้รับสิทธิ์ และจะสามารถดูเนื้อหาที่ถูกซ่อนไว้โดยผู้สอนได้

และในการตรวจสอบสิทธิ์ในการเข้าใช้งาน จะมีการตรวจสอบสิทธิ์ต่าง ๆ ทุกครั้งที่มีการส่ง Request โดยใช้ API Token ที่ผู้ใช้จะได้รับเมื่อเข้าสู่ระบบในการยืนยันตัวตนตามฟังก์ชันที่เรียกใช้งาน

3.5.3. ระบบจัดการลำดับการทำงาน

เป็นส่วนที่ทำหน้าที่ส่งข้อมูลระบบแกนกลางไปยัง Worker แต่ละชนิดผ่านตัวกลางทั้งหมด 3 แบบ โดยจะเลือกใช้งานในขั้นตอนการติดตั้งระบบ ได้แก่

1) RabbitMQ เหมาะสำหรับต้องการติดตั้งระบบใช้งานบนเครื่องแม่ข่ายส่วนตัว

2) AWS SQS เหมาะสำหรับไม่ต้องการติดตั้งระบบบนเครื่องแม่ข่ายส่วนตัว หรือติดตั้งระบบบนบริการ Cloud Computing

3) AWS Lambda เหมาะสำหรับต้องการติดตั้งระบบบนสถาปัตยกรรมแบบไร้เครื่องแม่ข่าย

3.5.4. การจัดการบัญชี AWS

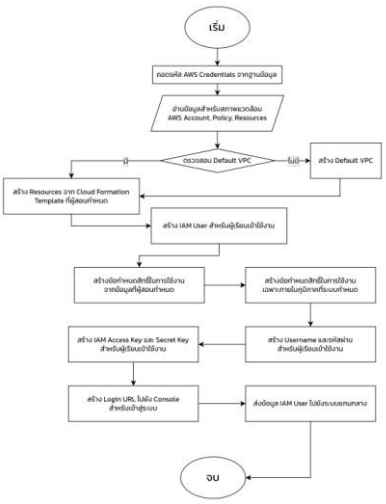
ในการจัดการบัญชีระบบจะจัดเก็บ Root Security Credentials ของแต่ละบัญชีไว้เพื่อใช้งานสร้างและทำลายสภาพแวดล้อม เพื่อป้องกันปัญหา

Lock-down หากผู้เรียนมีการตั้งค่า Resource Policy ที่ผิดพลาด [3] โดยอาจใช้ AWS Organization ในการช่วยบริหารจัดการบัญชีได้ [2] ขึ้นอยู่กับผู้ติดตั้งระบบ

3.5.5. Worker

1) สำหรับสร้างบัญชี AWS ใช้ภาษา Python ในการพัฒนา จะทำงานโดยการติดต่อไปยัง AWS เพื่อร้องขอการเปลี่ยนรหัสผ่าน และดำเนินการเปลี่ยนรหัสผ่านผ่านหน้าเว็บไซต์ และบันทึกข้อมูลลงฐานข้อมูล โดยก่อนการบันทึกจะมีการเข้ารหัสข้อมูลก่อนด้วย

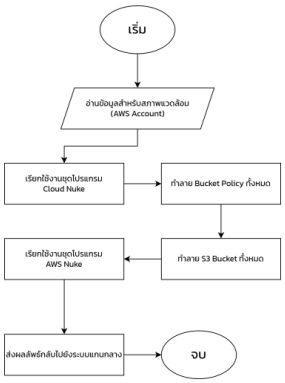
2) สำหรับสร้างสภาพแวดล้อม ในการฝึกปฏิบัติ โดยจะมีการสร้างทรัพยากรพื้นฐานด้วย Cloud Formation Template และสิทธิ์การเข้าใช้งานบริการต่าง ๆ ที่จะกำหนดโดยผู้สอนในแต่ละหัวข้อการฝึกปฏิบัติ พัฒนาด้วยภาษา Python พร้อมชุด AWS SDK สำหรับเรียกใช้บริการต่าง ๆ จาก AWS ดังรูปที่ 5



รูปที่ 5. แผนภาพการทำงานสร้างสภาพแวดล้อม

3) สำหรับทำลายสภาพแวดล้อม หลังการฝึกปฏิบัติ พัฒนาโดยใช้ชุดโปรแกรม Cloud Nuke ร่วมกับ AWS Nuke โดยจะมีการปรับแต่งให้ทำงานร่วมกันได้ดียิ่งขึ้น และครอบคลุมบริการ และทรัพยากรบน AWS ให้ได้มากที่สุด ดังรูปที่ 6

4) สำหรับตรวจสอบดูรายการค่าใช้จ่ายพัฒนาด้วยภาษา JavaScript โดยจะอ่านข้อมูลบัญชีทั้งหมดจากนั้นจะติดต่อไปยัง AWS เพื่อขอข้อมูลค่าใช้จ่ายผ่านบริการ Cost Explorer เพื่อนำมารวมเป็นค่าใช้จ่ายที่ใช้ภายในคอร์ส โดยจะมีการแจ้งเตือนหากค่าใช้จ่ายที่ใช้สูงกว่าที่กำหนดไว้

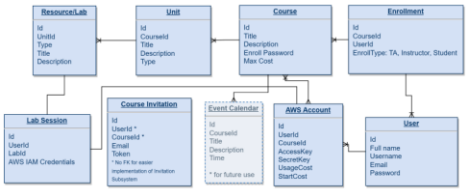


รูปที่ 6. แผนภาพลำดับการทำงานของระบบทำลายสภาพแวดล้อม

5) สำหรับตรวจสอบเวลาการทำงานของสภาพแวดล้อม โดยจะอ่านข้อมูลสิ่งแวดล้อมทั้งหมดที่ทำงานอยู่จากระบบแกนกลาง โดยหากมีสภาพแวดล้อมไหนที่หมดเวลาการทำงาน จะส่งข้อมูลไปยัง Worker เพื่อทำลายทรัพยากรออก

3.5.6. ฐานข้อมูล

ออกแบบฐานข้อมูลที่จำเป็นต้องใช้โดยมี Entity Relations Diagram ดังรูปที่ 7



รูปที่ 7. Entity Relations Diagram

3.6 พัฒนาระบบ

นำแผนที่ได้ออกแบบนำมาพัฒนาระบบ

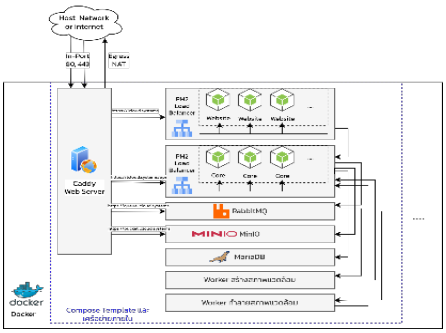
3.7 จัดการทดสอบ

มีการทดสอบแบ่งออกเป็นทั้งหมด 3 ครั้ง ดังนี้

1) ครั้งที่ 1 ทดสอบโดยคณะผู้จัดทำทำการทดสอบด้วยตนเอง

2) ครั้งที่ 2 ทดสอบโดยกลุ่มอาสาสมัครจากผู้เรียนในรายวิชา การพัฒนาคลาวด์แอปพลิเคชันประดับองค์กร

3.8 พัฒนาแม่แบบสำหรับนำระบบไปใช้งาน



รูปที่ 8. เครือข่ายและความสัมพันธ์ภายในคอนเทนเนอร์ที่ออกแบบ

หลังจากดำเนินการพัฒนาระบบเสร็จสิ้น จะนำเนินการสร้างแม่แบบสำหรับนำระบบไปใช้งานผ่านเทคโนโลยีคอนเทนเนอร์ด้วยชุดโปรแกรม Docker โดยจะจัดทำ Docker Compose Template มีส่วนประกอบต่าง ๆ ของระบบมีแผนภาพเครือข่าย และความสัมพันธ์ดังรูปที่ 8

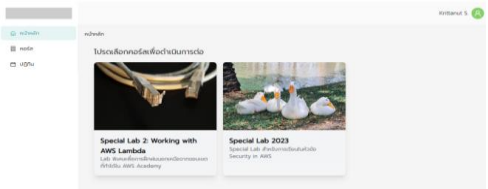
4. ผลการทดลอง และการพัฒนาระบบ

4.1 ผลการพัฒนาระบบ

4.1.1. หน้าเข้าสู่ระบบเว็บไซต์

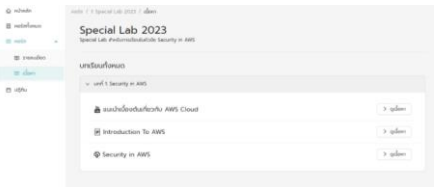
รูปที่ 9. หน้าเข้าสู่ระบบของเว็บไซต์

4.1.2. หน้าแสดงคอร์สทั้งหมดที่ลงทะเบียน



รูปที่ 10. หน้าเว็บแสดงคอร์สที่ลงทะเบียน

4.1.3. หน้าแสดงเนื้อหาภายในคอร์สเรียน

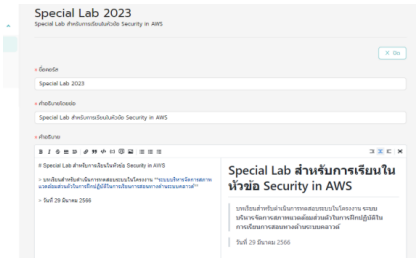


รูปที่ 11. หน้าเว็บแสดงเนื้อหาภายในคอร์ส

4.1.4. หน้าแสดงเนื้อหาภายในบทเรียนการฝึกปฏิบัติ

รูปที่ 12. หน้าเว็บแสดงเนื้อหาบทเรียนการฝึกปฏิบัติ

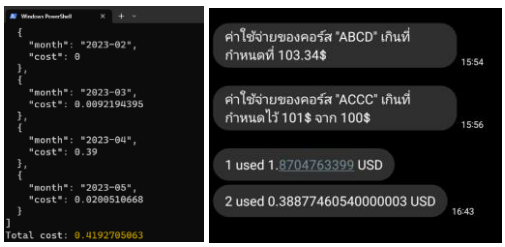
4.1.5. หน้าแก้ไข จัดการบทเรียน และจัดการรายชื่อนักเรียนภายในคอร์สสำหรับผู้สอน



รูปที่ 13. หน้าแก้ไขข้อมูลบทเรียนสำหรับผู้สอน

รูปที่ 14.หน้าดูรายชื่อนักเรียน และรายการค่าใช้จ่าย

4.1.6 ผลการทำงานของ Worker



รูปที่ 15. รายการรายงานค่าใช้จ่าย (ซ้าย) และการแจ้งเตือนค่าใช้จ่ายผ่านทาง SMS (ขวา)

4.2 การทดสอบระบบ

4.2.1. ครั้งที่ 1 ดำเนินการทดสอบด้วยตนเอง

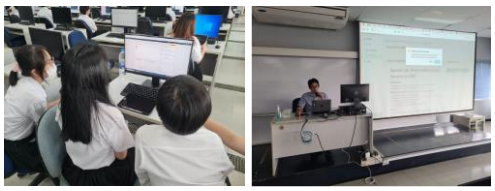
ดำเนินการทดสอบด้วยตนเอง โดยสร้างบทเรียนขึ้นมาสำหรับการทดสอบ โดยนำบทเรียน และข้อมูลการทำงานมาจาก AWS Academy เพื่อเปรียบเทียบการทำงาน

ตารางที่ 1. ตารางเปรียบเทียบระบบที่พัฒนากับ AWS Academy

หัวข้อ	ระบบที่พัฒนา	Academy
แก้ไขบทเรียน	ได้	ไม่ได้
เวลาเฉลี่ยในการโหลดเข้าเว็บไซต์	0.248s. (ICMP 16ms)	3.79s (ICMP 321ms)
เวลาเฉลี่ยนในการสร้างทรัพยากร	1:18 นาที	5:32 นาที
เวลาเฉลี่ยนในการลบทรัพยากร	1:32 นาที	8:40 นาที
สามารถตรวจความถูกต้องการฝึก	ไม่ได้	ได้
ดูรายละเอียดค่าใช้จ่าย	แยกรายบุคคลตามชนิดบริการ	แยกรายบุคคลตามหน่วยการเรียน
ทำแบบทดสอบ หรือข้อสอบ	ไม่ได้	ได้

4.2.2. ครั้งที่ 2 อาสาสมัครในรายวิชา พัฒนาคลาวด์แอปพลิเคชันระดับองค์กร

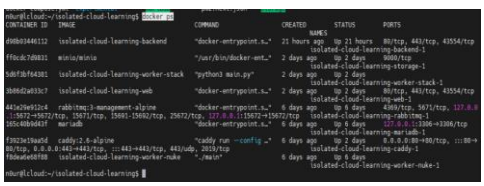
ดำเนินการทดสอบโดยมีอาสาสมัคร 5 คน เป็นนักเรียน 4 คน และผู้สอน 1 คน จากผลสำรวจความพึงพอใจหลังการทดสอบ พบว่ามีความพึงพอใจมากในการใช้งานระบบ



รูปที่ 16. ผู้สอนกำลังบรรยาย และผู้เรียนกำลังเรียนหัวข้อที่ใช้ในการทดสอบโดยใช้ระบบที่พัฒนาขึ้น

4.3 สร้างแม่แบบสำหรับการนำระบบไปใช้งาน

ดำเนินสร้างแม่แบบสำหรับการนำระบบไปใช้งานในรูปแบบของ Docker file และ Docker Compose ทำให้สามารถนำระบบไปใช้งานได้โดยง่าย โดยได้มีการทดลองติดตั้งบนเครื่องแม่ข่ายเสมือนส่วนตัว (VPS) เพื่อนำระบบไปใช้งาน สามารถนำไปใช้งานติดตั้งได้โดยง่าย สะดวก และมีความรวดเร็ว



รูปที่ 17. ผลลัพธ์เมื่อใช้งานคำสั่ง “docker ps” หลังจากการติดตั้งคอนเทนเนอร์จากไฟล์แม่แบบ

5. สรุปผล

จากการพัฒนาระบบบริหารจัดการสภาพแวดล้อมส่วนตัวในการฝึกปฏิบัติในการเรียนการสอนทางด้านระบบคลาวด์ ทำให้ได้ระบบที่สามารถใช้ในการฝึกปฏิบัติการเรียนการสอนทางด้านระบบคลาวด์ สามารถปรับแต่ง แก้ไขเนื้อหาของการฝึกปฏิบัติได้เอง และยังสามารถควบคุมค่าใช้จ่ายในการดำเนินการเรียนการสอนได้อย่างมีประสิทธิภาพ

หลังจากการพัฒนาระบบต้นแบบสำเร็จทางผู้จัดทำได้มีการจัดการทดสอบระบบขึ้น เพื่อทดสอบการทำงานของระบบ ซึ่งผลการทดสอบระบบนั้นแสดงให้เห็นว่าตัวระบบต้นแบบนั้นทำงานได้เป็นปกติตามที่ได้ทำการออกแบบไว้ มีความรวดเร็วในการทำงาน สามารถแก้ปัญหาของผู้สอนได้ตามขอบเขตที่ต้องการ ทั้งการสร้างทรัพยากร ทำลายทรัพยากร ปรับปรุงเนื้อหาที่ใช้ในการเรียนการสอน การตรวจสอบค่าใช้จ่ายจากการใช้งาน สามารถนำไปเป็นต้นแบบเพื่อใช้งานในการเรียนการสอนจริงได้จริงในอนาคต

เอกสารอ้างอิง

[1] Amazon Web Services. Setting Up Multi-User Environments in AWS (for Classroom Training and Research), Amazon Web Services, Tech. Rep., Accessed: 2022-09-15. [Online]. Available: https://docs.aws.amazon.com/pdfs/whitepapers/latest/setting-up-multi-user-environments/setting-up-multi-user-environments.pdf

[2] Amazon Web Services. (2022) I accidentally denied everyone access to my Amazon S3 bucket. How do I regain access? Accessed: 2022-10-21. [Online]. Available: https://repost.aws/knowledge-center/s3-accidentally-denied-access

การพัฒนาไฟร์วอลล์ที่ทำงานบนหน่วยประมวลผลกราฟฟิก

ธเนศ สุขได้พึ่ง[1] และ ธวัชชัย ฮานอน[2]

[1, 2]คณะเทคโนโลยีสารสนเทศ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง กรุงเทพฯ

Emails: 62070085@it.kmitl.ac.th 62070088@it.kmitl.ac.th

บทคัดย่อ

เนื่องจากความปลอดภัยของระบบเครือข่ายคอมพิวเตอร์นั้นจะมีโปรแกรมที่คอยดูแลการใช้งานคือ ไฟร์วอลล์ โดยปัจจุบันไฟร์วอลล์ทำงานบนหน่วยประมวลผลกลาง แต่การใช้งานเครือข่ายคอมพิวเตอร์ที่มากขึ้นทำให้ระบบไม่สามารถทำงานได้ตามปริมาณการใช้งานเครือข่ายและถูกจำกัดด้วยประสิทธิภาพของหน่วยประมวลผลกลาง ในงานวิจัยนี้เลยสร้างไฟร์วอลล์ที่ประมวลผลแบบขนานบนหน่วยประมวลผลกราฟฟิก มีหลักการทำงานคือทำการย้ายข้อมูลที่ถูกส่งมาในระบบเครือข่ายไปให้หน่วยประมวลผลกราฟฟิกคำนวณในแต่ละข้อมูลเทียบกับเงื่อนไขในการให้ข้อมูลนั้นผ่านได้หรือไม่ โดยการประมวลผลบนหน่วยประมวลผลกราฟฟิกจะใช้ภาษา OpenCL ซึ่งการประมวลผลจะต้องเป็นแบบขนานจึงมีการแบ่งการประมวลผลออกเป็นสองส่วนคือเปรียบเทียบเงื่อนไขของทุกกฎไฟร์วอลล์ จากนั้นทำการเลือกการตัดสินใจของกฎไฟร์วอลล์ที่ข้อมูลตรงเงื่อนไข โดยการประมวลผลแต่ละส่วนจะมีการแบ่งทรัพยากรบนหน่วยประมวลผลกราฟฟิกออกจากกัน ผลการทดสอบหลังการพัฒนาระบบพบว่าสามารถส่งข้อมูลจากเครือข่ายไปยังหน่วยประมวลผลกราฟฟิกและส่งผลลัพธ์กลับไปยังเครือข่ายเพื่อทำหน้าที่เป็นไฟร์วอลล์ได้ การกำหนดจำนวนข้อมูลที่ส่งไปประมวลผลบนหน่วยประมวลผลกราฟฟิกนั้น พบว่าเมื่อมีการส่งข้อมูลต่อครั้งมากขึ้น จะใช้เวลาในการประมวลผลต่อหน่วยข้อมูลลดลงจนใช้ข้อมูลเพิ่มขึ้นถึงจำนวนหนึ่งจึงมีแนวโน้มใช้เวลาเพิ่มขึ้นอีกครั้ง ประสิทธิภาพในการคัดกรองข้อมูลบนเครือข่ายเมื่อเทียบกับของดั้งเดิมพบว่าได้ประสิทธิภาพครึ่งหนึ่ง เนื่องด้วยการทดสอบนั้นไม่สามารถควบคุมสภาพแวดล้อมได้เหมาะสมจึงไม่สามารถใช้งานประสิทธิภาพของหน่วยประมวลผลกราฟฟิกได้เต็มที่ทำให้ผลในด้านประสิทธิภาพที่ยังไม่ชัดเจน หากสามารถควบคุมการทดสอบได้เชื่อว่าจะสามารถให้ประสิทธิภาพทัดเทียมกับของดั้งเดิมและการพัฒนาต่อคาดว่าสามารถลดทอนการย้ายข้อมูลไปมาในระบบปฏิบัติการและออกแบบการประมวลผลให้มีความฉลาดในการคัดกรองยิ่งขึ้น

คำสำคัญ – ไฟร์วอลล์; การประมวลผลแบบขนาน; หน่วยประมวลผลกราฟฟิก; ลินุกซ์

1. บทนำ

ในการทำงานของระบบและเครือข่ายเทคโนโลยีสารสนเทศในปัจจุบันนั้นจำเป็นต้องมี Firewall ควบคุมและป้องกันภัยในทั้งระบบเครือขายและบนคอมพิวเตอร์ End System ซึ่งแม้การทำงานของคอมพิวเตอร์จะมีประสิทธิภาพมากขึ้น ความต้องการทางการใช้งานก็มากขึ้นเช่นกัน การพัฒนา Firewall ที่ทำงานอยู่บน CPU แบบดั้งเดิมนั้นพัฒนาอย่างไรก็ไม่สามารถข้ามขีดจำกัดด้านความเร็วของ CPU ได้ ถึงแม้แบ่งการทำงานเป็น Thread ก็สามารถตรวจสอบได้แค่ทีละ Packet ต่อ Thread ซึ่งจำนวน Thread ที่ CPU เดียวสามารถใช้งานได้ก็ค่อนข้างน้อยเมื่อเทียบกับเทคโนโลยีทางด้าน Hybrid Computing Technology หรือ High Performance Computing Technology ในปัจจุบัน ผู้จัดทำจึงต้องการพัฒนา Firewall ที่นำความสามารถของหน่วยประมวลผลกราฟฟิกหรือ GPU ที่ประมวลผล Thread จำนวนมากแบบขนานมาใช้งาน

2. ทฤษฎีการย้าย Firewall จาก Linux Kernel ไปยังหน่วยประมวลผลกราฟฟิก

2.1. งานวิจัยที่เกี่ยวข้อง

ขณะที่ดำเนินงานวิจัยนี้ได้พบงานวิจัยด้านการใช้ GPU มาทำการเพิ่มประสิทธิภาพของ Linux Kernel ได้แก่การใช้ OpenCL หรือ CUDA ในการย้ายข้อมูลจาก Linux Kernel ไปประมวลผลใน CPU[1][2] , แนวทางการออกแบบ Parallel Firewall[3], การใช้ CUDA ในการเร่งการประมวลผล Packet Filtering[4][5]

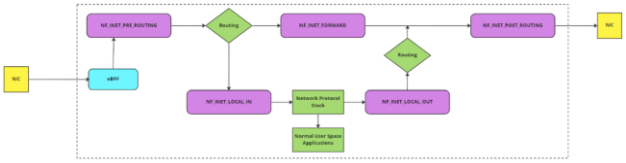
2.2. Linux

Linux หรือ GNU/Linux นั้นคือตระกูลของระบบปฏิบัติการที่มีต้นแบบจาก UNIX (UNIX-Like Operating System) ที่มีลักษณะเด่นคือมี Kernel หรือใจกลางของ

ระบบปฏิบัติการมีลักษณะเป็น OpenSource และเผยแพร่ Source Code ให้บุคคลทั่วไปใช้งานได้โดยไม่มีค่าใช้จ่าย โดย Linux ที่บุคคลทั่วไปนั้นใช้งานจะจำแนกเป็น Distro ซึ่งเป็น Linux Kernel รวมกับเครื่องมือและ Software ต่างๆที่พร้อมใช้งาน แต่ในงานวิจัยนี้หากพูดถึง Linux จะเน้นไปที่ Linux Kernel โดยเฉพาะ

2.3. Netfilter

Netfilter คือ Framework การจัดการ Packet ของ Linux ที่ทำงานส่งต่อเป็นท่อนๆหลังรับมาจาก Network Interface Card โดยมีจุดเชื่อมหรือ Hook ให้ใช้งานในการเรียก Packet มาจัดการ



รูปที่ 1. Netfilter Hooks

2.4. OpenCL

Open Computing Language เป็นภาษาที่ใช้ในการออกแบบโปรแกรมเพื่อประมวลผลแบบขนาน ถูกพัฒนาโดย Khronos OpenCL Working Group สามารถประมวลผลได้ทั้งบน หน่วยประมวลผลกลาง (CPU) และ หน่วยประมวลผลกราฟฟิก (GPU) และบนอุปกรณ์ที่สามารถประมวลผลภาษานี้ได้ รองรับการใช้งานตั้งแต่ Embedded จนถึง High Performance Computing

โครงสร้างของภาษาจะประกอบไปด้วย 4 โมเดลหลัก 1. Platform Model เป็นพื้นฐานในการตรวจสอบอุปกรณ์ว่ารองรับภาษาหรือไม่ 2. Memory Model ส่วนที่จัดการหน่วยความจำทั้งบน OpenCL Device หรือในที่นี้คือ GPU และ OpenCL Host หรือคอมพิวเตอร์ที่ควบคุมการประมวลผล เพื่อให้ทำการประมวลผลแบบขนานได้ 3. Execution Model จัดการการประมวลผลแบ่งการประมวลผลใน OpenCL Device โดยจะมีการแบ่งเป็น Work Group หรืองานทั้งหมดแบ่งเป็น Work item หรือหน่วยงานย่อยในการประมวลผล โดยในหน่วยย่อยสามารถ Sync หากันได้ ในการประมวลจะมีการจัด Queue การทำงานในหน่วยประมวลผล 4. Programming Model เป็นส่วนที่ใช้ในการออกแบบโปรแกรม

ลำดับการทำงานของ OpenCL นั้นจะเริ่มด้วยการเตรียมทรัพยากรจาก OpenCL Host เช่น เลือก Platform ในการประมวลผล กำหนดขนาด Memory ในการประมวลผลและกำหนดค่าใน Memory กำหนด Kernel File ในการประมวลผล เตรียม Queue เพื่อใช้ในการ Enqueue การประมวลผลต่างๆ จากนั้นจัดลำดับการประมวลผลและกำหนดการประมวลผลใน OpenCL Kernel ของ OpenCL Device ซึ่ง OpenCL Device Code จะแยกออกมาจาก Code ของ OpenCL Host ทำการโอนย้ายข้อมูลระหว่าง OpenCL Host และ OpenCL Device กำหนดขนาดในการทำงานทั้งหมดหรือ Work Group และขนาดการประมวลผลย่อย หรือ Work Item โดยขนาดสูงสุดจะถูกกำหนดโดย OpenCL Platform สุดท้ายคือการคืนทรัพยากรที่ใช้ในการประมวลผลทั้งหมดเพื่อให้พร้อมในการประมวลผลอื่นๆในโปรแกรม ทั้งนี้ในการเขียน OpenCL Host Program สามารถเขียนได้โดยใช้ภาษา C หรือ C++ ก็ได้โดยขึ้นอยู่กับการประมวลผลนั้นๆ ว่าเหมาะสมกับภาษาใด

2.5 Libnetfilter_queue

Libnetfilter_queue คือ User Space Library สำหรับการจัดการ Packet และเชื่อมต่อกับ Netfilter ใน User Space โดยใช้ Linux Firewall เช่น iptables, nftables, หรือ Netfilter Hooks ในการส่ง Packet เข้า Queue จากนั้นทำการอ่าน Queue ผ่าน Netlink Packet Data Handle (struct nfq_data *nfad) ใน User Space โดยการส่ง Packet เข้า Queue นั้นจะเป็นการดึง Packet ออกมาจาก Netfilter ไปเก็บไว้ในอีก Queue

3. การออกแบบระบบ

แยกระบบออกเป็น 2 Thread ได้แก่ recvThread เพื่อรับ Packet จาก Libnetfilter_queue และ verdictThread ที่เรียกใช้ OpenCL ในการประมวลผล Packet จากนั้นส่งผลการตัดสินใจกลับให้ Netfilter

3.1 การรับ Packets จาก Libnetfilter_queue

3.1.1 รูปแบบการเก็บ Packet Data

จัดเก็บ Packet Fields ต่างๆตามตารางที่ 1 โดยเมื่อรับข้อมูลมาจาก Libnetfilter_queue จำเป็นจะต้องแก้จาก Network Byte Order ซึ่งเป็น Big Endian ให้เป็น x86 Byte Order หรือ Little Endian ก่อนนำไปใช้งาน นอกจากนี้ได้รวม Source IP Address และ Destination IP Address เข้าด้วยกันเพื่อความสะดวกในการประมวลผลบน GPU และหากไม่มี Packet Field เช่น Source Port และ Destination Port จะเก็บค่าเป็น 0

ตารางที่ 1. Datatype ที่ใช้เก็บ Packet Data

Packet Field	Datatype
Source IP + Destination IP	uint64_t
Protocol	uint8_t
Source Port	uint16_t
Destination Port	uint16_t

3.1.2 การจัดเก็บ Packet

รับ Packet จาก Libnetfilter_queue มาเก็บไว้ใน Linked List แยกตาม Queue ที่ใช้เพื่อลดเวลาการรอ Packet ขณะที่ verdictThread ทำงานเรียกใช้ OpenCL

3.2 การควบคุม Rule

3.2.1 รูปแบบการเก็บ Rule

จัดเก็บ Packet Field ของ Rule ตามตารางที่ 2 โดยมีข้อมูลเพิ่มมาจาก Packet Field คือ Source Mask และ Destination Mask ซึ่งเก็บรวมกันในรูปแบบเดียวกับ IP Address และ Verdict หรือผลการตัดสินของ Rule โดยหาก Field ใหนเป็น Match All จะแทนค่าด้วย 0

ตารางที่ 2. Datatype ที่ใช้เก็บ Rule

Rule Field	Datatype
Source IP + Destination IP	uint64_t
Source Mask + Destination Mask	uint64_t
Protocol	uint8_t
Source Port	uint16_t
Destination Port	uint16_t
Verdict	int

3.2.2 แนวคิดการเทียบ Rule และ Packet

หาก Protocol, Source Port, หรือ Destination Port ของ Rule เป็น 0 จะทำการแปลง Packet Field ที่รับมาให้เป็น 0 ก่อนทำการเปรียบเทียบ จากนั้นทำการเทียบแต่ละ Packet Field ดัง Pseudo Code (1) โดยจะใช้งาน Verdict ของ Rule แรกที่เกิดการ Match หรือใช้งาน Drop Packet หรือ 0 หากไม่เกิดการ Match ขึ้น

Match = (Packet IP & Rule Mask == Rule IP)
& (Packet Protocol == Rule Protocol)
& (Packet Source Port == Rule Source Port)
& (Packet Dest Port == Rule Dest Port) *(1)*

3.3 OpenCL Program

การประมวลผลแบบขนานบน GPU ด้วย OpenCL ไม่เหมาะสมกับการ Jump เหมือนบน CPU จึงไม่สามารถทำการข้ามการประมวลผล Rule ได้ทำให้ต้องแบ่ง OpenCL Kernel ออกเป็น 2 ส่วนคือการเทียบ Packet กับ Rule และการสรุปผล

3.3.1 Compare Kernel

ทำการเทียบ Packet และ Rule ตาม Field ต่างๆตาม Pseudo Code (1) จากนั้นส่งผลการเทียบต่อให้ Sync Kernel

3.3.2 Sync Kernel

ทำการค้นหาผลลัพธ์การเทียบ Packet และ Rule ที่ Match กันเป็นอันดับแรกของแต่ละ Packet แล้วส่ง Verdict หรือผลการตัดสินใจของ Rule สำหรับแต่ละ Packet ออกมา

3.3.3 การจัดการ Buffer

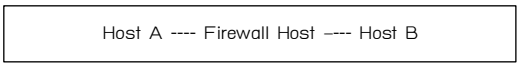
ได้แยก Buffer ออกเป็น 3 กลุ่มหลักคือ 1) Rule Buffer ที่จะสร้างและย้ายข้อมูลไป GPU เฉพาะตอนเปิดโปรแกรมเพียงครั้งเดียว 2) Packet Buffer และ Verdict Buffer ที่จะถูกสร้างเมื่อประมวลผลครั้งแรก แต่ถูกใช้ซ้ำในการโอนย้ายข้อมูลระหว่าง GPU และ OpenCL Host 3) GPU Only Buffer ที่ใช้ส่งข้อมูลกันระหว่าง Compare Kernel และ Sync Kernel

4. แนวทางการทดสอบระบบ

4.1. ระบบเครื่องคอมพิวเตอร์ที่ใช้ทำการทดสอบ

ได้ใช้เครื่องคอมพิวเตอร์ 3 เครื่องเชื่อมต่อกันผ่าน 10GE Fiber Optic ดังรูปที่ที่ 2 โดย Firewall Host ใช้ CPU Intel Core i5-4590, GPU NVIDIA GeForce GT1030 2GB และ

ระบบปฏิบัติการ Ubuntu 22.04.2 LTS และ Host A และ Host B ใช้ระบบปฏิบัติการ Windows 10



รูปที่ 2. Topology การทดสอบ

4.2. การทดสอบความถูกต้อง

ทดสอบโดยทำการเทียบ Packet และ Rule ด้วย Pseudo Code (1) บน CPU ตามลำดับ Rule และบน GPU แบบคู่ขนาน หากการประมวลผลแบบขนานถูกต้อง ผลลัพธ์ที่ได้จะต้องเหมือนการประมวลผลบนตามลำดับบน CPU

4.3. การทดสอบประสิทธิภาพการประมวลผล Firewall ด้วย GPU

ใช้ Host A ทำการสร้าง Packet ด้วย nping และ Colasoft Packet Player ส่งมายัง Firewall Host จากนั้นทำการวัดเวลาที่ใช้ในการประมวลผลด้วย GPU บน Firewall Host ในกรณีที่มีตัวแปรต่างกันได้แก่

- ตัวแปร N แทนจำนวน Packet ที่ส่งไปประมวลผลแต่ละครั้งบน GPU แต่แก่ 10 20 30 ... 90 100
- ตัวแปร M แทนจำนวน Rule ได้แก่ 1,000 2,000 3,000 ... 9,000 10,000

โดยทำการวัดแต่ละการทดลอง 1000 ครั้งจากนั้นนำผลที่ได้มาเฉลี่ยเป็นเวลาที่ใช้ประมวลผล M Rules บน GPU เฉลี่ยเมื่อแบ่งกลุ่มทีละ N Packet ซึ่งการทดลองนี้ไม่ได้ทำการวัดเวลาที่ใช้ย้าย Packet ผ่าน Libnetfilter_queue

4.4. การเทียบประสิทธิภาพกับระบบปัจจุบัน

ทำการสร้าง Firewall Rule ที่มีลักษณะคือจะไม่เกิดการ Match กับ Packet ที่ใช้ทดสอบ จนกว่าจะถึง Rule สุดท้าย เพื่อบังคับให้ iptables ทำการเทียบ Packet ต่อทุกๆ Rule จากนั้นทำการส่ง Packet จาก Host A ผ่าน Firewall Host ไปยัง Host B เพื่อทำการทดสอบดังนี้

ในส่วนของ iptables นั้นได้ใช้งาน Kernel Module เป็นกลุ่ม x_tables และ ip_tables และไม่ได้ใช้งาน nf_tables ซึ่งเป็น Kernel Module ของ Linux Firewall ใหม่หรือเครื่องมืออื่นๆที่ช่วยเพิ่มประสิทธิภาพ iptables เช่น ipset

4.4.1. การทดสอบ Throughput

ทำการวัด Throughput บน Tx ของ Host A และ Rx ของ Host B เมื่อ Firewall Host ใช้ iptables และ GPU Firewall ทำหน้าที่เป็น Firewall โดยใช้งานโปรแกรม FreeMeter ทำการ Sample ทุก 1/5 วินาที และการส่ง Packet ใช้งาน Colasoft Packet Player ส่ง Packet จาก Host A ไป Host B โดยสร้าง Packet ประเภท ICMP และ UDP ด้วย nping และทำการ Capture มาใช้งานด้วย Wireshark

4.4.2. ทดสอบ ICMP

ทำการส่ง ICMP Echo จาก Host A ผ่าน Firewall Host ไปยัง Host B 10,000 ครั้งโดยใช้ Delay ระหว่าง Packet 0ms และ 10ms และทำการวัด Round Trip Time และ Loss ของ ICMP Reply จาก Host B เมื่อใช้ iptables และ GPU Firewall ทำหน้าที่เป็น Firewall

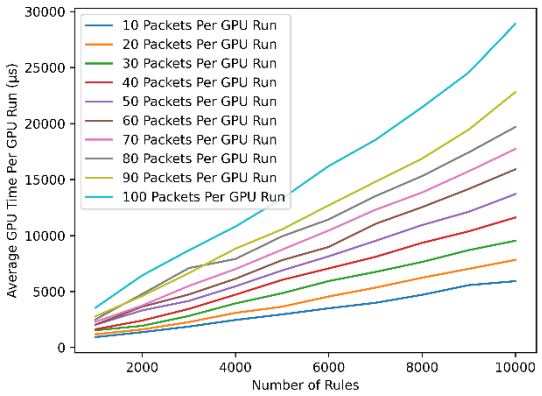
5. ผลการทดสอบระบบ

5.1. ความแม่นยำการประมวลผลบน GPU

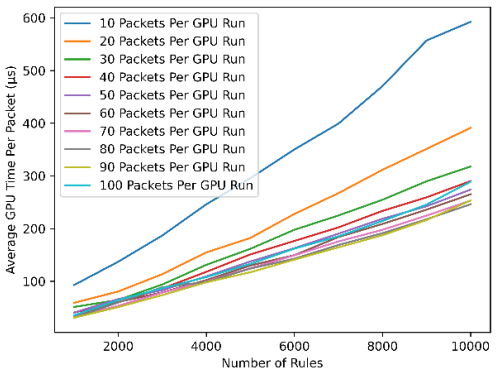
พบว่าสามารถประมวลผลได้ถูกต้องและเลือกผลการตัดสินของ Rule ที่ถูกต้องได้แม้ Match กับหลาย Rule หรือไม่ Match กับ Rule ใดก็ตาม

5.2. ประสิทธิภาพการประมวลผล Firewall ด้วย GPU

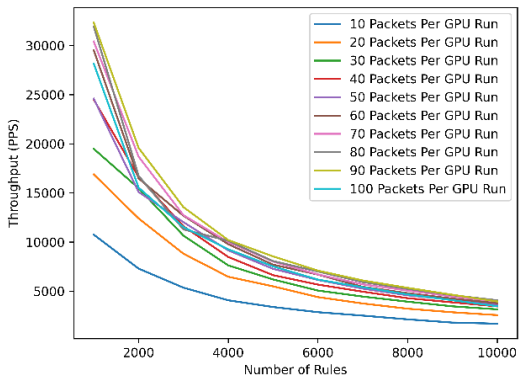
จากการทดลองได้เวลาประมวลผลผลดังรูปที่ 3 จากนั้นนำมาคำนวณเป็นเวลาประมวลผลเฉลี่ยต่อ Packet ดังรูปที่ 4 และ Throughput ในเฉพาะส่วนของการประมวลผลบน GPU ดังรูปที่ 5



รูปที่ 3. เวลาที่ใช้ประมวลผล Packet เฉลี่ยบน GPU ต่อการประมวลผลหนึ่งครั้ง



รูปที่ 4. เวลาที่ใช้ประมวลผล Packet เฉลี่ยบน GPU ต่อ Packet

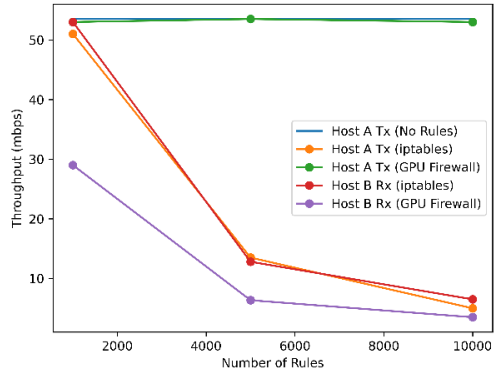


รูปที่ 5. Throughput (PPS) ในส่วนของการประมวลผลบน GPU

5.3. การเปรียบเทียบประสิทธิภาพกับระบบปัจจุบัน

5.3.1. Throughput

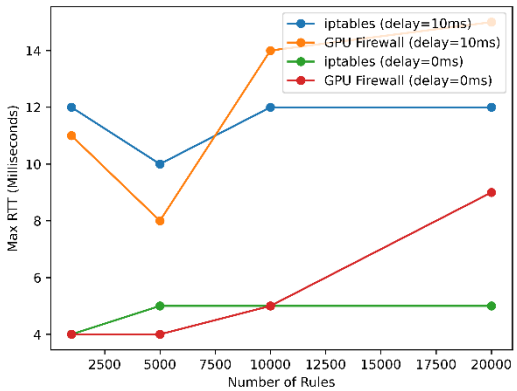
พบว่าโดยเฉลี่ยระบบที่พัฒนาให้ Throughput เพียงครึ่งหนึ่งของ iptables แต่เกิดจุดสังเกตขึ้นคือเมื่อ Rx ของ Host B ลดลง Tx ของ Host A จะลดลงหากใช้ iptables แต่หากใช้ระบบบน GPU จะไม่ส่งผลต่อ Tx ของ Host A คาดว่ามาจากการที่ Libnetfilter_queue สร้าง Queue แยกจึงไม่ส่งผลให้ Flow Control ของ 10GE NIC ทำงาน



รูปที่ 6. ผลการทดสอบ Throughput ระบบเทียบ iptables

5.3.2. Round Trip Time

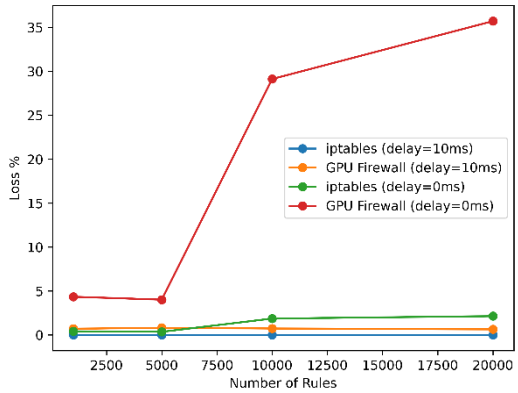
สาเหตุที่ใช้ Max Round Trip Time เป็นเพราะ nping แสดงผล Average และ Min Round Trip Time เป็น 0.00ms



รูปที่ 7. Max RTT ของการทดสอบ ICMP Echo

5.3.2. Loss

พบว่าระบบ GPU Firewall ที่ผู้จัดทำพัฒนาขึ้นส่งผลให้เกิด Loss มากกว่า iptables



รูปที่ 8. Loss ของการทดสอบ ICMP Echo

6. วิเคราะห์และสรุปผล

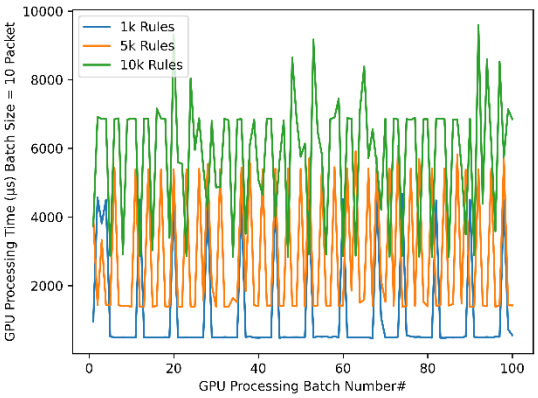
6.1. ความเร็วการประมวลผล Firewall ด้วย GPU

พบว่าเมื่อเพิ่มจำนวน Packet ที่ใช้ในการประมวลผลต่อครั้งจะทำให้เวลาประมวลผลโดยรวมเพิ่มขึ้นดังรูปที่ 3 แต่จะทำให้เวลาประมวลผลเฉลี่ยต่อ Packet ลดลงดังรูปที่ 4 แต่เมื่อจำนวน Packet มากกว่าค่าหนึ่งจะทำให้เวลาประมวลผลเฉลี่ยต่อ Packet มีแนวโน้มเพิ่มขึ้น

6.2. ประสิทธิภาพเทียบกับระบบปัจจุบัน

ไม่สามารถเทียบความเร็วประมวลผลได้ชัดเจนเนื่องจากเมื่อ Packet ใน Queue มี Source IP, Destination IP, และ IP Protocol เหมือนกันหรือ Sequence Number ต่อเนื่องกัน คาดว่า iptables มีแนวโน้มที่จะปล่อยผ่านพร้อมๆกันหลาย Packet ทำให้เวลาประมวลผล Packet ไม่เพิ่มมากเมื่อเพิ่ม Rule ที่ใช้ในการทดสอบด้วย ICMP Echo

การทดสอบ Throughput นั้นระบบ GPU Firewall ให้ Throughput เฉลี่ยครึ่งหนึ่งของ iptables แต่คาดว่าผลลัพธ์ที่ได้ยังไม่ชัดเจนเพราะเมื่อแสดงผลลัพธ์การประมวลผลของการทดลองที่ 5.2 ออกมาพบว่าเวลาที่ใช้ประมวลผลบน GPU มีแนวโน้มขึ้นลงดังรูปที่ 9 ผู้จัดทำคาดว่าอาจจะมาจากการที่ Firewall Host ได้ใช้ GPU ในการประมวลผลหน้าจอขณะทำหน้าที่เป็น Firewall เลยส่งผลให้การประมวลผล Firewall บน GPU ต้องรอการประมวลผลนานกว่าจำเป็นจึงใช้ประสิทธิภาพของ GPU ได้ไม่เต็มที่



รูปที่ 9. เวลาการประมวลผล 100 ครั้งแรกเมื่อประมวลผลครั้งละ 10 Packet

6.3. ปัญหาการจัดการ Queue

การที่ผู้จัดทำได้แยกระบบออกเป็น Thread รับข้อมูล (recvThread) และ Thread ประมวลผล (verdictThread) ได้ส่งผลในแง่ลบคือเมื่อ Packet มีจำนวนมาก ทำให้รับมาเก็บใน User Space และ Kernel Space พร้อมๆกันจึงใช้งาน Memory มากเกินจำเป็น

คาดว่า Libnetfilter_queue ได้สร้าง Queue แยกออกมาจาก Queue ของ NIC เนื่องจากเมื่อการใช้งาน Queue บน Firewall Host สูง Tx ของ Host A ไม่ได้ลดลงเหมือนกรณีที่ใช้ iptables ดังรูปที่ 6 เพราะ Libnetfilter_queue ใช้งาน Queue แยกจึงส่งผลให้ Kernel Driver ของ NIC ไม่ทำการส่ง Flow Control Frame

6.4. ระบบในภาพรวม

ระบบที่พัฒนาขึ้นสามารถย้าย Packet ออกมาจาก Linux Kernel ผ่าน Libnetfilter_queue มาประมวลผลบนหน่วยประมวลผลกราฟฟิกแล้วส่งการตัดสินใจหรือ Verdict กลับไปยัง Linux Kernel เพื่อ Drop หรือ Forward Packet ต่อไปยัง Network ได้

6.5. แนวทางการพัฒนาต่อ

6.5.1. แก้ไขระบบเป็น Daemon

ระบบที่พัฒนาได้ใช้เพียง main() เพื่อสร้าง Loop ให้โปรแกรมทำงานต่อ จึงไม่สามารถแก้ไข Rule ขณะ Runtime ได้ คาดว่าหากแก้ไขระบบเป็น Daemon จะทำให้ควบคุมการเปิด/ปิดโปรแกรมและการจัดการ Rule เหมาะสมขึ้น

6.5.2. Libnetfilter_queue

คาดว่าหากสามารถเปลี่ยนจากการใช้งาน Libnetfilter_queue ไปใช้ Kernel Space to User Space Shared Memory จะสามารถลด Delay การย้าย Packet ไป GPU ได้ส่วนหนึ่ง

6.5.3. OpenCL Kernel

หากสามารถแก้ไขการรวมผลลัพธ์ใน Sync Kernel ให้เป็น Parallel มากขึ้นหรือลดความจำเป็นในการแยก OpenCL Kernel สำหรับเทียบ Rule (Compare Kernel) และ Kernel รวมผลลัพธ์ (Sync Kernel) คาดว่าจะเพิ่มประสิทธิภาพได้อีกเพราะการแยก OpenCL Kernel ทำให้ต้องรอการประมวลผลของ Compare Kernel เสร็จก่อนและเพิ่มเวลาการย้ายข้อมูลจาก Compare Kernel มายัง Sync Kernel

เอกสารอ้างอิง

[1]Chia-Heng Tu, Te-Sheng Lin. “Augmenting Operating Systems with OpenCL Accelerators” ACM Transactions on Design Automation of Electronic Systems, Vol. 24, No. 3, Article 30. March 2019.

[2]Weibin Sun, Robert Ricci. “Augmenting Operating Systems With the GPU”[Online]. Available: https://arxiv.org/abs/1305.3345. 2013

[3]Errin W. Fulp. “Parallel Firewall Designs for High-Speed Networks” Proceedings IEEE INFOCOM

2006. 25TH IEEE International Conference on Computer Communications, April 2006

[4]Manoj Singh Gaur and Vijay Laxmi. “Parallel Firewalls on General-Purpose Graphics Processing Units”[Online]. Available: https://arxiv.org/abs/1312.4188. 2013

[5]K. Karimi, A. Ahmadi and M. Ahmadi. “Parallel Implementation of Linux Packet Filtering” The CSI Journal on Computer Science and Engineering, Vol. 11, No. 2 & 4(b), 2013.pp.24-30

[6]Michael Kerrisk. “The Linux Programming Interface”, No Starch Press, Inc. 2010

[7]Rusty Russell, Harald Welte. “Linux netfilter Hacking HOWTO”[Online]. Available: https://www.netfilter.org/documentation/HOWTO/netfilter-hacking-HOWTO.txt. 2002

[8]Khronos OpenCL Working Group. “The OpenCLTM Specification”[Online]. Available: https://registry.khronos.org/OpenCL/specs/3.0-unified/html/OpenCL_API.html. 2022

[9]The Netfilter webmasters. “Lilbnetfilter_queue Documentation”[Online]. Available: https://netfilter.org/projects/libnetfilter_queue/doxygen/html/. Retrieved 2022

การศึกษาหาวิธีการที่เหมาะสมสำหรับการแบ่งชุดข้อมูลด้วยอัลกอริทึม BITMAP INTERSECTION LOOKUP

สมภพ ศักดิ์ศรีชัย[1] และ อภิสร ตี้ฮ้อ[2]

[1, 2]คณะเทคโนโลยีสารสนเทศ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง กรุงเทพฯ

Emails: 62070191@it.kmitl.ac.th, 62070218@it.kmitl.ac.th

บทคัดย่อ

ในปัจจุบันมีข้อมูลในระบบเครือข่ายจำนวนมากทำให้การทำ Packet Classification มีความล่าช้า และทำให้ใช้พื้นที่หน่วยความจำมากในการจัดเก็บข้อมูล จึงมีจุดประสงค์ในการพัฒนา Algorithm Bitmap Intersection Lookup(BIL) ในการแบ่ง Segment เพื่อลดขนาดหน่วยความจำที่ใช้ และลดเวลาในการ update table เพื่อเพิ่มประสิทธิภาพการทำงาน ทางผู้จัดทำจึงได้เสนอแนวคิดการแบ่ง Segment ว่าควรแบ่งอย่างไรมีประสิทธิภาพสูงสุดต่อหน่วยความจำที่มี โดยวิธีการแบ่ง Segment ที่เสนอนั้นบ่งบอกว่าควรแบ่ง Segment อย่างไรจึงจะทำให้มีความเร็วการทำงานดีที่สุดและใช้ความจำน้อยที่สุด ดังนั้นผู้จัดทำจึงพิสูจน์ตามวิธีการแบ่ง Segment ที่เสนอ ในการทดลองเริ่มจากกำหนด Requirement คือขนาดความจำและจำนวนของกฎ โดยจะทดสอบหาค่าเฉลี่ยความเร็วในการทำงานและวัดหน่วยความจำที่ใช้ และนำผลลัพธ์ตาม Requirement ที่ได้นำไปเปรียบเทียบกับการแบ่ง Segment รูปแบบอื่น ๆ

จากผลการทดลองหากแบ่ง Segment มากๆ ทำให้ใช้ความจำน้อยลง และทำให้การทำงานเร็วขึ้น ในการทดลองตาม Requirement เมื่อนำไปเปรียบเทียบกับการแบ่งรูปแบบอื่น สรุปได้ว่าหากแบ่ง Segment โดยมีจำนวน Segment ที่เท่าๆ กันจะมีประสิทธิภาพดีที่สุดและใช้หน่วยความจำน้อยที่สุด แต่ข้อเสียการแบ่ง Segment มาก ๆจะทำให้ใช้เวลาในการค้นหามากขึ้นตามจำนวน Segment แต่จะห่างกันไม่มาก ดังนั้นการทดลองทั้งหมดเป็นไปตามแนวคิดที่ได้เสนอ สำหรับการนำไปใช้งานจริงควรคำนึงถึงขนาดความจำของเครื่องเพราะประสิทธิภาพของ BIL จะขึ้นอยู่กับขนาดความจำ ถ้ามีความจำมากประสิทธิภาพจะมากตาม

คำสำคัญ- BITMAP INTERSECTION LOOKUP, อัลกอริทึม, การแบ่งชุดข้อมูล

ABSTRACT

Therefore, the purpose of developing Algorithm Bitmap Intersection Lookup(BIL) is to divide segments to reduce the amount of memory used and reduce the time to update the table to increase work efficiency. Therefore, the author proved according to the proposed segmentation method. In the experiment, starting from the requirement is the memory size and number of rules, the test will average the working speed and measure the memory used, and the results according to the requirements will be compared with other types of segmentation..

According to the results of the experiment, if the segment is divided a lot, it will use less memory and make the work faster. In the experiment according to the requirement when compared with other forms of division. In conclusion, if you divide a segment with the same number of segments, it will have the best performance and use the least memory, but the disadvantage of splitting a lot of segments will cause it will take more time to search according to the number of segments, but not far apart. Therefore, all experiments are based on the

proposed concept. For practical use, the memory size of the machine should be taken into account, because the performance of BIL will depend on the memory size. If there is a lot of memory, the performance will be very followed.

1. บทนำ

Bitmap Intersection Lookup เป็น Algorithm ที่คิดขึ้นมาเพื่อจัดการแอปลิเคชันหรือสามารถนำไปใช้การจัดการแพ็คเก็ตบนอินเทอร์เน็ต เช่น การสื่อสารที่มีความเร็วสูงเป็นต้น ทำให้เกิดการกำหนดกฎการเข้าถึงที่เพิ่มขึ้นซึ่ง BIL คือ กฎที่มี Algorithm โครงสร้างที่ง่ายไม่ซับซ้อน มีความเร็วในการค้นหาปรับปรุงกฎ ที่ดีและสูงขึ้น อีกทั้งยังใช้เนื้อที่จัดเก็บข้อมูลต่ำ โดยจะจัดโครงสร้างของกฎลงใน BIL tables และนำกฎที่มีลักษณะเป็นช่วง แปลงค่าให้อยู่ในรูป Prefix และแบ่งเป็น Block ขนาดเล็กๆ เพื่อใช้สำหรับระบบ โดยผลลัพธ์ที่ได้ Algorithm จะเลือกกฎที่มีความสำคัญสูงสุดเป็นผลลัพธ์

แต่ยังไม่มีการทดสอบว่า Algorithm ที่ใช้หลักการ BIL นั้นมีวิธีการแบ่งชุดข้อมูลอย่างไรจึงจะมีประสิทธิภาพหรือเหมาะสมต่อหน่วยความจำที่มีตาม จึงมีจุดประสงค์เพื่อหาวิธีการแบ่งชุดข้อมูลที่เหมาะสมในรูปแบบเงื่อนไขที่แตกต่างกัน อีกทั้งเพื่อให้ทราบถึงปัญหาต่างๆ ของการแบ่งชุดข้อมูลที่อาจจะเกิดขึ้น

2. การทบทวนเอกสารวรรณกรรม

ผู้วิจัยได้รวบรวมศึกษาเอกสารและงานวิจัยที่เกี่ยวข้องที่เป็นความรู้พื้นฐาน ในด้านเนื้อหาและเทคโนโลยี เพื่อเป็นข้อมูลในการพัฒนา ส่วนประกอบ ในบทนี้ด้วย

2.1 Search Algorithm

เป็นขั้นตอนสำหรับการค้นหาข้อมูลตามที่เรากำหนด โดยค้นหาข้อมูลจากตัวเก็บข้อมูลที่อยู่ในรายการหรือ List การค้นหาข้อมูลนั้นสามารถเป็นได้ทั้งข้อมูลที่เรียงลำดับ และไม่เรียงลำดับ ขึ้นอยู่กับประเภทข้อมูลของ Search Algorithm ที่ใช้ในการค้นหา ในตัวอย่างนี้จะยก Linear Search และ Binary Search มาเป็นตัวอย่าง

2.2.1 Linear Search

เป็นการค้นหาแบบง่าย เป็นการค้นหาแบบเรียงลำดับหรือไม่เรียงลำดับก็ได้ โดยค้นทีละตัวไปจนครบ โดยเริ่มจากตัวแรกไปยังตัวสุดท้ายจนกระทั้งพบข้อมูลที่ต้องการค้นหา

ยกตัวอย่างเช่น ถ้าต้องการหาตัวเลข 39 โดยมีข้อมูลทั้งหมด 7 ตัว แล้วเลขที่ 39 อยู่ในตำแหน่งที่ 5 โดยวิธีนี้ใช้ขั้นตอนในการค้นหา 5 ขั้นตอนจึงพบข้อมูลที่ต้องการ

2.2.2 Binary Search

Binary Search เป็นการการค้นหาข้อมูลที่มีความเร็ว และไม่ซับซ้อนเหมาะกับข้อมูลที่มีปริมาณไม่มาก โดยข้อมูลที่จะนำไปค้นหาต้องถูกเรียงลำดับจากน้อยไปมากก่อน วิธีค้นหาคือ หาค่ากึ่งกลาง = [โดยที่นำค่าที่น้อยที่สุด + ค่าที่มากที่สุด] แล้วหารด้วย 2 แล้วตรวจสอบดูว่าค่ากลึ่งกลางนั้นมากกว่า หรือน้อยกว่าค่าที่เราต้องการค้นหา หากค่าที่ต้องการหาน้อยกว่าค่ากึ่งกลาง จะตัดนำแหน่งข้อมูลหมดที่มีค่ามากกว่าตำแหน่งกึ่งกลาง หรือถ้าหากค่าที่ต้องการหามากกว่าค่ากึ่งกลาง จะตัดนำแหน่งข้อมูลหมดที่มีค่าที่น้อยกว่าตำแหน่งกึ่งกลาง ทำแบบนี้วนซ้ำๆ จนพบค่าที่ต้องการค้นหา

2.2 หลักการทำงานของ BIL

การทำงานของ Bitmap Intersection Lookup หรือ BIL จะอยู่ในส่วนของ Bitmap Table โดยที่ Bitmap Table จะมีการสร้างตารางไว้เปรียบเทียบไว้อยู่แล้ว ในตารางจะประกอบไปด้วย Index และ Bitmap โดย Index จะมีขนาดเท่ากับจำนวนข้อมูลที่รับเข้าเท่ากับ 2 กำลัง N bit ในส่วนของ Bitmap คือคู่ค้นหาของค่า Value และ Masking ที่มีการแมชท์ (Match) กับ Rule โดยในข้อมูลแต่ละ field จะมีตำแหน่งของกฎซึ่งจะถูกแทนที่ด้วย Bit หากมี Bit ตรงกับกฎข้อใดให้นำค่า 1 ไปอัพเดทในตาราง หากไม่ตรงให้นำค่า 0 ไปอัพเดททำจนครบทุก Index แล้วใส่ใน Bitmap Table ในการค้นหาใน Firewall ที่นำ BIL มาใช้ เช่นหากมีข้อมูลที่รับเข้ามา หากอยากรู้ว่าค่าที่ต้องการค้นหาตรงกับ

Firewall Rule ข้อใด ให้นำ Packet header ของแต่ละ Field มาเปรียบเทียบ โดยใน Firewall Rule จะมี Rule ID จากนั้นทำการ Lookup หรือดึงข้อมูลที่รับเข้ามาดูตรวจสอบว่าตรงกับกฎข้อใดใน Bitmap Table แล้วทำการ List รายออกมา แล้วกำหนดค่าความสำคัญที่สุดแล้วนำกฎข้อนั้น ๆ มาใช้โดยในแต่ละกฎจะระบุว่ามี Action อย่างไร เช่น อนุญาตให้ใช้งาน (Allow) หรือ ไม่อนุญาตใช้งาน (Denied)

2.3 แนวคิดการจัดประเภทแพ็คเก็ตและการแบ่งชุดข้อมูล

การจัดประเภทแพ็คเก็ตมีจุดประสงค์เพื่อให้สามารถใช้งานระบบเครือข่ายได้อย่างมีประสิทธิภาพ เพื่อเพิ่มความปลอดภัย และเพื่อให้ระบบเครือข่ายมีความเร็วมากยิ่งขึ้น จึงต้องมีการพัฒนาอุปกรณ์ต่าง ๆ มาช่วยเพิ่มประสิทธิภาพ ซึ่งจำเป็นต้องมีการจัดชุดข้อมูลเพื่อสามารถแบ่งแยกแพ็คเก็ตในแต่ละตัวที่เข้ามาถูกจัดประเภทและแบ่งชุดข้อมูลได้อย่างเหมาะสม โดยการจัดประเภทแพ็คเก็ตนั้นมีความเร็วไม่มากพอ และยังมีผลกระทบที่เกิดจากการแบ่งชุดข้อมูล ในด้านของความถูกต้องของข้อมูล รวมไปถึงส่งผลกระทบต่อหน่วยความจำทีมีหากไม่มีการแบ่งชุดข้อมูลให้เหมาะสม

ในปัจจุบันการพัฒนาอุปกรณ์เครือข่ายให้มีความเร็วสูงแล้วเสถียรมากขึ้น โดยการนำเทคนิคการประมวลผล การจัดประเภทแพ็คเก็ตและการแบ่ง ชุดข้อมูล มาพัฒนาอุปกรณ์ Hardware เพื่อให้สามารทำงานด้วยความเร็วสูงได้อย่างมีประสิทธิภาพ โดยการจัดประเภทแพ็คเก็ตที่ดีนั้น ประสิทธิภาพในการค้นหาจะไม่ขึ้นอยู่กับจำนวนข้อมูล ในกรณีนี้จะระบุเป็นการจัดประเภทแพ็คเก็ตบน Firewall โดยการจัดประเภทแพ็คเก็ตบน Firewall คือการนำข้อมูลที่เข้ามาให้ตรงกับ Firewall Rules เป็นต้น และการแบ่ง ชุดข้อมูล คือการแบ่งชุดข้อมูลให้เหมาะสมกับข้อมูลที่รับเข้ามา การจัดประเภทข้อมูลบน Firewall Rule มีหน้าที่ตรวจสอบข้อมูลที่รับเข้ามาแล้วตรวจสอบเงื่อนไข โดยการส่งข้อมูลเข้าไปในเครือข่ายซึ่งมี Firewall Rules จำนวนมากและถูกแบ่งไว้เพื่อทำการตรวจสอบ คัดกรองข้อมูล การจัดประเภทข้อมูลและการแบ่ง ชุดข้อมูล จึงเป็นปัจจัยที่สำคัญที่ช่วยจัดการข้อมูลที่มากเกินไป ที่อาจทำให้เกิดปัญหาในการค้นหาล่าช้าได้ รวมไปถึงความถูกต้องของข้อมูลจากการค้นหาเมื่อเปรียบเทียบกับ Firewall Rules เป็นต้น

3. การออกแบบ Algorithm และทดสอบประสิทธิภาพ

การที่ค้นหาข้อมูลต่างๆ จะมี Search Algorithm ที่เป็นพื้นฐานในการค้นหา โดยการค้นหาใน Firewall Rule จะมี Search Algorithm พื้นฐานเหมือนกัน โดยยกตัวอย่างการทำงานในการค้นหา Firewall Rules โดยกรณีที่ใช้ Linear Search ในการค้นหาจะใช้เวลาค้นหาที่นาน ซึ่งระบบงานดังกล่าวจะมีความล่าช้าเนื่องจากจำเป็นต้องตรวจสอบค่าที่รับเข้ามาที่ละตัว โดยระบบงานเดิมนั้นการทำงานต่างๆ จะใช้เวลาในการค้นหาขึ้นอยู่กับจำนวนข้อมูลที่เข้ามา

3.1 แนวคิดการแบ่ง Segment

การแบ่ง Segment นั้นทำเพื่อลดขนาดหน่วยความจำที่มากเกินไป เพื่อใช้ลดระยะเวลาในการ Update Table ทำให้ขนาดตารางในแต่ละ Segment มีขนาดเล็กลง เพื่อทำให้สามารถสร้างตารางได้เร็วขึ้น เช่น 1 field มีขนาด 8 Bits มีขนาด 256 แบ่งเป็น 4 Segment โดยแบ่งเป็น Segment ละ 4 Bit จะมีขนาด 32 จะเห็นได้ว่าการแบ่ง Segment ทำให้ขนาดหน่วยความจำลดลง และเพิ่มความเร็วในการสร้างตาราง แต่ในเรื่องผลกระทบที่เกิดจากแบ่ง Segment นั้นอาจเกิดปัญหาได้ เช่น หากมีการแบ่ง Segment มากเกินไปอาจทำให้เกิดความล่าช้าในการค้นหาข้อมูลเพราะมีจำนวน BIL Table จำนวนมาก

3.2 หลักการแบ่ง Segment

การแบ่ง Segment นั้นต้องคำนึงถึงขนาดหน่วยความจำที่มีอยู่ ยิ่งแบ่งออกมากหน่วยความจำที่ต้องใช้จะยิ่งน้อยลงและขนาดของ Bit ในแต่ละ Segment เท่ากันจะใช้เวลาในการสร้าง และขนาดความจำน้อยที่สุดของจำนวน Segment นั้นๆ

3.3 การพัฒนา Algorithm

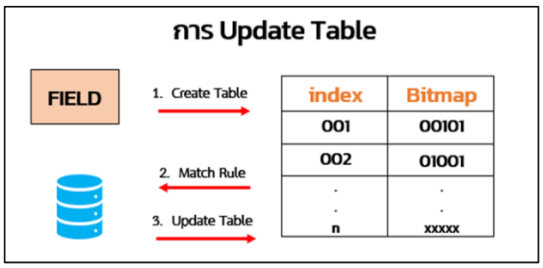
การสร้างอัลกอริทึมจะแบ่งเป็น 2 วิธีหลัก คือ

3.3.1. การ Update Table

1. Create Table สร้างตาราง Bitmap Table เช่น มีข้อมูลขนาด 8 Bits สร้าง Table 1 field มีค่า index 2^4

2. Match Rule กำหนดคู่ Pair โดยใน Pair จะประกอบด้วย value กับ mask โดยจะนำค่า Index ไป match rules โดยตรวจสอบทีละ Rules ตั้งแต่ Rule ข้อ 1 ไปจนถึงข้อสุดท้าย

3. Update Table จากขั้นตอนที่ 2 ในกรณีที่ Rule Match กับ Index ให้นำค่า 1 ไปอัพเดทใน Table ในกรณีที่ไม่ Match ให้นำค่า 0 ไปอัพเดทใน Table แล้วทำไปเรื่อย ๆ จนกว่าจะค้นหาครบทุก Index



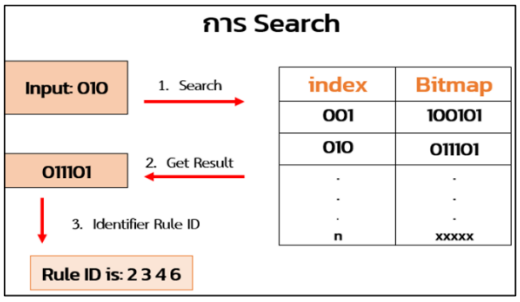
รูปที่ 1 ขั้นตอนการทำงานของ BIL ในการ Update Table

3.3.2. การ Search ในการแบ่งชุดข้อมูลแต่ละรูปแบบ

1. Get input & Search Rule นำค่า input ที่รับมาไปค้นหาใน Bitmap Table

2. นำผลลัพธ์ที่ได้มาระบุ Rule ID

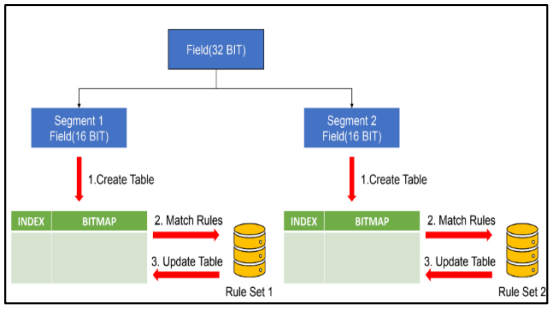
3. ทำการทดสอบความเร็วในการ Search โดยกำหนดขนาดในการแบ่งชุดข้อมูลแต่ละรูปแบบ



รูปที่ 2 ขั้นตอนการทำงานของ BIL ในการ Search

3.4 การสร้างตารางแบ่ง Segment

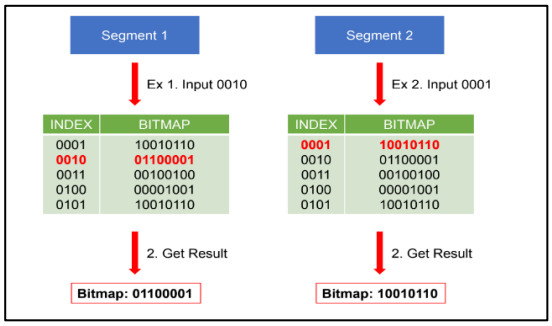
การสร้าง BIL Table ต้องกำหนดจำนวน BIT แต่ละ Field โดยกำหนดไม่เท่ากันได้ ในการแบ่งชุดข้อมูลจะนำจำนวน BIT ใน Field มาแบ่งชุดข้อมูลก่อนแล้วนำมาสร้างเป็น BIL Table



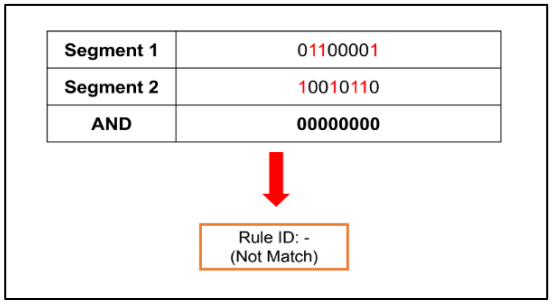
รูปที่ 3 BIL Search สำหรับการแบ่งชุดข้อมูล

3.5 วิธีการค้นหาแบ่ง Segment

การค้นหาแบบแบ่ง Segment ในขั้นตอนแรกจะรับ Input เข้ามาแล้วนำไปตรวจสอบในแต่ละ Table โดยเริ่มจาก Table แรก จนหมดจำนวนTable ตามจำนวนที่แบ่งชุดข้อมูลจากนั้นนำผลลัพธ์ทั้งที่ค้นหาเจอของแต่ละ Table นำมาทำตรรกะ AND กันหาก Bit ไหนตรง มีค่าเท่ากับ 1 ก็จะถูกเซ็ทเป็น หาก Bit ต่างกันหรือเป็น 0 ก็จะเท่ากับ 0 แล้วจะได้ผลลัพธ์สุดท้ายว่ามี Rule ID ตรงกับข้อใด



รูปที่ 4 การค้นหาใน BIL 2 Table



รูปที่ 5 ผลลัพธ์การค้นหาใน BIL 2 Table การ AND

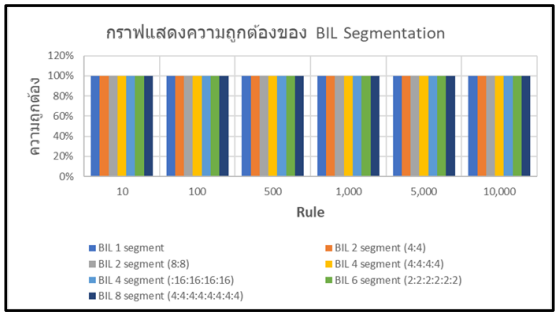
4. การทดลอง

การทดลองของ Bitmap Intersection Lookup จะแบ่งออกได้ดังนี้ โดยเริ่มจากทดสอบความถูกต้อง จากนั้นทดสอบประสิทธิภาพการทำงานโดยวัดความเร็วการทำงาน หน่วยความจำที่ใช้ สุดท้ายเป็นการ

ทดสอบว่าควรแบ่ง Segment อย่างไรจึงจะเหมาะสมที่สุด ในอัลกอริทึมที่เสนอนั้นเป็นการบ่งบอกว่าต้องแบ่งเท่า ๆ กันทุกจึงจะได้หน่วยความจำน้อยที่สุด และมีความเร็วการทำงานดีที่สุด และเหมาะสมต่อหน่วยความจำ หรือตาม Requirement ที่กำหนด จากนั้นนำไปเปรียบเทียบกับทุกรูปแบบที่มีการแบ่ง Segment หากมีหน่วยความจำที่เท่ากัน แล้วการแบ่ง Segment แบบเท่ากันจะมีประสิทธิภาพดีที่สุดหรือไม่

4.1 ความถูกต้อง

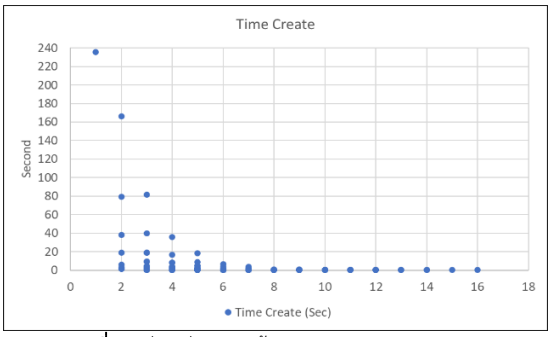
การทดสอบของ Algorithm Linear Search และ BIL Search แบบการแบ่งเพียง 1 Segment และเป็นการค้นหาของกฎเพียงคู่ค้นหาของ Value/Mask เพียงคู่เดียว ทดสอบด้วย 10 rule, 100 rule, 1000 rule, 5000 rule, 10,000 rule



รูปที่ 6 ความถูกต้อง BIL Search แต่ละ Segment

จากการทดลอง 4.1 จากกราฟภาพรวมของความถูกต้องระหว่าง BIL Algorithm ที่มีการแบ่ง Segment เมื่อเปรียบเทียบกับ Linear Search นั้นเห็นได้ว่ามีความถูกต้องเหมือนกันทั้งหมด ทำให้เราสามารถนำอัลกอริทึมนี้ไปพัฒนาเพื่อทดสอบในการแบ่ง Segment ในขั้นต่อไป

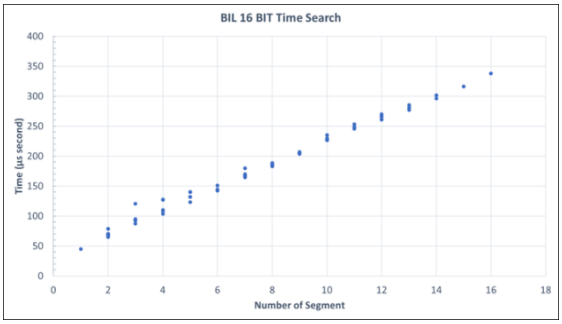
4.2 ความเร็วสร้างตารางทุก Segment



รูปที่ 7 เปรียบเทียบเวลาสร้างตารางทุก Segment 16 Bir

จากการทดลองที่ 4.2 เห็นได้ว่า 1 Segment มีรูปแบบเดียวคือ16 Bit มีเวลาในการสร้างตารางนานที่สุดเนื่องจากใช้ขนาดหน่วยความเท่ากับ 2 กำลัง N Bit หากมี Rule มากจะยิ่งใช้เวลาสร้างตารางมากตาม หากเราแบ่ง Segment มากขึ้นจะเห็นได้ว่าเวลาที่ใช้ในการสร้างตารางจะลดลงตามเนื่องจากมีจำนวน Bit น้อยลง เช่น แบ่ง 2 Segment แบบ [8:8] จะมีขนาดเท่ากับ (2^8) 2 ตารางจะเท่ากับ 512 Index

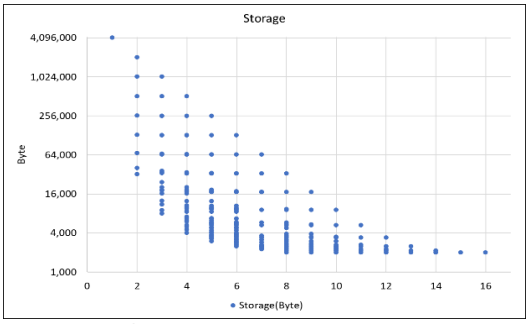
4.3 ความเร็วการค้นหาทุก Segment



รูปที่ 8 เปรียบเทียบความเร็วการค้นหาทุก Segment 16 Bit

จากรูปที่ 8 เห็นได้ว่าการค้นหา 1 Segment จะเร็วที่สุด และจะช้าลงเมื่อมีการแบ่ง Segment มากขึ้นตามลำดับ เนื่องจากการค้นหาตั้งแต่ 2Segment ขึ้นไปต้องนำค่าทุก Segment มา AND กันจึงจะได้คำตอบ ทำให้เวลาช้ากว่า 1 Segment มากขึ้นเรื่อย ๆ ตาม Segment ที่ต้อง AND กัน จะกราฟเห็นได้ว่าทุก Segment เวลาจะไม่ขึ้นอยู่กับจำนวนข้อมูลมากนักแต่จะใกล้เคียงกัน

4.4 หน่วยความจำที่ใช้ทุก Segment



รูปที่ 9 เปรียบเทียบความจำทุก Segment 16 Bit

จากการทดลองที่ 4.4 12 การใช้หน่วยความจำของ 1 Segment จะมีขนาดจำนวน Byte ที่มากที่สุดคือ 4,096,0000 Byte หากแบ่ง Segment ให้จำนวน Bit เล็กลงไปจนถึง 16 Segment ขนาดความจำที่ใช้จะลดลง

ตามลำดับ ความจำที่ใช้ต่ำที่สุดในการแบ่ง Segment ตั้งแต่ 8 Bit ถึง 16 Bit จะใช้ความจำเพียง 2,000 Byte เท่านั้น เราจะได้เห็นได้การแบ่ง Segment นั้นมีผลต่อหน่วยความจำและประสิทธิภาพของ BIL

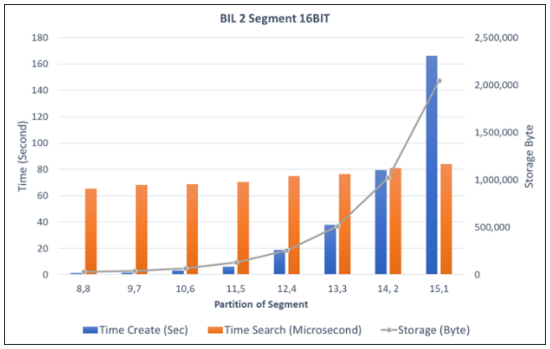
4.5 ทดสอบตาม Requirement

เป็นการทดลองตาม Algorithm ที่เสนอนั้นจะเป็นการบ่งบอกว่าแบ่ง Segment เป็นกี่ Bit แล้วแบ่งจำนวน Segment เท่าใด ซึ่ง Algorithm ที่เสนอนั้นจะเป็นการบ่งบอกว่าแบ่ง Segment เท่าๆ กันจะดีที่สุด จะทำให้ใช้หน่วยความจำน้อยที่สุดและมีประสิทธิภาพในการทำงานทำดีที่สุดทั้งความเร็วการค้นหา ความเร็วการสร้างตาราง ซึ่งเราจะพิสูจน์ตาม Requirement และนำไปเปรียบเทียบกับรูปแบบอื่นที่มีขนาดหน่วยความจำเท่ากันว่าเป็นไปตาม Algorithm ที่เสนอหรือไม่ โดยตัวอย่างโจทย์ที่ใช้ในการทดลอง ดังนี้

1. อุปกรณ์ Arduino มีความจำ 32,000 Byte, 1 Field 16 Bit จำนวน 500 Rule จะแบ่ง Segment อย่างไรจึงมีประสิทธิภาพมากที่สุด

2. มีขนาดหน่วยความจำ 2,000 Byte ต้องแบ่ง Segment เท่าใดจึงจะมีประสิทธิภาพมากที่สุด

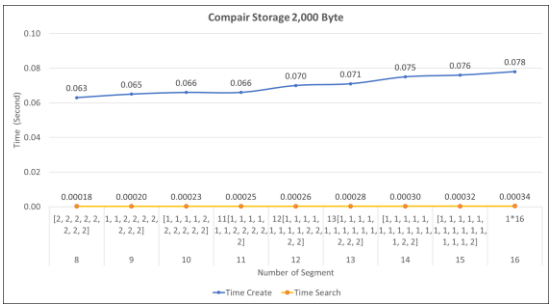
ผลลัพธ์ที่ได้จาก Requirement คือ แบ่งแบบ 2 Segment ขนาดของ Bit เท่ากันจะได้ แบบ [8:8] ใช้หน่วยความจำ 32,000 Byte แล้วนำไปเปรียบเทียบกับการแบ่ง 2 Segment รูปแบบอื่นที่แบ่งไม่เท่ากันจะได้ผลลัพธ์ดังรูปที่ 10



รูปที่ 10 ผลลัพธ์การแบ่ง Segment ตาม Requirement

จากผลการทดลองที่ 4.5 ผลลัพธ์ที่ได้จากอัลกอริทึมที่เสนอนั้น จากโจทย์ Requirement ที่กำหนด ทำให้ได้คำตอบคือ แบ่ง2 Segment โดยแบ่งแบบ (8:8) มีเวลาค้นหา 65 microsecond หรือเท่ากับ 0.000065 วินาที เวลาในการสร้างตาราง 1.24 วินาที และมีความจำ 32,000 Byte ซึ่งใช้เท่ากับหน่วยความจำตามที่กำหนด จากกราฟเมื่อเปรียบเทียบกับรูปแบบอื่น ๆ ทั้งหมดของ 2 Segment ทำให้เราทราบว่าเวลาในการทำงานดีที่สุดและมีหน่วยความน้อยที่สุดซึ่งผลลัพธ์ที่ได้เป็นไปตามอัลกอริทึมที่ได้เสนอ

4.6 เปรียบเทียบ Segment ที่มีความจำเท่ากัน



รูปที่ 11 กราฟเปรียบเทียบ Segment ความจำเท่ากัน

จากกราฟเห็นได้ว่าเวลาในการทำงานที่เร็วที่สุดเมื่อมีหน่วยความจำเท่ากัน คือแบ่ง 8 Segment แบบ [2:2:2:2:2:2:2:2] มีเวลาในการค้นหา 0.00018 วินาที และเวลาสร้างตาราง 0.063 หากแบ่ง 9 Segment ไปจนถึง 16 Segment ที่มีหน่วยความจำเท่ากัน ความเร็วในการทำงานจะใช้เวลามากขึ้นตามลำดับ แต่จะห่างกันไม่มากดังรูปที่ 11

การผลการทดลองทั้งหมดจะเห็นว่าการแบ่ง Segment มีผลต่อหน่วยความและประสิทธิภาพการทำงานของ BIL ในการทดลองตาม Requirement จะได้ผลลัพธ์ที่ดีที่สุดคือแบ่ง Segment เท่าๆ กันและจำนวน Bit เท่ากันจะดีที่สุด

5. สรุปผล

จากผลการทดลองทั้งหมดสรุปได้ว่าวิธีที่เสนอในการแบ่ง Segment ที่บอกว่าหากแบ่ง Segment เท่า ๆ กันและจำนวน Bit เท่าๆ กันจะทำให้ใช้หน่วยความจำน้อยที่สุดและเวลาการทำงานทั้งเวลาการสร้างตาราง เวลาการค้นหาดีที่สุด จะเห็นได้จากการทดลองที่กำหนด Requirement ซึ่งผลลัพธ์ที่ได้จาก Algorithm ที่เสนอทำให้ได้การแบ่ง Segment ที่มีประสิทธิภาพมากที่สุดแต่จะไม่เกินความจำที่กำหนดตาม Requirement ซึ่งเป็นไปตามอัลกอริทึมที่เสนอ โดย BIL Algorithm สามารถนำไปใช้กับ Firewall เพื่อหาตัวที่ Match กันได้ หรือนำไปใช้ในการจัดการทรัพยากรในระบบต่าง ๆที่มี

จำกัดและยังนำไปใช้ในจัดกลุ่มข้อมูลเพื่อลดเวลาในการทำงานและเพิ่มประสิทธิภาพและรวดเร็วได้ ถ้าจะนำไปใช้งานจริงควรคำนึงถึงขนาดความจำของเครื่องเพราะประสิทธิภาพของ BIL จะขึ้นอยู่กับขนาดความจำถ้ามีความจำมากประสิทธิภาพจะมากตาม

เอกสารอ้างอิง

[1] Akharin Khunkitti, Nuttachot Promrit. "**Bitmap Intersection Lookup (BIL) : A Packet Classification 's Algorithm with Rules Updating**." [online]. available: https://www.koreascience.or.kr/article/CFKO200533239340120.pdf. 2005

[2] วรเชษฐ์นิ่มเจริญ, สุภัควี สุโพธิ์ และ อัครินทร์ คุณกิตติ. "**A Study and Development of Bitmap Intersection Lookup Algorithm for Packet Classification.**" The 18th National Conference on Computing and Information Technology. ปีที่ 18. ฉบับที่ 1. 19-20 พฤษภาคม 2022 หน้าที่ 448 - 453

[3] Javatpoint. "**Linear Search Algortithm.**" [online]. available: https://www.javatpoint.com/linear-search. 2011

[4] Tutorialspoint. "**C-Programming Tutorial**." [online]. available: https://www.tutorialspoint.com/cprogramming/c_strings.htm.

[5] Benznest. "**เขียนโปรแกรมภาษา C แบบพื้นฐาน.**" [ออนไลน์]. เข้าถึงได้จาก: https://benzneststudios.com/blog/c-programming/c-programming-basic-3/. 2559

[6] Python Tutorial. "**Basic python.**" [online]. Available. https://www.geeksforgeeks.org g/python-lists/. 2023

[7] GeeksforGeeks. "**Python Lists**". [online]. Available: https://www.geeksforgeeks.org/python-lists/. 2023

[8] Bartosz Zaczynski. "**Bitwise Operator in Python**". [online]. Available: https://realpython.com/python-bitwise-operators/. 2021

[9] John Strutz. "**Dictionaries in Python.**". [online]. Available: https://realpython.com/python-dicts/. 2019

การพัฒนาระบบตรวจจับการบุกรุกสำหรับบ้านอัจฉริยะ

กฤติยาณี โรจนประดิษฐ์[1] และ มนัสวี พึ่งสุนทรบัตร์[2]

[1]คณะเทคโนโลยีสารสนเทศ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง กรุงเทพฯ

Emails: 62070004@it.kmitl.ac.th, 62070156@it.kmitl.ac.th

บทคัดย่อ

ในปัจจุบันมีการนำอุปกรณ์อิเล็กทรอนิกส์เข้ามาช่วยเพิ่มประสิทธิภาพการดูแลความปลอดภัยของที่อยู่อาศัยกันอย่างแพร่หลาย ผู้พัฒนาจึงเล็งเห็นว่าสามารถเพิ่มประสิทธิภาพให้กับกล้องวงจรปิดประเภท IP Camera ในการตรวจจับผู้บุกรุกได้ โดยการนำมาเชื่อมต่อกันเป็นระบบ Smart Home ผ่านซอฟต์แวร์เพื่อใช้เพิ่มระดับความปลอดภัยให้กับที่อยู่อาศัยโดย ในโครงงานนี้ผู้พัฒนาได้พัฒนาโปรแกรมตรวจจับผู้บุกรุก โดยเริ่มจากการเก็บข้อมูลชื่อและภาพใบหน้าของสมาชิกภายในบ้านแล้วจึงนำข้อมูลที่ได้ไปใช้ในการฝึกโมเดลด้วยอัลกอริทึมของ Local Binary Patterns Histogram หลังจากนั้นจึงจัดเก็บโมเดลที่ได้ลงเครื่องเพื่อไว้เปรียบเทียบกับบุคคลที่เข้ามาในบ้านซึ่งในการตรวจจับผู้บุกรุกจะรับภาพสตรีมจากกล้องวงจรปิดผ่าน RTSP Protocol เมื่อมีคนเข้ามาในบ้านจะใช้เทคโนโลยี Dlib ในการแยกแยะใบหน้าว่าเป็นบุคคลที่ทำกาลงทะเบียนเข้ามาหรือไม่ แต่ในกรณีที่สภาพแวดล้อมในขณะนั้นค่อนข้างมืดหรือในกรณีที่บุคคลที่เข้ามามีการปกปิดใบหน้า ทำให้ระบบไม่สามารถจับใบหน้าเพื่อตรวจสอบว่าเป็นใครได้ ดังนั้นผู้พัฒนาจึงเพิ่มทาง เลือกให้กับระบบโดยการใช้โมเดลของ Caffe มาช่วยในการตรวจจับวัตถุเพื่อใช้ระบุว่าเป็นมนุษย์ที่เข้ามาในบ้านหรือไม่และผลลัพธ์ทั้งหมดจะทำการแจ้งเตือนไปยังแอปพลิเคชันไลน์

จากการพัฒนาและการทดสอบประสิทธิภาพของระบบในการตรวจจับผู้บุกรุกนั้นก็จะเห็นว่าระบบสามารถนำไปใช้งานได้จริง ซึ่งจากผลการทดสอบพบว่าระบบมีความแม่นยำในการตรวจจับประมาน 93.3% และในกรณีที่ต้องการจะตรวจจับใบหน้าในสภาพแวดล้อมที่ค่อนข้างมืดอาจจะทำให้ไม่สามารถตรวจจับได้ ดังนั้นการจะนำระบบไปใช้งานควรจะต้องเพิ่มความสว่างให้กับพื้นที่บริเวณที่จะติดตั้งกล้องวงจรปิด เพื่อให้ระบบของเราสามารถตรวจจับผู้บุกรุกได้อย่างมีประสิทธิภาพ

คำสำคัญ – บ้านอัจฉริยะ; ตรวจจับผู้บุกรุก; ระบบรักษาความปลอดภัย; *DETECTION SYSTEM; SMART HOME*

1. บทนำ

ในปัจจุบันการมีอุปกรณ์อิเล็กทรอนิกส์ที่ช่วยเพิ่มความสะดวกสบายและความปลอดภัยในบ้านมีการใช้งานกันอย่างแพร่หลาย แต่ถ้าหากใช้อุปกรณ์หลายๆ ชิ้นแทนที่จะสะดวกสบายกลับทำให้ยากต่อการควบคุม เนื่องจากอุปกรณ์แต่ละชิ้นไม่ได้เชื่อมต่อถึงกันหรือบางครั้งมีฟังก์ชันการใช้งานที่ไม่เพียงพอต่อความต้องการ ดังนั้นผู้จัดทำจึงเล็งเห็นว่าสามารถนำอุปกรณ์อิเล็กทรอนิกส์ต่าง ๆ มาเชื่อมต่อกันเป็นระบบ Smart Home โดยเชื่อมต่อกันผ่านซอฟต์แวร์ เพื่อให้สามารถตรวจสอบและควบคุมได้ง่ายขึ้น โดยจะนำมาประยุกต์มาใช้ในการตอบสนองความต้องการในการเพิ่มระดับความปลอดภัยมากขึ้นในการตรวจจับผู้บุกรุก และมีการเตือนภัยในบ้านทำให้ผู้อยู่อาศัยภายในบ้านมีการเตรียมตัวรับมือกับผู้บุกรุกได้ทันที ทั้งนี้เสียงที่เตือนภัยยังจะทำให้ผู้ที่อาศัยอยู่ในบริเวณใกล้เคียงมาช่วยเหลือได้อีกด้วย

2. ทฤษฎีและเทคโนโลยีที่ใช้ในการพัฒนาระบบตรวจจับผู้บุกรุก

2.1. การประมวลผลภาพ

การประมวลผลภาพ (Image Processing) เป็นการประยุกต์ใช้งานการประมวลผลสัญญาณบนสัญญาณ 2 มิติ เช่นภาพนิ่ง หรือ ภาพวีดีทัศน์ โดยใช้คอมพิวเตอร์ในการประมวลผล โดยมีวัตถุประสงค์เพื่อปรับปรุงคุณภาพของรูปภาพ

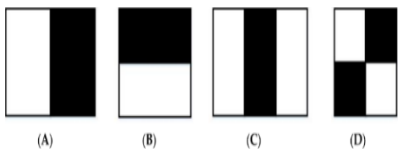
2.2. การตรวจจับวัตถุ (Object Detection)

การตรวจจับวัตถุ (Object Detection) คือ เทคโนโลยีที่เกี่ยวข้องกับ Computer Vision และ Image Processing ที่ใช้ในงาน AI ตรวจจับวัตถุชนิดที่กำหนด เช่น รถยนต์ อาคาร มนุษย์ ที่อยู่ในวิดีโอ หรือรูปภาพ โดยใช้ Caffe เป็นเครื่องมือสำหรับ deep learning ที่ได้รับความนิยมที่ออกแบบมาสำหรับการสร้างแอพ

2.3. การตรวจจับใบหน้า (Face Detection)

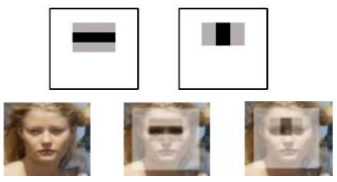
การตรวจจับใบหน้า (Face Detection) เป็นการค้นหาใบหน้าบุคคลจากรูปภาพต่างๆ หรือวีดิโอที่ได้รับมาจากนั้นนำมาวิเคราะห์ค้นหาแค่ใบหน้าต่อเพื่อให้ง่ายต่อการจำแนกในขั้นถัดไป โดยใช้เทคนิคการตรวจจับใบหน้าใหม่ โดยใช้การจำลองรูปแบบ Haar-like ซึ่งเป็นตัวตรวจจับทำการปรับขนาดของตัวตรวจจับแทนการปรับขนาดของภาพ

การตรวจจับและตีความวัตถุที่อยู่ในภาพจะใช้วิธีการสร้างรูปเหลี่ยม (Feature) ที่แสดงถึงผลต่างระหว่างพื้นที่ส่วนที่เป็นสีขาวและส่วนที่เป็นสีดำ รูปสี่เหลี่ยมที่สร้างขึ้นสามารถเปลี่ยนแปลงขนาดและตำแหน่งได้ ใช้สำหรับการตรวจจับลักษณะเด่นบนภาพแบบต่างๆ การตรวจจับและตีความหมาย



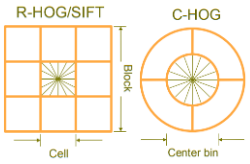
รูปที่ 1 ตัวอย่างการจำลองรูปแบบ Haar-like

ถัดมานำการจำลองรูปแบบ Haar-like ที่ได้จากขั้นตอนแรกมาเข้ากระบวนการการเรียนรู้ เรียกว่า “Adaptive Boost” ซึ่งแนวคิดหลักของ AdaBoost คือการเปลี่ยน Weak Model ให้กลายเป็น Strong Model ประมวลผลวิเคราะห์ข้อมูลกลุ่ม Data Set เดียวกันโดยใช้โมเดลเดียวกัน



รูปที่ 2 ตัวอย่างการจำแนกได้จากกระบวนการ AdaBoost

และเทคโนลียีที่นำมาปรับใช้กับ model การตรวจจับใบหน้าด้วย Histogram of Oriented Gradients (HOG) วิธีการสกัดคุณลักษณะเด่นของภาพใบหน้าโดยการวิเคราะห์ค่าความถี่ของทิศทางตามค่าเกรเดียนท์ โดยใช้การจะจายตัวของความเข้มข้นของทิศทางเส้นขอบ แล้วแบ่งภาพเป็นภาพย่อย โดยวิธีการแบ่งภาพเป็นภาพย่อยแบ่งได้ 2 แบบ คือแบบสี่เหลี่ยม และแบบวงกลม

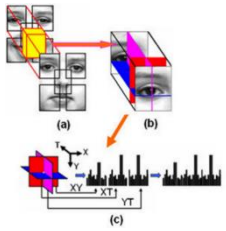


รูปที่ 3 แสดงการแบ่งภาพย่อย

2.4. การรู้จำใบหน้า (Face Recognition)

การรู้จำใบหน้า (Face Recognition) เป็นการนำภาพใบหน้าจากการวิเคราะห์ของขั้นตอน ตรวจจับใบหน้า (Face Detection) ที่ได้มาเปรียบเทียบกับฐานข้อมูลที่ได้ทำการบันทึกไว้ ว่าใบหน้าที่ตรวจจับได้ตรงกับใบหน้าไหนในฐานข้อมูลที่บันทึก โดยใช้เทคโนโลยี Biometrics ในการแยกแยะลักษณะต่างๆบนใบหน้า

เทคนิคการระบุตัวตนโดยอาศัย Local binary pattern โดยเป็นพื้นผิวแบบสองมิติมีหลักการทำงานโดยแบ่งภาพออกเป็นส่วนๆ และเปรียบเทียบสเกลสีเทา โดยจะพิจารณาเป็นเลขฐานสองคำนวณค่าไบนารีความยาว 8 บิท โดยนำค่าที่คำนวนได้มาทำ Histogram



รูปที่ 4 การแปลงจากค่าพิกเซลเป็นค่า LBP

2.5. Rapid Spanning Tree Protocol (RSTP)

เป็นโปรโตคอลที่ใช้รูปแบบ client/server ที่ถูกออกแบบเพื่อใช้ในการแสดงสื่อมัลติมีเดีย สำหรับ Real Server เวอร์ชั่นใหม่ RTSP จะสนับสนุน SureStreamTM

ซึ่งจะสามารถเลือกที่จะส่งข้อมูลที่อัตราความเร็วสูงที่สุดในขณะนั้นโดยอัตโนมัติ

2.6. Machine Learning

การสอนให้ระบบคอมพิวเตอร์ทำการเรียนรู้ได้ด้วยตนเองโดยการใช้ข้อมูล โดยมี Data Scientist เป็นผู้ออกแบบ ส่วนการเรียนรู้ของเครื่อง ถูกใช้งานเสมือนเป็นสมอง เรียนรู้จากสิ่งที่เราส่งเข้าไปกระตุ้น แล้วจดจำเอาไว้เป็นมันสมอง ส่งผลลัพธ์ออกมาเป็นตัวเลข หรือ Code ที่ส่งต่อไปแสดงผลสามารถเอาไปใช้งานได้หลายรูปแบบ

2.7. OpenCV (Open source Computer Vision)

เป็นไลบรารีสำหรับการประมวลผลภาพขั้นพื้นฐาน ที่ถูกพัฒนาขึ้นโดยบริษัท Intel เพื่อใช้ในการพัฒนาระบบ Open Source ที่นำมาใช้สร้าง Machine Learning หรือ AI โดยส่วนใหญ่จะมุ่งเป้าไปที่การแสดงผลด้วยคอมพิวเตอร์แบบ

2.8. SQLite

เป็น Library ภาษา C สำหรับจัดการ Database ถือว่าเป็นโปรแกรมฐานข้อมูลที่มีขนาดเล็กเนื่องจากมีขนาดไม่ถึง 1 MB เก็บฐานข้อมูลเป็นไฟล์โดยไม่จำเป็นต้องมีเซิร์ฟเวอร์ โดยจะเป็นเก็บเป็นไฟล์นามสกุล .db โดยมีความน่าเชื่อถือสูง มีคุณสมบัติครบถ้วน มีความเสถียรในการใช้ข้ามแพลตฟอร์ม SQLite เป็นเอ็นจินฐานข้อมูล SQL แบบฝังตัว อ่านและเขียนโดยตรงไปยังไฟล์ทั่วไป มีการระหว่างการใช้แลกเปลี่ยนหน่วยความจำได้รวดเร็ว

2.9. Tkinter

เป็น Library สำหรับการพัฒนา GUI ที่มากับ python เป็น standard library และเป็น GUI library ที่ได้รับความนิยมเนื่องจากสามารถใช้ได้ทั้งในระบบปฏิบัติการ MacOS, Linux, Windows และสามารถเรียนรู้ได้ง่าย ไม่ซับซ้อน ทำงานได้รวดเร็ว Tkinter เป็นวิธีที่ง่ายในการสร้างแอปพลิเคชัน GUI มีวิดเจ็ตในการควบคุมมากมาย มีข้อเสียคือต้องใช้หลายขั้นตอนในการออกแบบ

2.10. Line notify

เป็นบริการรับการแจ้งเตือนจากบัญชีทางการในรูปแบบ API สำหรับนักพัฒนาซอฟต์แวร์ นำไปใช้ต่อยอดพัฒนาโปรเจคต่างๆ เชื่อมต่อกับเว็บเซอร์วิส เช่น GitHub IFTTT และ Mackerl สร้างการแจ้งเตือนแบบข้อความไปยังกลุ่มหรือบัญชีส่วนตัวได้โดยไม่เสียค่าใช้จ่าย ยกเว้นกรณีที่เชื่อมต่อกับเว็บเซอร์วิสอื่นๆ

2.11. NumPy

เป็น Library ที่เป็นโอเพ่นซอร์สที่มีฟังก์ชันไว้คำนวนตัวเลข ด้วย Python โดยส่วนมากจะใช้ในการคำนวนเมทริกซ์ และจัดการอาร์เรย์ทำงานได้รวดเร็ว และการประหยัด Memory ทำงานใกล้เคียง MATLAB นิยมในการใช้งานใน Data Science และ Machine Learning

2.12. Pillow

เป็น Library ที่เป็นโอเพ่นซอร์สมีโมดูลจัดการและประมวลผลรูปภาพหลายรูปแบบบน Python สามารถใช้งานได้ทั้งบน Windows, Mac OS X และ Linux และ Pillow ถูกออกแบบมาเพื่อให้เข้าถึงข้อมูลที่ถูกจัดเก็บในรูปแบบพิกเซลได้อย่างรวดเร็ว

2.13. DLIB

DLIB คือไลบรารี่แบบ Open-Source ที่เก็บอัลกอริทึมที่เกี่ยวข้องกับการเรียนรู้ของเครื่อง (Machine Learning) การประมวลผลภาพ (Image Processing) พีชคณิตเชิงเส้น (Linear Algebra) การทำเหมืองข้อมูล (Data Mining) ด้าน เครือข่ายระบบ (Networking) การบีบอัดข้อมูล (Data Compression) การสร้างกราฟิกบนส่วนติดต่อกับผู้ใช้งาน (Graphical User Interfaces) และอัลกอริทึมเชิงตัวเลข (Numerical Algorithms)

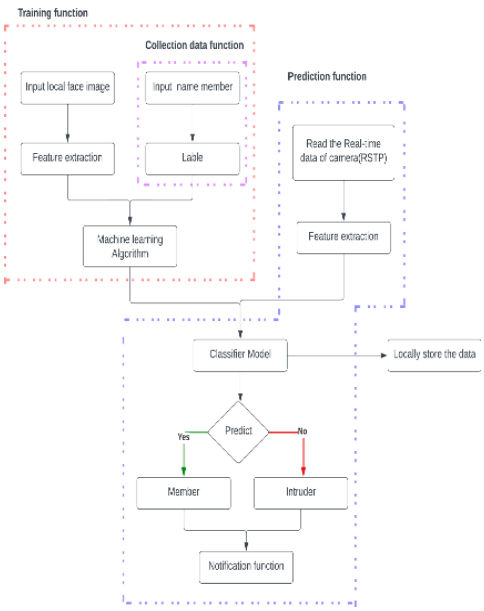
3. การออกแบบระบบและความต้องการของระบบตรวจจับผู้บุกรุก

3.1. ความต้องการของระบบตรวจจับผู้บุกรุก

การพัฒนาระบบเริ่มมาจากปัญหาในการถูกโจรกรรมทรัพย์สินภายในบ้าน ดังนั้นจึงมีความต้องการที่จะเพิ่มระดับความปลอดภัยให้มากขึ้นในการตรวจจับผู้บุกรุก โดยจะให้มีการเตือนภัยไปยังผู้อยู่อาศัยภายในบ้านโดยตรง จะได้มีการเตรียมตัวรับมือกับผู้บุกรุกได้ทันที

โดยระบบทำให้เราสามารถทราบได้เมื่อมีคนเข้ามาบริเวณบ้านของเรา และสามารถระบุตัวตนได้ว่าบุคคลที่เข้ามาเป็นบุคคลใด เป็นคนในบ้านหรือไม่ หากไม่ใช่ให้สันนิฐานว่าเป็นผู้บุกรุกไว้จากนั้นแจ้งเตือนไปยังไลน์ แต่ในกรณีที่มีคนเข้ามาบริเวณบ้านของเรา แต่ถ้าสภาพแวดล้อมในขณะนั้นค่อนข้างมืดหรือในกรณีที่บุคคลที่เข้ามามีการปกปิดใบหน้า เช่น สวมแมส หรือ หมวกกันน็อค ทำให้ระบบไม่สามารถจับใบหน้าได้อาจจะทำให้ระบบของเราไม่สามารถระบุตัวตนได้ว่าคนที่เข้ามาเป็นใคร แต่ยังมีการแจ้งเตือนไปยังไลน์ว่ามีบุคคลที่ไม่สามารถระบุตัวตนได้เข้ามาในบ้านทำให้คนในบ้านสามารถทราบได้ว่ามีคนเข้ามา

3.2. ภาพรวมของระบบ



รูปที่ 5 ภาพรวมของระบบ

3.2.1 Collection data function

การทำงานของ Collection function จะเริ่มต้นจากการทำการกรอกชื่อผู้ใช้ผ่านหน้า Interface หลังจากนั้นจะเก็บข้อมูลที่กรอกเข้ามาไปทำการบันทึกลงฐานข้อมูล ซึ่งฐานข้อมูลที่เรานำมาใช้คือ SQLite หลังจากที่ระบบทำการตรวจสอบว่าผู้ใช้งานได้กรอกข้อมูลถูกต้องตามเงื่อนไขคือต้องเป็นตัวอักษรเท่านั้น ระบบจึงจะสามารถดำเนินงานต่อไป หลังจากที่ทำการกรอกข้อมูลผ่านแล้วตัวระบบจะทำการเปิดกล้องอัตโนมัติ จากนั้นระบบจะไปโหลด HAAR CASCADE มาเพื่อใช้ในการตีกรอบใบหน้า หลังจากเพื่อทำการบันทึกใบหน้าที่ตรวจจับได้ลงโฟลเดอร์และไปการเรียกใช้ Training function ในลำดับถัดไป

3.2.2. Training function

ในฟังกชันนี้จะไปทำการเรียกโฟลเดอร์ที่เก็บรูปมาลูป หลังจากนั้นก็ทำการแยกชื่อ path เพื่อดึง Id ออกมา แล้วจึงนำรูปภาพแต่ละรูปในโฟลเดอร์มาแปลงเป็น Array โดยจะทำการ Append Id เข้าไปด้วยเพื่อใช้ในการฝึกโมเดลหลังจากที่ฝึกเสร็จก็จะจัดเก็บโมเดลลงเครื่องในรูปแบบไฟล์ของ .yml

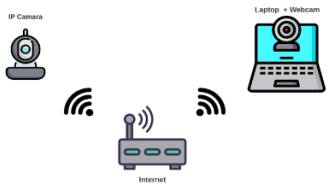
3.2.3. Prediction function

หลักการทำงานก็คือรับค่าจากกล้องวงจรปิดเข้ามา แล้วทำการ Feature Extraction หลังจากนั้นก็โหลดโมเดลมาใช้งาน เพื่อใช้ในการคาดเดาว่าใบหน้าที่ตรวจจับได้เป็นสมาชิกภายในบ้านหรือเป็นผู้บุกรุก แล้วจึงส่งไปที่ Notification function เพื่อทำการแจ้งเตือนไปยังแอพพลิเคชันไลน์

3.2.4. ทางเลือกอื่นของระบบ

ในกรณีที่สภาพแวดล้อมในขณะนั้นค่อนข้างมืดหรือในกรณีที่บุคคลที่เข้ามามีการปกปิดใบหน้า ทำให้ระบบไม่สามารถจับใบหน้าเพื่อตรวจสอบว่าเป็นใครได้ ผู้จัดทำจึงได้คิดหาวิธีแก้ไขโดยแทนที่จะให้ตัวระบบจับใบหน้าเพื่อแยกแยะก็ให้ไปตรวจสอบแทนว่าพบวัตถุที่เป็น ‘มนุษย์’ เข้ามาในบ้านหรือไม่แทน แล้วจึงทำการแจ้งเตือนไปยังแอพพลิเคชันไลน์ โดยโมเดลที่นำมาใช้จะเป็นโมเดลสำเร็จรูปของ Caffe ที่ใช้ในการตรวจจับวัตถุ

3.3. ทรัพยากรระบบการตรวจจับผู้บุกรุก



รูปที่ 6 อุปกรณ์ในระบบการตรวจจับผู้บุกรุก

ตารางที่ 1 คุณสมบัติของระบบการตรวจจับผู้บุกรุก

ลำดับ	อุปกรณ์
1	เครื่องโน๊ตบุ๊ค + Webcam สำหรับประมวลผล จำนวน 1 เครื่อง • CPU : AMD Ryzen 5 3550H • RAM : 8GB DDR4 • Storage : HDD 1TB • OS : Windows 10
2	กล้องวงจรปิด – IP Camera จำนวน 1 ตัว • ความคมชัด : 5MP • การเชื่อมต่อ : Wi-Fi 802.11bgn • สามารถเห็นใบหน้าได้ภายใน 10 เมตร
3	Router wifi – สำหรับเชื่อมต่อ Internet

Library ที่ต้องติดตั้งลงเครื่อง

Python == 3.10.8

numpy==1.19.5

opencv-contrib-python==4.6.0.66

opencv-python==4.6.0.66

Pillow==8.4.0

line-notify==0.1.4

dlib==19.22.99

emoji==2.2.0

cmake==3.26.3

wheel==0.37.1

โปรแกรมสำหรับการพัฒนาระบบการตรวจจับผู้บุกรุก

โปรแกรม VSCODE (Visual Studio Code) ใช้ในการพัฒนาระบบการตรวจจับผู้บุกรุกโดยนำมาเป็นชุดควบคุมโดยใช้ภาษา Python .ในการเขียนคำสั่งในการควบคุมการทำงานของระบบตรวจจับผู้บุกรุกเนื่องจากมี interface ที่ใช้งานง่ายและสะดวก

3.5. การจัดเก็บข้อมูลระบบการตรวจจับผู้บุกรุก

การจัดเก็บข้อมูลจะมีการแบ่งข้อมูลออกเป็น 2 ส่วน

1. ในส่วนของ ชื่อผู้ใช้ในการลงทะเบียน โดยในส่วนนี้จะมีการเก็บที่ SQLite ซึ่งจะมีรหัสผู้ใช้จะเก็บเป็นตัวเลขจำนวนเต็มและชื่อผู้ใช้จะเก็บเป็นตัวอักษรภาษาอังกฤษ

2. ในส่วนของ รูปภาพและวีดีโอ โดยในส่วนนี้จะมีการเก็บไปที่ Computer Memory โดยรูปภาพจะมีการเก็บเป็นไฟล์ .jpg ก่อนจะนำไปแปลงเป็นอาร์เรย์เพื่อนำไปใช้ต่อ

3.4. การพัฒนาระบบการตรวจจับผู้บุกรุก

ตารางที่ 2 ตารางการพัฒนาระบบการตรวจจับผู้บุกรุก

1	GUI Assets	- Entry - Button
2	เก็บข้อมูล Users	- กรอกชื่อ - เปิดกล้องเก็บรูปลงโฟลเดอร์
3	Config (RTSP)	- กรอก link RTSP
4	Config Token Line	- กรอก token line
5	Save Data (Config)	- นำข้อมูลที่ได้จากการ config มาใช้งาน
6	Config Mac Address	- ดึงจาก txt -> list
7	Bluetooth	- กรอก mac address - scan หาเครื่องที่เปิดบลูทูธใกล้เคียง
8	Detection Face	- ตรวจจับใบหน้าว่าเป็น 'ใคร' เข้ามาในบ้าน
9	Model	- เอารูปกับข้อมูลชื่อ tag กันมาเทรนเป็น model
10	Predict	- ทำนายว่าเป็นสมาชิกหรือผู้บุกรุก
11	Notification	- แจ้งเตือนไปยังไลน์
12	Manage Data	- เปลี่ยนชื่อ - ลบข้อมูล
13	Interface (All)	- ทุกฟังก์ชันมี interface รองรับ
14	Model (Object)	- model ที่ใช้แยกวัตถุ (สำเร็จรูป ของ caffe)
15	Detection Object	- ตรวจจับว่ามีคนเข้ามาในบ้าน
16	Setup Library	- เซ็ตไลบลารี่ต่างๆ

ในการพัฒนาในส่วนของ GUI Assets ก็จะทำการออกแบบใน Figma ก่อน โดยข้อมูลที่ต้องจัดเก็บเข้าระบบ หลังจากนั้นในข้อ 8 และ 15 จากตารางที่ 2 ตารางการพัฒนาระบบการตรวจจับผู้บุกรุก จะเป็นการนำ Dlib มาใช้ในการตรวจจับโดยจะมีโมเดลทั้งที่ฝึกเองและที่เป็นสำเร็จรูปมาช่วยในการตรวจสอบหาผู้บุกรุกและจึงทำการแจ้งเตือนไปยังแอพพลิเคชันไลน์

4. การทดสอบและการวิเคราะห์

4.1. วิธีในการทดสอบประสิทธิภาพ

แบ่งออกเป็น 2 ชุด

<u>การทดสอบชุดที่ 1</u> การทดสอบการตรวจจับใบหน้า

เป็นการทดสอบเพื่อเปรียบเทียบเทคโนโลยี Dlib โดยใช้เทคนิค HOG และ เทคโนโลยี HAAR CASCADE โดยใช้เทคนิค LBPH เพื่อนำมาใช้ใน model ให้มีประสิทธิภาพสูงสุดในการตรวจจับผู้บุกรุก โดยทดสอบการตรวจจับใบหน้า ว่าสามารถตรวจจับใบหน้าในอิริยาบทต่างๆ และมีจำนวนคนในกล้องมากกว่า 1 คน สามารถจับได้หรือไม่มาประยุกต์ใช้ในโมเดลที่ได้สร้างขึ้น

O ขั้นตอนในการทดสอบ

1. เตรียมรูปภาพสำหรับการทดสอบทั้งหมด 15 รูปภาพ ที่มีคนอยู่ในรูปภาพไม่น้อยกว่า 1 คน
2. นำรูปภาพทั้ง 15 รูปภาพมาแสดงในโปรแกรม
3. ใช้เทคโนโลยี Haar cascade โดยใช้เทคนิค LBPH ในการทดสอบก่อน
4. ใช้เทคโนโลยี Dlib โดยใช้เทคนิค HOG ในการทดสอบ
5. บันทึกผลการทดสอบการตรวจจับใบหน้า โดยถ้าจับได้จะขึ้นกรอบรูปบริเวณใบหน้า
6. สรุปผลการทดสอบการตรวจจับใบหน้าระหว่างเทคโนโลยี Dlib และ Haar cascade

<u>การทดสอบชุดที่ 2</u> การทดสอบการตรวจจับผู้บุกรุก

เป็นการทดสอบประสิทธิภาพในการตรวจจับผู้บุกรุก และผู้ที่ได้ลงทะเบียนว่ามีความแม่นยำมากน้อยเพียงใด โดยใช้เทคโนโลยีที่ได้ทำการเปรียบเทียบว่ามีประสิทธิภาพดังนั้นจึงนำมาปรับใช้กับ model

oขั้นตอนในการทดสอบ

1. ลงทะเบียนบุคคลในบ้านโดยกรอกชื่อและแสดงใบหน้าผ่านกล้อง Webcam
2. โปรแกรมจะทำการจับใบหน้าและครอปภาพเพื่อไป TRAIN MODEL
3. เปิดตัวสตรีมที่รับภาพแบบเรียลไทม์ผ่านกล้องวงจรปิดเพื่อจับภาพใบหน้า
4. โดยให้ผู้ที่ลงทะเบียน(คนในบ้าน) เดินผ่านกล้องจำนวน 50 ครั้ง
5. ถัดมาให้ผู้ที่ไม่ลงทะเบียน(ผู้บุกรุก) เดินผ่านกล้องจำนวน 50 ครั้ง
6. ระบบจะทำการจับใบหน้าและแจ้งเตือนผ่านไลน์ว่าเป็นใคร
7.บันทึกผลการทดลอง

4.2. ผลการทดสอบ

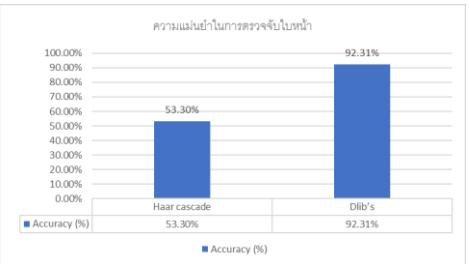
<u>การทดสอบชุดที่ 1</u> การทดสอบการตรวจจับใบหน้า

เป็นการทดสอบการตรวจจับใบหน้าโดยเปรียบเทียบระหว่างเทคโนโลยี Dlib และ เทคโนโลยี HAAR CASCADE เพื่อเปรียบเทียบประสิทธิภาพในการตรวจจับใบหน้า เพื่อนำมาใช้ใน model ให้มีความแม่นยำในการตรวจจับใบหน้า

ตารางที่ 3 ตารางผลการทดลองระหว่างเทคโนโลยี

เครื่องมือที่ใช้ตรวจจับใบหน้า / ใบหน้าที่ตรวจจับได้	Face Detection : Haar cascade [15]	Face Detection : Dlib [15]
จับใบหน้าได้ครบถ้วน	7	10
จับใบหน้าได้บางส่วน	1	2
จับใบหน้าไม่ได้เลย	7	1
Accuracy (%)	53.3	92.31

รูปที่ 6 กราฟแสดงความแม่นยำในการตรวจจับใบหน้า

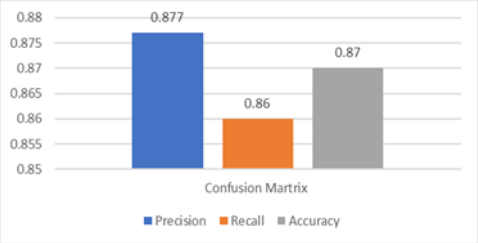


ค่าจากกราฟเป็นการคำนวนจากสมการ ค่าความแม่นยำ (Accuracy) จากผลที่ได้จากการทดลอง

<u>การทดสอบชุดที่ 2</u> การทดสอบการตรวจจับผู้บุกรุกเป็นการทดสอบการตรวจจับใบหน้าของผู้ที่ลงทะเบียน และไม่ได้ลงทะเบียนจะเท่ากับเป็นผู้บุกรุก เป็นการทดสอบประสิทธิภาพของ MODEL เพื่อทดสอบประสิทธิภาพในการตรวจจับผู้บุกรุกว่า MODEL มีการประสิทธิภาพเพียงใด โดยใช้เทคโนโลยี DLIB

ตารางที่ 4 ตารางผลการทดลองการตรวจจับผู้บุกรุก

ภาพที่ป้อนเข้ามา / ภาพที่ตรวจจับได้	ผู้บุกรุก [50]	สมาชิก [50]
ผู้บุกรุก	43	6
สมาชิก	7	44
Accuracy (%)	93.3%	
Recognition Rate (%)	76.6%	80%



รูปที่ 7 กราฟแสดงความแม่นยำในการตรวจผู้บุกรุก

4.3. วิเคราะห์ผลการทดสอบประสิทธิภาพ

จากการทดสอบชุดที่ 1

เป็นการเปรียบเทียบประสิทธิภาพระหว่างระหว่างเทคโนโลยี DLIB และ เทคโนโลยี HAAR CASCADE ได้ผลสรุปว่า เทคโนโลยี DLIB มีความแม่นยำในการตรวจจับใบหน้ามากกว่าเทคโนโลยี HAAR CASCADE เพราะ เทคโนโลยี DLIB มีการตรวจจับในกรณีที่บุคคลมีการเคลื่อนไหวที่รวดเร็ว ซึ่งตรงข้ามกับเทคโนโลยี HAAR CASCADE ซึ่งเป็นไปตามที่คาดไว้จากการศึกษาหลักการทำงานของแต่ละเทคโนโลยีซึ่งเทคโนโลยี DLIB มีเทคนิค HOG ที่มีการตรวจใบหน้าที่ละเอียดกว่า เทคโนโลยี HAAR CASCADE

จากการทดสอบชุดที่ 2

พบว่าการตรวจจับใบหน้าของเทคโนโลยี DLIB ที่นำมาใช้ใน MODEL ค่อนข้างมีความแม่นยำสามารถใช้ในการตรวจจับผู้บุกรุกได้ แต่อาจจะมีข้อผิดพลาดเล็กน้อย การทดสอบประสิทธิภาพของโครงงานในการตรวจจับผู้บุกรุกโดยนำผลการทดสอบที่ได้มาคำนวนในสมการได้ประสิทธิภาพความแม่นยำประมาน 93.3% จากการใช้เทคโนโลยี DLIB ที่ใช้เทคนิค HOG ซึ่งอาจจะมีความคลาดเคลื่อนจากที่คาดไว้เพียงเล็กน้อย แต่เนื่องจากระบบของโครงงานเรามีกล้อง IP CAMARA เป็นเซ็นเซอร์ในการตรวจจับผู้บุกรุกเพียงอย่างเดียว ทำให้ประสิทธิภาพของการตรวจจับผู้บุกรุกยังมีข้อผิดพลาดเล็กน้อยซึ่งเป็นที่ยอมรับได้

4.4. สรุปผลการทดสอบประสิทธิภาพ

จากการทดสอบประสิทธิภาพในด้านการตรวจจับบุคคลทำให้เราเลือกใช้เทคโนโลยี DLIB ที่ใช้เทคนิค HOG ในการตรวจจับใบหน้าเนื่องจากมีความแม่นยำมากกว่าเทคโนโลยี HAAR CASCADE ที่ใช้เทคนิค LBPH แต่ประสิทธิภาพในการตรวจจับใบหน้าของเทคโนโลยี DLIB ที่ใช้เทคนิค HOG จากการทดสอบพบว่าสามารถตรวจจับใบหน้าและระบุว่าเป็นบุคคลใดได้เกือบทั้งหมดโดยที่บุคคลที่เข้ามานั้นต้องไม่มีการปิดบังใบหน้าซึ่งจากการทดสอบทั้งหมดเราสามารถนำเทคโนโลยี DLIB ที่ใช้เทคนิค HOG มาใช้ในโครงงานของเราในการตรวจจับใบหน้าได้ เพราะถือว่ามีข้อผิดพลาดเพียงเล็กน้อย ถ้าหากมีการ TRAIN MODEL ที่มากขึ้นก็จะทำให้ตรวจจับใบหน้าได้แม่นยำมากยิ่งขึ้น

5. สรุปผลโครงงาน

5.1. สรุปผลโครงงาน

ในการตรวจจับผู้บุกรุกนั้น สามารถนำมาประยุกต์ใช้ได้ในหลายๆสถานที่ เช่น ครัวเรือนหรือหอพักเพื่อรักษาความปลอดภัยให้สถานที่นั้นๆ ซึ่งในปัจจุบันมีการใช้เทคโนโลยีการตรวจจับใบหน้า อย่างแพร่หลาย เช่น ใช้ในการระบุตัวตน แต่ยังมีส่วนน้อยที่ใช้ในการรักษาความปลอดภัยให้แก่เคหะสถานซึ่งในการพัฒนาระบบตรวจจับผู้บุกรุก ในช่วงแรกของการพัฒนาได้ใช้เทคโนโลยี HAAR CASCADE โดยใช้เทคนิค LOCAL BINARY PATTERN (LBPH) ในการตรวจจับใบหน้า และ TRAIN MODEL โดยใช้ กล้อง WEBCAM ซึ่งในขณะนั้นสามารถตรวจจับใบหน้า และจดจำใบหน้าพร้อมระบุตัวตนได้ในระบบแต่ในปัจจุบันได้มีการเปลี่ยนอุปกรณ์ในการรับภาพวีดีโอแบบ REALTIME โดยใช้กล้องวงจรปิด IP CAMERA ซึ่งทำให้การจับใบหน้าและการวิเคราะห์มีข้อผิดพลาดจำนวนมากซึ่งไม่เป็นที่ยอมรับหากนำมาใช้ในการรักษาความปลอดภัย

จึงได้มีการเปลี่ยนเทคโนโลยีการตรวจจับใบหน้าเป็นเทคโนโลยี DLIB โดยใช้เทคนิค HISTOGRAM OF ORIENTED GRADIENTS (HOG) ซึ่งทำให้มีความแม่นยำมากยิ่งขึ้น มีข้อผิดพลาดน้อยลง แต่อาจจะยังมีข้อผิดพลาดอยู่บ้างเนื่องจากผลการทดสอบประสิทธิภาพยังมีส่วนที่ทดสอบผิดพลาดแต่สามารถนำระบบที่พัฒนาไปใช้รักษาความปลอดภัยได้ดีในระดับนึง แต่ทำให้ยังไม่สามารถเพิ่ม SENSOR BLUETOOTH ที่เราได้ตั้งเป้าหมายไว้เพื่อเพิ่มประสิทธิภาพในการตรวจจับให้มีความแม่นยำมากขึ้นในช่วงระหว่างการพัฒนา เนื่องจากการทำงานที่ซ้อนทับกับของ LIBRARY และเวลาที่ไม่เอื้ออำนวย แต่ในภาพรวมของโครงงานและระบบที่เราได้พัฒนาขึ้นมาสามารถใช้ตรวจจับผู้บุกรุกได้ดีในระดับหนึ่ง

ซึ่งคำนวนความแม่นยำออกมาได้ประมาน 93% และสามารถแจ้งเตือนเมื่อมีคนเข้ามาในบ้าน และระบุว่าเป็นใครไปยังคนที่อาศัยอยู่ภายในบ้านผ่านแอพพลิเคชั่นไลน์

ในการพัฒนาระบบของเราทั้งหมดเราสามารถตรวจจับใบหน้าบุคคลที่เข้ามาในบริเวณบ้าน และสามารถระบุตัวตนได้ผ่านกล้องวงจรปิด IP CAMARA ซึ่งเป็นไปตามเป้าหมายที่ได้ตั้งไว้ คือสามารถตรวจจับผู้บุกรุกและแจ้งเตือนไปยังคนในบ้าน แต่ในส่วนที่ทำไม่สำเร็จจะเป็นในส่วนของการเพิ่ม SENSOR ที่ช่วยเพิ่มการยืนยันตัวตนของคนในบ้าน ซึ่งในช่วงแรกมีการคิดที่จะเพิ่มในส่วนของการจับ BLUETOOTH เพื่อนำ MAC ADDRESS มายืนยันตัวตนเพิ่มมาเช็คว่าเป็นคนในบ้านหรือไม่ซึ่งในอนาคตอาจจะมีผู้สนใจที่จะนำไปพัฒนาต่อเพื่อให้ระบบไม่มีข้อผิดพลาดและความแม่นยำกลายเป็น 100% เพราะนั่นหมายถึงความปลอดภัยของคนที่อาศัยภายในบ้าน

5.2 แนวทางการพัฒนาต่อ

1. ปรับปรุงหรือเปลี่ยนเทคโนโลยีไปใช้เทคโนโลยีตรวจจับใบหน้าตัวอื่น เช่น TENSORFLOW, PYTORCH
2. เพิ่ม SENSOR ตัวอื่นเข้ามาช่วยเพิ่มเงื่อนไขในการตรวจจับผู้บุกรุก เช่น BLUETOOTH, WI-FI

5.3 อุปสรรคที่พบเจอ

1. การแยกแยะระหว่างสมาชิกและผู้บุกรุกที่ยังคลาดเคลื่อน เนื่องจากการเปลี่ยนอุปกรณ์กล้องเป็นกล้องวงจรปิด
2. ภาพกระตุกเนื่องจากการรับภาพจากกล้องวงจรปิดมีความ DELAY จากภาพ REALTIME
3. การติดตั้ง LIBRARY ที่ต้องใช้มีความยุ่งยากและต้องเปลี่ยนบ่อยเนื่องจากการเปลี่ยนอุปกรณ์รับภาพวีดีโอและการลองเปลี่ยนเทคโนโลยีต่างๆให้เหมาะสมกับการการนำไปใช้ของระบบ
4. มีความรู้ไม่เพียงพอต่อการพัฒนา จึงต้องหาความรู้ในระหว่างพัฒนาระบบ ทำให้เสียเวลาในการศึกษาและพัฒนามากพอสมควร
5. การเปลี่ยนแปลง FLOW การทำงานของระบบย่อยบ่อย เนื่องจากมีการเปลี่ยนเทคโนโลยีที่ใช้บ่อย

เอกสารอ้างอิง

[1] Imarisha Thamali. “Sampling & Quantization in Digital Image Processing” [Online]. Available: https://wimarshikathamali1995.medium.com/sampling-quantization-in-digital-image-processing-8c4490357039. 2020

[2] MathWorks. “Image Types in the Toolbox” [Online].Available: https://www.mathworks.com/help/images/image-types-in-the-toolbox.html. Retrieved 2022

[3] Tech Talk 2 Apply. “สร้าง Smart Home ด้วย Home Assistant” [Online].Available: https://techtalk2apply.com/smart-home-with-home-assistant/. 2021

[4] Nuttakan Chuntra. “OpenCV คืออะไร?” [Online].Available: https://medium.com/@nut.ch40/opencv-คืออะไร-8771e2a4c414. 2018

[5] Nuttakan Chuntra. “opencv-python เบื้องต้น” [Online].Available: https://phyblas.hinaboshi.com/oshi01. 2021

[6] Tutorialspoint. “Python - GUI Programming (Tkinter)” [Online].Available: https://www.tutorialspoint.com/python/python_gui_programming.htm. Retrieved 2022

[7] Bruno Dufour & Wen Hsin Chang. “An Introduction to Tkinter” [Online].Available: https://www.cs.mcgill.ca/~hv/classes/MS/TkinterPres/. Retrieved 2022

[8] phyblas. “การจัดการไฟล์ด้วย python โดยใช้มอดูล os และ shutil” [Online].Available: https://phyblas.hinaboshi.com/20200319. 2021

[9] Robert E. Schapire. “Explaining AdaBoost” [Online].Available: http://rob.schapire.net/papers/explaining-adaboost.pdf. Retrieved 2023

[10] Kelvin Salton. “Face Recognition: Understanding LBPH Algorithm” [Online].Available: https://towardsdatascience.com/face-recognition-how-lbph-works-90ec258c3d6b. 2017

[11] Arun Ponnusamy. “CNN based face detector from dlib” [Online].Available: https://towardsdatascience.com/cnn-based-face-detector-from-dlib-c3696195e01c. 2018

[12] Amolik Vivian Paul. “SSD Object Detection in Real Time (Deep Learning and Caffe” [Online]. Available: https://medium.com/acm-juit/ssd-object-detection-in-real-time-deep-learning-and-caffe-f41e40eea968. 2020

[13] Mahbubul Islam. “COMPARISON OF STOP SIGN DISTANCE DETECTION USING 2D AND 3D CAMERAS” Journal of UAB ECTC, vol.15, no. l, June 2016.

[14] รุสลี่ สุทธวีร์กูล,วิไลพรแซ่ลี้. “Haar-like Face Detection based-on Haar-like Features” SWU Engineering Journal, ปีที่ 6, ฉบับที่ 2, กรกฎาคม 2554. หน้า 34-43