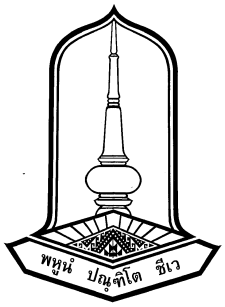


การขจัดวิกลภาพของกฎไฟร์วอลล์โดยใช้แผนผังต้นไม้ และการตัดสินใจแบบโดเมนเดี่ยว

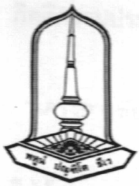
อธิพงศ์ คำสีลา

เสนอต่อมหาวิทยาลัยมหาสารคาม เพื่อเป็นส่วนหนึ่งของการศึกษาตามหลักสูตร
ปริญญาวิทยาศาสตรมหาบัณฑิต สาขาวิชาวิทยาการคอมพิวเตอร์
ตุลาคม 2557

การขจัดวิกลภาพของกฎไฟร์วอลล์โดยใช้แผนผังต้นไม้ และการตัดสินใจแบบโดเมนเดี่ยว

อธิพงศ์ คำสีลา

เสนอต่อมหาวิทยาลัยมหาสารคาม เพื่อเป็นส่วนหนึ่งของการศึกษาตามหลักสูตร
ปริญญาวิทยาศาสตรมหาบัณฑิต สาขาวิชาวิทยาการคอมพิวเตอร์
ตุลาคม 2557



คณะกรรมการสอบวิทยานิพนธ์ ได้พิจารณาวิทยานิพนธ์ของนายอธิพงศ์ คำสีลา แล้วเห็นสมควรรับเป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิทยาศาสตรมหาบัณฑิต สาขาวิชาวิทยาการคอมพิวเตอร์ ของมหาวิทยาลัยมหาสารคาม

คณะกรรมการสอบวิทยานิพนธ์

.....	ประธานกรรมการ
(ผศ.ดร.ฉัตรเกล้า เจริญผล)	(กรรมการบัณฑิตศึกษาประจำคณะ)
.....	กรรมการ
(อาจารย์ ดร.สมนึก พ่วงพรพิทักษ์)	(อาจารย์ที่ปรึกษาวิทยานิพนธ์หลัก)
.....	กรรมการ
(ผศ.ดร.พนิดา ทรงรัมย์)	(อาจารย์ที่ปรึกษาวิทยานิพนธ์ร่วม)
.....	กรรมการ
(อาจารย์ ดร.คำรณ สุนัติ)	(ผู้ทรงคุณวุฒิ)

มหาวิทยาลัยอนุมัติให้รับวิทยานิพนธ์ฉบับนี้ เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิทยาศาสตรมหาบัณฑิต สาขาวิชาวิทยาการคอมพิวเตอร์ ของมหาวิทยาลัยมหาสารคาม

.....

(รศ.เทียนศักดิ์ เมฆพรรณโอภาส)
รักษาการคณบดีคณะวิทยาการสารสนเทศ

.....

(ศ.ดร.ประดิษฐ์ เทอดทูล)
คณบดีบัณฑิตวิทยาลัย
วันที่ 31 เดือน ก.ค. พ.ศ. 2557

กิตติกรรมประกาศ

วิทยานิพนธ์ฉบับนี้ ได้รับทุนจาก “โครงการพัฒนาศักยภาพทางการวิจัย ประจำปีการศึกษา 2555 และประจำปีการศึกษา 2556”

วิทยานิพนธ์ฉบับนี้สำเร็จสมบูรณ์ได้ด้วยความกรุณาและความช่วยเหลืออย่างสูงยิ่งจาก อาจารย์ ดร.สมนึก พ่วงพรพิทักษ์ ประธานกรรมการควบคุมวิทยานิพนธ์ ผู้ช่วยศาสตราจารย์ ดร.พนิดา ทรงรัมย์ กรรมการควบคุมวิทยานิพนธ์ อาจารย์ ดร.ฉัตรเกล้า เจริญผล ประธานกรรมการสอบ และ อาจารย์ ดร.คำรณ สุนัติ กรรมการผู้ทรงคุณวุฒิ

ขอขอบพระคุณ อาจารย์ ดร.สมนึก พ่วงพรพิทักษ์ อาจารย์ที่ปรึกษาผู้เชี่ยวชาญผู้ให้การช่วยเหลือเครื่องมือและสนับสนุนการวิจัย

ขอขอบพระคุณพระคุณบิดามารดา และครอบครัว ซึ่งเปิดโอกาสให้ได้รับการศึกษาเล่าเรียน ตลอดจนคอยช่วยเหลือและให้กำลังใจผู้วิจัยเสมอมาจนสำเร็จการศึกษา

อธิพงศ์ คำสีลา

ชื่อเรื่อง การขจัดวิกลภาพของกฎไฟร์วอลล์โดยใช้แผนผังต้นไม้และการตัดสินใจแบบโดเมนเดี่ยว
ผู้วิจัย นายอธิพงศ์ คำสีลา
ปริญญา วิทยาศาสตรมหาบัณฑิต **สาขาวิชา** วิทยาการคอมพิวเตอร์
กรรมการควบคุม อาจารย์ ดร.สมนึก พ่วงพรพิทักษ์
ผู้ช่วยศาสตราจารย์ ดร.พนิดา ทรงรัมย์
มหาวิทยาลัย มหาวิทยาลัยมหาสารคาม **ปีที่พิมพ์** 2557

บทคัดย่อ

ไฟร์วอลล์มีบทบาทสำคัญในควบคุมการเข้าถึงระบบเครือข่าย โดยประสิทธิภาพของไฟร์วอลล์ขึ้นอยู่กับการจัดการกฎไฟร์วอลล์ที่ผู้ดูแลระบบได้กำหนดไว้ การจัดการกฎของไฟร์วอลล์ที่ไม่ดี (ซึ่งอาจสาเหตุมาจาก วิกลภาพของกฎ เช่น การขัดแย้ง การทับซ้อน หรือ การคาบเกี่ยวกัน ระหว่างกฎ) จะส่งผลให้ประสิทธิภาพการทำงานของไฟร์วอลล์และระบบเครือข่ายลดลง ดังนั้น ในวิทยานิพนธ์นี้ จึงได้ออกแบบและพัฒนา แนวคิดไฟร์วอลล์ต้นแบบขึ้นมาใหม่ ซึ่งปราศจากวิกลภาพของกฎโดยสิ้นเชิง โดยเสนอแนวคิด การตัดสินใจแบบโดเมนเดี่ยว (Single Domain Decision: *SDD*) ร่วมกับการใช้แผนภาพแบบต้นไม้ (Tree Diagram, *TD*) เป็นโครงสร้างข้อมูล ด้วยแนวคิด *SDD* ทำให้กฎไฟร์วอลล์สามารถสลับตำแหน่งได้อย่างอิสระ โดยไม่ส่งผลกระทบต่อความหมายด้านการรักษาความปลอดภัยของระบบเครือข่าย นอกจากนี้โครงสร้างแบบ *TD* ยังสามารถเพิ่มประสิทธิภาพ ในการผ่านกฎของไฟร์วอลล์ได้เร็วยิ่งขึ้น ไฟร์วอลล์ต้นแบบตามแนวคิดใหม่นี้ ได้ถูกทดลองเพื่อวัดประสิทธิภาพบนเครือข่ายทดสอบ (Test-bed) ซึ่งผลการทดลองแสดงให้เห็นว่า การออกแบบด้วยแนวคิดที่นำเสนอนี้ สามารถขจัดวิกลภาพของกฎ ไฟร์วอลล์ ได้อย่างสมบูรณ์ และเพิ่มประสิทธิภาพในการผ่านกฎของไฟร์วอลล์อีกด้วย

คำสำคัญ : การจัดการกฎไฟร์วอลล์; วิกลภาพของกฎไฟร์วอลล์; การผ่านกฎของไฟร์วอลล์

TITLE Anomaly Elimination for Firewall Rules using Tree Diagram and Single Domain Decision

AUTHOR Mr.Atipong Khumseela

DEGREE Master of Science **MAJOR** Computer Science

ADVISORS Somnuk Puangpronpitag, Ph.D.

Panida Songram, Ph.D.

UNIVERSITY Mahasarakham University **DATE** 2014

ABSTRACT

Firewall is an important system to control the access of networks. The firewall performance depends significantly on firewall rule management. Bad firewall rule management (causing by firewall rule anomalies such as *conflict*, *redundancy, overlap*) can reduce the efficiency of both firewall & network. So, in this thesis, we design and develop a new prototyped firewall that can completely eliminate firewall rule anomalies by proposing a *Single Domain Decision (SDD)* scheme together with a *Tree Diagram (TD)* structure. By using the *SDD* scheme, the firewall can freely swap its rules with no effect to the overall rule meaning. Also, the *TD* structure can increase the efficiency of firewall rule parsing. The prototyped firewall has been experimented on a test-bed. The Experimental results have demonstrated that our proposed design can completely get rid of firewall rule's anomalies, and improve the efficiency of firewall rule parsing.

Key Words: Firewall rule management; Firewall rule anomaly; Firewall rule parsing

สารบัญ

สารบัญตาราง

สารบัญรูป

สารบัญรูป

หน้า

บทที่ 1

บทนำ

1.1 หลักการและเหตุผล

ในปัจจุบันระบบอินเทอร์เน็ต (Internet) มีบทบาทที่สำคัญต่อชีวิตประจำวันของเราเป็นอย่างมาก เพราะเป็นศูนย์รวมการเชื่อมต่อของระบบเครือข่ายทั่วโลกที่รวบรวมสารสนเทศทุกประเภทเข้าด้วยกัน เมื่อระบบเครือข่ายขององค์กรได้ทำการเชื่อมต่อเข้ากับระบบเครือข่ายอินเทอร์เน็ตแล้วจะทำให้ผู้ใช้งานระบบอินเทอร์เน็ตจากสถานที่ต่างๆทั่วโลกสามารถเข้าถึงข้อมูลภายระบบเครือข่ายขององค์กรได้ ปัญหาที่จะตามมาก็คือมาตรการการรักษาความปลอดภัยของข้อมูล ซึ่งอาจจะถูกขโมยข้อมูลหรือสวมสิทธิการใช้งานผู้ดูแลหรืออาจถูกปิดกั้นการให้บริการข้อมูลขององค์กรจากพฤติกรรมอันไม่พึงประสงค์ของผู้ใช้งานในระบบอินเทอร์เน็ต และอาจส่งผลเสียต่อระบบเครือข่ายขององค์กรได้ในภายหลัง ยกตัวอย่างเช่น เมื่อองค์กรให้บริการเว็บไซต์บนระบบอินเทอร์เน็ต อาจมีผู้ไม่หวังดีปิดกั้นการให้บริการเว็บไซต์ขององค์กรโดยใช้เทคนิควิธีการต่างๆ ซึ่งจะส่งผลให้ผู้ใช้งานเว็บไซต์ขององค์กรไม่สามารถเข้าถึงข้อมูลได้ ดังนั้นระบบเครือข่ายขององค์กรจำเป็นจะต้องมีไฟร์วอลล์ (Firewall) เพื่อควบคุมการเข้าถึงข้อมูลและการให้บริการขององค์กร ไฟร์วอลล์เปรียบเสมือนประตูกั้นระหว่างระบบอินเทอร์เน็ตกับระบบเครือข่ายขององค์กร ซึ่งจะทำหน้าที่ตรวจสอบและกำหนดสิทธิข้อมูลที่วิ่งผ่านเข้าออกระบบเครือข่ายให้สอดคล้องกับนโยบายการรักษาความปลอดภัยขององค์กร ข้อมูลที่วิ่งผ่านเข้าออกไฟร์วอลล์จะถูกเปรียบเทียบเข้ากับเงื่อนไขของกฎไฟร์วอลล์ซึ่งเป็นหัวใจสำคัญในการทำงานของไฟร์วอลล์โดยสามารถที่จะกำหนดสิทธิให้กับข้อมูลที่เข้าออกได้สองแบบคือ ยอมรับ (Accept) หรือ ปฏิเสธ (Deny)

กฎของไฟร์วอลล์จะถูกสร้างให้สอดคล้องกับนโยบายการรักษาความปลอดภัยขององค์กรหรือระบบเครือข่ายนั้นๆโดยผู้ดูแลระบบ ในระบบเครือข่ายที่มีมาตรการการรักษาความปลอดภัยที่สูงขึ้นจะส่งผลให้จำนวนกฎของไฟร์วอลล์มีจำนวนเพิ่มขึ้นอีกด้วย ปัญหาที่ตามมาก็คือประสิทธิภาพในการทำงานของไฟร์วอลล์จะลดลง เนื่องจากลักษณะข้อมูลที่วิ่งผ่านกฎของไฟร์วอลล์มีการทำงานแบบตามลำดับขั้น (Sequence) เมื่อกฎของไฟร์วอลล์เพิ่มขึ้นจะทำให้ระยะเวลาในการผ่านกฎของไฟร์วอลล์เพิ่มขึ้น และนอกจากนี้การจัดการกฎของไฟร์วอลล์ก็จะมีความซับซ้อนมากขึ้นตามไปด้วย หากผู้ดูแลระบบขาดความรู้ความเข้าใจในระบบเครือข่ายแล้ว การออกแบบและการจัดการกฎของไฟร์วอลล์อาจเกิดวิกลภาพของกฎภายในระบบไฟร์วอลล์ (Firewall Rule Anomaly) ซึ่งจะส่งผลเสียต่อการทำงานของไฟร์วอลล์ที่ยากจะแก้ไข เช่น อาจทำให้ระบบเครือข่ายขององค์กรมีช่องโหวเกิดขึ้นหรือมีการปิดกั้นการจราจรของข้อมูลที่ถูกต้อง พบว่ามีหลายงานวิจัยที่นำเสนอวิธีการแก้ไขปัญหาวิกลภาพของกฎเหล่านี้ Al-Shaer และคณะ [1] ได้กำหนดรูปแบบวิกลภาพของกฎไฟร์วอลล์ไว้ 4 รูปแบบที่เกิดขึ้นได้กับองค์กร

ขนาดเล็กถึงองค์กรขนาดใหญ่ และได้นำเสนอวิธีการตรวจสอบวิกลภาพของกฎไฟร์วอลล์ด้วยแผนภาพสถานะความสัมพันธ์เสมือน (Similar State Diagram) พบว่าวิธีการนี้สามารถตรวจสอบวิกลภาพของกฎไฟร์วอลล์ได้เพียงกฎต่อกฎเท่านั้น โดย Chomsiri [2] ได้แนะนำขั้นตอนการตรวจสอบวิกลภาพของกฎไฟร์วอลล์ที่มีประสิทธิภาพมากขึ้นโดยใช้หลักพีชคณิตเชิงสัมพันธ์ (Relational Algebra) พบว่าสามารถวิเคราะห์วิกลภาพของกฎที่เกิดขึ้นในระบบไฟร์วอลล์ได้ครบถ้วน อย่างไรก็ตาม การตรวจสอบด้วยวิธีการนี้อยู่ภายใต้ระบบไฟร์วอลล์ที่มีการตรวจสอบแบบตามลำดับเท่านั้น Liu [3] ได้เสนอวิธีการตรวจสอบวิกลภาพของกฎไฟร์วอลล์บนระบบโครงสร้างของไฟร์วอลล์แบบต้นไม้พบว่ามีประสิทธิภาพในการตรวจสอบที่ดียิ่งขึ้น จากงานวิจัยที่กล่าวมาสามารถตรวจสอบหาวิกลภาพของกฎไฟร์วอลล์ได้ดีทั้งหมด แต่ยังไม่มีงานวิจัยใดที่ขจัดวิกลภาพของกฎได้อย่างสมบูรณ์

วิทยานิพนธ์นี้จึงเสนอวิธีการแก้ไขปัญหาการจัดการกฎของไฟร์วอลล์ที่มุ่งเน้นเป้าหมายเพื่อขจัดวิกลภาพของกฎไฟร์วอลล์โดยใช้แนวคิด การตัดสินใจแบบโดเมนเดี่ยว (Single Domain Decision: *SDD*) เนื่องจากต้นเหตุวิกลภาพของกฎไฟร์วอลล์เกิดจากการที่สมาชิกของกฎใดๆ ตั้งแต่สองกฎขึ้นไปมีผลของการกระทำที่แตกต่างกันแต่ถูกกระทำในเวลาเดียวกัน ดังนั้นไฟร์วอลล์ใดๆจะต้องมีการตัดสินใจเพียงอย่างใดอย่างหนึ่งเท่านั้นคือ ยอมรับทั้งหมด (Accept All) หรือ ปฏิเสธทั้งหมด (Deny All) ซึ่งจะทำให้ไฟร์วอลล์ปราศจากวิกลภาพของกฎไฟร์วอลล์ได้อย่างสมบูรณ์ นอกจากนี้วิทยานิพนธ์ยังได้ปรับปรุงโครงสร้างของไฟร์วอลล์โดยใช้โครงสร้างข้อมูลแผนภาพแบบต้นไม้ (Tree Diagram, *TD*) ซึ่งจะช่วยให้การการผ่านกฎของไฟร์วอลล์มีประสิทธิภาพเพิ่มมากขึ้น

1.2 วัตถุประสงค์ของการวิจัย

งานวิจัยนี้มีวัตถุประสงค์เพื่อนำเสนอต้นแบบของไฟร์วอลล์ใหม่ที่สามารถแก้ไขปัญหาวิกลภาพกฎไฟร์วอลล์โดยใช้แนวความคิดการตัดสินใจแบบโดเมนเดียวและใช้โครงสร้างแผนภาพแบบต้นไม้

1.3 ความสำคัญของการวิจัย

1. ได้แนวคิดของไฟร์วอลล์ใหม่ที่สามารถขจัดปัญหาวิกลภาพของกฎไฟร์วอลล์ทั้ง 4 รูปแบบ ได้แก่ กฎที่ถูกบัง (Shadowing Anomaly), กฎที่เกี่ยวพันกัน (Correlation Anomaly), กฎที่ถูกครอบคลุม (Generalization Anomaly) และกฎที่กระทำซ้ำซ้อน (Redundancy Anomaly) ได้อย่างสมบูรณ์และมีประสิทธิภาพในการผ่านกฎของไฟร์วอลล์ได้ดีกว่าไฟร์วอลล์โดยทั่วไป

2. ได้ต้นแบบของไฟร์วอลล์ที่สามารถนำไปต่อยอดและพัฒนาใช้กับไฟร์วอลล์โอเพนซอร์สเพื่อลดค่าใช้จ่ายในการนำเข้าไฟร์วอลล์จากต่างประเทศ ซึ่งจะช่วยให้องค์กรขนาดเล็ก เช่น โรงเรียน ห้างร้านต่างๆ สามารถใช้ไฟร์วอลล์ที่มีประสิทธิภาพโดยลดต้นทุนให้การนำเข้าได้

1.4 ขอบเขตของการวิจัย

1. พัฒนาต้นแบบของไฟร์วอลล์ที่มีการตัดสินใจแบบโดเมนเดี่ยว (*SDD*) เพื่อขจัดวิกลภาพของกฎไฟร์วอลล์โดยมีรูปแบบวิกลภาพของกฎที่สนใจได้แก่ กฎที่ถูกบัง, กฎที่เกี่ยวพัน, กฎที่ถูกครอบคลุม และกฎที่กระทำซ้ำซ้อน

2. พัฒนาต้นแบบของไฟร์วอลล์การตัดสินใจแบบโดเมนเดี่ยว (*SDD*) ที่สนับสนุนหมายเลขไอพีรุ่น 4 (IPV4)

3. การทดสอบประสิทธิภาพทำการทดลองบน Test-bed

1.5 นิยามศัพท์เฉพาะ

1. ไฟร์วอลล์ (Firewall) คือ เครื่องมือที่อาจจะเป็นซอฟแวร์หรือฮาร์ดแวร์ ซึ่งใช้จำกัดการเข้าถึงข้อมูลหรือการให้บริการ เพื่อป้องกันการเข้าถึงข้อมูลอันไม่พึงประสงค์จากผู้ไม่หวังดี การทำงานของไฟร์วอลล์ขึ้นอยู่กับการจัดการกฎที่ผู้ดูแลระบบได้กำหนดไว้

2. โดเมน (Domain) คือ เซตของตัวแปรที่ถูกกำหนดลงในฟังก์ชันที่ได้นิยามไว้แล้ว ซึ่งในวิทยานิพนธ์นี้ได้กำหนดให้ โดเมนคือเซตของตัวแปรที่ประกอบไปด้วย หมายเลขไอพีต้นทาง (Source IP Address: *SIP*), หมายเลขไอพีปลายทาง (Destination IP Address: *DIP*), หมายเลขพอร์ตต้นทาง (Source Port: *SPT*), หมายเลขพอร์ตปลายทาง (Destination Port: *DPT*) และโพรโทคอล (Protocol: *PRO*)

3. การตัดสินใจแบบโดเมนเดี่ยว (Single Domain Decision: *SDD*) คือ แนวคิดการออกแบบไฟร์วอลล์มีสิทธิการตัดสินใจของโดเมนนั้นอย่างใดอย่างหนึ่งคือ ยอมรับทั้งหมด (Accept All) หรือปฏิเสธทั้งหมด (Deny All) เท่านั้น

4. กฎไฟร์วอลล์ (Firewall Rule) คือ เงื่อนไขการทำงานของไฟร์วอลล์จะเปรียบเทียบกับข้อมูลหรือแพ็กเก็ตที่วิ่งผ่านเข้าออกไฟร์วอลล์เพื่อกำหนดสิทธิการตัดสินใจให้กับข้อมูลหรือแพ็กเก็ตนั้นๆ ซึ่งจะมีผลการตัดสินใจอยู่สองแบบคือ ยอมรับ (Accept) หรือปฏิเสธ (Deny) ในวิทยานิพนธ์นี้ได้กำหนดลำดับของกฎไฟร์วอลล์โดยใช้ตัวย่อ Fwr_x เช่น กฎลำดับที่ 1 แทนด้วย Fwr_1 เป็นต้น

5. วิกลภาพของกฎไฟร์วอลล์ (Firewall Rules Anomaly) คือ ความผิดปรกติของกฎไฟร์วอลล์ โดยมีกฎตั้งแต่สองกฎขึ้นไปให้ความหมายที่ขัดแย้งกันหรือให้ความหมายที่ซ้ำซ้อน คาบเกี่ยวกัน ซึ่งมีผลทำให้ (1) มีกฎที่ไม่มีความจำเป็นอยู่ในระบบ และอาจส่งผลต่อประสิทธิภาพการทำงานของไฟร์วอลล์ (2) เกิดความสับสนในการตีความหมายจากความขัดแย้งระหว่างกฎของไฟร์วอลล์

6. การจัดการกฎของไฟร์วอลล์ (Firewall Rule Management) คือ การวิเคราะห์ออกแบบและแก้ไขกฎของไฟร์วอลล์ให้สอดคล้องกับนโยบายมาตรการรักษาความปลอดภัยของระบบเครือข่ายองค์กร ซึ่งการออกแบบกฎไฟร์วอลล์ให้สอดคล้องกับนโยบายใดๆอาจมีหลายรูปแบบขึ้นอยู่กับเทคนิควิธีการและความสามารถของผู้ดูแลระบบ ซึ่งหากมีการจัดการกฎของไฟร์วอลล์ที่ดีจะส่งผลให้ประสิทธิภาพการทำงานของไฟร์วอลล์ดีขึ้นตามไปด้วย

7. การผ่านกฎของไฟร์วอลล์ (Firewall Rule Parsing) คือ การนำข้อมูลหรือแพ็กเก็ตที่วิ่งผ่านเข้าออกไฟร์วอลล์มาเปรียบเทียบกับเงื่อนไขของกฎไฟร์วอลล์เพื่อกำหนดสิทธิ์การเข้าออกระบบเครือข่าย ซึ่งประสิทธิภาพการผ่านกฎไฟร์วอลล์จะลดลงเมื่อมีจำนวนการเปรียบเทียบเงื่อนไขของกฎไฟร์วอลล์หรือจำนวนกฎของไฟร์วอลล์เพิ่มขึ้น

8. แผนภาพแบบต้นไม้ (Tree Diagram) คือ โครงสร้างของข้อมูลที่มีลักษณะการเรียงตัวเป็นกิ่งก้านสาขาโดยจะไม่มีวงวน โยงในสมาชิกตัวต่างๆ โดยสมาชิกจะถูกเก็ยไว้ในประเภทข้อมูลที่เรียกว่าโหนด (ในวิทยานิพนธ์จะเปรียบเทียบโหนดใดๆ คือ ฟิลด์เงื่อนไขของกฎไฟร์วอลล์) ซึ่งโหนดเริ่มต้นของโครงสร้างเรียกว่า ราก ถูกแทนด้วยฟิลด์เงื่อนไขของหมายเลขไอพีต้นทาง (*SIP*), และมีกิ่งก้านเชื่อมโยงโหนดต่างได้แก่ หมายเลขพอร์ตต้นทาง (*SPT*), หมายเลขไอพีปลายทาง (*DIP*), หมายเลขพอร์ตปลายทาง (*DPT*), โพรโทรคอล (*PRO*) ตามลำดับ

บทที่ 2

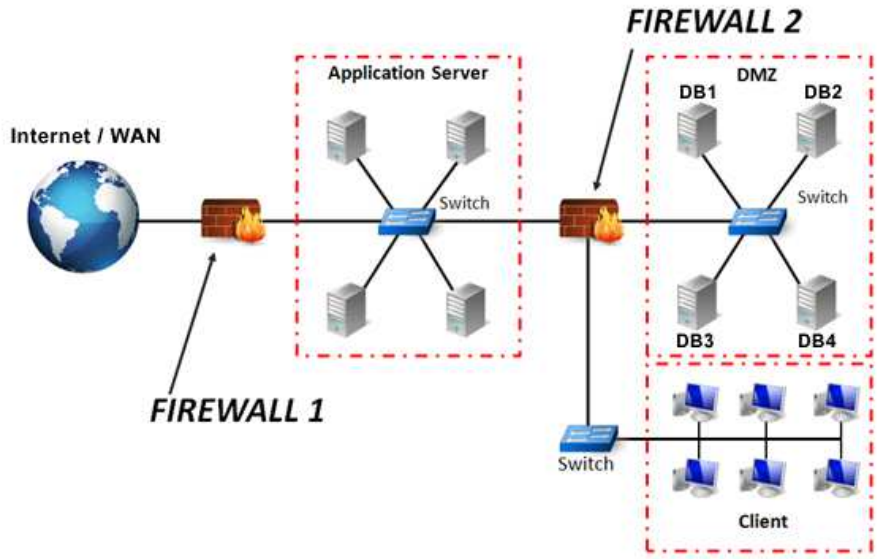
ทฤษฎีและงานวิจัยที่เกี่ยวข้อง

ในบทนี้จะอธิบายถึงทฤษฎีพื้นฐานที่จำเป็นสำหรับงานวิจัยซึ่งจะแบ่งออกเป็น 2 ส่วน คือ ส่วนแรก จะกล่าวถึงทฤษฎีที่เกี่ยวข้องกับการทำงานของไฟร์วอลล์ และทฤษฎีที่นำมาใช้ปรับปรุงโครงสร้างการทำงานของไฟร์วอลล์ เพื่อให้มีประสิทธิภาพที่ดียิ่งขึ้น ส่วนที่สอง จะกล่าวถึงปัญหาที่เกิดขึ้นกับไฟร์วอลล์และงานวิจัยที่นำเสนอแนวทางการแก้ไขปัญหาซึ่งสามารถแบ่งได้ดังนี้ ปัญหาวิกลภาพของกฎไฟร์วอลล์ (Firewall Rule Anomaly) ปัญหาด้านการจัดการกฎไฟร์วอลล์ (Firewall Rule Management) และปัญหาด้านประสิทธิภาพในการผ่านกฎไฟร์วอลล์ (Firewall Rule Parsing)

2.1 ไฟร์วอลล์

2.1.1 ไฟร์วอลล์ (Firewall)

ไฟร์วอลล์ [4] เป็นอุปกรณ์รักษาความปลอดภัยที่มีความสำคัญต่อระบบเครือข่าย ซึ่งจะทำหน้าที่ตรวจสอบการจราจรของข้อมูล (โดยปรกติจะเรียกข้อมูลนี้ว่า แพ็กเก็ต (Packet)) ที่วิ่งผ่านเข้าออกระบบเครือข่าย โดยทั่วไปแล้วไฟร์วอลล์จะนำไปวางกั้นระหว่างเครือข่ายภายนอกหรือเครือข่ายสาธารณะ (Wide Area Network: *WAN*) กับเครือข่ายภายในหรือเครือข่ายขององค์กร (Local Area Network: *LAN* หรือ Metropolitan Area Network: *MAN*) เพื่อป้องกันการเข้าถึงข้อมูลอันไม่พึงประสงค์และภัยคุกคามต่างๆโดยผู้ไม่หวังดีจากระบบเครือข่ายภายนอก หรือวางกั้นระหว่างเครือข่ายภายในองค์กรเพื่อจำกัดการเข้าถึงข้อมูลของบุคลากรตามแผนกต่างๆ เช่น แผนกบัญชีสามารถเข้าถึงข้อมูลการเงินของลูกค้าและข้อมูลบัญชีของบริษัท ซึ่งแผนกอื่นที่ไม่มีส่วนเกี่ยวข้องจะไม่สามารถเข้าถึงข้อมูลนี้ได้ เพื่อกำหนดขอบเขตความรับผิดชอบของแผนกนั้นๆ เมื่อข้อมูลมีการรั่วไหลเกิดขึ้น



รูปที่ 2.1 การวางตำแหน่งของไฟร์วอลล์ในระบบเครือข่าย

จากรูปที่ 2.1 *FIREWALL1* จะตรวจสอบแพ็กเก็ตที่จราจรเข้าออกระบบเครือข่ายภายในและเครือข่ายสาธารณะ (Internet) เพื่อกำหนดสิทธิการเข้าถึงและการให้บริการข้อมูลขององค์กรสู่สาธารณะ (Application Server) ยกตัวอย่างเช่น การให้บริการเว็บไซต์ (Web Server) การให้บริการอีเมล์ (Mail Server) เป็นต้น ซึ่งต่างจาก *FIREWALL2* ที่จะทำหน้าที่ในการกำหนดสิทธิเข้าถึงของบุคลากร (Client) กับฐานข้อมูลขององค์กรให้เหมาะสม (Demilitarized Zone: *DMZ*) เพื่อกำหนดขอบเขตการรับผิดชอบต่อการเข้าถึงข้อมูลนั้นๆ ยกตัวอย่าง การเข้าถึงฐานข้อมูลลูกค้า (Database1: *DB1*) การเข้าถึงฐานข้อมูลการเงินบริษัท (Database2: *DB2*) กำหนดให้แผนกบัญชีเข้าถึงได้เท่านั้น

2.1.2 กฎไฟร์วอลล์ (Firewall Rule)

ไฟร์วอลล์ทำหน้าที่ตรวจสอบแพ็กเก็ตที่วิ่งผ่านเข้าออกไฟร์วอลล์โดยเปรียบเทียบเข้ากับกลุ่มของกฎไฟร์วอลล์ที่กำหนดโดยผู้ดูแลระบบ เพื่อให้สอดคล้องกับนโยบายการรักษาความปลอดภัยขององค์กร ซึ่งอาจจะเรียกว่า โพลีซี (Policy) และเรียกกฎของไฟร์วอลล์แต่ละบรรทัดว่า กฎไฟร์วอลล์ (Firewall Rule: *Fwr*) โดยกฎข้อแรกของกฎไฟร์วอลล์จะแทนด้วย Fwr_1 เป็นต้น ส่วนประกอบของกฎไฟร์วอลล์จะประกอบด้วยฟิลด์ข้อมูลหลัก 6 ฟิลด์ด้วยกันคือ หมายเลขไอพีต้นทาง (Source IP Address: *SIP*) หมายเลขไอพีปลายทาง (Destination IP Address: *DIP*) หมายเลขพอร์ตต้นทาง (Source Port: *SPT*) หมายเลขพอร์ตปลายทาง (Destination Port: *DPT*) โพรโทคอล (Protocol: *PRO*) และผลการกระทำของกฎ (Action: *ACT*) ดังตารางที่ 2.1

ตารางที่ 2.1 กฎของไฟร์วอลล์

No	*SIP*	*SPT*	*DIP*	*DPT*	*PRO*	*ACT*
Fwr_1	140.192.37.20	All	All	80	TCP	Deny
Fwr_2	140.192.37.20	All	All	80	TCP	Accept
Fwr_3	All	All	161.120.33.40	80	TCP	Accept
Fwr_4	140.197.37.0/24	All	161.120.33.40	80	TCP	Deny
Fwr_5	140.192.37.30	All	All	21	TCP	Deny
Fwr_6	140.197.37.0/24	All	All	21	TCP	Accept
Fwr_7	140.197.37.0/24	All	161.120.33.40	21	TCP	Accept
Fwr_8	All	All	All	All	TCP	Deny
Fwr_9	140.197.37.0/24	All	161.120.33.40	53	UDP	Accept
Fwr_{10}	All	All	161.120.33.40	53	UDP	Accept
Fwr_{11}	All	All	All	All	UDP	Deny

จากตารางที่ 2.1 เมื่อไฟร์วอลล์ตรวจสอบแพ็กเก็ตที่ผ่านเข้าออกจะตรวจสอบข้อมูลโดยเปรียบเทียบฟิลด์ข้อมูลของกฎไฟร์วอลล์กับข้อมูลบนเฮดเดอร์แพ็กเก็ต (TCP/IP Header) [5] เริ่มต้นจากกฎลำดับที่ 1 (Fwr_1) ถ้าหากข้อมูลในฟิลด์เงื่อนไขทุกฟิลด์สอดคล้องกันจะกระทำตามผลการกระทำ (ACT) ของกฎนั้นๆ ซึ่งถ้าผลการกระทำเป็น ยอมรับ (Accept) จะอนุญาตให้แพ็กเก็ตที่ถูกตรวจสอบผ่านเข้าออกระบบเครือข่ายได้ตามปรกติ แต่ถ้าผลของการกระทำเป็น ปฏิเสธ (Deny) แพ็กเก็ตจะถูกปฏิเสธหรือถูกทิ้งในทันที และถ้าหากแพ็กเก็ตที่ตรวจสอบไม่สอดคล้องกับฟิลด์เงื่อนไข ฟิลด์ใดฟิลด์หนึ่งแพ็กเก็ตจะถูกตรวจสอบกับกฎลำดับถัดไปตามลำดับ และเมื่อแพ็กเก็ตตรวจสอบกับกฎของไฟร์วอลล์ลำดับสุดท้ายพบว่ายังไม่สอดคล้องกับฟิลด์เงื่อนไข แพ็กเก็ตจะถูกปฏิเสธโดยปริยาย (Default Deny) ซึ่งเป็นกฎที่ถูกกำหนดโดยอัตโนมัติเพื่อป้องกันการไหลผ่านของแพ็กเก็ตที่คาดไม่ถึงซึ่งอาจเป็นอันตรายต่อระบบเครือข่ายหรือข้อมูลต่างๆภายในองค์กรอีกด้วย

กฎของไฟร์วอลล์ใช้ในการพิจารณาและตรวจสอบแพ็กเก็ตที่วิ่งผ่านเข้าออกไฟร์วอลล์หากขาดการออกแบบกฎที่ดีและการสร้างกฎที่ผิดพลาดหรือเกิดกฎวิกลภาพ อาจนำไปสู่ปัญหาหลายประการเช่น เปิดช่องโหว่ของระบบเครือข่ายโดยไม่คาดคิด และอาจส่งผลให้ประสิทธิภาพในการผ่านกฎลดลงอีกด้วย ซึ่งในหัวข้อต่อไปจะอธิบายองค์ประกอบต่างๆ ที่เกี่ยวข้องกับการสร้างกฎของไฟร์วอลล์

2.1.3 หมายเลขไอพี (IP Address)

หมายเลขไอพี (IP Address) [5] ย่อมาจาก Internet Protocol Address คือลำดับหมายเลขที่กำหนดให้กับอุปกรณ์แต่ละชนิด เช่น เครื่องคอมพิวเตอร์ เครื่องพิมพ์ โทรศัพท์มือถือหรืออุปกรณ์ต่างๆ ที่มีส่วนร่วมอยู่ในระบบเครือข่ายคอมพิวเตอร์หนึ่งๆ ที่ใช้อินเทอร์เน็ตโพรโทคอลในการ

สื่อสาร หมายเลขไอพีทำหน้าที่สำคัญสองอย่าง ได้แก่ การระบุแม่ข่ายหรือส่วนเชื่อมต่อประสานเครือข่ายและกำหนดที่อยู่ให้ตำแหน่งที่ตั้งของอุปกรณ์นั้นๆ

แต่เดิมผู้ออกแบบหมายเลขไอพี ได้กำหนดเลขที่อยู่ไอพีให้เป็นตัวเลข 32 บิตซึ่งเป็นที่รู้จักในชื่อ หมายเลขไอพีรุ่น 4 (IPV4) และระบบนี้ยังคงมีการใช้งานอยู่ในปัจจุบัน อย่างไรก็ตามเนื่องจากอินเทอร์เน็ตมีการเจริญเติบโตขึ้นอย่างมหาศาลและมีการคาดการณ์ว่า หมายเลขไอพีรุ่น 4 กำลังจะถูกใช้หมดไป จึงได้พัฒนาขึ้นในปี ค.ศ. 1995 คือหมายเลขไอพีรุ่น 6 (IPV6) ซึ่งใช้ตัวเลข 128 บิตและถูกกำหนดเป็นมาตรฐานใน RFC 2460 เมื่อปี ค.ศ. 1998 ซึ่งจะถูกนำมาใช้จริงตั้งแต่กลางคริสต์ทศวรรษ 2000

หมายเลขไอพีเป็นระบบเลขฐานสอง แต่จะแสดงผลและจัดเก็บบันทึกด้วยสัญกรณ์ที่มนุษย์สามารถอ่านเข้าใจได้ง่าย ตัวอย่างเช่น 172.16.254.1 (IPV4) และ 2001:db8:0:1234:0:567:8:1 (IPV6) เป็นต้น ในวิทยานิพนธ์นี้ได้ทำการทดสอบและปรับปรุงโครงสร้างของไฟร์วอลล์โดยใช้หมายเลขไอพีรุ่น 4 ในการทดลองในส่วนของฟิลด์ข้อมูล หมายเลขไอพีต้นทาง (Source IP Address: *SIP*) และหมายเลขไอพีปลายทาง (Destination IP Address: *DIP*) ซึ่งจะช่วยให้สามารถเข้าใจการตรวจสอบข้อมูลกับกฎของไฟร์วอลล์ได้ง่ายขึ้น อย่างไรก็ตาม โครงสร้างและผลการทดลองที่ได้จากวิทยานิพนธ์นี้สามารถนำไปประยุกต์ใช้งานกับหมายเลขไอพีรุ่น 6 (IPV6) ได้เช่นเดียวกัน

หมายเลขไอพีรุ่น 4 (IPV4) ประกอบด้วยเลข 32 บิต ซึ่งสามารถรองรับที่อยู่ที่ไม่ซ้ำกันมากที่สุดเท่าที่จะเป็นไปได้คือ 4,294,967,296 หรือ 2^{32} หมายเลข แต่หมายเลขไอพีรุ่น 4 ได้สงวนบางเลขหมายไว้สำหรับใช้งานตามจุดประสงค์พิเศษ เช่น หมายเลขไอพีเครือข่ายส่วนตัว (Private IP Address) ประมาณ 18 ล้านหมายเลข และหมายเลขไอพีสำหรับการแพร่สัญญาณเฉพาะกลุ่ม (IP Multicast) ประมาณ 270 ล้านหมายเลข

หมายเลขไอพีรุ่น 4 เขียนแทนด้วยเลขสัญกรณ์จุดฐานสิบแบบบัญญัติ ซึ่งประกอบด้วยเลขฐานสิบ 4 จำนวน แต่ละจำนวนมีค่าตั้งแต่ 0 ถึง 255 และคั่นด้วยจุด ตัวอย่างเช่น 172.16.254.1 เป็นต้น แต่ละส่วนของหมายเลขแทนกลุ่มของเลข 8 บิต (Octet) ผู้ดูแลระบบเครือข่ายแปลงหมายเลขไอพีเป็นสองส่วนคือ ส่วนของหมายเลขเครือข่าย (Network Number) และส่วนของหมายเลขแม่ข่าย (Host Identifier) และได้นำมาใช้กำหนดหมายเลขภายในองค์กรในเครือข่ายหนึ่งๆ

ตารางที่ 2.2 สถาปัตยกรรมเครือข่ายแบบคลาส [5]

คลาส	บิตขึ้นต้น	จำนวนบิตเครือข่าย	จำนวนบิตแม่ข่าย	จำนวนไอพีเครือข่าย	จำนวนไอพีแม่ข่าย	เลขที่อยู่เริ่มต้น	เลขที่อยู่สิ้นสุด
A	0	8	24	128 (2^{7})	16,777,216 (2^{24})	0.0.0.0	127.255.255.255
B	10	16	16	16,384 (2^{14})	65,536 (2^{16})	128.0.0.0	192.255.255.255
C	110	24	8	2,097,152 (2^{21})	256 (2^{8})	192.0.0.0	223.255.255.255
D (Multicast)	1110	ไม่ได้กำหนดไว้	ไม่ได้กำหนดไว้	ไม่ได้กำหนดไว้	ไม่ได้กำหนดไว้	224.0.0.0	239.255.255.255
E (Reserved)	1111	ไม่ได้กำหนดไว้	ไม่ได้กำหนดไว้	ไม่ได้กำหนดไว้	ไม่ได้กำหนดไว้	240.0.0.0	255.255.255.255

จากตารางที่ 2.2 เครือข่ายแบบคลาส (Classful Network) ได้ออกแบบให้สามารถกำหนดเครือข่ายเอกเทศได้จำนวนมากขึ้นและสามารถออกแบบเครือข่ายย่อย (Subnetwork) โดยละเอียดได้ 4 บิตแรกของออกเตตที่มีนัยสำคัญมากที่สุดของหมายเลขไอพี ถูกนิยามว่าเป็นคลาส โดยที่คลาส A, B และ C ได้นิยามขึ้นเพื่อการกำหนดที่อยู่สำหรับการแพร่สัญญาณเฉพาะราย (Unicast) โดยสากล ซึ่งคลาส D ได้นิยามไว้เพื่อใช้สำหรับงานแพร่สัญญาณในเครือข่าย (IP Multicast) มักใช้สำหรับงานประเภทสตรีมมิ่ง (Streaming) สื่อบนระบบเครือข่ายสาธารณะหรือเครือข่ายส่วนตัว ส่วนคลาส E จะนิยามไว้เป็นหมายเลขไอพีสงวน (Reserved IP Address) เพื่อใช้ในงานวิจัยต่าง ซึ่งจะไม่ถูกนำมาใช้ในเครือข่ายสาธารณะ

2.1.4 หมายเลขพอร์ต (Port Number)

หมายเลขพอร์ต (Port Number) [6] เป็นส่วนหนึ่งของข้อมูลที่ใช้ระบุตัวตนของผู้ส่งและผู้รับข้อความ ในระบบเครือข่ายคอมพิวเตอร์นอกจากจะใช้หมายเลขไอพีเป็นตัวกำหนดที่อยู่ของอุปกรณ์เชื่อมต่อสื่อสารแล้ว หมายเลขพอร์ตก็มีส่วนสำคัญในการกำหนดช่องทางการสื่อสารของอุปกรณ์ให้มีช่องทางการรับส่งข้อมูลที่ชัดเจนขึ้น โดยมีผู้ที่กำหนดลักษณะและมาตรฐานของหมายเลขพอร์ตคือองค์กรกำหนดหมายเลขอินเทอร์เน็ต (Internet Assigned Number Authority: IANA) ซึ่งเป็นผู้กำหนดมาตรฐานการใช้งานให้กับหมายเลขไอพีเช่นเดียวกัน

หมายเลขพอร์ตเป็นเลขฐานสองที่มีจำนวน 16 บิตหรือจำนวน 65,536 หมายเลข โดยมีการกำหนดแบ่งออกเป็นช่วงๆ ได้แก่ Well-Known, Registered

1. Well-Known Port เป็นช่วงของหมายเลขพอร์ตที่ถูกควบคุมมาตรฐานจาก IANA และถูกกำหนดให้ใช้โดยผู้ใช้งานที่มีสิทธิพิเศษ (Privileged User) พอร์ตเหล่านี้จะถูกใช้สำหรับการติดต่อสื่อสารกับเครื่องอุปกรณ์ที่มีระบบเวลาที่ยาวนาน (Server) เพื่อวัตถุประสงค์ในการให้บริการแก่ผู้ใช้ที่ไม่ทราบพอร์ตการเชื่อมต่อบริการ โดยทั่วไปนิยมใช้ในการเชื่อมต่อ TCP/IP ซึ่งพอร์ต Well-Known ได้ถูกกำหนดไว้ทั้งหมด 1,024 พอร์ต เริ่มตั้งแต่ 0 – 1,023 จากตารางที่ 2.3 ได้ยกตัวอย่างพอร์ตที่ได้รับความนิยมและเป็นที่รู้จักโดยทั่วไป

ตารางที่ 2.3 ตัวอย่างหมายเลขพอร์ต [6]

ชื่อโพรโทคอล	หมายเลขพอร์ต	โพรโทคอล TCP/IP	วัตถุประสงค์
Echo	7	TCP/UDP	ใช้ตรวจสอบสถานะของการเชื่อมต่อระหว่างอุปกรณ์
Discard	9	TCP/UDP	ส่งคำขอเพื่อยกเลิกการเชื่อมต่อกับอุปกรณ์ภายในเครือข่าย
FTP-data	20	TCP	ใช้รับและส่งข้อมูลระหว่างอุปกรณ์
FTP	21	TCP	ใช้ควบคุมคำสั่งการรับส่งข้อมูลระหว่างอุปกรณ์
SSH	22	TCP	ใช้เข้ารหัสและถอดรหัสการเข้าใช้งานระบบ
Telnet	23	TCP	ใช้ควบคุมอุปกรณ์ทางระยะไกล
SMTP	25	TCP	ให้บริการการรับส่งอีเมล์ระหว่างอุปกรณ์
HTTP	80	TCP	ให้บริการการเข้าถึงข้อมูลเว็บไซต์
POP3	110	TCP	ใช้เข้าถึงอีเมล์ของผู้ให้บริการโดยการคัดลอกข้อมูล
IMAP	143	TCP	ใช้เข้าถึงอีเมล์ของผู้ให้บริการโดยตรงผ่านระบบเครือข่าย

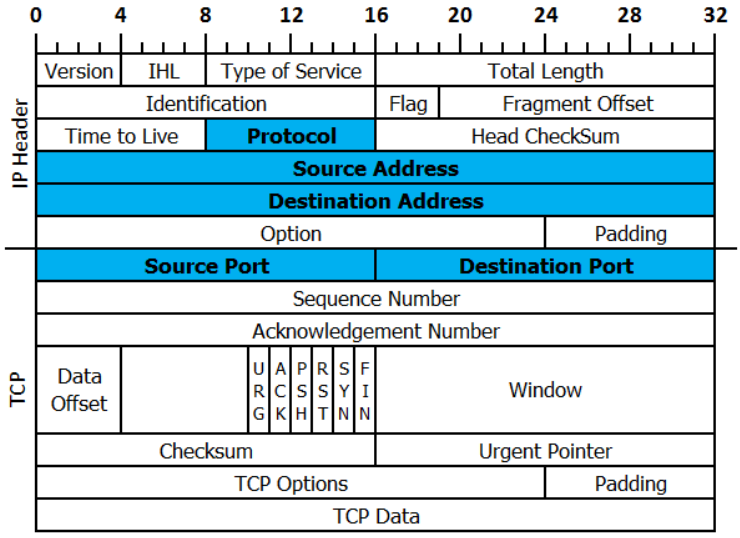
2. Registered Port เป็นช่วงของหมายเลขพอร์ตตั้งแต่ 1,024 – 65,535 ที่ทาง IANA จะไม่มีการควบคุม โดยส่วนใหญ่จะถูกใช้งานในระบบของผู้ใช้สามัญทั่วไปหรือโปรแกรมที่ดำเนินการโดยผู้ใช้สามัญ วัตถุประสงค์ในการใช้งานหมายเลขพอร์ตนี้จะเป็นลักษณะการเชื่อมต่อแบบเฉพาะเจาะจงซึ่งผู้ใช้อาจจะไม่ทราบหมายเลขพอร์ตการให้บริการเหล่านี้

2.1.5 โพรโทคอล (Protocol)

โพรโทคอล คือ ข้อกำหนดหรือข้อตกลงในการสื่อสารระหว่างคอมพิวเตอร์หรือเป็นภาษาในการสื่อสารระหว่างคอมพิวเตอร์ด้วยกัน ช่วยให้ระบบคอมพิวเตอร์สองระบบที่แตกต่างกันสามารถสื่อสารกันอย่างเข้าใจได้ โดยกำหนดข้อตกลงเกี่ยวกับการสื่อสารระหว่างคอมพิวเตอร์ต่างๆ ทั้งวิธีการรับส่งข้อมูล วิธีการตรวจสอบข้อผิดพลาดของการรับส่งข้อมูล การแสดงผลข้อมูลเมื่อมีรับและส่งข้อมูลกันระหว่างสองเครื่อง ซึ่งจะช่วยให้การสื่อสารเป็นมาตรฐานที่ชัดเจนยิ่งขึ้น

2.1.6 แพ็กเก็ตเฮดเดอร์

แพ็กเก็ตเฮดเดอร์ (TCP/IP Header) [7] คือ บล็อกโครงสร้างที่เก็บรายละเอียดต่างๆของแพ็กเก็ตไว้เพื่อใช้ในการรับและส่งข้อมูลในระบบเครือข่าย ในบทที่แล้วได้กล่าวถึงความหมายของหมายเลขไอพีรุ่น4 ดังนั้นเมื่อกล่าวถึงแพ็กเก็ตเฮดเดอร์ในบทนี้จะต้องรองรับหมายเลขไอพีรุ่น 4 เช่นเดียวกัน ซึ่งรายละเอียดต่างๆจะอธิบายดังต่อไปนี้



รูปที่ 2.2 แพ็กเก็ตเฮดเดอร์ไอพีรุ่น 4 [5, 7]

จากรูปที่ 2.2 เป็นโครงสร้างของแพ็กเก็ตเฮดเดอร์จะถูกแบ่งออกเป็น 2 ส่วนด้วยกันคือ IP Header และ TCP Header เมื่อกฎของไฟร์วอลล์มีการตรวจสอบแพ็กเก็ตที่ผ่านเข้าออกระบบเครือข่าย ไฟร์วอลล์จะเปรียบเทียบข้อมูล ไอพีต้นทาง (Source Address) 32 บิต ไอพีปลายทาง (Destination Address) 32 บิต พอร์ตต้นทาง (Source Port) 16 บิต พอร์ตปลายทาง (Destination Port) 16 บิต และโพรโทคอล (Protocol) 8 บิต

2.2 ไฟร์วอลล์โอเพนซอร์ส

2.2.1 IPtables

IPtables [8] คือ โปรแกรมที่ใช้ในการคัดกรองแพ็กเกตได้รับความนิยมมากในการใช้พัฒนาไฟร์วอลล์โอเพนซอร์สเป็นซอฟแวร์ที่พัฒนาสำหรับระบบปฏิบัติการลินุกซ์ทำงานในส่วน userspace คิดค้นโดย Rusty Russell และถูกพัฒนาภายใต้กลุ่มเน็ตฟิตเตอร์ (Netfilter) ในปี ค.ศ. 1998 แรกเริ่มได้ใช้ชื่อว่า IPfwadm ซึ่งรันบน Kernel 2.0 และได้พัฒนาต่อมาเป็น IPchain ทำงานบน

Kernel 2.2 และเวอร์ชันปัจจุบัน IPtables เวอร์ชัน 1.4.21 released (เมื่อวันที่ 9 สิงหาคม 2557) ซึ่งรันบน Kernel 2.4.x และ 2.6.x หลักการทำงานของ IPtables เป็นตารางของชุดข้อกำหนดหรือรูลเซ็ต (Rule Set) โดยแต่ละข้อกำหนดหรือรูลที่ผูกกับไอพีประกอบด้วยไอพีคลาสสิฟายเออร์ (IP classifier) ได้แก่ ไอพีต้นทาง (Source Address), ไอพีปลายทาง (Destination Address) เป็นต้น และเงื่อนไขการทำงาน

คุณสมบัติหลักของเน็ตฟิลเตอร์

1. รองรับการทำงาน Stateless packet filtering (IPV4 และ IPV6)
2. รองรับการทำงาน Stateful packet filtering (IPV4 และ IPV6)
3. รองรับทุกการทำงานของ NAT และ NAPT (IPV4)
4. มีโครงสร้างการทำงานพื้นฐานที่มีความยืดหยุ่นและสามารถปรับขยายได้

ตัวอย่างคำสั่งการใช้งาน IPtables

1. *#iptables -A FORWORD -s 192.168.1.0/24 -d 0/0 -p tcp --dport 21 --syn -Accept*
2. *#iptables -A FORWORD -s 192.168.1.0/24 -d 0/0 -p tcp --dport 23 --syn -j REJECT*
3. *#iptables -A FORWORD -s 0/0 -d 0/0 -t tcp --dport 25 --syn -j ACCEPT*

โดยคำสั่งแต่ละบรรทัดมีการทำงานดังนี้

1. อนุญาตให้ไอพีต้นทาง 192.168.1.1 - 192.168.1.254 ใช้งาน FTP (Port 21) กับทุกๆปลายทางได้
2. ไม่อนุญาตให้ไอพีต้นทาง 192.168.1.1 – 192.168.1.254 ใช้งาน Telnet (Port 23) กับทุกๆปลายทางได้
3. อนุญาตให้ไอพีต้นทางทุกไอพีใช้งาน SMTP กับทุกๆปลายทางได้

คำสั่งในการทำงานของไฟร์วอลล์ IPtable สามารถจำแนกตามตารางที่ 2.4

ตารางที่ 2.4 การทำงานของ IPtables [8]

หน้าที่	การทำงาน	รายการ
Filter	INPUT	กรองแพ็กเกตที่ส่งมายังเครื่องโลคอลโฮสต์ไม่ว่าจะผ่านอินเตอร์เฟสใดก็ตาม
	OUTPUT	กรองแพ็กเกตที่จะส่งออกจากเครื่องโลคอลโฮสต์
	FORWARD	กรองแพ็กเกตที่มีการส่งต่อโดยไม่ผ่านโลคอลโพรเซส
Nat	PREROUTING	สำหรับทำ DNAT และเป็นการเลี่ยงเทเบิ้ลฟิลเตอร์ในบางกรณี
	OUTPUT	สำหรับทำ NAT เข้ากับแพ็กเกตที่ส่งออกเครื่องโลคอลเอง
	POSTROUTING	สำหรับทำ SNAT
Mangle	PREROUTING	แก้ไขแพ็กเกตเฮดเดอร์เช่น TOS และเป็นจุดเริ่มต้นในการเก็บคอนเน็คชัน
	INPUT	แก้ไขแพ็กเกตหลังจากมีการเราท์แพ็กเกตก่อนส่งไปให้โลคอลโพเซส
	OUTPUT	หลังจากผ่านโลคอสโพรเซสจะเราท์อีกครั้งเพื่อค้นหาเส้นทางการส่งต่อของแพ็กเกตผ่านอินเตอร์เฟสใดและเป็นจุดเริ่มต้นใหม่
	FORWARD	แก้ไขแพ็กเกตหลังจากมีการเราท์และไม่ส่งต่อให้กับโลคอลโพเซส
	POSTROUTING	แก้ไขข้อมูลในแพ็กเกตหลังจากเราท์ก่อนจะส่งผ่านอินเตอร์เฟส

2.2.2 IPset

IPset [9] คือ โปรแกรมที่ใช้ในการคัดกรองแพ็กเกต ช่วยสนับสนุนการทำงานของ IPtables เป็นซอฟแวร์ที่พัฒนาสำหรับระบบปฏิบัติการลีนุกซ์ ทำงานในส่วนของ userspace และ kernel module ซึ่งรันบน Linux 2.4.x และ Linux 2.6.x ถูกพัฒนาโดย Jozsef Kadlecsik เวอร์ชันปัจจุบันคือ เวอร์ชัน 6.11-2 ข้อดีของ IPset คือความเร็วในการตรวจสอบแพ็กเกตโดยทำผ่านทาง IPTables Rule และสามารถความคุม Rule ได้แบบ ไดนามิคโดยไม่ต้องโหลด IPTables Rule ใหม่ ซึ่งสามารถแก้ IP Address และ MAC Address ผ่านทาง IPset ได้โดยตรง

ตัวอย่างการใช้งาน IPset ในการสร้างเซตที่นี้ใช้รูปแบบ macipmap

#ipset -N [setnam] [settype] --from [ip1] --to [ip2]

#ipset -N myset macipmap --from 192.168.0.10 --to 192.168.0.250

การเพิ่ม IP และ MAC เข้าไปในเซต

#ipset -A myset 192.168.0.11:AA:BB:CC:DD:EE:FF

IPset เป็นส่วนสนับสนุนการใช้งาน IPtables ให้มีความเร็วในการผ่านกฎของไฟร์วอลล์เพิ่มขึ้นโดยมีตัวอย่างการใช้งานร่วมกันดังนี้

#iptables -A FORWARD -m set --set myset src -j ACCEPT

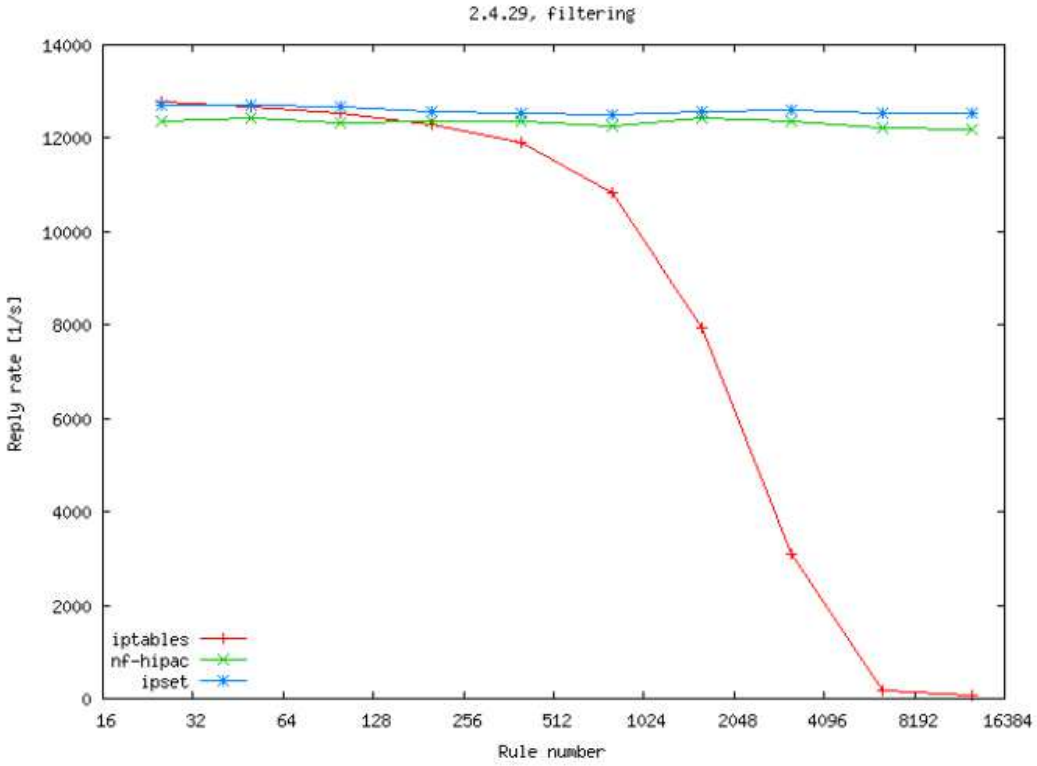
การลบ IP และ MAC ออกจาก set

#ipset -D myset 192.168.0.11:AA:BB:CC:DD:EE:FF

2.2.3 nf-HiPAC

nf-HiPAC [10] คือ โปรแกรมที่ใช้ในการคัดกรองแพ็กเกตที่โดดเด่นสำหรับระบบปฏิบัติการลินุกซ์ ทำงานในส่วนของ userspace สนับสนุนการทำงานของ IPtables รันบน Linux 2.4.x และ Linux 2.6.x เวอร์ชันล่าสุดคือ nf-HiPAC v0.9.1 เมื่อวัน 11 ตุลาคม 2005 ลักษณะการทำงานคล้ายกับ IPset เพื่อเพิ่มความสามารถในการตรวจสอบเข้าถึงกฎของไฟร์วอลล์ได้เร็วยิ่งขึ้น

เปรียบเทียบความสามารถของ IPtables, IPset และ nf-HiPAC จะได้ดังรูปที่ 2.3



รูปที่ 2.3 เปรียบเทียบการตอบกลับแพ็กเกตของ IPtables, IPset และ nf-HiPAC [9]

จากรูปที่ 2.3 จะเห็นได้ว่า ประสิทธิภาพการทำงานของ IPtables จะแปรผกผันกับจำนวนกฎของไฟร์วอลล์ คือเมื่อจำนวนกฎไฟร์วอลล์ของ IPtables มีจำนวนเพิ่มมากขึ้นจะทำให้ประสิทธิภาพการทำงานลดลง ซึ่งแตกต่างจาก IPset และ nf-HiPAC จะเห็นว่า ประสิทธิภาพในจำนวนกฎช่วง 16 – 32 กฎจะมีประสิทธิภาพเทียบเท่ากันทั้ง 3 ในช่วงที่จำนวนกฎของไฟร์วอลล์ที่มากกว่า 256 กฎ การทำงานของ IPset และ nf-HiPAC ยังคงมีประสิทธิภาพคงเดิม จึงสรุปโดยคราวว่าการทำงานของ IPset และ nf-HiPAC มีประสิทธิภาพเทียบเท่ากันและจำนวนกฎของไฟร์วอลล์ส่งผลต่อประสิทธิภาพการผ่านกฎน้อยมาก

2.3 หลักการของการจำกัดสิทธิ

หลักการของการจำกัดสิทธิ (Principle of Least Privilege) [11] หมายถึง หลักการที่ให้สิทธิกับผู้ใช้มีสิทธิตามความจำเป็นของการทำงานเท่านั้น ซึ่งหลักการนี้เป็นส่วนหนึ่งในการจัดการบริหารความเสี่ยง (Risk Management) การบริหารจัดการความเสี่ยงเป็นกระบวนการในการระบุช่องโหว่และภัยคุกคามทรัพยากรข้อมูลในองค์กรเพื่อใช้บรรลุวัตถุประสงค์ในการค้นหามาตรการป้องกันและแก้ไข

หลักการของการจำกัดสิทธิได้รับการยอมรับอย่างกว้างขวางว่าเป็นการพิจารณาการออกแบบที่มีความสำคัญในการเสริมสร้างการป้องกันข้อมูล กำหนดขอบเขตการทำงานที่อาจเกิดจากความผิดพลาดและพฤติกรรมที่เป็นอันตราย

ดังนั้นหากนำหลักการของการจำกัดสิทธิมาประยุกต์ใช้กับระบบไฟร์วอลล์โดยจำกัดการเข้าถึงข้อมูลและบริการของผู้ใช้ให้น้อยที่สุดก่อนจะส่งผลประโยชน์ดังต่อไปนี้

1. ระบบมีเสถียรภาพที่ดีขึ้น (Better System Stability) เมื่อมีการจำกัดการเข้าถึงข้อมูลตามขอบเขตของการทำงาน จะช่วยให้สามารถกำหนดขอบเขตความรับผิดชอบของผู้ใช้งานเมื่อเกิดผลกระทบต่างๆต่อทรัพยากรข้อมูลหรือการให้บริการต่างๆ และป้องกันการทำงานของผู้ใช้ที่ผิดพลาดเพื่อไม่ให้ส่งผลกระทบต่อข้อมูลอื่นที่ทำงานอยู่ในระบบเดียวกัน

2. ระบบมีการรักษาความปลอดภัยที่ดีขึ้น (Better System Security) เมื่อระบบมีการจำกัดการเข้าถึงข้อมูลเป็นโดยแบ่งกลุ่มรับผิดชอบ หากพบว่ากลุ่มของผู้ใช้ใดมีช่องโหวหรืออาจถูกโจมตีโดยสปายแวร์หรือไวรัส ระบบจะสามารถสลายการโจมตีและกำหนดขอบเขตการแพร่กระจายเพื่อไม่ให้เกิดผลกระทบต่อระบบโดยรวมได้อีกด้วย

3. ความง่ายดายในการใช้งาน (Ease of Deployment) โดยทั่วไปพบว่าในระบบการทำงานที่มีสภาพแวดล้อมขนาดใหญ่ จะทำให้ผู้ใช้คำนึงถึงขอบเขตและผลกระทบของการทำงานต่อระบบอื่นที่เกี่ยวข้องมากขึ้น จากประโยชน์ของสองข้อแรกนั้นจะช่วยให้ผู้ใช้สามารถจัดการขอบเขตการทำงานได้ชัดเจนและสะดวกยิ่งขึ้น

การประเมินความเสี่ยงเป็นการกำหนดมูลค่าเชิงปริมาณหรือคุณภาพของความเสี่ยงที่เกี่ยวข้องกับสถานการณ์ที่เป็นรูปธรรมและเป็นภัยคุกคามที่ได้รับการยอมรับ โดยคำนวณจากองค์ประกอบสององค์ประกอบด้วยกันคือ ขนาดของความสูญเสียที่อาจจะเกิดขึ้น และความน่าจะเป็นอาจจะเกิดขึ้น ดังนั้นหากมีการจำกัดการเข้าถึงข้อมูลโดยคำนึงถึงสิทธิน้อยที่สุดก่อน จะทำให้ความเสี่ยงที่อาจจะเกิดขึ้นกับระบบมีผลกระทบต่อองค์กรที่น้อยลงหรืออาจจะเป็นที่ยอมรับได้

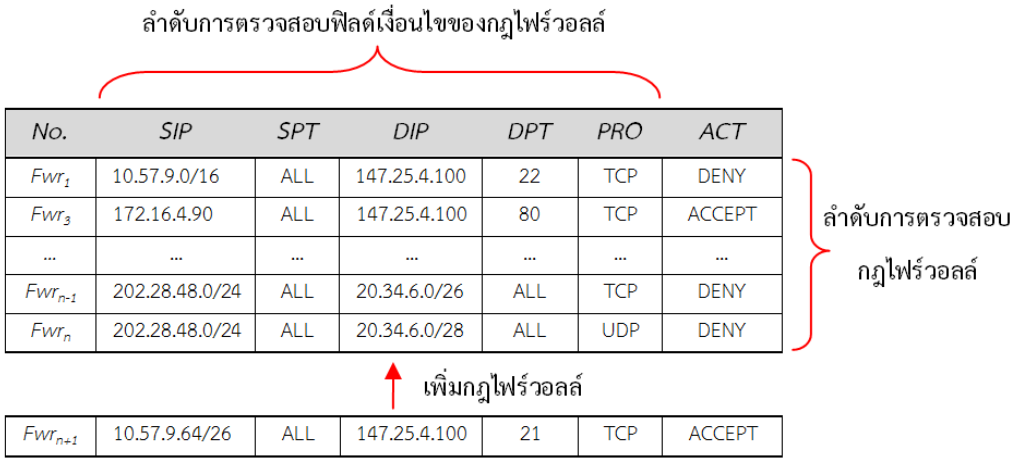
2.4 โครงสร้างข้อมูล

โครงสร้างข้อมูล (Data Structure) เป็นวิธีการจัดเก็บข้อมูลในคอมพิวเตอร์เพื่อให้สามารถใช้งานได้อย่างมีประสิทธิภาพ การเลือกโครงสร้างข้อมูลที่เหมาะสมจะทำให้สามารถเลือกใช้ขั้นตอนวิธีที่มีประสิทธิภาพไปพร้อมกันได้ โครงสร้างข้อมูลที่ออกแบบมาเป็นอย่างดีจะสามารถรองรับการประมวลผลของข้อมูลที่มีประมาณมากๆได้โดยใช้ทรัพยากรที่น้อยที่สุดเท่าที่จะเป็นไปได้ ทั้งในแง่ของเวลาและหน่วยความจำ

ในวิทยานิพนธ์นี้จะเลือกโครงสร้างข้อมูลที่เหมาะสมกับการทำงานของไฟร์วอลล์ ซึ่งคุณสมบัติหลักที่สำคัญที่ต้องคำนึงถึง อย่างแรกคือ โครงสร้างที่สามารถเข้าถึงข้อมูลได้อย่างรวดเร็ว เนื่องจากการทำงานหลักของไฟร์วอลล์คือการตรวจสอบข้อมูลที่ผ่านเข้าออกระบบเครือข่าย การเข้าถึงข้อมูลที่รวดเร็วจึงเป็นส่วนสำคัญที่สุดของไฟร์วอลล์ อย่างที่สองคือ การเก็บจัดเก็บข้อมูลของกฎไฟร์วอลล์เนื่องจากขอบเขตของข้อมูลกฎไฟร์วอลล์มีจำนวนมาก (104 บิต) ดังนั้นจะต้องเลือกโครงสร้างข้อมูลที่สามารถรองรับข้อมูลของกฎไฟร์วอลล์ให้ดีอีกด้วย และสุดท้ายคือระยะเวลาในการดำเนินการเพิ่ม ลบ และแก้ไขข้อมูลภายในโครงสร้าง ซึ่งโดยส่วนใหญ่การดำเนินการกับกฎของไฟร์วอลล์ไม่ถูกปรับปรุงและแก้ไขบ่อยครั้ง ขึ้นอยู่กับการเปลี่ยนแปลงนโยบายรักษาความปลอดภัยของระบบเครือข่าย ในหัวข้อต่อไปนี้จะอธิบายโครงสร้างที่สามารถนำมาใช้กับไฟร์วอลล์ มีการเปรียบเทียบประโยชน์และผลกระทบของการนำมาประยุกต์ใช้กับโครงสร้างของไฟร์วอลล์ดังต่อไปนี้

2.4.1 โครงสร้างข้อมูลแบบเมตริก

โครงสร้างข้อมูลแบบเมตริก [12] เป็นโครงสร้างของไฟร์วอลล์ที่ใช้งานในระบบปัจจุบัน (Traditional Firewall) มีการแบ่งแถวและคอลัมน์ชัดเจน กำหนดให้แต่ละคอลัมน์คือ ฟิลด์เงื่อนไขของกฎไฟร์วอลล์โดยปรกติไฟร์วอลล์จะมีทั้งหมด 6 ฟิลด์ด้วยกัน แต่จะมีเพียง 5 ฟิลด์เท่านั้นที่ใช้ในการตรวจสอบกับแพ็กเก็ตที่ไหลผ่านเข้าออกระบบ ได้แก่ หมายเลขไอพีต้นทาง *(SIP)*, หมายเลขไอพีปลายทาง *(DIP)*, หมายเลขพอร์ตต้นทาง *(SPT)*, หมายเลขพอร์ตปลายทาง *(DPT)*, โพรโทคอล *(PRO)* และฟิลด์ที่ 6 เป็นผลการตัดสินใจของกฎ *(ACT)* ดังรูปที่ 2.4



No.	SIP	SPT	DIP	DPT	PRO	ACT
Fwr_1	10.57.9.0/16	ALL	147.25.4.100	22	TCP	DENY
Fwr_3	172.16.4.90	ALL	147.25.4.100	80	TCP	ACCEPT
...	...	...	...	...	...	...
Fwr_{n-1}	202.28.48.0/24	ALL	20.34.6.0/26	ALL	TCP	DENY
Fwr_n	202.28.48.0/24	ALL	20.34.6.0/28	ALL	UDP	DENY
Fwr_{n+1}	10.57.9.64/26	ALL	147.25.4.100	21	TCP	ACCEPT

รูปที่ 2.4 โครงสร้างข้อมูลแบบเมตริก

การเปรียบเทียบข้อมูลโครงสร้างแบบเมตริก จะใช้คุณสมบัติการเปรียบเทียบแบบตามลำดับขั้น (Sequentially Matching) จากรูปที่ 2.4 โดยเริ่มเปรียบเทียบจากกฎลำดับที่ 1 *(Fwr_1)* ในฟิลด์ข้อมูลลำดับที่ 1 *(SIP)* จนถึงฟิลด์ข้อมูลลำดับที่ 5 *(PRO)* หากแพ็กเก็ตที่เปรียบเทียบและพบว่าไม่ตรงกับทุกฟิลด์เงื่อนไขของกฎไฟร์วอลล์ลำดับที่ 1 แพ็กเก็ตก็จะทำการเปรียบเทียบกับกฎในลำดับต่อไป เมื่อแพ็กเก็ตที่เปรียบเทียบตรงกับทุกฟิลด์เงื่อนไขของกฎไฟร์วอลล์ลำดับใดๆ จะแสดงผลการกระทำ *(ACT)* ของกฎๆนั้น คือ ยอมรับ (Accept) หรือ ปฏิเสธ (Deny) โดยที่ประสิทธิภาพในการผ่านกฎของไฟร์วอลล์เท่ากับ $O(d \times n)$ โดยที่ d คือ จำนวนฟิลด์เงื่อนไขของกฎไฟร์วอลล์ (โดยปรกติจะมีจำนวน 5 ฟิลด์เงื่อนไข) และ n คือ จำนวนกฎทั้งหมดของไฟร์วอลล์

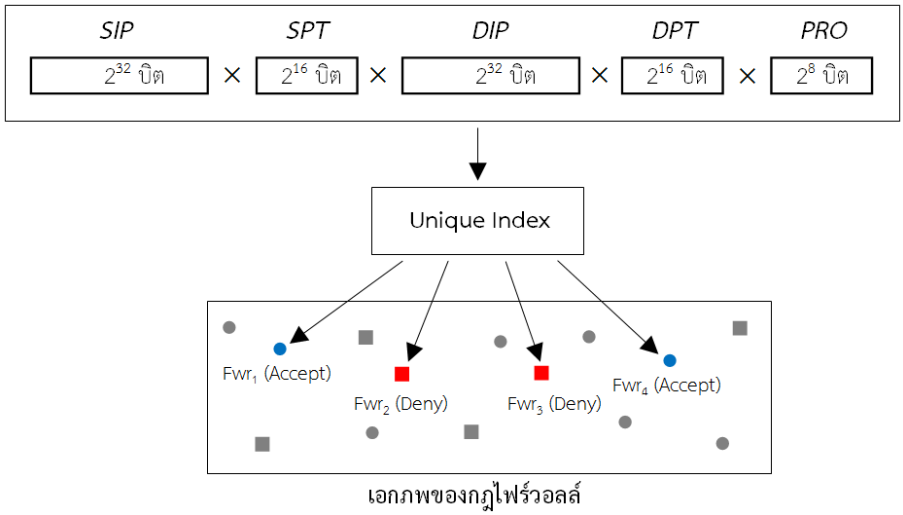
ในการดำเนินการของโครงสร้างข้อมูลแบบเมตริกจะเพิ่มกฎใหม่ไว้ลำดับหลังสุดของโครงสร้างโดยมีประสิทธิภาพเท่ากับ $O(1)$ เนื่องจากไม่มีการตรวจสอบใดๆ ซึ่งในการดำเนินการกับโครงสร้างแบบเมตริกจึงสามารถทำได้ง่าย เนื่องจากโครงสร้างไม่มีความซับซ้อนและจัดการกฎได้สะดวกและรวดเร็ว จึงเป็นที่นิยมในปัจจุบัน แต่ปัญหาหลักของโครงสร้างนี้คือประสิทธิภาพการเปรียบเทียบแพ็กเก็ตและกฎวิกลภาพที่เกิดขึ้นกับกฎของไฟร์วอลล์ซึ่งจะอธิบายรายละเอียดของปัญหาในหัวข้อต่อไป

2.4.2 โครงสร้างข้อมูลแบบผลคูณคาร์ทีเซียน

ผลคูณคาร์ทีเซียน (Cartesian Product) [12] คือการดำเนินการทางคณิตศาสตร์ที่ดำเนินการกับเซตหลายเซตได้ผลเป็นเซต (หรือ เซตผลคูณ) สำหรับเซต A และ B ผลคูณคาร์ทีเซียน $A \times B$ เป็นเซตคู่อันดับ (a,b) ที่ $a \in A$ และ $b \in B$

การประยุกต์นำแนวคิดนี้มาใช้กับโครงสร้างของไฟร์วอลล์ เพื่อเพิ่มประสิทธิภาพในการเข้าถึงข้อมูลให้รวดเร็ว ซึ่งแน่นอนว่ามีประสิทธิภาพเท่ากับ $O(1)$ กำหนดให้ข้อมูลของแพ็กเก็ตทั้ง 5

ฟิลด์เงื่อนไขเป็นสมาชิกของเซตจำนวน 5 เซต โดยเริ่มต้นจากการแจกแจงผลคูณคาร์ทีเซียนทุกคู่อันดับเซตเงื่อนไข ซึ่งจะได้เซตของผลคูณหรือเรียกว่า เอกภพของกฎไฟร์วอลล์ ประกอบด้วยตัวเลขของข้อมูลจำนวนไม่ซ้ำกัน (ตัวเลขดังกล่าวจะแสดงค่าบนเซตเอกภาพของกฎไฟร์วอลล์) พร้อมกันนั้นก็ทำให้การตรวจสอบตัวเลขจำนวนเต็มที่อาจจะมีโอกาสอ้างเกินกว่าสองครั้งเนื่องมาจากการสร้างกฎที่ซ้ำซ้อนกัน ซึ่งสามารถแสดงได้ ดังรูปที่ 2.5



รูปที่ 2.5 โครงสร้างข้อมูลแบบคูณคาร์ทีเซียน

จากรูปที่ 2.5 การแจกแจงเซตของกฎไฟร์วอลล์สามารถทำได้โดยการนำจำนวนสมาชิกทั้งหมดของฟิลด์เงื่อนไข *SIP*, *DIP*, *SPT*, *DPT*, *PRO* และ ACT มาคูณแบบคาร์ทีเซียนดังนี้

$$SIP \times SPT \times DIP \times DPT \times PRO \times ACT = \{(a,b,c,d,e,f) \mid a \in SIP,\ b \in SPT,\ c \in DIP,\ d \in DPT,\ e \in PRO,\ f \in ACT\}$$

กำหนดให้ a เป็นสมาชิกของเซตหมายเลขไอพีต้นทาง *(SIP)*

b เป็นสมาชิกของเซตหมายเลขพอร์ตต้นทาง *(SPT)*

c เป็นสมาชิกของเซตหมายเลขไอพีปลายทาง *(DIP)*

d เป็นสมาชิกของเซตหมายเลขพอร์ตปลายทาง *(DPT)*

e เป็นสมาชิกของเซตโพรโทคอล (*PRO*)

f เป็นสมาชิกของผลการกระทำของกฎ (*ACT*)

สมมุติให้กฎของไฟร์วอลล์ *(Fwr)* มีคุณสมบัติดังต่อไปนี้

$$Fwr:\{\ \{\{100,101\}\wedge\{80\}\wedge\{200,300\}\wedge\{80,443\}\wedge\{0\}\}\ \rightarrow Accept\ \}$$

โดยที่ $\{100,101\}$ คือ หมายเลขไอพีต้นทาง (*SIP*)

$\{80\}$ คือ หมายเลขพอร์ตต้นทาง (*SPT*)

$\{200,301\}$ คือ หมายเลขไอพีปลายทาง (*DIP*)

$\{80,443\}$ คือ หมายเลขพอร์ตปลายทาง (*DPT*)

$\{0\}$ คือ โพรโทคอล (*PRO*)

Accept คือ ผลการกระทำของกฎ (*ACT*)

จากตัวอย่างได้กำหนดค่าหมายเลขไอพีเป็นเลขฐานสิบให้ง่ายต่อการอธิบาย ซึ่งสามารหาจำนวนสมาชิกทั้งหมดที่สามารถแจกแจงด้วยผลคูณคาร์ทีเซียนดังนี้

$$n(SIP\times SPT\times DIP\times DPT\times PRO\times ACT) = n(SIP)\times n(SPT)\times n(DIP)\times n(DPT)\times n(PRO)\times n(ACT)$$

เมื่อแทนค่าเข้าสมการข้างต้นจะได้จำนวนสมาชิกทั้งหมดเท่ากับ $2\times1\times2\times2\times1\times1=8$ คู่อันดับซึ่งแจกแจงได้ดังนี้

คู่อันดับที่ 1 $(100\wedge80\wedge80\wedge200\wedge80\wedge0)\ \rightarrow$ Accept

คู่อันดับที่ 2 $(100\wedge80\wedge80\wedge200\wedge443\wedge0)\ \rightarrow$ Accept

คู่อันดับที่ 3 $(100\wedge80\wedge80\wedge300\wedge80\wedge0)\ \rightarrow$ Accept

คู่อันดับที่ 4 $(100\wedge80\wedge80\wedge300\wedge443\wedge0)\ \rightarrow$ Accept

คู่อันดับที่ 5 $(101\wedge80\wedge80\wedge200\wedge80\wedge0)\ \rightarrow$ Accept

คู่อันดับที่ 6 $(101\wedge80\wedge80\wedge200\wedge443\wedge0)\ \rightarrow$ Accept

คู่อันดับที่ 7 $(101\wedge80\wedge80\wedge300\wedge80\wedge0)\ \rightarrow$ Accept

คู่อันดับที่ 8 $(101\wedge80\wedge80\wedge300\wedge443\wedge0)\ \rightarrow$ Accept

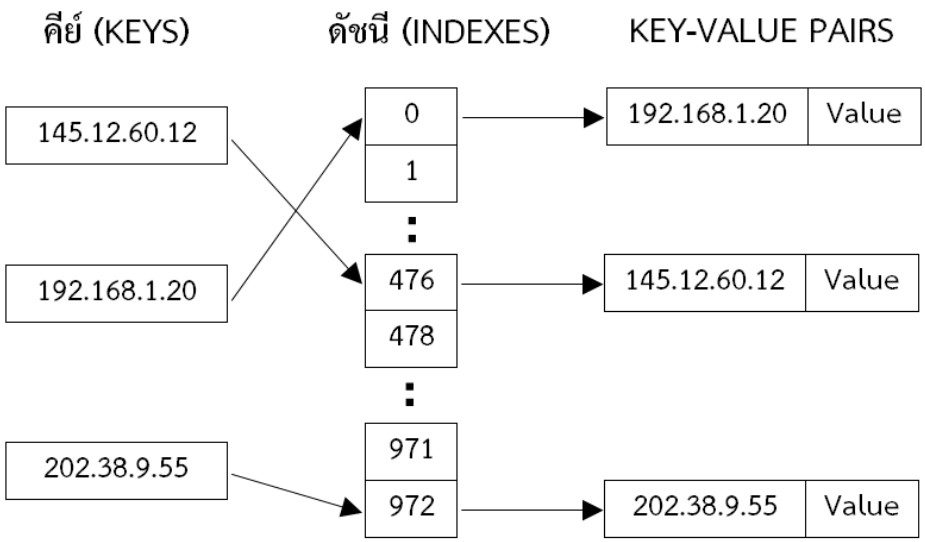
ขั้นต่อไปให้นำคู่อันดับในเซตที่ผ่านกระบวนการแจกแจงเป็นเลขฐานสองเรียงลำดับต่อกัน (เพื่อกำหนดให้ขอบเขตของเงื่อนไขกฎนั้นๆมีตำแหน่งที่ไม่ซ้ำกันหรือเรียกว่า Unique Index)

ในการเก็บข้อมูลของฟิลด์เงื่อนไขจะมีขอบเขตตามจำนวนบิตที่กำหนด (SIP 32 บิต, SPT 16 บิต, DIP 32 บิต, DPT 16 บิต และ PRO 8 บิต) ดังนั้นจำนวนข้อมูลทั้งหมดในเอกภพของโครงสร้างไฟร์วอลล์มีจำนวนเท่ากับ 104 บิตหรือจำนวน 2^{104} ซึ่งในทางปฏิบัติไม่เหมาะสมที่จะลงทุนเพื่อเก็บข้อมูลตัวเลขมากมายในหน่วยความจำได้ ดังนั้นปัญหาของโครงสร้างแบบผลคูณคาร์ทีเซียนจึงยังขนาด

ความสามารถในการจัดเก็บข้อมูล แต่สามารถแบ่งลำดับการตรวจสอบตามเซตของฟิลด์เงื่อนไขเพื่อลดประมาณการใช้หน่วยความจำ ในทางเดียวกันการผ่านกฎของไฟร์วอลล์ก็จะใช้เวลาเพิ่มขึ้นอีกด้วย

2.4.3 โครงสร้างข้อมูลแบบตารางแฮช

ตารางแฮช (Hash Table) [12] เป็นโครงสร้างข้อมูลในรูปแบบตาราง ซึ่งมักจะใช้แถวลำดับหรือแผนที่ขนาดใหญ่เพื่อใช้ในการจัดเก็บข้อมูลจำนวนมาก โดยมีลักษณ์การเก็บแบบดัชนี (Indexing) วิธีการเก็บนั้นจะนำข้อมูลมาผ่านฟังก์ชันฟังก์ชันหนึ่งซึ่งเรียกฟังก์ชันว่า ฟังก์ชันแฮช แล้วจะได้เลขจำเพาะกับข้อมูลนั้น (เรียกข้อมูลที่จะผ่านฟังก์ชันว่า คีย์ (Key)) กล่าวคือ ข้อมูลแต่ละตัวเมื่อผ่านฟังก์ชันแฮชแล้ว จะได้เลขที่แตกต่างกัน แล้วจึงนำข้อมูลไปเก็บไว้ในตารางแถวลำดับหรือแผนที่ที่กำหนดไว้ดังรูปที่ 2.6



รูปที่ 2.6 โครงสร้างข้อมูลแบบตารางแฮช

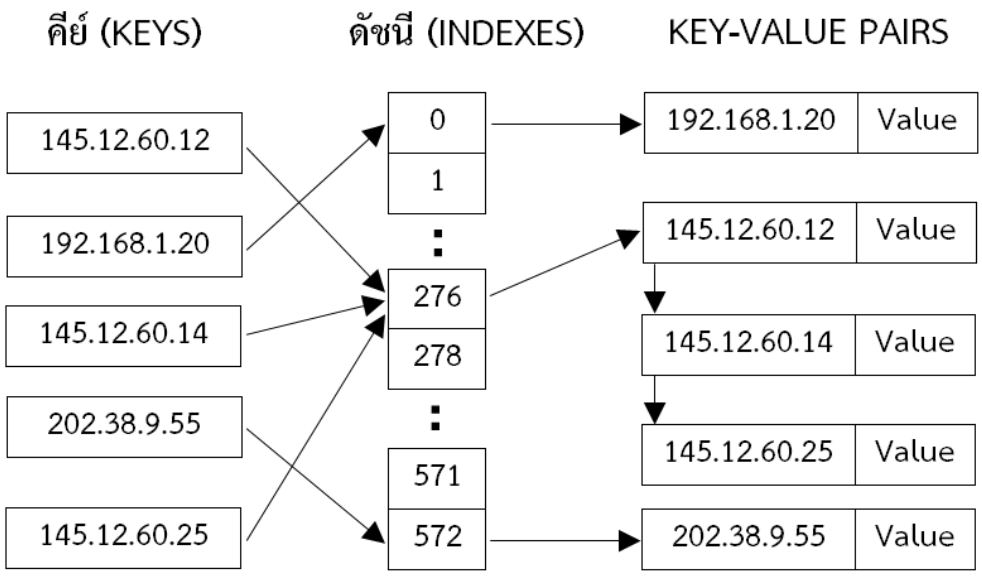
จากรูปที่ 2.6 เป็นการเก็บข้อมูลหมายเลขไอพี กำหนดให้หมายเลขไอพีเป็นคีย์ ซึ่งสามารถตรวจสอบได้โดยนำคีย์ผ่านฟังก์ชันแฮช ก็จะทราบถึงตำแหน่งที่อยู่ของหมายเลขไอพีว่าอยู่ในช่องใดทันที ซึ่งฟังก์ชันแฮชเปรียบเทียบได้กับสารบัญหรือดรรชนีของหนังสือ

ประสิทธิภาพในการเข้าถึงข้อมูลของตารางแฮชเน้นการเข้าถึงข้อมูลอย่างรวดเร็วเป็นเวลาคงที่ $O(1)$ แต่ข้อมูลเหล่านั้นจะต้องไม่มีลำดับและไม่ซ้ำกันเท่านั้น ข้อเสียของโครงสร้างข้อมูลแบบตารางแฮชคือ ขนาดของตารางมีขนาดใหญ่เท่ากับ $O(n)$ ซึ่งเมื่อนำมาเปรียบเทียบกับข้อมูลของกฎ

ไฟร์วอลล์แล้วในกรณีที่ไม่ซ้ำกัน n จะมีค่าเท่ากับจำนวนกฎของไฟร์วอลล์ 2^{104} ซึ่งในทางปฏิบัติไม่เหมาะสมที่จะนำมาลงทุน

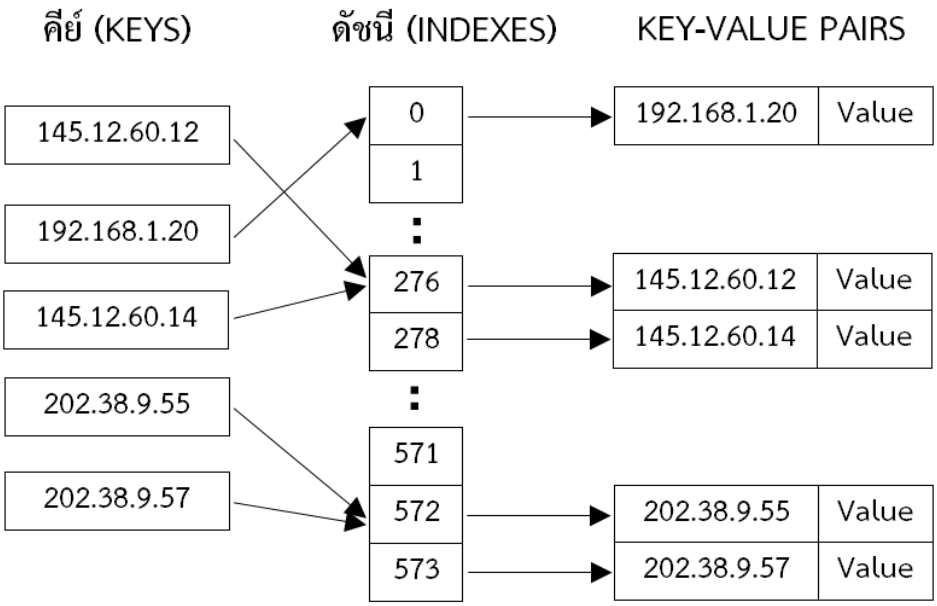
แต่การลดหน่วยความจำในการเก็บข้อมูลตารางแฮชก็สามารถทำได้เช่นกัน ซึ่งผลกระทบเมื่อลดจำนวนขนาดของตารางแฮชลงแล้ว จะทำให้ฟังก์ชันแฮชมีโอกาสที่คีย์ใดๆซ้ำกันในตำแหน่งเดียวกันได้ และวิธีที่นิยามใช้กันมากมีสองวิธีคือ การแฮชแบบรายการแยก (Separate Chaining) และการแฮชแบบเปิด (Open Addressing) โดยมีรายละเอียดดังนี้

1. การแฮชแบบรายการแยก คือ การใช้รายการหรือแถวลำดับขยายขนาดได้แทนการเก็บสมาชิกโดยตรง ตารางแฮชแต่ละช่องก็จะเก็บข้อมูลในลักษณะรายการ เมื่อคีย์ใดที่ต้องถูกเก็บในตารางแฮชตำแหน่งเดียวกัน ก็จะถูกเก็บไว้ในรายการนี้ต่อไปเรื่อยๆ ดังรูปที่ 2.7



รูปที่ 2.7 โครงสร้างข้อมูลตารางแฮชแบบรายการแยก

2. การแฮชแบบเปิด คือ การหาช่องในตารางแฮชใหม่โดยกระโดดไปจากที่เดิมเป็นฟังก์ชันหนึ่งจนกว่าจะหาช่องว่างเยอะจึงจะใส่ค่าลงไป ฟังก์ชันที่มักจะกระโดดมีสามแบบคือ การตรวจเชิงเส้น (Linear Probing), การตรวจกำลังสอง (Quadratic Probing) และการแฮชสองชั้น (Double Hashing) ซึ่งในวิทยานิพนธ์นี้จะไม่ขอนำเสนอรายละเอียดของวิธีการต่างๆ เพียงแต่นำเสนอให้เห็นถึงขั้นตอนการทำงานของการแฮชแบบเปิดเท่านั้น ดังรูปที่ 2.8



รูปที่ 2.8 โครงสร้างข้อมูลตารางแฮชแบบเปิด

ถึงแม้ว่าวิธีดังกล่าวข้างต้นจะช่วยให้สามารถใช้ตารางขนาดเล็กได้ แต่การให้เกิดการชนกันจนรายการยาวเกินไปหรือหนาแน่นเกินไป จะทำให้การเข้าถึงข้อมูลต้องเสียเวลาค้นหาในการรายการมากขึ้นและอาจผิดจุดประสงค์ความเป็นตารางแฮชที่ต้องการเข้าถึงข้อมูลอย่างรวดเร็วได้ ดังนั้นจึงนิยามค่าสัดส่วนบรรจุ (Load Factor: λ) มีค่าเท่ากับจำนวนข้อมูล (N) หารด้วยขนาดตาราง ($Tablesize$) สามารถนิยามได้ดังนี้

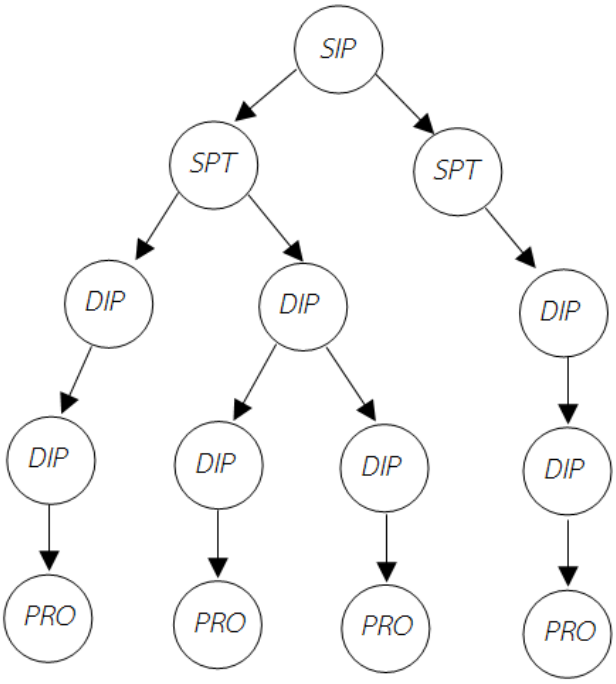
$$\lambda = \frac{N}{Tablesize} \qquad (1)$$

ในวิธีการแฮชแบบรายการแยกจะสามารถประมาณความยาวเฉลี่ยของรายการในแต่ละช่อง สำหรับวิธีการแฮบแบบเปิดจะต้องมีค่าสัดส่วนบรรจุน้อยกว่าหนึ่งอยู่เสมอ (ส่วนใหญ่อยู่ที่ 0.5)

2.4.4 โครงสร้างข้อมูลแบบต้นไม้

โครงสร้างข้อมูลแบบต้นไม้ (Tree Structure) [12] คือ โครงสร้างข้อมูลที่มีลักษณะเป็นกิ่งก้านสาขาแตกแขนงออกไป จะไม่มีวง (Loop) ที่โยงสมาชิกตัวต่างๆเข้าด้วยกัน โดยสมาชิกจะถูกเก็บไว้ในประเภทข้อมูลชนิดวัตถุ (Object) หรือโครงสร้าง (Structure) เรียกว่า โหนด (Node) ซึ่งจะมีตัวแปรซึ่งเก็บตัวชี้ (Pointer) ไปยังโหนดอื่นๆได้

โครงสร้างข้อมูลแบบต้นไม้เน้นการเข้าถึงข้อมูลที่รวดเร็วโดยการตัดทอนกิ่งที่ไม่สนใจ ซึ่งโครงสร้างแบบต้นไม้นี้มีหลายลักษณะพิเศษด้วยกัน เช่น ต้นไม้ค้นหาแบบทวิภาคที่สามารถไล่ตามกิ่งของต้นไม้โดยการเข้าถึงข้อมูลจะมีจำนวนเท่ากับความสูงของต้นไม้ซึ่งมีประสิทธิภาพเท่ากับ $O(\log n)$ และมีลักษณะพิเศษมากมาย ซึ่งในวิทยานิพนธ์นี้ได้ให้ความสนใจกับลักษณะโครงสร้างต้นไม้แบบไม่สมดุล (Unbalance Tree) เป็นโครงสร้างที่ไม่มีข้อจำกัดเรื่องขอบเขตของข้อมูลใดๆ ซึ่งสามารถเปรียบเทียบได้กับโครงสร้างแบบลิงค์ลิส (Linked List) เนื่องจากมีคุณสมบัติที่คล้ายคลึงกัน แต่ในวิทยานิพนธ์นี้จะนำเสนอในรูปแบบของโครงสร้างต้นไม้ไม่สมดุลแทนเพื่อให้การอธิบายเป็นไปโดยง่ายดังรูปที่ 2.9



รูปที่ 2.9 โครงสร้างข้อมูลแบบต้นไม้ไม่สมดุล

จากรูปที่ 2.9 จะเห็นว่าโครงสร้างแบบต้นไม้มีการแบ่งลำดับชั้นชัดเจน โดยเริ่มจากโหนดของหมายเลขไอพีต้นทาง (*SIP*) เรียกโหนดนี้ว่า ราก (Root) ที่มีเส้นเชื่อมออกไปยังโหนดต่างๆ ในระดับล่าง ได้แก่ โหนดของหมายเลขพอร์ตต้นทาง (*SPT*), โหนดของหมายเลขไอพีปลายทาง (*DIP*), โหนดของหมายเลขพอร์ตต้นปลายทาง (*DPT*) และโหนดของโพรโทคอล (*PRO*) ตามลำดับ

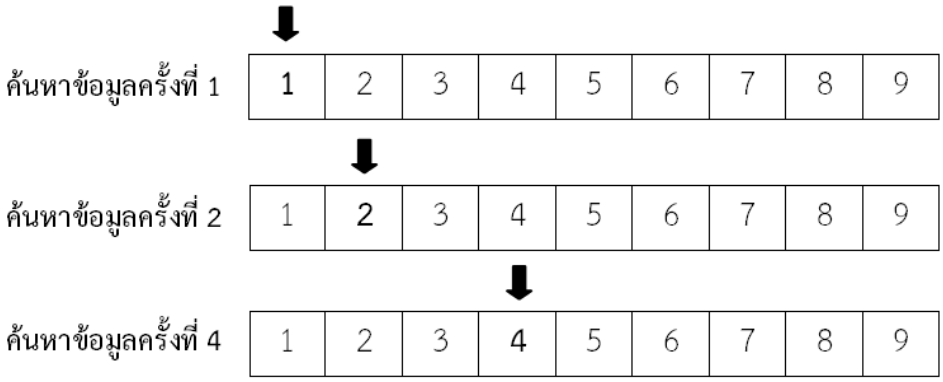
ประสิทธิภาพการเข้าถึงข้อมูลของโครงสร้างแบบต้นไม้มีค่าเท่ากับ $O(d \times (2n-1))$ กำหนดให้ d คือจำนวนเงื่อนไขของกฎไฟร์วอลล์หรือจำนวนโหนด (โดยปรกติจะมีค่าเท่ากับ 5) และ n คือจำนวนกฎทั้งหมดของไฟร์วอลล์

2.5 การค้นหาข้อมูล

แนวคิดในการค้นหาข้อมูลมีหลายรูปแบบด้วยกันขึ้นอยู่กับการนำไปใช้ให้เหมาะสมกับงานต่างๆ ซึ่งในวิทยานิพนธ์นี้ได้ใช้แนวคิดในการค้นหาข้อมูล 2 แบบด้วยกัน ซึ่งแต่ละรูปแบบได้จำแนกให้เหมาะสมกับการค้นหาในขั้นตอนต่างๆ

2.5.1 การค้นหาแบบเชิงเส้น

การค้นหาแบบเชิงเส้นหรือแบบตามลำดับ (Linear Search or Sequential Search) [12] เป็นการค้นหาข้อมูลที่มีรูปแบบเรียบง่าย และมีการนำไปใช้กับงานได้หลากหลาย ขั้นตอนในการค้นหาแบบเชิงเส้นนั้น เริ่มต้นจากการค้นหาข้อมูลตั้งแต่ข้อมูลที่ 1 ของรายการไปไปจนถึงข้อมูลท้ายสุด ดังรูปที่ 2.10 ค้นหาข้อมูล 4 ในรายการ

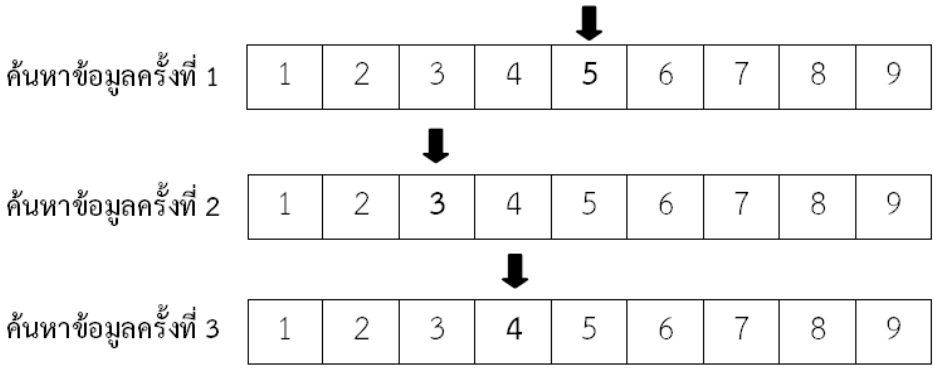


รูปที่ 2.10 การค้นหาข้อมูลแบบเชิงเส้น

จากรูปที่ 2.10 จะเห็นได้ว่าการค้นหาข้อมูลแบบเชิงเส้น จะไม่สนใจลำดับการจัดเรียงของข้อมูล เนื่องจากการค้นหานี้จะค้นหาข้อมูลทุกข้อมูลในรายการ (Array) ประสิทธิภาพของการค้นหาแบบเชิงเส้นจะเท่ากับ $O(n)$ โดยที่ n คือจำนวนของข้อมูลในรายการ

2.5.2 การค้นหาแบบทวิภาค

การค้นหาแบบทวิภาค (Binary Search) [13] คือ ขั้นตอนวิธีการค้นหาตำแหน่งของข้อมูลที่ต้องการ (ข้อมูลที่นำเข้าหรือคีย์ (Key)) โครงสร้างของข้อมูลที่ใช้ค้นหาจะต้องมีการจัดเก็บข้มูลเป็นแนวเส้นตรง (Linear Structure) และข้อมูลที่ค้นหาจะต้องมีการลำดับข้อมูลไว้ก่อนแล้ว ซึ่งในวิทยานิพนธ์นี้พบว่าเมื่อกฎของไฟร์วอลล์ปราศจากกฎวิกลภาพแล้วกฎของไฟร์วอลล์จะสามารถเปลี่ยนตำแหน่งได้โดยอิสระ แต่ความหมายของไฟร์วอลล์ยังคงทำงานดังเดิมดังรูปที่ 2.11 ค้นหาข้อมูล 4 ในรายการ



รูปที่ 2.11 การค้นหาข้อมูลแบบทวิภาค

ขั้นตอนการค้นหาแบบทวิภาคเริ่มต้นจากกำหนดค่า จุดต่ำสุด (Lower Point) คือข้อมูลตัวแรกของชุดข้อมูลและกำหนด จุดสูงสุด (Higher Point) คือข้อมูลตัวสุดท้ายของชุดข้อมูล จากทำการเปรียบเทียบข้อมูลที่อยู่กึ่งกลางระหว่าง จุดต่ำสุดกับจุดสูงสุด กรณีที่เปรียบเทียบแล้วพบว่าข้อมูลตรงกัน (กรณีที่ดีที่สุดมีค่า $O(1)$) ก็จะหยุดการค้นหาทันที แต่ถ้าข้อมูลไม่ตรงกันจะตรวจสอบว่า คีย์มีค่ามากกว่าหรือน้อยกว่าข้อมูลที่เปรียบเทียบ กรณีที่คีย์มากกว่าข้อมูลกำหนดให้จุดต่ำสุดเป็นข้อมูลที่อยู่กึ่งกลาง และถ้ากรณีที่คีย์น้อยกว่าข้อมูลกำหนดให้จุดมากสุดเป็นข้อมูลที่อยู่กึ่งกลาง จากนั้นให้ทำการค้นหาข้อมูลที่อยู่ตำแหน่งกึ่งกลางของ จุดต่ำสุดกับจุดสูงสุด และนำมาเปรียบเทียบกับคีย์ต่อไปเรื่อยๆ จนกว่าจะพบข้อมูลหรือจุดต่ำสุดเท่ากับจุดสูงสุด (ไม่พบข้อมูล) แล้วจึงจะจบการค้นหา

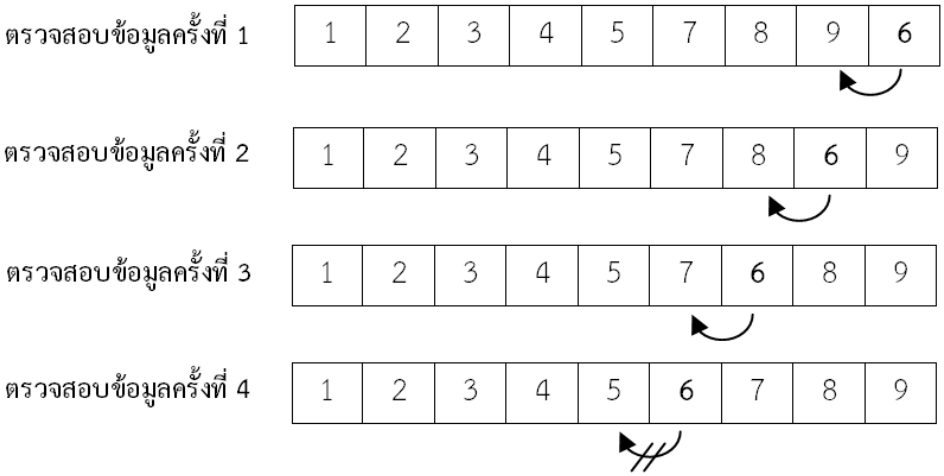
ตารางที่ 2.5 เปรียบเทียบประสิทธิภาพในการค้นหาข้อมูล

กรณีการค้นหาข้อมูล	การค้นหาแบบเชิงเส้น	การค้นหาแบบทวิภาค
ประสิทธิภาพการค้นหากรณีที่เลวร้ายที่สุด (Worst Case)	$O(n)$	$O(\log n)$
ประสิทธิภาพการค้นหาโดยเฉลี่ย (Average Case)	$O(n)$	$O(\log n)$
ประสิทธิภาพการค้นหากรณีที่ดีที่สุด (Best Case)	$O(1)$	$O(1)$
หน่วยความจำที่ใช้ในการค้นหา (Space Complexity)	$O(1)$	$O(1)$

2.6 การเรียงลำดับข้อมูล

การเรียงลำดับแบบแทรก (Insertion Sort) [13] เป็นวิธีการเรียงลำดับที่ทำการเพิ่มสมาชิกใหม่เข้าไปในเซตที่มีการเรียงลำดับอยู่ก่อนแล้ว และทำให้เซตใหม่ที่ได้มีสมาชิกทุกตัวเรียงลำดับด้วยการเรียงลำดับในวิธีนี้เป็นวิธีที่ง่ายไม่ซับซ้อน เปรียบเสมือนวิธีการจัดเรียงลำดับไพ่ในมือ นั่นคือในขณะเล่น

ไพ่เรามักเรียงไพ่ในมือตามลำดับตัวเลข และเมื่อหยิบไพ่ใหม่เข้ามาเพิ่มแทรกลงไประหว่างไพ่ที่เรียงไว้แล้ว โดยจะได้ไพ่ที่เรียงลำดับตัวเลขเช่นกัน ดังรูปที่ 2.12



รูปที่ 2.12 การเรียงลำดับแบบแทรก

จากรูปที่ 2.12 ขั้นตอนการเรียงลำดับข้อมูลเริ่มจากการตรวจสอบข้อมูลที่เพิ่มเข้ามาใหม่ในเซตของข้อมูล {6} โดยนำมาเปรียบเทียบกับข้อมูลที่อยู่ลำดับก่อนหน้าของข้อมูล {9} ถ้าหากข้อมูลที่นำเข้ามาน้อยกว่าข้อมูลที่ตรวจสอบในรายการ ให้สลับลำดับของข้อมูล {6, 9} แล้วทำการตรวจสอบข้อมูลไปเรื่อยจนกว่าจะเจอข้อมูลที่มากกว่า {5} จึงจะสิ้นสุดการเรียงลำดับข้อมูล

2.7 หลักการประมาณด้วยสัญกรณ์โอใหญ่

สัญกรณ์โอใหญ่ (Big-O Notation) [13] เป็นสัญกรณ์คณิตศาสตร์ที่ใช้บรรยายพฤติกรรมเชิงเส้นกำกับของฟังก์ชัน โดยระบุเป็นขนาด (Magnitude) ของฟังก์ชัน ใช้ในการเขียนเพื่อประมาณพจน์ในทางคณิตศาสตร์ และถูกใช้ประยุกต์ในทางวิทยาการคอมพิวเตอร์เพื่ออธิบายประสิทธิภาพการทำงานของโปรแกรมและอัลกอริทึมในกรณีที่ต้องประมวลผลข้อมูลจำนวนมาก และเพื่อใช้อธิบายประสิทธิภาพของขั้นตอนวิธีหรือโครงสร้างข้อมูลนั้นๆ ซึ่งในวิทยานิพนธ์ได้นำหลักการนี้มาประมาณเวลาที่ใช้ในการเข้าถึงข้อมูลของโครงสร้างไฟร์วอลล์และระยะเวลาที่ใช้ในการสร้างกฎเพื่อเปรียบเทียบประสิทธิภาพการทำงานของไฟร์วอลล์โครงสร้างต่างๆ

สัญกรณ์โอใหญ่จะระบุลักษณ์ของฟังก์ชันตามอัตราการเติบโต ถึงแม้ฟังก์ชันจะแตกต่างกันแต่หากมีการเจริญเติบโตเท่ากันก็จะมีสัญกรณ์โอใหญ่เท่ากัน สำหรับสัญกรณ์โอใหญ่แล้วจะพิจารณา

เฉพาะขอบเขตบนของอัตราการเติบโตของฟังก์ชัน เช่น ฟังก์ชัน n^2+n และ $n+4$ ล้วนมีอัตราการเติบโตน้อยกว่าหรือเท่ากับ n^2 นั่นคืออัตราการเติบโตของฟังก์ชัน n^2 เป็นขอบเขตบนของ n^2+n และ $n+4$ จึงกล่าวได้ว่า n^2+n และ $n+4$ เป็นสมาชิกของฟังก์ชัน $O(n^2)$

สัญกรณ์โอใหญ่มีการใช้งานในสองกรณีด้วยกันได้แก่ กรณีเส้นกำกับอนันต์ และ เส้นกำกับกณิกนันต์ ความแตกต่างระหว่างสองกรณีนี้เป็นความแตกต่างในขั้นการประยุกต์ใช้ อย่างไรก็ตาม นิยามเชิงรูปนัยของ “โอใหญ่” นั้นเหมือนกันในทั้งสองกรณี มีเพียงลิมิตสำหรับค่าตัวแปรของฟังก์ชันเท่านั้นที่แตกต่างกัน โดยจะอธิบายการใช้งานของทั้งสองกรณีได้ดังนี้

2.7.1 เส้นกำกับอนันต์ (Infinite asymptotic)

สัญกรณ์โอใหญ่มีประโยชน์ในการใช้วิเคราะห์ขั้นตอนวิธีการ เพื่อหาประสิทธิภาพของขั้นตอนวิธีการ ตัวอย่าง สมมุติให้เวลา (จำนวนขั้นตอน) ที่ใช้ในการแก้ไขปัญหาขนาด n ฟังก์ชันเป็น

$$T(n)=4n^2-2n+2 \qquad (2)$$

เมื่อ n มีค่ามากขึ้น พจน์ n^2 จะใหญ่ขึ้นครอบงำพจน์อื่นๆ จนกระทั่งสามารถละเลยพจน์อื่นๆได้ ที่สำคัญสัมประสิทธิ์ของแต่ละพจน์จะขึ้นอยู่กับรายละเอียดปลีกย่อยของการนำขั้นตอนวิธีไปปฏิบัติ ตลอดจนฮาร์ดแวร์ที่ใช้ในการดำเนินการ ฉะนั้นจึงสามารถละเลยได้เช่นกัน สัญกรณ์โอใหญ่จะเก็บเฉพาะส่วนที่เหลือจากที่ละเลยได้ข้างต้นจึงเขียนได้ว่า $T(n)\in O(n^2)$

2.7.2 เส้นกำกับกณิกนันต์ (Infinitesimal asymptotic)

สัญกรณ์โอใหญ่ยังใช้เพื่อแสดงพจน์ของค่าคลาดเคลื่อนโดยประมาณในฟังก์ชันทางคณิตศาสตร์ ตัวอย่างเช่น

$$e^x=1+x+x^2+\frac{x^2}{2}+O(x^3) \quad as \quad x\to 0 \qquad (3)$$

หมายความว่า เมื่อ x มีค่าเข้าใกล้ศูนย์ผลต่างของฟังก์ชัน e^x กับ $1+x+x^2+\frac{x^2}{2}$ (หรืออาจกล่าวอีกในหนึ่งว่าเป็นความคลาดเคลื่อนของสองฟังก์ชันนี้) จะมีอยู่ในสับเซตของ $O(x^3)$ นั่นเองหรือเขียนเป็นสัญลักษณ์ว่า

$$\left|e^x-\left(1+x+x^2+\frac{x^2}{2}\right)\right|\in O(x^3) \quad as \quad x\to 0 \qquad (4)$$

ในบางครั้งสัญกรณ์โอใหญ่อาจมีการครอบคลุมมากเกินไป เช่น $O(n^2) \subset O(n^3)$ เป็นต้น จึงทำให้สำหรับฟังก์ชันใดๆอาจอยู่ในเซตของสัญกรณ์โอใหญ่หลายค่า จึงมีการกำหนดรูปแบบฟังก์ชันอย่างง่าย ให้ตอบในรูปสัญกรณ์โอใหญ่มาตราฐานน้อยสุด (Little-o Notation) กล่าวคือตอบในรูปแบบมาตรฐานที่เล็กที่สุด ซึ่งมักจะอนุโลมให้ใช้สัญลักษณ์เท่ากับ $(=)$ แทนสัญลักษณ์สมาชิก $(\in)$ เมื่อใช้กับรูปสัญกรณ์โอใหญ่มาตรฐานน้อยสุดนี้ เช่น $n^2 + 4 = O(n^2)$

ทางวิทยาการคอมพิวเตอร์ การทำงานที่มีสัญกรณ์โอใหญ่มาตรฐานน้อยสุดมีขนาดเล็กเท่าใดแสดงว่ามีประสิทธิภาพในการทำงานยิ่งเร็วเท่านั้น สัญกรณ์โอใหญ่มาตราฐานเรียงจากขนาดเล็กไปใหญ่ ให้ k เป็นค่าคงที่ใดๆ ที่มากกว่าศูนย์ และ n เป็นโดเมนของฟังก์ชัน ดังตารางที่ 2.4

ตารางที่ 2.6 สัญกรณ์โอใหญ่มาตราฐาน

สัญกรณ์โอใหญ่มาตราฐาน	ชื่อฟังก์ชัน	ตัวอย่าง
$O(1)$	ค่าคงที่	การตรวจสอบเลขฐานสองเป็นเลขคู่หรือเลขคี่
$O(\log n)$	ลอการิทึม	การค้นหาข้อมูลที่มีการเรียงลำดับด้วยโดยการค้นหาแบบทวิภาค (Binary Search) หรือการค้นหาในโครงสร้างต้นไม้สมดุล (Balance Tree Search)
$O(n^k), 0 < k < 1$	เอกซ์โพเนนเชียลฐานเศษส่วนแท้	การค้นหาข้อมูลในโครงสร้างต้นไม้เคดี (kd-Tree Search)
$O(n)$	สมการเชิงเส้น	การค้นหาข้อมูลในรายการข้อมูลที่ไม่มีการเรียงลำดับในโครงสร้างต้นไม้ไม่สมดุลหรือในอาเรย์ (กรณีเลวร้ายที่สุด)
$O(n \log n)$	สมการอนุพันธ์ย่อยกึ่งเชิงเส้น	การเรียงลำดับแบบเร็ว (Quick Sort) (ในกรณีที่ดีที่สุด)
$O(n^2)$	สมการกำลังสอง	การเรียงลำดับแบบฟอง (Bubble Sort) หรือการคูณตัวเลข n หลักโดยวิธีการง่ายๆ
$O(n^k), k > 1$	โพลีโนเมียล หรือสมการพีชคณิตเชิงเส้น	การแยกโครงสร้างต้นไม้ไวยากรณ์ (Tree-Adjoin grammar) หรือ การเปรียบเทียบกราฟสองส่วน
$O(k^n), k > 1$	เอกซ์โพเนนเชียล	แก้ไขปัญหาการเดินทางของพนักงานขาย (Traveling Salesman Problem) โดยใช้กำหนดการพลวัต (Dynamic Programming)
$O(n!)$	แฟกทอเรียล	แก้ไขปัญหาการเดินทางของพนักงานขาย (Traveling Salesman Problem) โดยใช้การค้นหาทุกกรณี (Brute Force Search) หรือ การแจกแจงพาร์ทิชันทั้งหมดของเซต

2.8 งานวิจัยที่เกี่ยวข้อง

ในปัจจุบันมีงานวิจัยที่ได้แก้ไขปัญหาของไฟร์วอลล์วอลล์หลายด้านด้วยกัน ซึ่งส่วนใหญ่มุ่งเน้นไปที่ปัญหาด้านวิกลภาพของกฎไฟร์วอลล์และประสิทธิภาพในการผ่านกฎไฟร์วอลล์ ซึ่งสาเหตุของปัญหาโดยส่วนใหญ่เกิดจาก การสร้างกฎที่ซับซ้อนเข้าใจยากทำให้มีการจัดลำดับของกฎไม่เหมาะสม การที่สร้างกฎเกิดความจำเป็น และการสร้างกฎที่มีกฎวิกลภาพบนระบบไฟร์วอลล์ จากปัญหาที่กล่าวมาพบว่า ปัญหาที่ส่งผลกระทบมากที่สุดต่อการทำงานของไฟร์วอลล์คือ ปัญหาด้านวิกลภาพของกฎไฟร์วอลล์ เนื่องจากส่งผลกระทบโดยตรงด้านความปลอดภัยของระบบเครือข่าย มีงานวิจัยทีนำเสนอวิธีการตรวจสอบและนำเสนอวิธีการแก้ไขปัญหาต่างๆดังนี้

2.8.1 วิกลภาพของกฎไฟร์วอลล์

ความผิดปรกติของกฎไฟร์วอลล์ โดยมีกฎตั้งแต่สองกฎขึ้นไปให้ความหมายที่ขัดแย้งกันหรือให้ความหมายที่ซ้ำซ้อน คาบเกี่ยวกัน โดย Al-Shaer และคณะ [1] ได้กำหนดรูปแบบวิกลภาพของกฎไฟร์วอลล์ไว้ 4 รูปแบบด้วยกัน ยกตัวอย่างได้จากตารางที่ 2.7 ดังนี้

ตารางที่ 2.7 กฎของไฟร์วอลล์ที่เกิดวิกลภาพ

No	*SIP*	*SPT*	*DIP*	*DPT*	*PRO*	*ACT*
Fwr_1	202.28.34.234	All	All	80	TCP	Deny
Fwr_2	202.28.34.0/24	All	All	80	TCP	Accept
Fwr_3	All	All	140.192.17.0/24	80	TCP	Deny
Fwr_4	120.192.27.0/16	All	140.192.17.0/26	21	TCP	Accept
Fwr_5	120.192.27.30	All	140.192.17.40	21	TCP	Accept
Fwr_6	All	All	All	All	TCP	Deny

1. กฎถูกบัง (Shadowing Anomaly) คือ การที่กฎใดๆถูกบังโดยกฎอื่นที่อยู่ลำดับก่อนหน้า เมื่อทุกฟิลด์ของ Fwr_y เป็นสมาชิกของ Fwr_x และทั้งสองกฎมีผลการกระทำที่ต่างกัน โดยที่ x<y

$$\forall i : Fwr_y[i] \subseteq Fwr_x[i] \wedge Fwr_x[\text{ACT}] \neq Fwr_y[\text{ACT}]$$

กำหนดให้ $i \in \{SIP, SPT, DIP, DPT, PRO\}$

จากตัวอย่างจะได้ว่า Fwr_4 เป็นกฎที่ถูกบังจาก Fwr_3 ดังนั้น สมาชิกของ Fwr_4 จะไม่ถูกกระทบกับแพ็กเก็ตใดๆเลย

2. กฎเกี่ยวพันธ์ (Correlation Anomaly) คือ การที่กฎใดๆมีการเกี่ยวพันกับกฎอื่น เมื่อบางฟิลด์ของ Fwr_x เป็นสมาชิกของ Fwr_y และบางฟิลด์ของ Fwr_y เป็นสมาชิกของ Fwr_x ทั้งสองกฎมีผลการกระทำที่แตกต่างกัน โดยที่ x<y

$$\forall i : Fwr_x[i] \Diamond Fwr_y[i] \wedge \exists i : Fwr_x[i] \subset Fwr_y[i] \wedge \exists j : Fwr_x[j] \supset Fwr_y[j] \wedge Fwr_x[\text{ACT}] \neq Fwr_y[\text{ACT}] \wedge i \neq j$$

กำหนดให้ $\Diamond \in \{\subset, \supset, =\}$, $i, j \in \{SIP, SPT, DIP, DPT, PRO\}$

จากตัวอย่างจะได้ว่า Fwr_2 เป็นกฎที่เกี่ยวพันธ์กับ Fwr_3 ดังนั้น สมาชิกที่ $Fwr_2 \cap Fwr_3$ ของ Fwr_3 จะไม่ถูกกระทบกับแพ็กเก็ตใดๆเลย

3. กฎคลุมเครือ (Generalization Anomaly) คือการที่กฎใดๆครอบคลุมกฎอื่น เมื่อทุกฟิลด์ของ Fwr_x เป็นสมาชิกของแต่ละฟิลด์บน Fwr_y ทั้งสองกฎมีผลการกระทำที่แตกต่างกัน โดยที่ x<y

$$\forall i : Fwr_x[i] \subseteq Fwr_y[i] \wedge \exists j : Fwr_x[j] \neq Fwr_y[j] \wedge Fwr_x[\text{ACT}] = Fwr_y[\text{ACT}]$$

กำหนดให้ $i, j \in \{SIP, SPT, DIP, DPT, PRO\}$

จากตัวอย่างจะได้ว่า Fwr_1 เป็นกฎที่ครอบคลุมกับ Fwr_2 สมาชิกที่ $Fwr_2 \cap Fwr_3$ ของ Fwr_3 จะไม่ถูกกระทบกับแพ็กเก็ตใดๆเลย

4. กฎซ้ำซ้อน (Redundancy Anomaly) คือ การที่กฎใดๆซ้ำซ้อนกับกฎอื่น เมื่อทุกฟิลด์ของ Fwr_y เป็นสมาชิกของ Fwr_x ทั้งสองมีผลการกระทำที่เหมือนกัน โดยที่ x<y

$$\forall i : Fwr_y[i] \subseteq Fwr_x[i]$$

กำหนดให้ $i \in \{SIP, SPT, DIP, DPT, PRO, ACT\}$

จากตัวอย่างจะได้ว่า Fwr_5 เป็นกฎที่กระทำซ้ำซ้อนกับ Fwr_4 ดังนั้น Fwr_5 จะไม่ถูกกระทบกับแพ็กเก็ตใดๆเลย

2.8.2 งานวิจัยที่มุ่งเน้นการแก้ไขปัญหาวิกลภาพของกฎไฟร์วอลล์

Al-Shaer และคณะ [1] ได้นิยามรูปแบบวิกลภาพของกฎไฟร์วอลล์ไว้ 4 รูปแบบและได้นำเสนอวิธีการตรวจสอบวิกลภาพที่เกิดขึ้นกับบนระบบไฟร์วอลล์โดยใช้แนวคิดแผนภาพสถานะความสัมพันธ์เสมือน (Similar State Diagram) พบว่าวิธีการนี้สามารถตรวจสอบวิกลภาพของกฎไฟร์วอลล์ได้เพียงกฎต่อกฎเท่านั้น ดังนั้นหากต้องการวิเคราะห์วิกลภาพทั้งระบบไฟร์วอลล์ในกรณีที่กฎมีจำนวนมากจะทำให้ยากต่อการสรุปวิธีการแก้ไขปัญหาวิกลภาพที่เกิดขึ้น

Chomsiri [2] ได้นำเสนอวิธีการตรวจสอบวิกลภาพของกฎไฟร์วอลล์โดยใช้หลักการพีชคณิตเชิงสัมพันธ์ (Relational Algebra) พบว่าวิธีการนี้สามารถตรวจพบวิกลภาพของกฎไฟร์วอลล์ที่เกิดขึ้นบนระบบไฟร์วอลล์ได้ทันที พร้อมทั้งนำเสนอวิธีการกำจัดกฎที่เกิดวิกลภาพเพื่อเพิ่มประสิทธิภาพของกฎไฟร์วอลล์ เช่น การลบกฎที่ถูกบังและกระทำซ้ำซ้อนออก เป็นต้น การกระทำดังกล่าวจะทำให้ไฟร์วอลล์มีประสิทธิภาพเพิ่มขึ้นจริงแต่ต้นเหตุของความขัดแย้งหรือกฎที่ถูกขจัดอาจเป็นกฎที่ต้องการจะเพิ่มเข้าไปเพื่อแก้ไขกฎเก่าก็เป็นได้ และการนำเสนอแนวคิดนี้อยู่ภายใต้การตรวจสอบแบบตามลำดับซึ่งมีประสิทธิภาพเท่ากับ $O(d \times n)$ โดยที่ d คือจำนวนเงื่อนไขของกฎไฟร์วอลล์ และ n คือจำนวนกฎทั้งหมดบนระบบไฟร์วอลล์

Liu [3,14-15] ได้คิดค้นวิธีการตรวจสอบวิกลภาพของกฎไฟร์วอลล์ โดยประยุกต์โครงสร้างของไฟร์วอลล์แบบต้นไม้มาใช้กับระบบไฟร์วอลล์ซึ่งในงานวิจัยเรียกโครงสร้างนี้ว่าแผนภาพการตัดสินใจ (Firewall Decision Diagram: *FDD*) พบว่าแนวคิดนี้สามารถตรวจสอบกฎวิกลภาพได้อย่างครบถ้วน และประสิทธิภาพการผ่านกฎของไฟร์วอลล์เท่ากับ $O(k \times d \log(n))$ โดยที่ d คือจำนวนเงื่อนไขของกฎไฟร์วอลล์ n คือจำนวนกฎของไฟร์วอลล์ซึ่งแทนค่าด้วย $2n-1$ ในกรณีที่กฎมีการแบ่งกิ่งก้านมากที่สุด และ k คือค่าคงที่ในการตรวจสอบขอบเขตของช่วงข้อมูลซึ่งมีค่าน้อยมาก

Chomsiri และคณะ [16] ได้แก้ไขโครงสร้างของไฟร์วอลล์โอเพนซอร์ส IPtables จากโครงสร้างแบบเมตริกเป็นโครงสร้างแบบต้นไม้และสามารถจัดการลำดับการตรวจสอบเงื่อนไขของฟิลด์ต่างๆได้โดยอิสระตามรูปแบบของแพ็กเก็ตการให้บริการขององค์กร เช่น กรณีที่ทราบว่า 95% ของแพ็กเก็ตที่เข้าสู่ระบบเครือข่ายใช้งานโพรโทรคอล HTTP (พอร์ต 80) ดังนั้นจะทำการตรวจสอบเงื่อนไขหมายเลขพอร์ตปลายทาง (*DPT)* ก่อนเพื่อกรองแพ็กเก็ตที่สนใจส่งผลให้ประสิทธิภาพในการทำงานเร็วขึ้น แต่งานวิจัยนี้มีการเปรียบเทียบประสิทธิภาพกับ Iptables แบบดั้งเดิมซึ่งเป็นที่ทราบกันดีอยู่แล้วว่าโครงสร้างของไฟร์วอลล์แบบต้นไม้ทำงานได้ดีกว่าแบบเมตริก ซึ่งหากจะให้ดีกว่านี้อาจจะต้องนำไปเปรียบเทียบกับไฟร์วอลล์ IPset ที่ใช้โครงสร้างของไฟร์วอลล์แบบตารางแฮช

Booth และคณะ [17] นำเสนอวิธีการเพิ่มกฎของไฟร์วอลล์โดยใช้หลักการเปลี่ยนแปลงความเชื่อ (Belief Revision) พบว่าเมื่อกฎที่ถูกเพิ่มเข้าไปเกิดความขัดแย้งกับกฎเดิมในระบบไฟร์วอลล์จะดำเนินการโดยแก้ไขกฎเดิมในระบบและแทนด้วยกฎใหม่ทันที ซึ่งจะทำให้กฎในระบบไฟร์วอลล์ปราศจากกฎวิกลภาพ แต่ปัญหาของงานวิจัยจะเกิดขึ้นเมื่อกฎที่ถูกเพิ่มเข้าไปเป็นกฎที่เกิดจากความผิดพลาดของผู้ดูแลระบบ ก็จะส่งผลให้ระบบมีช่องโหว่ที่เกิดขึ้นจากกระกระทำที่รู้เท่าไม่ถึงการณ์ ซึ่งหลักการแนวคิดวิธีนี้เป็นวิธีที่ช่วยขจัดปัญหาวิกลภาพของกฎไฟร์วอลล์ได้ในระดับหนึ่งเท่านั้น

Khummanee และคณะ [18] ได้ออกแบบโครงสร้างของไฟร์วอลล์และการตัดสินใจแบบใหม่โดยมีชื่อว่า ไฟร์วอลล์การตัดสินใจแบบโดเมนเดี่ยว (Single Decision Domain, *SDD*) พบว่า

งานวิจัยนี้ได้มุ่งเน้นในการแก้ไขปัญหาวิกลภาพของกฎไฟร์วอลล์ โดยกำหนดให้ระบบไฟร์วอลล์ใดๆจะมีการตัดสินใจได้เพียงหนึ่งเดียวเท่านั้นคือ ยอมรับเท่านั้น (Close Firewall System, *CFS*) หรือ ปฏิเสธเท่านั้น (Open Firewall System, *OFS*) เท่านั้น ซึ่งแน่นอนว่า ด้วยหลักการนี้จะทำให้ไฟร์วอลล์ปราศจากกฎวิกลภาพ

2.8.3 งานวิจัยที่มุ่งเน้นในการเพิ่มประสิทธิภาพการผ่านกฎไฟร์วอลล์

El-Alfy [19] ได้ปรับปรุงประสิทธิภาพการผ่านกฎของไฟร์วอลล์โดยใช้หลักการคำนวนกลุ่มความซับซ้อนของปัญหาเอ็นพีแบบยาก (NP-Hard) ที่ใช้ประยุกต์ใช้การค้นหาแบบฮิวริสติก (Heuristic Search) บนพื้นฐานขั้นตอนวิธีเชิงพันธุกรรม (Genetic Algorithm) พบว่าวิธีที่จัดลำดับความสำคัญของกฎไฟร์วอลล์ให้สอดคล้องกับแพ็กเก็ตที่เข้าออกระบบเพื่อลดขั้นตอนการตรวจสอบและในการจัดลำดับนี้ได้ให้ความสำคัญกับความสัมพันธ์ของกฎไฟร์วอลล์เพื่อไม่ให้กระทบกับนโยบายการรักษาความปลอดภัยของไฟร์วอลล์

ศุภพร รัฐอาจและคณะ [20] ได้ปรับปรุงประสิทธิภาพการผ่านกฎของไฟร์วอลล์โอเพนซอร์ส Iptables [8], Ipset [9] ซึ่งเป็นซอฟแวร์ที่ในการตรวจสอบแบบตารางแฮช โดยการใช้งานมีขอบเขตการสร้างเซตของแฮชอยู่ที่ 2^{16} บิตเท่านั้น เนื่องจากจะช่วยลดปัญหาการจัดการหน่วยความจำ ซึ่งพบว่างานวิจัยนี้ได้ออกแบบและจัดการกฎให้เหมาะสมภายใต้ขอบเขตซึ่งทำให้ประสิทธิภาพการทำงานของไฟร์วอลล์เพิ่มขึ้นอย่างเห็นได้ชัด แต่การจัดการกฎด้วย Ipset จะต้องคำถึงหน่วยความจำเป็นสำคัญ

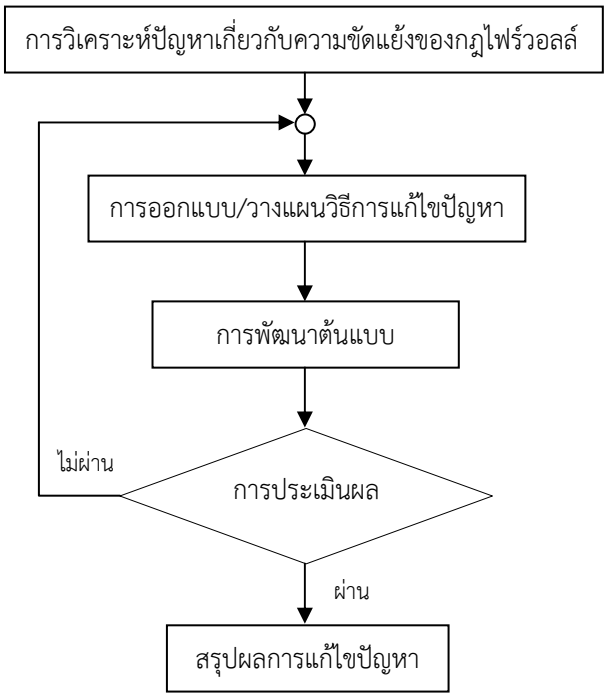
จากงานวิจัยที่กล่าวมาจะเห็นได้ว่ามีงานวิจัยคิดค้นวิธีการในการตรวจสอบวิกลภาพของกฎไฟร์วอลล์หลากหลายวิธี เพื่อมุ่งเน้นให้ไฟร์วอลล์มีการทำงานที่สอดคล้องกับนโยบายความปลอดภัยของระบบเครือข่าย แม้ว่าจะได้ผลที่ดี แต่ต้นเหตุของปัญหายังคงไม่ได้ถูกแก้ไขอย่างสมบูรณ์ เนื่องจากสาเหตุของปัญหาที่แท้จริงคือ การที่สมาชิกของกฎใดๆมากกว่าสองกฎ ที่เป็นสมาชิกซึ่งกันและกัน และมีผลของการกระทำที่แตกต่างกันในเวลาเดียวกัน

บทที่ 3

วิธีดำเนินการวิจัย

การดำเนินการวิจัยของวิทยานิพนธ์ในบทนี้ เป็นการนำเสนอหลักการวิเคราะห์และออกแบบแนวคิดในการพัฒนาไฟร์วอลล์การตัดสินใจแบบโดเมนเดี่ยว ได้แก่ ออกแบบ ขั้นตอนการดำเนินการ เครื่องมือที่ใช้ในการทดลอง การกำหนดองค์ประกอบของการทดลอง รวมถึงการทดลองในมุมมองต่างๆ เพื่อให้ได้ผลการทดลองที่มีประสิทธิภาพในทุกๆด้าน โดยมีขั้นตอนการดำเนินการดังนี้

3.1 แผนการดำเนินงาน



รูปที่ 3.1 แผนภาพการดำเนินงานวิจัย

จากรูปที่ 3.1 เป็นแผนภาพการดำเนินงานวิจัยที่อธิบายการดำเนินงานทั้งหมดขั้นตอนแรกจะวิเคราะห์ถึงปัญหาที่เกิดขึ้นเพื่อกำหนดทิศทางในการแก้ไขปัญหา จุดประสงค์ของการทำงานและขอบเขตอย่างชัดเจนเมื่อได้ปัญหาและทิศทางของงานวิจัย ขั้นต่อไปจะเป็นการออกแบบและวางแผนวิธีการแก้ไขปัญหา โดยรวบรวมข้อมูลต่างๆที่เกี่ยวข้องจากการค้นคว้างานวิจัยแล้วดำเนินงานตามแผนที่วางเอาไว้ตามลำดับ เมื่อจบขั้นตอนการดำเนินการ จะต้องประเมินผลของการดำเนินงานเพื่อทดสอบ

ประสิทธิภาพและเมื่อผลของการประเมินผลมีประสิทธิภาพตามที่ได้ออกแบบไว้ แล้วข้ามไปสู่ขั้นตอนการสรุปผลการแก้ไขปัญหา แต่ถ้าประสิทธิภาพการทำงานไม่เป็นไปตามที่คาดการจะต้องกลับไปทำงานออกแบบและวางแผนใหม่ เพื่อแก้ไขให้ดีขึ้นจนกว่าผลการประเมินจะผ่าน โดยในวิทยานิพนธ์นี้มีรายละเอียดในการทำงานแต่ละขั้นตอนดังต่อไปนี้

3.2 วิเคราะห์ปัญหาของไฟร์วอลล์

วิทยานิพนธ์นี้ได้เล็งเห็นความสำคัญของการจัดการกฎของไฟร์วอลล์ เนื่องจากไฟร์วอลล์เป็นอุปกรณ์หลักสำคัญที่ใช้ในการจำกัดการเข้าถึงข้อมูลและการให้บริการของระบบเครือข่าย สิ่งสำคัญที่สุดของการทำงานไฟร์วอลล์มีผลมาจากการจัดการกฎ ซึ่งเมื่อมีการออกกฎและจัดการกฎไม่ดีแล้ว จะส่งผลต่อประสิทธิภาพการทำงานลดลงและยังส่งผลต่อมาตราการรักษาความปลอดภัยของเครือข่ายอีกด้วย

จากการสำรวจงานวิจัยที่เกี่ยวข้องกับไฟร์วอลล์ พบว่างานวิจัยส่วนใหญ่มุ่งประเด็นไปที่การเพิ่มประสิทธิภาพในการผ่านกฎและการขจัดวิกลภาพของกฎไฟร์วอลล์ ซึ่งในบทนี้ได้วิเคราะห์ปัญหาของไฟร์วอลล์เป็น 2 ส่วนด้วยกันดังนี้

3.2.1 วิเคราะห์ปัญหาประสิทธิภาพการผ่านกฎของไฟร์วอลล์

การทำงานของไฟร์วอลล์แบบดั้งเดิมมีการให้ความสำคัญกับความสัมพันธ์ของลำดับกฎ เมื่อเพิ่มกฎใหม่เข้าไปในระบบไฟร์วอลล์ กฎจะอยู่ลำดับท้ายสุด ซึ่งการผ่านกฎของไฟร์วอลล์จะมีการตรวจสอบแบบตามลำดับ แน่นอนว่าประสิทธิภาพจะลดลงเมื่อกฎมีจำนวนมากขึ้น สิ่งที่ควรคำนึงถึงอีกข้อหนึ่งคือการออกแบบและสร้างกฎให้สอดคล้องกับนโยบายการรักษาความปลอดภัยใดๆนั้นสามารถเขียนได้หลายรูปแบบ ทั้งนี้ที่จะเขียนกฎให้มีประสิทธิภาพได้นั้นอาจขึ้นอยู่กับประสบการณ์และความเชี่ยวชาญของผู้ดูแลระบบอีกด้วย

ไฟร์วอลล์ 1
Fwr_1: $[30,50] \wedge [21,40] \wedge [20,22] \rightarrow Deny$
Fwr_2: $[20,60] \wedge [10,25] \wedge [20,22] \rightarrow Accept$
Fwr_3: $[70,70] \wedge [20,40] \wedge [20,30] \rightarrow Deny$

ไฟร์วอลล์ 2
Fwr_1: $[20,60] \wedge [10,20] \wedge [20,22] \rightarrow Accept$
Fwr_2: $[20,21] \wedge [10,25] \wedge [20,22] \rightarrow Accept$
Fwr_3: $[50,60] \wedge [10,25] \wedge [20,22] \rightarrow Accept$
Fwr_4: $[70,70] \wedge [20,40] \wedge [20,30] \rightarrow Deny$

รูปที่ 3.2 เปรียบเทียบรูปแบบการสร้างกฎของไฟร์วอลล์

จากรูปที่ 3.2 จะเห็นว่าการสร้างกฎไฟร์วอลล์ 1 นั้นมีจำนวนกฎที่น้อยกว่ากฎไฟร์วอลล์ 2 แต่มีมาตรการรักษาความปลอดภัยเท่ากัน ซึ่งไฟร์วอลล์ 1 จะตรวจผ่านกฎได้เร็วกว่าไฟร์วอล์ 2

เนื่องจากจำนวนกฎมีผลโดยตรงต่อการเข้าถึงข้อมูล การสร้างกฎบนระบบไฟร์วอลล์แบบดั้งเดิมนั้นไม่มีรูปแบบการสร้างกฎที่เป็นกลางขึ้นอยู่กับความรู้และความเข้าใจของผู้ดูแลระบบ เมื่อกฎมีการถ่ายทอดจากผู้ดูแลระบบต่อไป การสร้างกฎที่เกิดจากความเชี่ยวชาญที่แตกต่างกันอาจทำให้ยากต่อการเข้าใจเพิ่มขึ้นอีกด้วย

3.2.2 วิเคราะห์วิกลภาพของกฎไฟร์วอลล์

ปัญหาวิกลภาพของกฎไฟร์วอลล์สืบเนื่องมาจากสาเหตุเดียวกันคือ กฎของไฟร์วอลล์ให้ความสำคัญกับลำดับ ซึ่งขอบเขตข้อมูลของกฎไฟร์วอลล์มีจำนวนมากทำให้กฎเกิดความสัมพันธ์ที่ซับซ้อนต่างๆ ขึ้นมากมายส่งผลให้ผู้ดูแลระบบจัดการกฎด้วยความผิดพลาด มีงานวิจัยที่ได้นำเสนอวิธีการแก้ไขปัญหาวิกลภาพของกฎแม้ว่าจะได้ผลที่ดี แต่ต้นเหตุของปัญหายังไม่ได้ถูกแก้ไขอย่างจริงจัง ซึ่งต้นเหตุที่สำคัญคือการที่กฎตั้งแต่สองกฎขึ้นไปให้ความหมายที่ขัดแย้งกันหรือให้ความหมายที่ซ้ำซ้อนคาบเกี่ยวกัน ซึ่งโดยที่กฎในระบบไฟร์วอลล์มีความสัมพันธ์ในลักษณะต่างๆสามารถนิยามได้ดังนี้

<u>นิยาม 1</u> กำหนดให้ Fwr_x และ Fwr_y เป็นกฎของไฟร์วอลล์ที่ **ไม่เกี่ยวพันธ์โดยสมบูรณ์** เมื่อสมาชิกทุกฟิลด์เงื่อนไขของ Fwr_x ไม่เป็นสมาชิกของ Fwr_y ตามฟิลด์เงื่อนไขที่สอดคล้อง

$$\forall i : Fwr_x[i] \not\bowtie Fwr_y[i]$$

กำหนดให้ $\bowtie \in \{\subset, \supset, =\}$, $i \in$ {*SIP, SPT, DIP, DPT, PRO*}

<u>นิยาม 2</u> กำหนดให้ Fwr_x และ Fwr_y เป็นกฎของไฟร์วอลล์ที่ **เท่ากันอย่างแน่ชัด** เมื่อสมาชิกทุกฟิลด์เงื่อนไขของ Fwr_x เท่ากับสมาชิกของ Fwr_y ตามฟิลด์เงื่อนไขที่สอดคล้อง

$$\forall i : Fwr_x[i] = Fwr_y[i]$$

กำหนดให้ $i \in$ {*SIP, SPT, DIP, DPT, PRO*}

<u>นิยาม 3</u> กำหนดให้ Fwr_x และ Fwr_y เป็นกฎของไฟร์วอลล์ที่ **ถูกครอบคลุม** เมื่อ Fwr_x และ Fwr_y ไม่เป็นกฎที่เท่ากันอย่างแน่ชัด และทุกฟิลด์เงื่อนไขของ Fwr_x เป็นสมาชิกหรือเท่ากับ Fwr_y ตามฟิลด์เงื่อนไขที่สอดคล้อง ซึ่งจะเรียก Fwr_x ว่า **กฎย่อย** (Subset rule) และเรียก Fwr_y ว่า **กฎหลัก** (Superset rule)

$$\forall i : Fwr_x[i] \subseteq Fwr_y[i] \wedge \exists j : Fwr_x[i] \neq Fwr_y[i]$$

กำหนดให้ $i, j \in$ {*SIP, SPT, DIP, DPT, PRO*}

<u>นิยาม 4</u> กำหนดให้ Fwr_x และ Fwr_y เป็นกฎของไฟร์วอลล์ที่ **เกี่ยวพันธ์** เมื่อฟิลด์เงื่อนไขของ Fwr_x มีเป็นสมาชิก Fwr_y ตามฟิลด์เงื่อนไขที่สอดคล้องอย่างน้อยหนึ่งเงื่อนไข และฟิลด์เงื่อนไข Fwr_y เป็นสมาชิกของ Fwr_x ในฟิดล์เงื่อนไขที่เหลือ

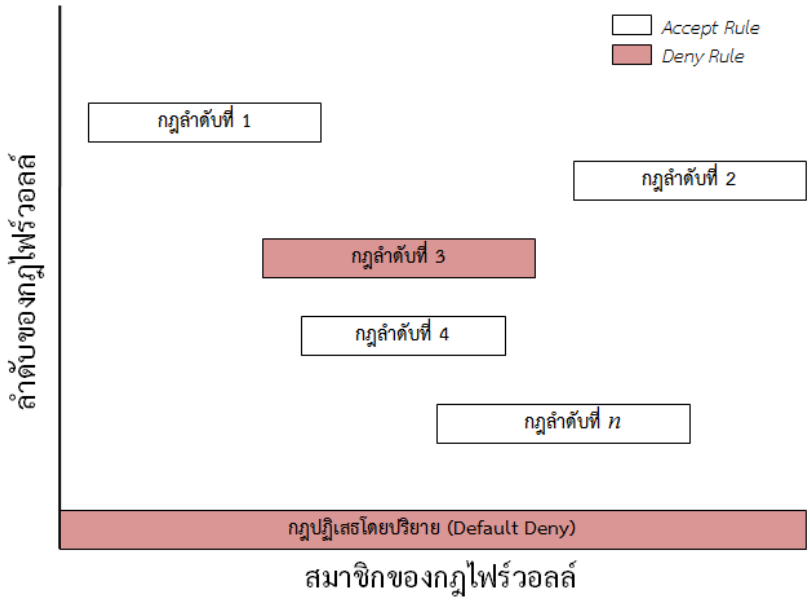
$$\forall i : Fwr_x[i] \bowtie Fwr_y[i] \wedge \exists i, j : Fwr_x[i] \subset Fwr_y[i] \wedge Fwr_x[i] \subset Fwr_y[i] \wedge i \neq j$$

กำหนดให้ $\bowtie \in \{\subset, \supset, =\}$, $i, j \in$ {*SIP, SPT, DIP, DPT, PRO*}

นิยาม 5 กำหนดให้ Fwr_x และ Fwr_y ที่มีความสัมพันธ์ของกฎไฟร์วอลล์แบบไม่เกี่ยวพันธ์โดยสมบูรณ์, เท่ากันอย่างแน่ชัด, ถูกครอบคลุม และเกี่ยวพันธ์ อย่างน้อยหนึ่งกรณีและกำหนดให้ผลการกระทำของกฎไฟร์วอลล์แตกต่างกัน จะได้ว่า Fwr_x และ Fwr_y เป็น **กฎไฟร์วอลล์ที่เกิดความขัดแย้ง**

นิยาม 6 กำหนดให้ Fwr_x และ Fwr_y ที่มีความสัมพันธ์ของกฎไฟร์วอลล์แบบถูกครอบคลุม และผลการกระทำของกฎไฟร์วอลล์ทั้งสองเหมือนกัน จะได้ว่า Fwr_x และ Fwr_y เป็น **กฎไฟร์วอลล์ที่กระทำซ้ำซ้อน**

จากนิยามความสัมพันธ์กฎไฟร์วอลล์ข้างต้น พบว่าต้นเหตุความความขัดแย้งของกฎไฟร์วอลล์เกิดจากการที่กฎมีผลการกระทำที่แตกต่างกัน ดังรูปที่ 3.3 ในระบบไฟร์วอลล์แบบดั้งเดิม (Traditional Firewall) มีการตรวจผ่านกฎแบบตามลำดับหากพบว่า แพ็กเก็ตที่ถูกตรวจผ่านเข้ากับกฎของไฟร์วอลล์ไม่กระทบกับกฎใดๆ เลยจะกระทบกับกฎที่ถูกปฏิเสธโดยอัตโนมัติ (Default Deny) แต่เมื่อผู้ดูแลระบบได้เพิ่มกฎที่มีผลการกระทำเท่ากับปฏิเสธอยู่ระหว่างรายการของกฎไฟร์วอลล์จะส่งผลให้กฎไฟร์วอลล์มีโอกาสขัดแย้งกับกฎอื่นๆ ดังรูปที่ 3.3



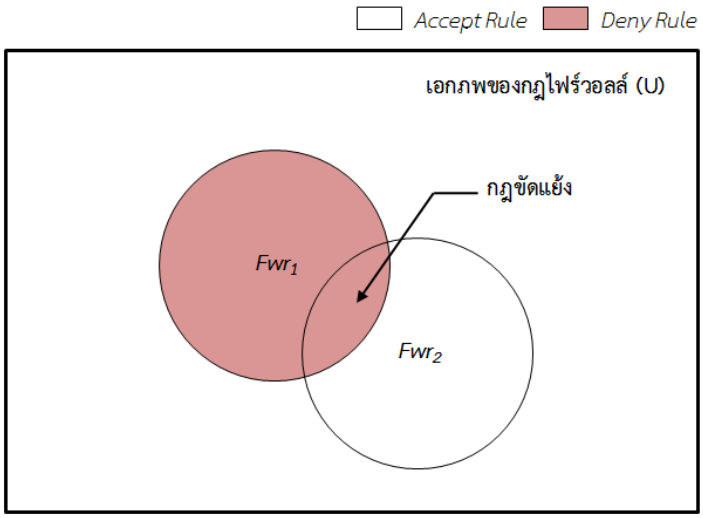
รูปที่ 3.3 กฎปฏิเสธโดยปริยายและกฎที่มีผลการกระทำปฏิเสธ

กฎของไฟร์วอลล์จะมีกฎปฏิเสธโดยปริยาย เพื่อป้องกันแพ็กเก็ตที่ไม่ได้รับอนุญาติเข้าสู่ระบบเครือข่าย ดังนั้นกฎที่จะแสดงผลบนระบบไฟร์วอลล์ควรจะเป็นกฎที่มีผลของการกระทำเป็นอนุญาติเท่านั้น แต่การเพิ่มกฎที่มีผลของการกระทำปฏิเสธ บนระบบไฟร์วอลล์ มีจุดประสงค์หลักคือเพื่อต้องการยืนยันว่า แพ็กเก็ตที่อยู่ในเซตของกฎปฏิเสธแน่ชัดจะไม่สามารถผ่านเข้ามาได้ แม้ว่าจะมีการ

เพิ่มกฎที่มีผลการกระทำยอมรับในภายหลัง เช่น กรณีที่ผู้ดูแลระบบต้องการปิดกั้นการแพร่กระจายไวรัสคอมพิวเตอร์จากระบบเครือข่ายภายนอกเป็นต้น

แต่ข้อเสียของไฟร์วอลล์ที่มีผลการตัดสินใจที่แตกต่างกัน จะทำให้กฎมีความขัดแย้งและเมื่อกฎมีความขัดแย้งทับซ้อนกันเป็นจำนวนมากแล้วผู้ดูแลระบบต้องการจะแก้ไขผลการกระทำของสมาชิกเซตใดๆด้วยเหตุผลที่ชัดเจนซึ่งจะทำให้การแก้ไขนั้นเป็นไปได้ยาก

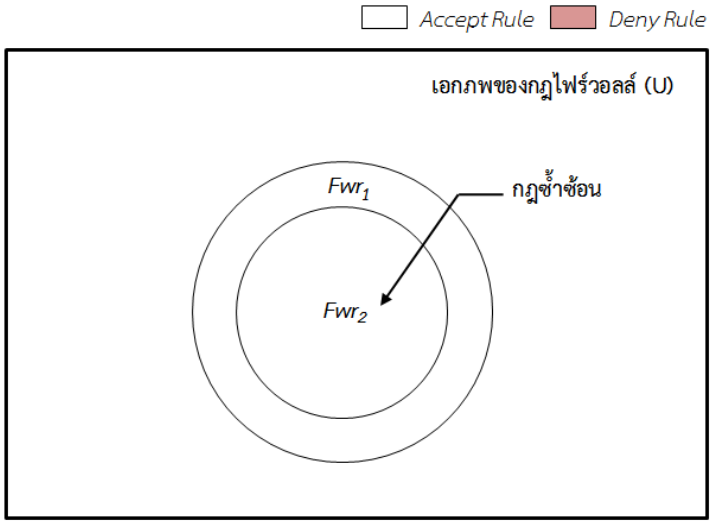
สมมุติให้ Fwr_1 และ Fwr_2 เป็นกฎของไฟร์วอลล์มีความสัมพันธ์แบบเกี่ยวพันธ์ และกฎทั้งสองมีผลการกระทำที่แตกต่างกันจะสามารถเขียนเป็นแผนภาพของเวนน์ (Venn diagram) แสดงได้ดังนี้



รูปที่ 3.4 แผนภาพของเวนน์แสดงกฎที่ขัดแย้ง

จากรูปที่ 3.4 จะได้ว่า $Fwr_1 \cap Fwr_2$ เป็นส่วนที่ขัดแย้งหรือขัดแย้งกันในการตัดสินใจของไฟร์วอลล์ เนื่องจากหากผู้ดูแลระบบต้องการที่จะแก้ไขกฎในส่วนของ $Fwr_1 \cap Fwr_2$ ให้มีผลของการกระทำคือ ยอมรับ (Accept) จาก Fwr_2 ซึ่งก่อนหน้านี้คือ ปฏิเสธ (Deny) จาก Fwr_1 จะไม่สามารถทำได้เนื่องจากกฎของไฟร์วอลล์ให้สำคัญกับลำดับของกฎไฟร์วอลล์ ดังนั้นหากต้องการจะแก้ไขกฎผู้ดูแลระบบจะต้องลบกฎ Fwr_1 ก่อนแล้วเพิ่มกฎ Fwr_2 ก่อนจึงจะสามารถแก้ไขได้ ในกรณีที่กฎบนระบบไฟร์วอลล์มีจำนวนหลายพันกฎ ปัญหานี้จึงเป็นปัญหาที่ยากจะแก้ไข นอกจากนี้ยังมีปัญหาที่กฎเกิดจากกฎกระทำซ้ำซ้อนซึ่งจะทำให้กฎมากเกินความจำเป็นดังตัวอย่าง

สมมุติให้ Fwr_1 และ Fwr_2 เป็นกฎของไฟร์วอลล์ที่กระทำซ้ำซ้อนสามารถเขียนเป็นแผนภาพของเวนน์ (Venn diagram) แสดงได้ดังนี้



รูปที่ 3.5 แผนภาพของเวนน์แสดงกฎที่กระทำซ้ำซ้อน

จากรูปที่ 3.5 จะได้ว่า $Fwr_2 \subset Fwr_1$ เมื่อมีการผ่านกฎไฟร์วอลล์ Fwr_2 จะไม่มีทางกระทบกับแพ็กเก็ตใดๆเลยเนื่องจาก Fwr_1 กระทำก่อนหน้าแล้ว ในกรณีที่ไฟร์วอลล์มีกฎกระทำซ้ำซ้อนมากๆ จะส่งผลให้ไฟร์วอลล์ตรวจผ่านกฎช้าลงโดยไม่จำเป็น ดังนั้นกฎที่กระทำซ้ำซ้อนควรจะถูกยุบให้เหลือเพียงกฎเดียว

3.3 แนวคิดในการแก้ไขปัญหา

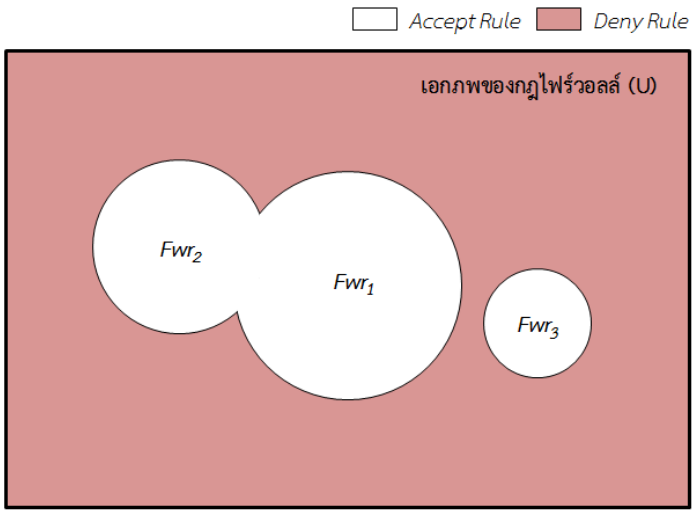
ไฟร์วอลล์ในปัจจุบันยอมให้สร้างกฎได้อย่างเป็นอิสระโดยการที่กฎใดๆสามารถกำหนดให้มีผลของการกระทำที่ยอมรับและปฏิเสธบนระบบไฟร์วอลล์ และเมื่อเกิดปัญหาก็จะตามมาแก้ไขในภายหลัง เช่น การวิเคราะห์หากฎวิกลภาพแล้วนำเสนอวิธีการต่างๆเพื่อนำมาอธิบายว่าการตัดสินใจที่เกิดความขัดแย้งกันระหว่างยอมรับและปฏิเสธว่าควรจะผลการกระทำใดสมเหตุสมผลมากกว่ากัน พบว่าหากต้องการจะแก้ปัญหาให้ตรงประเด็นที่สุดระบบไฟร์วอลล์จะต้องมีผลของการตัดสินใจเพียงอย่างเดียวเท่านั้น ซึ่งในวิทยานิพนธ์นี้ได้นำเสนอแนวคิดที่เรียกว่า การตัดสินใจแบบโดเมนเดี่ยว (Single Domain Decision) ซึ่งวิธีดังกล่าวจะช่วยไม่ให้เกิดกฎวิกลภาพตั้งแต่เริ่มต้นสร้างกฎโดยมีรายละเอียดดังต่อไปนี้

พื้นฐานของระบบไฟร์วอลล์มี 2 ประเภทคือ ระบบไฟร์วอลล์แบบปิด (Close Firewall System: *CFS*) และระบบไฟร์วอลล์แบบเปิด (Open Firewall System: *OFS*)

3.3.1 ระบบไฟร์วอลล์แบบปิด

ระบบไฟร์วอลล์แบบปิดจะเป็นระบบที่จำกัดการไหลผ่านของแพ็กเก็ตทุกประเภทเมื่อเริ่มต้นระบบหรือกรณีที่ไม่มีกฎบนระบบไฟร์วอลล์ และหากต้องการกำหนดให้แพ็กเก็ตไหลผ่านได้จึงเปิดใช้งานโดยเพิ่มกฎเข้าไปในระบบ ซึ่งกฎที่อยู่บนระบบไฟร์วอลล์เมื่อตรวจสอบทุกเงื่อนไขแล้วจะมี

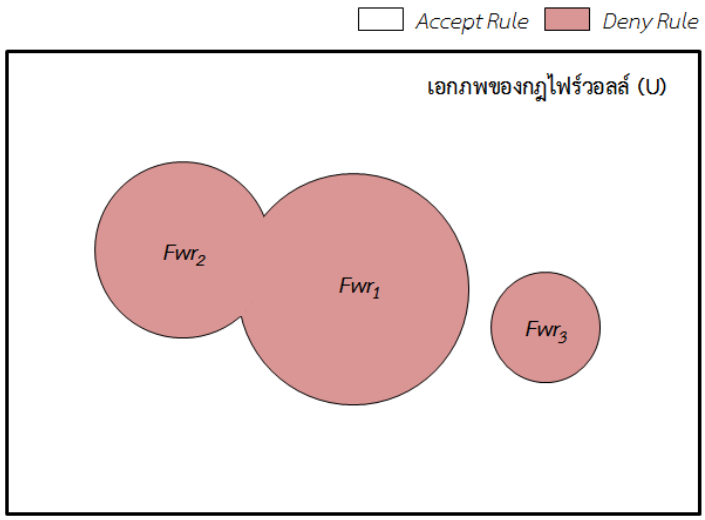
ผลของการกระทำเป็น ยอมรับเท่านั้น (Accept All) จะไม่มีกฎที่มีผลของการกระทำเป็นปฏิเสธแม้แต่กฎเดียวดังรูปที่ 3.6



รูปที่ 3.6 แผนภาพของเวนน์แสดงโครงสร้างระบบไฟร์วอลล์แบบปิด

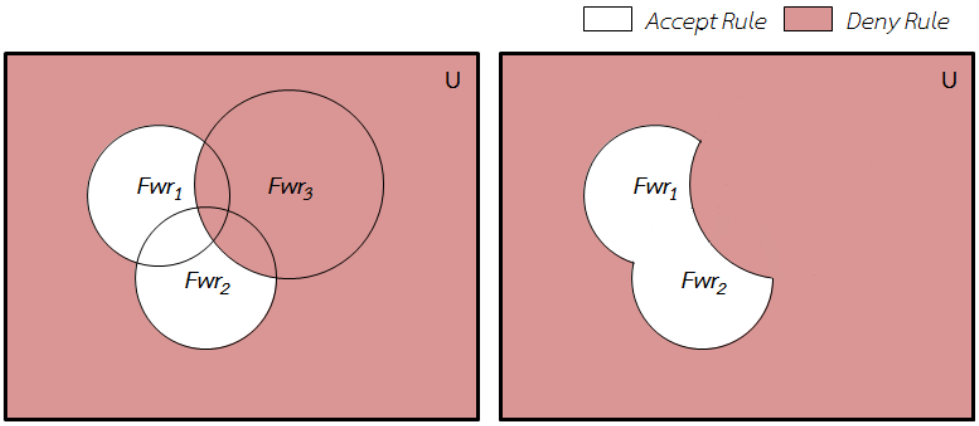
3.3.2 ระบบไฟร์วอลล์แบบเปิด

ระบบไฟร์วอลล์แบบเปิดจะเป็นระบบที่ยอมให้แพ็กเก็ตทุกประเภทไหลผ่านเข้าระบบเครือข่ายเมื่อเริ่มต้นระบบหรือกรณีที่กฎบนระบบไฟร์วอลล์ และหากต้องการปิดกั้นการไหลผ่านของแพ็กเก็ตจึงกำหนดกฎเข้าไปในระบบ ซึ่งกฎที่อยู่บนระบบไฟร์วอลล์เมื่อตรวจสอบทุกเงื่อนไขแล้วจะมีผลการกระทำเป็น ปฏิเสธเท่านั้น (Deny All) จะไม่มีกฎที่มีผลของการกระทำเป็นยอมรับแม้แต่กฎเดียว ดังรูปที่ 3.7



รูปที่ 3.7 แผนภาพของเวนน์แสดงโครงสร้างระบบไฟร์วอลล์แบบเปิด

การสร้างกฎของไฟร์วอลล์ด้วยแนวคิดการตัดสินใจแบบโดเมนเดี่ยวจะไม่เกิดกฎวิกลภาพระหว่างกฎของไฟร์วอลล์ และสามารถยุบรวมกฎที่ต่อเนื่องกันได้ทันที เนื่องจากมีส่วนที่ตัดสินใจเป็นชนิดเดียวกันทั้งหมด นอกจากนี้ลำดับความสัมพันธ์ของกฎจะไม่มีผลต่อนโยบายการรักษาความปลอดภัยซึ่งจะช่วยลดความซับซ้อนของกฎไฟร์วอลล์ได้อีกด้วย



รูปที่ 3.8 แผนภาพของเวนน์แสดงการขจัดวิกลภาพของกฎไฟร์วอลล์

รูปที่ 3.8 กำหนดให้ $Fwr_1 \cup Fwr_2$ เป็นกฎของไฟร์วอลล์ที่มีผลของการกระทำเท่ากับยอมรับ กำหนดให้ Fwr_3 เป็นกฎของไฟร์วอลล์ที่มีผลของการกระทำเท่ากับปฏิเสธ และทั้งสามกฎมีสมาชิกบางตัวทับซ้อนกันบนระบบไฟร์วอลล์แบบปิด สมาชิกของกฎไฟร์วอลล์ที่ถูกเพิ่มเข้าไปล่าสุดจะถูกตัดทิ้งไปพร้อมกับสมาชิกที่ทับซ้อนกันด้วย จะได้ว่าสมาชิกใน $Fwr_3 \cap (Fwr_1 \cup Fwr_2)$ จะไม่พบบนระบบไฟร์วอลล์

จะสรุปได้ว่า แนวคิดการตัดสินใจแบบโดเมนเดี่ยว (Single Domain Decision) เมื่อพบกฎไฟร์วอลล์ที่มีกฎวิกลภาพ ผลการกระทำของกฎไฟร์วอลล์เดิมที่มีสมาชิกทับซ้อนหรือขัดแย้งกันจะถูกแทนที่ด้วยผลการกระทำของกฎไฟร์วอลล์ใหม่ทันที ซึ่งเชื่อว่ากฎไฟร์วอลล์ที่เพิ่มบนระบบไฟร์วอลล์จะเป็นกฎที่ผ่านการปรับปรุงองค์ความรู้ใหม่ [17] ดังนั้นเมื่อกฎของไฟร์วอลล์เดิมถูกแทนที่ด้วยผลการกระทำของกฎไฟร์วอลล์ใหม่แล้ว ส่งผลให้ระบบไฟร์วอลล์ปราศจากกฎวิกลภาพและสามารถจัดการกฎได้โดยไม่ต้องคำนึงถึงลำดับการทำงานของกฎอีกด้วย

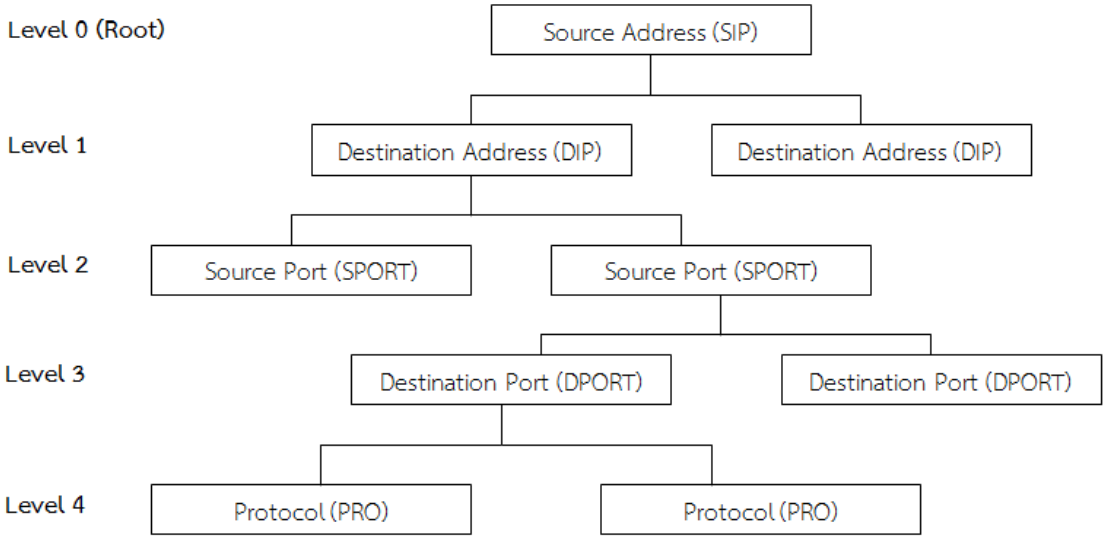
3.4 ออกแบบโครงสร้างและจัดเก็บข้อมูลของไฟร์วอลล์ *SDD*

3.4.1 ออกแบบลำดับชั้นโครงสร้างข้อมูลต้นไม้

ไฟร์วอลล์ที่มีการออกแบบด้วยแนวคิดการตัดสินใจแบบโดเมนเดี่ยว *(SDD)* จะใช้คุณสมบัติของโครงสร้างแบบต้นไม้แบบไม่สมดุลหรืออาจมองในมุมมองของโครงสร้างแบบลิงค์ลีสต์ได้

โดยวิทยานิพนธ์นี้จะขออธิบายในรูปแบบของโครงสร้างต้นไม้ไม่สมดุล เนื่องจากโครงสร้างนี้ไม่มีการกำหนดเงื่อนไขให้ซับซ้อนยุ่งยาก สามารถออกแบบได้อย่างอิสระ

โครงสร้างของไฟร์วอลล์ที่เก็บข้อมูลในรูปแบบต้นไม้นี้จะมีลำดับความลึกของต้นไม้ที่คงที่มีความลึกทั้งหมด 5 ระดับคือ ลำดับที่ 1 คือรากหรือ Root แทนด้วยเซตของหมายเลขไอพีต้นทาง (*SIP*) ลำดับที่ 2 คือหมายเลขไอพีปลายทาง *(DIP)* ลำดับที่ 3 คือหมายเลขพอร์ตต้นทาง (*SPT*) ลำดับที่ 4 คือหมายเลขพอร์ตปลายทาง (*DPT*) และลำดับสุดท้ายคือโพรโทคอล (*PRO*) ดังรูปที่ 3.9



รูปที่ 3.9 การแบ่งลำดับชั้นของโครงสร้าง *SDD*

3.4.2 ออกแบบการจัดเก็บข้อมูล

การจัดเก็บข้อมูลของโครงสร้าง *SDD* ได้มีการแบ่งลำดับชั้นในการตรวจสอบตั้งแต่โหนดรากไปจนถึงระบบของโหนดสุดท้าย การเก็บข้อมูลในแต่ละช่วงของโหนดเงื่อนไขนี้จะแบ่งช่วงของการตรวจสอบเป็น ช่วงขอบเขตบนและขอบเขตล่างของข้อมูล เพื่อลดปริมาณการใช้หน่วยความจำและยังลดเวลาที่ใช้ในการตรวจสอบได้อีกด้วย ซึ่งสามารถเขียนได้ลักษณะดังนี้

1. กำหนดช่วงข้อมูลให้กับหมายเลขไอพีแบบเดี่ยว

ยกตัวอย่าง หมายเลขไอพี 192.168.1.100 ซึ่งประกอบด้วยข้อมูลจำนวน 1 หมายเลขไอพี จะสามารถเขียนเป็นช่วงขอบเขตของข้อมูลได้หนึ่งช่วงข้อมูลประกอบด้วย ข้อมูลขอบเขตบนหรือข้อมูลเริ่มต้นเท่ากับ 192.168.1.100 และข้อมูลขอบเขตล่างหรือข้อมูลสิ้นสุดขเท่ากับ 192.168.1.100 การเก็บข้อมูลหมายเลขไอพีหนึ่งหมายเลขจะใช้หน่วยความจำ 32 บิต ดังนั้นในการเก็บข้อมูลแบบช่วงข้อมูลจะใช้หน่วยความจำเท่ากับ 64 บิต ซึ่งอาจจะใช้ปริมาณข้อมูลมากกว่าการเก็บ

หมายเลขเดียว แต่การออกแบบเช่นนี้จะมีผลดีกับการกำหนดให้กับช่วงของกลุ่มข้อมูลที่ต่อเนื่อง เป็นอย่างมาก

2. กำหนดช่วงข้อมูลให้กับหมายเลขไอพีแบบกลุ่ม

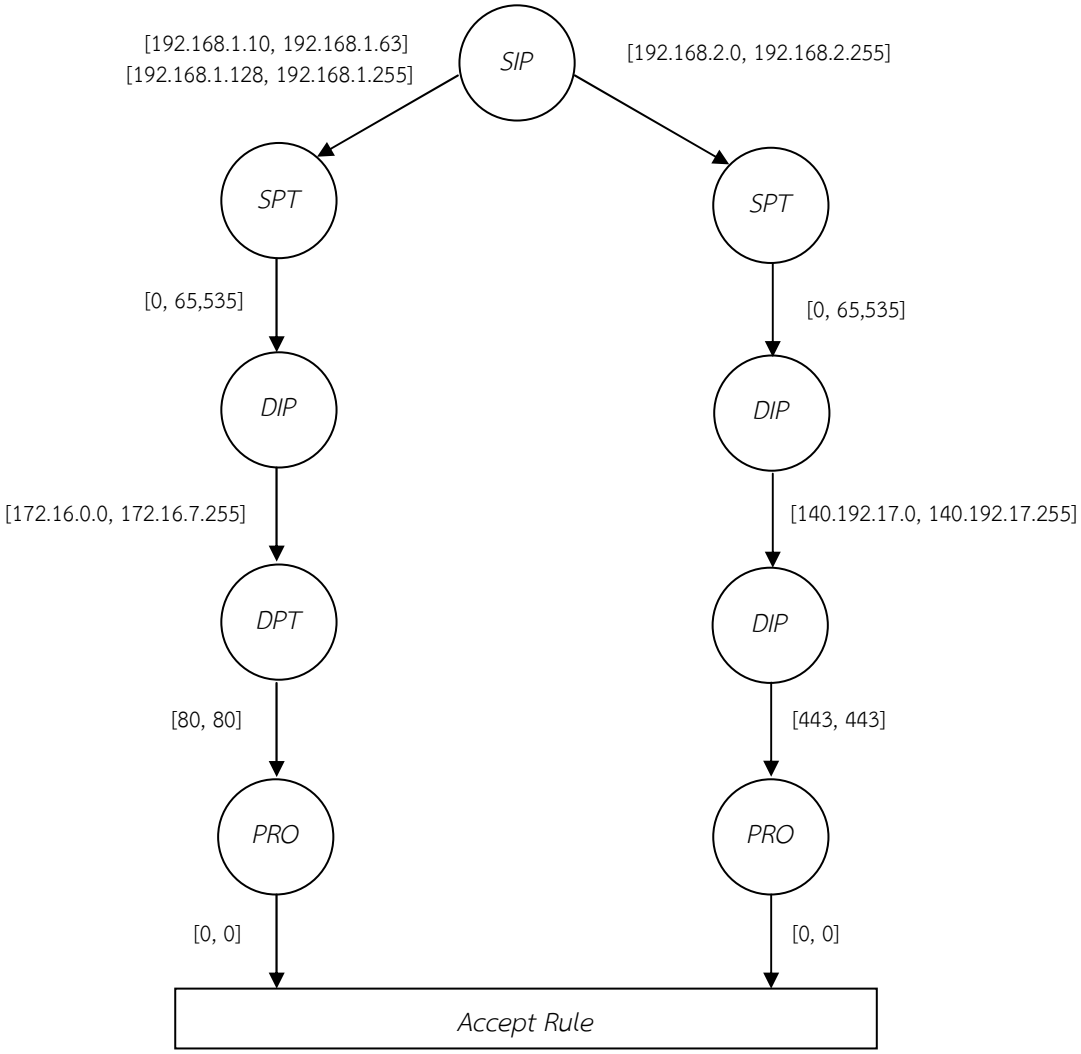
ยกตัวอย่าง หมายเลขไอพี 192.168.1.0/24 ประกอบด้วยข้อมูลจำนวน 2^8 หมายเลข ตั้งแต่หมายเลข 192.168.1.0 – 192.168.1.255 จะสามารถเขียนเป็นช่วงขอบเขตของข้อมูลได้ ข้อมูลขอบเขตบนหรือข้อมูลเริ่มต้นเท่ากับ 192.168.1.0 และข้อมูลขอบเขตล่างหรือข้อมูลสิ้นสุดเท่ากับ 192.168.1.255 โดยข้อมูลยังคงใช้หน่วยความจำเท่ากับ 64 บิต เช่นเดียวกับการกำหนดช่วงข้อมูลแบบเดี่ยว

หลังจากออกแบบการเก็บข้อมูลแบบช่วงแล้วจะสามารถนำมาใช้กับโครงสร้างที่แบ่งลำดับชั้นข้อมูลเพื่อใช้ในการตรวจสอบ โดยในแต่ละระดับจะแบ่งเป็นช่วงของข้อมูลตามรูปแบบของข้อมูล เช่น หมายเลขไอพีต้นทางและปลายทาง (*SIP, DIP*) จะมีขอบเขตเริ่มตั้งแต่หมายเลขไอพี 0.0.0.0 ถึง 255.255.255.255 โดยจะเขียนให้อยู่ในรูปแบบดังนี้ [0.0.0.0, 255.255.255.255] หมายเลขพอร์ตต้นทางและปลายทาง (*SPT, DPT*) จะมีขอบเขตเริ่มตั้งแต่ 0 ถึง 65,535 โดยจะเขียนให้อยู่ในรูปแบบดังนี้ [0, 65535] และโพรโทคอล (*PRO*) จะมีขอบเขตเริ่มตั้งแต่ 0 ถึง 255 เขียนย่อได้ดังนี้ [0, 255]

ตารางที่ 3.1 กฎที่ใช้ในการสร้างไฟร์วอลล์ *SDD*

No	*SIP*	*SPT*	*DIP*	*DPT*	*PRO*	*ACT*
Fwr_1	192.168.1.0/24	All	172.16.0.0/20	80	TCP	Accept
Fwr_2	192.168.1.64/26	All	172.16.0.0/20	80	TCP	Deny
Fwr_3	192.168.2.0/24	All	140.192.17.0/24	443	TCP	Accept

จากตารางที่ 3.1 สามารถนำกฎของไฟร์วอลล์มาสร้างไฟร์วอลล์ *SDD* บนระบบไฟร์วอลล์แบบปิดได้ดังนี้



รูปที่ 3.10 โครงสร้างของไฟร์วอลล์ *SDD* ระบบปิด (*CFS)*

จากรูปที่ 3.10 เมื่อได้โครงสร้างข้อมูล *SDD* แล้วจะเห็นว่าแต่ละโหนดเงื่อนไขจะมีการกำหนดช่วงขอบเขตของข้อมูลซึ่งแต่ละโหนด อาจจะมีหลายช่วงขอบเขตยกตัวอย่างเช่น โหนดราก (*SIP*) ประกอบไปด้วย 3 ช่วงด้วยกันได้แก่ [192.168.1.0, 192.168.1.63], [192.168.1.0, 192.168.1.63] และ [192.168.2.0, 192.168.2.255] ซึ่งข้อมูลแต่ละช่วงจะไม่ทับซ้อนกัน ดังนั้นในการตรวจสอบช่วงขอบเขตจะใช้คุณสมบัติการตรวจผ่านกฎแบบทวิภาคโดยรายละเอียดวิธีการจะกล่าวถึงในหัวข้อต่อไป

3.4.3 การดำเนินการบนระบบไฟร์วอลล์ SDD

1. การเพิ่มกฎบนระบบไฟร์วอลล์ *SDD* คือ การเพิ่มกฎที่มีผลของการกระทำตรงข้ามกับผลของการกระทำระบบเช่น ในระบบไฟร์วอลล์แบบ CFS กฎที่จะเพิ่มจะต้องมีผลของการกระทำเป็นยอมรับ (Accept) เท่านั้น

Algorithm 1: การเพิ่มกฎใหม่
SET SDD_d = เอกภพของกฎไฟร์วอลล์เงื่อนไข d กำหนดให้ $d \in \{ SIP, SPT, DIP, DPT, PRO \}$
SET Fwr_d = กฎของไฟร์วอลล์ใหม่เงื่อนไข d กำหนดให้ $d \in \{ SIP, SPT, DIP, DPT, PRO \}$
LOOP 1: for every element i in SDD_d
IF (INTERSECT($SDD_d(i)$,Fwr_d)) **THEN**
  IF ($SDD_d(i)$!= Fwr_d) **THEN**
    Set $S1_{data1}$ <-- min($SDD_d(i)_{data1}$, $Fwr_{d\ data1}$)
    IF ($SDD_d(i)_{data1}$ == $Fwr_{d\ data1}$) **THEN**
      Set $S1_{data2}$ <-- min($SDD_d(i)_{data2}$, $Fwr_{d\ data2}$) - 1
      Set $S2_{data1}$ <-- min($SDD_d(i)_{data2}$, $Fwr_{d\ data2}$)
    ELSE THEN
      Set $S1_{data2}$ <-- min(max($SDD_d(i)_{data1}$, $Fwr_{d\ data1}$)) – 1
      Set $S2_{data1}$ <-- min(max($SDD_d(i)_{data1}$, $Fwr_{d\ data1}$))
    END
      Set $S2_{data2}$ <-- max($SDD_d(i)_{data2}$, $Fwr_{d\ data2}$))
      IF (INTERSECT ($SDD_d(i)$, S1)) AND (INTERSECT ($SDD_d(i)$, S2)) THEN
        Set $SDD_d(i)_{data1}$ <--$S1_{data2}$
        Add S2 in $SDD_d(i)$
      ELSE IF (INTERSECT ($SDD_d(i)$, S2)) **THEN**
        Add S1 in $SDD_d(i)$
      END
      IF ($S2_{data2}$ – $S1_{data2}$!= 0) **THEN**
        Set $Fwr_{d\ data1}$ <-- min($SDD_d(i)_{data2}$, $Fwr_{d\ data2}$) + 1
        Set $Fwr_{d\ data2}$ <-- max($SDD_d(i)_{data2}$, $Fwr_{d\ data2}$)
      END
    END
  END
END LOOP 1
  IF (Fwr is not Empty) THEN
    ADD Fwr_d in $SDD_d(i)$
  END INSERTIONSORT(**$SDD_d(i)$**)

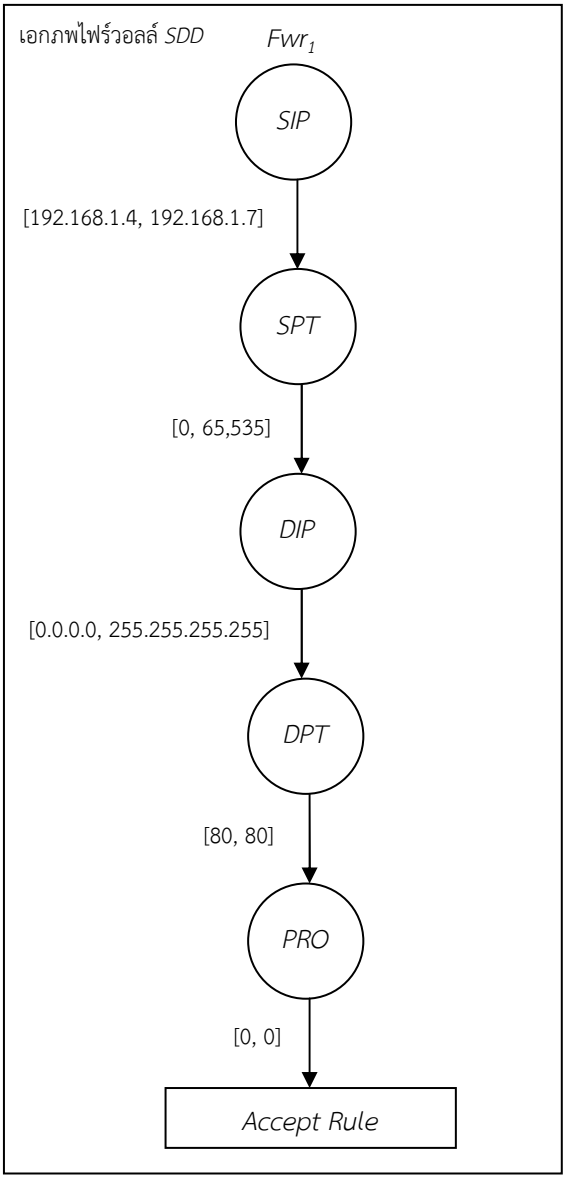
จาก Algorithm1 เป็นตัวอย่างแนวคิดการเพิ่มกฎโดยเช็คเริ่มจาก *SIP* เท่านั้นซึ่งในความเป็นจริงแล้ว กฎของไฟร์วอลล์ประกอบด้วยกัน 5 เงื่อนไขคือ *SIP, SPT, DIP, DPT* และ *PRO* ซึ่งจะมีการทำงานเช่นเดียวกันทั้งหมด โดยเมื่อทำการเช็คและเพิ่มทุกเงื่อนไขแล้วจะได้ตัวอย่างดังนี้

ตัวอย่างการเพิ่มกฎลงในระบบไฟร์วอลล์แบบปิด (Close Firewall System)

ตารางที่ 3.2 กฎที่ใช้ในการเพิ่มเข้าระบบไฟร์วอลล์ *SDD*

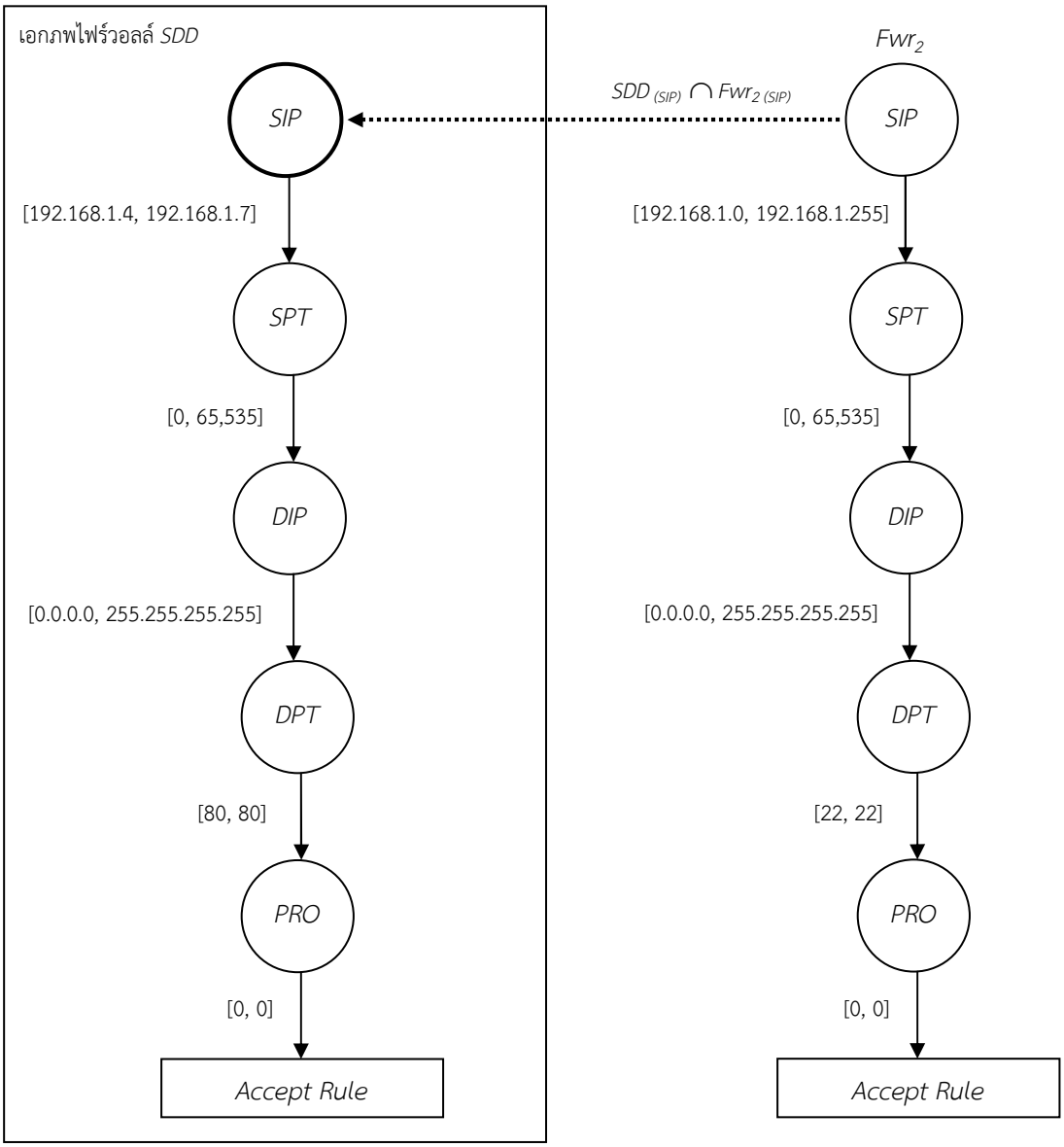
No	*SIP*	*SPT*	*DIP*	*DPT*	*PRO*	*ACT*
Fwr_1	192.168.1.4/30	All	All	80	TCP	Accept
Fwr_2	192.168.1.0/24	All	All	22	TCP	Accept

จากตารางที่ 3.2 จะสามารถสร้างนำกฎของไฟร์วอลล์เพิ่มในระบบไฟร์วอลล์ *SDD* ได้ดังนี้



รูปที่ 3.11 การสร้างเพิ่มกฎบนระบบไฟร์วอลล์แบบปิด *CFS* (1)

จากรูปที่ 3.11 กำหนดให้ *SDD* เป็นเซตของกฎไฟร์วอลล์ในระบบไฟร์วอลล์ เมื่อเพิ่ม Fwr_1 เข้าไปในระบบไฟร์วอลล์ซึ่งยังไม่มีกฎที่กำหนดไว้ก่อนหน้า ดังนั้นจะได้ว่า $SDD \cap Fwr_1 = \phi$ หมายความว่า Fwr_1 ไม่สัมพันธ์กับกฎใดๆบนระบบไฟร์วอลล์ เนื่องจากการเพิ่มกฎของไฟร์วอลล์เข้าระบบจะต้องมีการตรวจสอบความสัมพันธ์ของกฎก่อนเพิ่มเข้าระบบไฟร์วอลล์ เมื่อ Fwr_1 ไม่สัมพันธ์กับกฎใดๆให้ทำการเพิ่มกฎ Fwr_1 เข้าไปในระบบไฟร์วอลล์ *SDD* ได้ทันที แล้วทำการต่อไปโดยเพิ่มกฎ Fwr_2 เข้ามายังระบบไฟร์วอลล์ *SDD*



รูปที่ 3.12 การสร้างเพิ่มกฎบนระบบไฟร์วอลล์แบบปิด *CFS* (2)

รูปที่ 3.12 เมื่อเพิ่มกฎ Fwr_2 เข้ามาในระบบไฟร์วอลล์ SDD จะต้องทำการตรวจสอบสมาชิกของโหนดเงื่อนไขโดยเริ่มจากฟิลด์ *SIP* ซึ่งสามารถเขียนได้ว่า *SIP* ของ *SDD* เท่ากับ $SDD_{(SIP)}$ และ *SIP* ของ Fwr_2 เท่ากับ $Fwr_{2\,(SIP)}$ ดังนั้น เมื่อ $Fwr_{2\,(SIP)} \cap SDD_{(SIP)} \neq \phi$ จะต้องหาช่วงของสมาชิกดังนี้

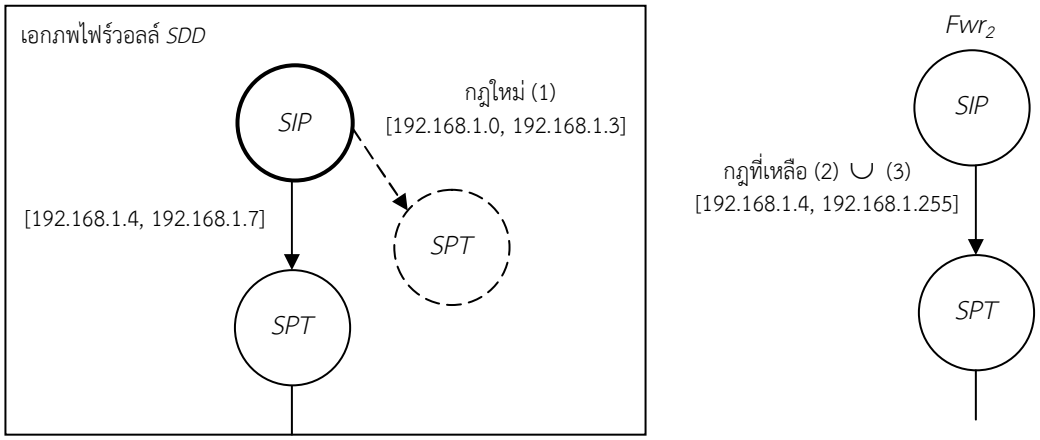
ช่วงของสมาชิก $SDD_{(SIP)}$ เท่ากับ[192.168.1.4, 192.168.1.7] และ $Fwr_{2\,(SIP)}$ เท่ากับ [192.168.1.0, 192.168.1.255] จะมีความสัมพันธ์ 3 ช่วงด้วยกันดังนี้

ข้อมูลช่วงที่ 1 ประกอบด้วยข้อมูลที่เริ่มต้นตั้งแต่ 192.168.1.0 ถึง 192.168.1.3 เป็นช่วงสมาชิกของ $Fwr_{2\,(SIP)}$ เป็น โหนดข้อมูลใหม่

ข้อมูลช่วงที่ 2 ประกอบด้วยข้อมูลที่เริ่มตั้งแต่ 192.168.1.4 ถึง 192.168.1.7 เป็นช่วงสมาชิกของ $SDD_{(SIP[0])} \cap Fwr_{2\,(SIP)}$ เป็น โหนดเงื่อนไขข้อมูลมีสมาชิกร่วมกับโหนดข้อมูลเดิม

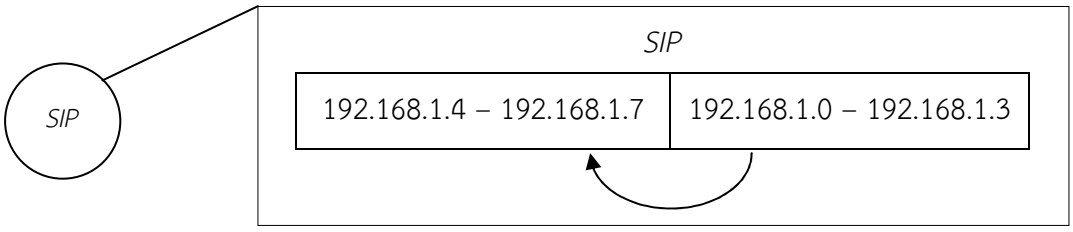
ข้อมูลช่วงที่ 3 ประกอบด้วยข้อมูลเริ่มตั้งแต่ 192.168.1.8 ถึง 192.168.1.255 เป็นช่วงสมาชิกของ $Fwr_{2\,(SIP)}$ เป็น โหนดข้อมูลใหม่

ข้อมูลช่วงที่ 1 เป็นช่วงของสมาชิกที่เป็นโหนดใหม่จะสามารถเพิ่มข้อมูลเข้าในระบบไฟร์วอลล์ได้ทันทีและทุกครั้งที่มีการเพิ่มกฎของไฟร์วอลล์ใหม่เข้าระบบและจัดเก็บข้อมูลของทั้ง 2 เซตสุดท้าย (192.168.1.4 –192.168.1.255) ไว้ใน Fwr_2 ดังรูปที่ 3.13



รูปที่ 3.13 การสร้างเพิ่มกฎบนระบบไฟร์วอลล์แบบปิด *CFS* (3)

เมื่อเพิ่มกฎเข้าไปในระบบไฟร์วอลล์ *SDD* แล้วจะต้องมีการเรียงลำดับข้อมูลใหม่โดยใช้แนวคิดการเรียงลำดับแบบแทรก (Insertion Sort) ซึ่งจะต้องเรียงลำดับข้อมูลจากน้อยไปหามาก ดังรูปที่ 3.14



รูปที่ 3.14 การสร้างเพิ่มกฎบนระบบไฟร์วอลล์แบบปิด *CFS* (4)

ช่วงของข้อมูลเงื่อนที่ถูกเพิ่มเข้าในระบบไฟร์วอลล์จะถูกเพิ่มในตำแหน่งท้ายสุดของอาเรย์ ดังนั้นข้อมูลจะต้องจัดการเรียงลำดับข้อมูลใหม่จากน้อยไปมากโดยมีวิธีการดังนี้

กำหนดให้ *SIP* เป็นอาเรย์ของข้อมูลที่เก็บหมายเลขไอพี ซึ่งมีสมาชิกทั้งหมด 2 ช่วงข้อมูล ด้วยกันคือ

$SIP_{[0]}$ คือช่วงข้อมูลข้อมูลในตำแหน่งที่ 1 ประกอบด้วย 192.168.1.4 ถึง 192.168.1.7

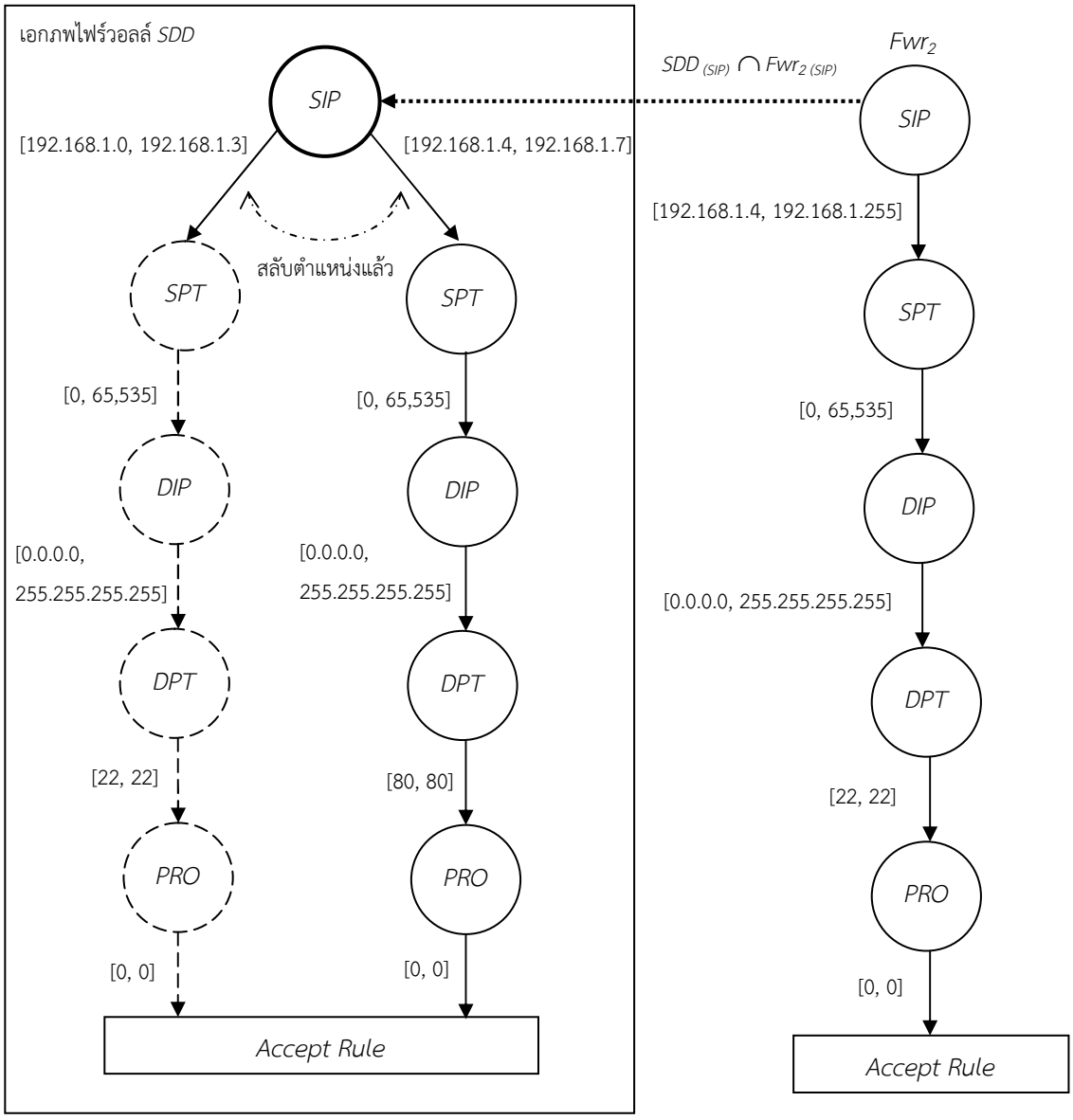
$SIP_{[1]}$ คือช่วงข้อมูลข้อมูลในตำแหน่งที่ 2 ประกอบด้วย 192.168.1.0 ถึง 192.168.1.3

จากนั้นให้เปรียบเทียบช่วงของข้อมูลที่เพิ่มเข้ามาใหม่ในระบบ

ถ้า ขอบเขตล่างของ $SIP_{[1]}$ น้อยกว่า ขอบเขตบนของ $SIP_{[0]}$ แสดงว่า ช่วงของข้อมูล $SIP_{[1]}$ น้อยกว่า $SIP_{[0]}$ ดังนั้นจะต้องสลับตำแหน่งของช่วงข้อมูล

แต่ถ้า ขอบเขตล่างของ $SIP_{[1]}$ มากกว่า ขอบเขตบนของ $SIP_{[0]}$ แสดงว่า ช่วงของข้อมูล $SIP_{[1]}$ มากกว่ากว่า $SIP_{[0]}$ ดังนั้นไม่ต้องสลับตำแหน่งของช่วงข้อมูล

หลังจากเพิ่มโหนดเงื่อนไขใหม่และเรียงลำดับข้อมูลเรียบร้อยแล้วให้เพิ่มโหนดในระดับที่ต่ำกว่า จากตัวอย่าง เพิ่มโหนดเงื่อนไขใหม่ในระดับ 0 (ราก) ดังนั้นโหนดเงื่อนไขในระดับต่ำกว่าได้แก่ *SPT*, *DIP*, *DPT* และ *PRO* ดังรูปที่ 3.15



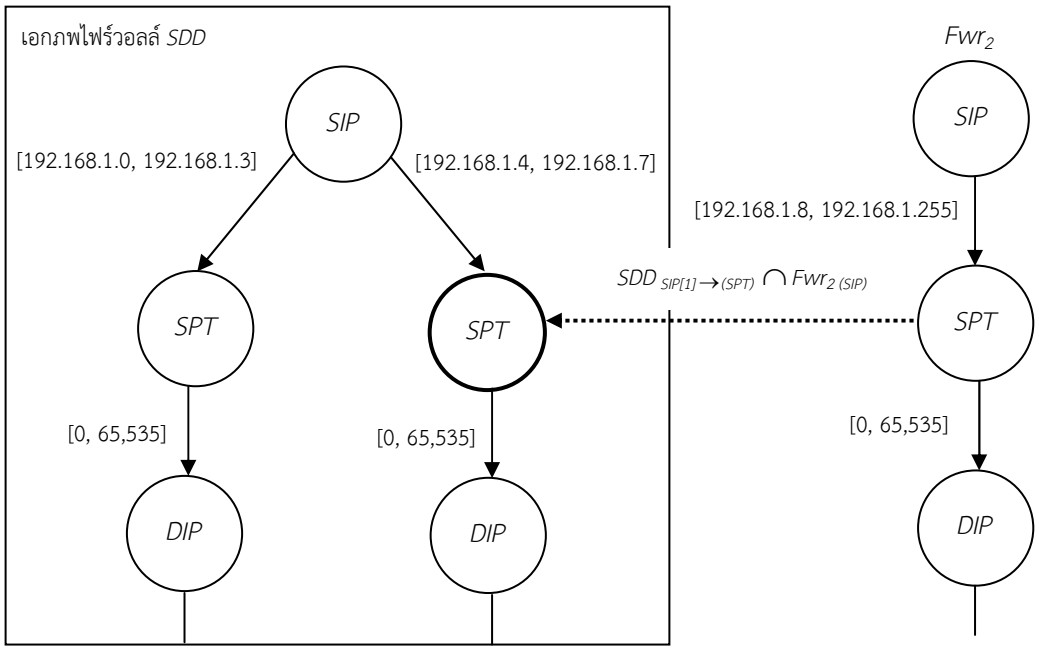
รูปที่ 3.15 การสร้างเพิ่มกฎบนระบบไฟร์วอลล์แบบปิด *CFS* (5)

หลังจากได้เพิ่มกฎใหม่ลงไปในระบบไฟร์วอลล์ *SDD* แล้วให้ตรวจสอบข้อมูลที่อยู่ใน Fwr_2 ว่าเป็นค่าว่างหรือไม่ถ้าไม่เป็นค่าว่างแสดงให้ทำการเปรียบเทียบข้อมูลในระบบไฟร์วอลล์ *SDD* ต่อแต่ถ้าเป็นค่าว่างให้จบการทำงาน ซึ่งในกรณีตัวอย่างยังมีข้อมูลเหลืออยู่จะต้องตรวจสอบความสัมพันธ์ เมื่อ $Fwr_{2\,(SIP)} \cap SDD_{(SIP)} \neq \phi$ ซึ่งจะต้องหาช่วงของสมาชิก $SDD_{(SIP)}$ เท่ากับ [192.168.1.4, 192.168.1.7] และ $Fwr_{2\,(SIP)}$ เท่ากับ [192.168.1.4, 192.168.1.255] ซึ่งได้สมาชิกดังนี้

ข้อมูลช่วงที่ 1 ประกอบด้วยข้อมูลที่เริ่มตั้งแต่ 192.168.1.4 ถึง 192.168.1.7 เป็นช่วงสมาชิกของ $SDD_{(SIP[0])} \cap Fwr_{2\,(SIP)}$ เป็น โหนดเงื่อนไขข้อมูลมีสมาชิกร่วมกับโหนดข้อมูลเดิม

ข้อมูลช่วงที่ 2 ประกอบด้วยข้อมูลเริ่มตั้งแต่ 192.168.1.8 ถึง 192.168.1.255 เป็นช่วงสมาชิกของ $Fwr_{2\ (SIP)}$ เป็น โหนดข้อมูลใหม่

ในกรณีที่โหนดข้อมูลใหม่มีสมาชิกร่วมกับโหนดข้อมูลเดิมให้เลื่อนอันดับการตรวจสอบไปที่โหนดระดับต่ำลงไป และจัดเก็บเซตของข้อมูลของเซตสุดท้ายไว้ที่ Fwr_2 ดังรูปที่ 3.16



รูปที่ 3.16 การสร้างเพิ่มกฎบนระบบไฟร์วอลล์แบบปิด *CFS* (6)

ในการตรวจสอบโหนดเงื่อนไขทุกๆโหนดจะมีการตรวจสอบแบบเดียวกันกับโหนดรากคือจะต้องหาความสัมพันธ์ของเซตโหนด ในระดับต่อไป $SDD_{SIP[1]\ \rightarrow(SPT)}$ คือ โหนดเงื่อนไขหมายเลขพอร์ตปลายทางที่แบ่งเส้นเชื่อมจากหมายเลขไอพีต้นทาง ตำแหน่งที่ 2 ดังรูปที่ 3.16

กำหนดให้ โหนดเงื่อนไข *SPT* ของ Fwr_2 เท่ากับ $Fwr_{2\ (SPT)}$ ดังนั้น เมื่อ $SDD_{SIP[1]\rightarrow(SPT)} \cap Fwr_{2\ (SPT)} \neq \phi$ ให้หาเซตของความสัมพันธ์ของโหนดเงื่อนไขซึ่งจะได้ดังนี้

ช่วงของสมาชิก $SDD_{SIP[1]\rightarrow(SPT)}$ เท่ากับ [0, 65,535] และ $Fwr_{2\ (SPT)}$ [0, 65,535] จะมีความสัมพันธ์หนึ่งช่วงข้อมูลดังนี้

ข้อมูลช่วงที่ 1 ประกอบด้วยข้อมูลที่เริ่มตั้งแต่ 0 ถึง 65,535 เป็นสมาชิกของ $SDD_{SIP[1]\ \rightarrow(SPT[0])} \cap Fwr_{2\ (SPT)}$ เป็นโหนดข้อมูลใหม่มีสมาชิกร่วมกับโหนดข้อมูลเดิม

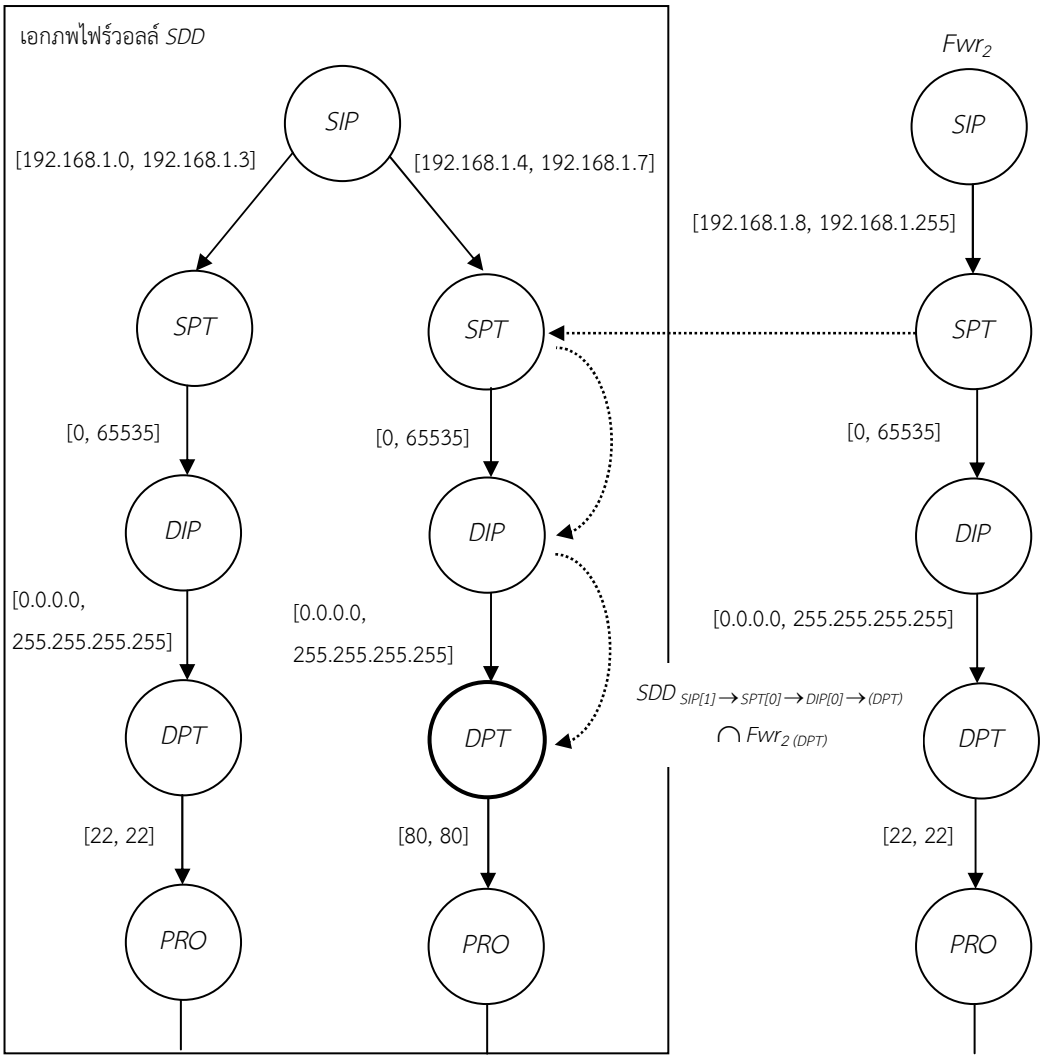
ในกรณีที่โหนดข้อมูลใหม่มีสมาชิกร่วมกับโหนดข้อมูลเดิมให้เลื่อนอันดับการตรวจสอบไปที่โหนดระดับต่ำลงไป และจัดเก็บเซตของข้อมูลของเซตสุดท้ายไว้ที่ Fwr_2 จากรูปที่ 3.17 จะเห็นว่าโหนดข้อมูลที่มีสมาชิกร่วมนี้จะมีลำดับดังนี้

สมาชิกของโหนด *SIP* ของไฟร์วอลล์ *SDD* เขียนสัญลักษณ์ได้เป็น $SDD_{(SIP)}$

สมาชิกของโหนด *SPT* ของไฟร์วอลล์ *SDD* ที่เชื่อมกับ *SIP* ตำแหน่งที่ 2 เขียนสัญลักษณ์ได้เป็น $SDD_{SIP[1] \rightarrow (SPT)}$

สมาชิกของโหนด *DIP* ของไฟร์วอลล์ *SDD* ที่เชื่อมกับ *SPT* เขียนสัญลักษณ์ได้เป็น $SDD_{SIP[1] \rightarrow SPT[0] \rightarrow (DIP)}$

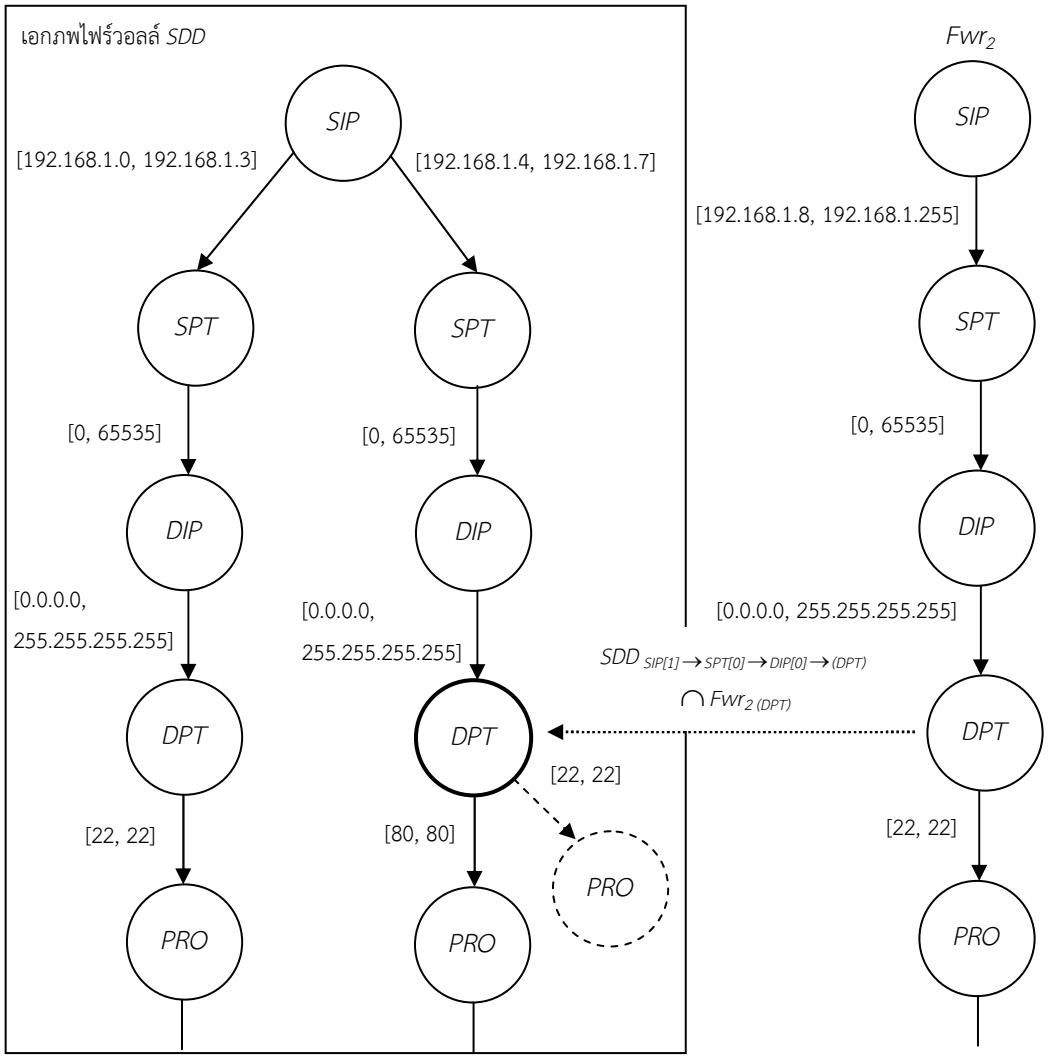
สมาชิกของโหนด *DPT* ของไฟร์วอลล์ *SDD* ที่เชื่อมกับ *DIP* เขียนสัญลักษณ์ได้เป็น $SDD_{SIP[1] \rightarrow SPT[0] \rightarrow (DIP[0]) \rightarrow (DPT)}$ ซึ่งนำมาเปรียบเทียบกับ *DPT* ของ Fwr_2 หรือ $Fwr_{2(DPT)}$ ดังนี้



รูปที่ 3.17 การสร้างเพิ่มกฎบนระบบไฟร์วอลล์แบบปิด *CFS* (7)

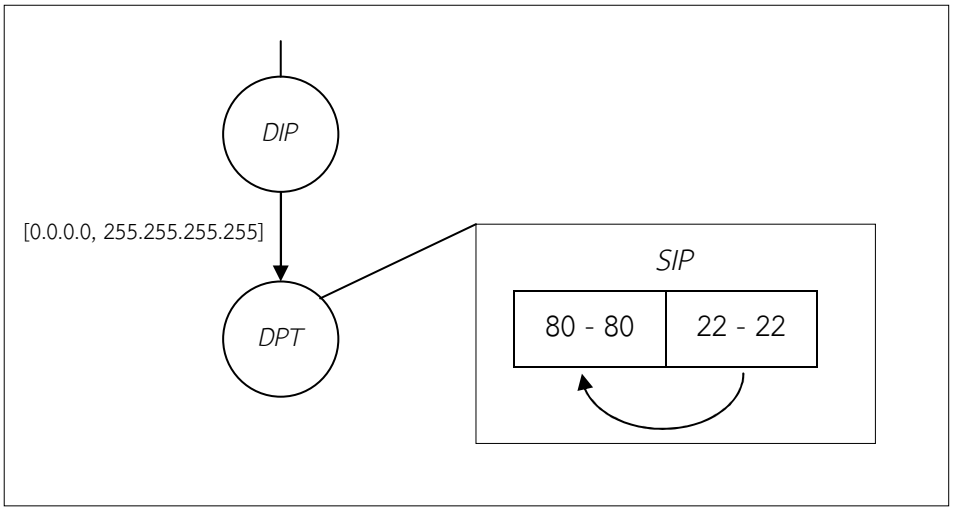
เมื่อมีการตรวจสอบความสัมพันธ์ของโหนดเงื่อนไขจนถึง $SDD_{SIP[1] \rightarrow SPT[0] \rightarrow DIP[0] \rightarrow (DPT)}$ ซึ่งมีสมาชิกเท่ากับ [80, 80] และ $Fwr_{2\ (DPT)}$ [22, 22]

จะได้ว่า เมื่อ $SDD_{SIP[1]\rightarrow SPT[0]\rightarrow DIP[0]\rightarrow (DPT)} \cap Fwr_{2\ (DPT)} = \phi$ แสดงว่าโหนดเงื่อนไข $Fwr_{2\ (DPT)}$ เป็นโหนดเงื่อนไขใหม่ จะต้องแบ่งเส้นเชื่อมออกมาจากโหนด $SDD_{SIP[1]\rightarrow SPT[0]\rightarrow DIP[0]\rightarrow (DPT)}$ ดังรูปที่ 3.18



รูปที่ 3.18 การสร้างเพิ่มกฎบนระบบไฟร์วอลล์แบบปิด *CFS* (8

เมื่อเพิ่มโหนดเงื่อนไขเข้าระบบไฟร์วอลล์ *SDD* แล้ว จะต้องทำการเรียงลำดับโหนดใหม่ทุกครั้งเนื่องจากในการค้นหาของไฟร์วอลล์ *SDD* จะใช้การค้นหาแบบทวิภาค รายงานข้อมูลในแต่ละปุ่มจะต้องมีการเรียงลำดับเสมอ ดังนั้นโหนด $SDD_{SIP[1]\rightarrow SPT[0]\rightarrow DIP[0]\rightarrow (DPT)}$ จะเรียงลำดับข้อมูลจากน้อยไปมากดังรูปที่ 3.19



รูปที่ 3.19 การสร้างเพิ่มกฎบนระบบไฟร์วอลล์แบบปิด *CFS* (9)

จากรูปที่ 3.1.9 ได้มีการประยุกต์ขั้นตอนการเรียงลำดับข้อมูลแบบแทรกให้สามารถเรียงลำดับข้อมูลแบบช่วงของข้อมูลได้ โดยมีวิธิการคิดดังนี้

กำหนดให้ *DPT* เป็นอาเรย์ของข้อมูลที่เก็บหมายพอร์ตปลายทางซึ่งมีสมาชิกทั้งหมด 2 ช่วงข้อมูลด้วยกันคือ

$DPT_{[0]}$ คือช่วงข้อมูลข้อมูลในตำแหน่งที่ 1 ประกอบด้วย 80 ถึง 80

$DPT_{[1]}$ คือช่วงข้อมูลข้อมูลในตำแหน่งที่ 2 ประกอบด้วย 22 ถึง 22

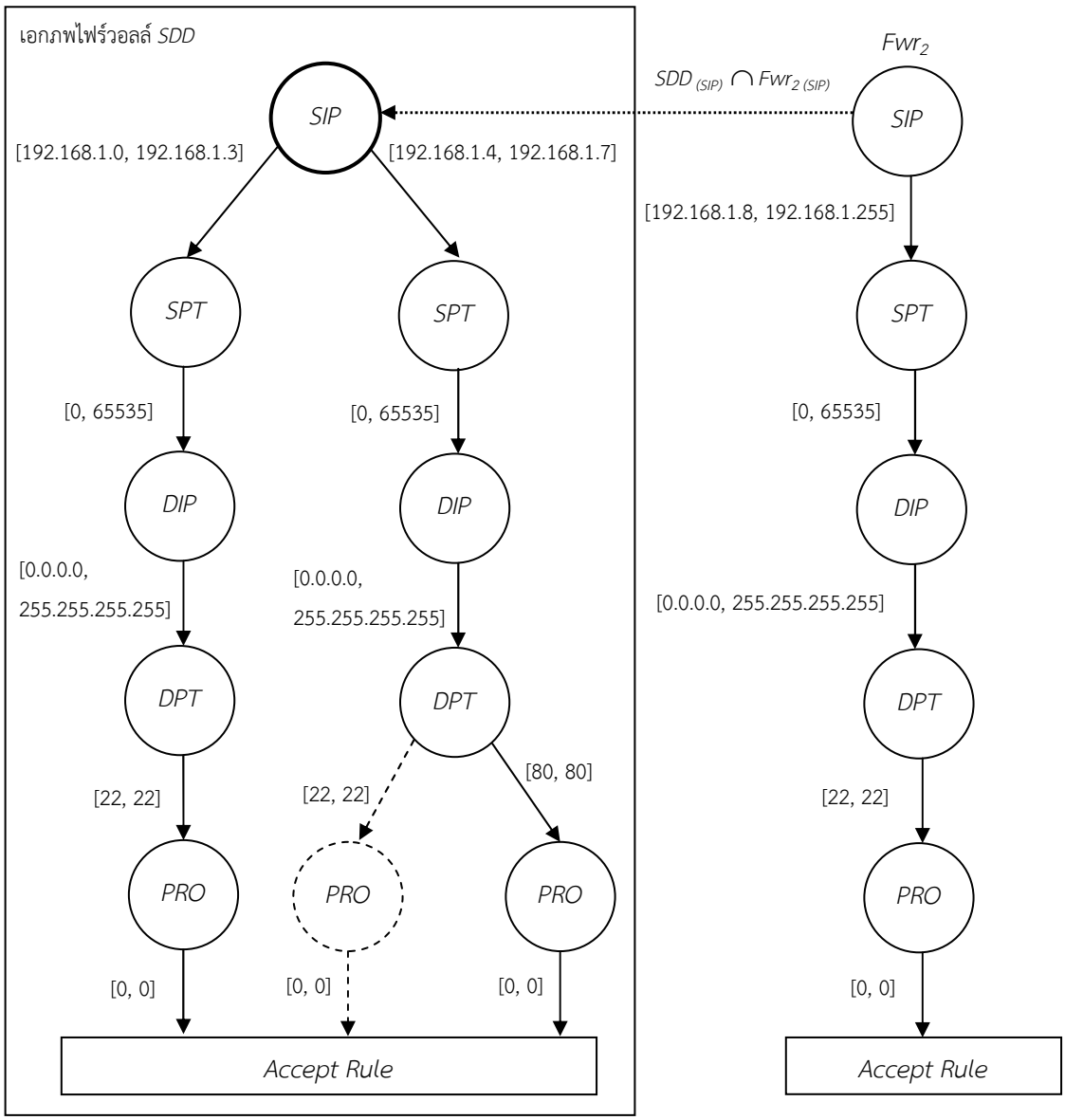
จากนั้นให้เปรียบเทียบช่วงของข้อมูลที่เพิ่มเข้ามาใหม่ในระบบ

ถ้า ขอบเขตล่างของ $DPT_{[1]}$ น้อยกว่า ขอบเขตบนของ $DPT_{[0]}$ แสดงว่า ช่วงของข้อมูล $DPT_{[1]}$ น้อยกว่า $DPT_{[0]}$ ดังนั้นจะต้องสลับตำแหน่งของช่วงข้อมูล

แต่ถ้า ขอบเขตล่างของ $DPT_{[1]}$ มากกว่า ขอบเขตบนของ $DPT_{[0]}$ แสดงว่า ช่วงของข้อมูล $DPT_{[1]}$ มากกว่ากว่า $DPT_{[0]}$ ดังนั้นไม่ต้องสลับตำแหน่งของช่วงข้อมูล

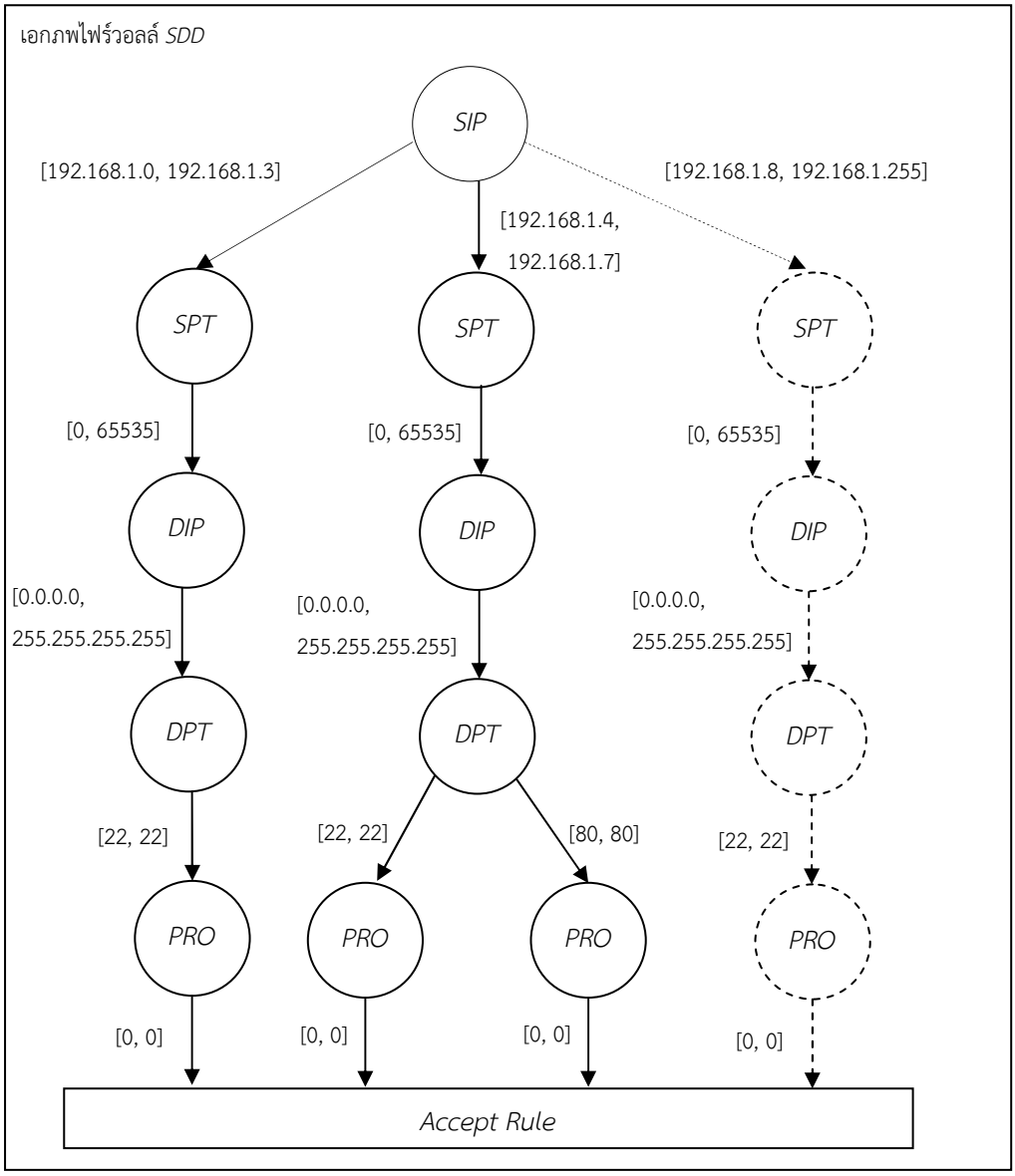
ขอบเขตบนของช่วงข้อมูลเดิมจะไม่มีทางเท่ากับขอบเขตล่างของช่วงข้อมูลใหม่

เมื่อได้ตำแหน่งที่ถูกต้องของช่วงข้อมูลใหม่แล้วให้ทำการเพิ่มโหนดลูกของโหนดเงื่อนไข *DPT* จาก Fwr_2 ดังรูปที่ 3.20



รูปที่ 3.20 การสร้างเพิ่มกฎบนระบบไฟร์วอลล์แบบปิด CFS (10)

หลังจากเพิ่มโหนดเงื่อนไขและโหนดลูกของข้อมูลดังรูปที่ 3.20 แล้วให้ตรวจสอบกฎที่อยู่ใน Fwr_2 ว่ายังมีข้อมูลอยู่หรือไม่ ในกรณีที่ข้อมูลยังไม่หมด ให้ทำการตรวจสอบความสัมพันธ์ของโหนดเงื่อนไข *SIP* บนระบบไฟร์วอลล์ *SDD* และโหนดเงื่อนไข *SIP* ของ Fwr_2 ดังนั้น เมื่อ $SDD_{(SIP)} \cap Fwr_{2\ (SIP)} = \phi$ ให้ทำการเพิ่มโหนดข้อมูลเข้าระบบไฟร์วอลล์แล้วเรียงลำดับข้อมูลใหม่ หลังจากที่เรียงลำดับข้อมูลให้เพิ่มโหนดเงื่อนไขที่อยู่ในระดับต่ำโดยแบ่งเส้นเชื่อมจากโหนดเงื่อนไขใหม่ดังรูปที่ 3.21



รูปที่ 3.21 การสร้างเพิ่มกฎบนระบบไฟร์วอลล์แบบปิด CFS (11)

จากรูปที่ที่ 3.21 โครงสร้างของไฟร์วอลล์ *SDD* ที่มีการเพิ่มกฎ Fwr_1 และ Fwr_2 จะเห็นได้ว่า เส้นที่แบ่งออกมาแต่ละเส้นในโหนดใดๆโหนดหนึ่ง จะไม่มีทางที่เซตของสมาชิกในเส้นเชื่อมนั้นทับซ้อนหรือซ้ำกัน และพบว่าโครงสร้างไฟร์วอลล์ *SDD* เมื่อเพิ่มกฎวิกลภาพบนระบบไฟร์วอลล์ โหนดของข้อมูลที่เกิดกฎวิกลภาพ จะเพิ่มจำนวนเส้นเชื่อมเพื่อขจัดปัญหากฎวิกลภาพ โดยจำนวนเส้นโหนดที่แบ่งเส้นเชื่อมจะมีจำนวนไม่เกิน $2n-1$ ของแต่ละโหนดเงื่อนไข ซึ่งกำหนดให้ n คือจำนวนกฎทั้งหมดของไฟร์วอลล์ที่เพิ่มเข้ามาในระบบ

ตัวอย่างพบว่ากฎที่เพิ่มเข้ามาในระบบไฟร์วอลล์มีจำนวนกฎทั้งหมด 2 กฎดังนั้นโหนดแต่ละโหนดจะมีเส้นเชื่อมได้มากสุด $2(2)-1=3$ เส้น ซึ่งจะไม่มีทางที่โหนดใดๆจะมีเส้นเชื่อมมากว่า 3 เส้น

การเพิ่มกฎของไฟร์วอลล์ *SDD* สามารถเป็นขั้นตอนย่อยๆ ได้ 3 ขั้นตอน แต่ละขั้นตอนสามารถหาระยะเวลาที่ใช้ในการคำนวนได้ดังต่อไปนี้

1. การตรวจสอบความสัมพันธ์ของกฎ ใช้แนวคิดการตรวจสอบแบบเชิงเส้น ซึ่งมีประสิทธิภาพเท่ากับ $O(n)$ ในกรณีที่กฎมีการแตกตัวจำนวนกฎในแต่ละโหนดจะแตกตัวได้ไม่เกิน $2n-1$ และกฎใดๆหนึ่งกฎมีจำนวนโหนดเงื่อนไข d จำนวน จะได้ $O(d\times(2n-1))$ ต่อการเพิ่มกฎไฟร์วอลล์จำนวนหนึ่งกฎ

2 การเรียงลำดับข้อมูลของกฎ ใช้แนวคิดการเรียงลำดับแบบแทรก ซึ่งประสิทธิภาพในกรณีที่แย่ที่สุด (Worst Case) มีค่าเท่ากับ $O(n^2)$ แต่ในการนำมาใช้กับการเรียงลำดับเงื่อนไขของกฎข้อมูลก่อนเพิ่มกฎมีการเรียงลำดับก่อนหน้าแล้วดังนั้น ประสิทธิภาพการเรียงลำดับแบบแทรกจะเท่ากับ $O(n)$ เนื่องจากเป็นประสิทธิภาพในกรณีที่ดีที่สุด เพราะข้อมูลที่เพิ่มเข้าในระบบไฟร์วอลล์จะไม่มีทางที่ข้อมูลจะแทรกระหว่างช่วงของข้อมูลอื่น กฎมีการแบ่งจำนวนโหนดได้ไม่เกิน $2n-1$ และกฎใดๆหนึ่งกฎมีจำนวนโหนดเงื่อนไข d จำนวน จะได้ $O(d\times(2n-1))$

ดังนั้น เมื่อเพิ่มกฎใดหนึ่งกฎเข้าระบบไฟร์วอลล์จะมีประสิทธิภาพเท่ากับ $O(d\times(2n-1)$ + $d\times(2n-1))$ แต่กรณีที่เพิ่มกฎของไฟร์วอลล์ที่มีคุณสมบัติเท่ากับ

$$\sum_{i=0}^{n} \exists j : SDD_{[i][j]} \subseteq Fwr_{[j]} \text{ กำหนดให้ } j \in \{SIP, SPT, DIP, DPT, PRO, ACT\}$$

จะทำให้กฎที่ถูกเพิ่มมีการหแตกกิ่งเท่ากับ $2n$ จำนวนโหนดข้อมูลเท่ากับ d ดังนั้นจะได้ว่าประสิทธิภาพในการเพิ่มกฎของไฟร์วอลล์ *SDD* มีค่าเท่ากับ $O((2n\times d)+2n(d\times(2n-1))\times k)$ กำหนดให้ k คือ ค่าที่ใช้ในการตรวจสอบขอบเขตของข้อมูลในแต่ละครั้ง

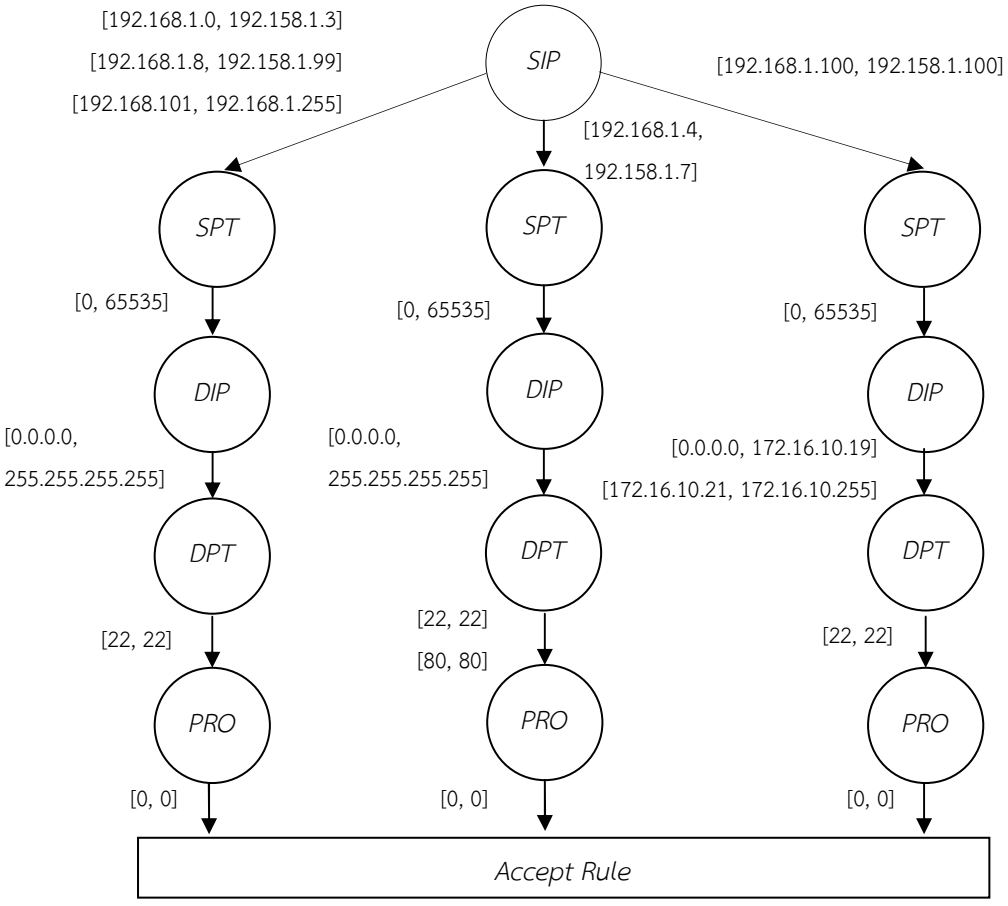
2. การตรวจสอบกฎบนระบบไฟร์วอลล์ SDD

ในการตรวจสอบกฎไฟร์วอลล์ SDD ได้ใช้แนวคิดการค้นหาข้อมูลแบบทวิภาค ซึ่งใช้ในการเปรียบเทียบแพ็กเก็ตข้อมูลเข้ากับกฎไฟร์วอลล์ โดยการตรวจสอบเริ่มต้นจากโหนดราก (*SIP*) ของระบบไฟร์วอลล์และโหนดเงื่อนไขในระดับล่างลงมาเรื่อยจนถึงโหนดเงื่อนไขสุดท้าย (*SPT, DIP, DPT, PRO*) ตามลำดับ

ตารางที่ 3.3 ตัวอย่างกฎไฟร์วอลล์ที่ใช้ในการตรวจสอบบนระบบ SDD

No	*SIP*	*SPT*	*DIP*	*DPT*	*PRO*	*ACT*
Fwr_1	192.168.1.4/30	All	All	80	TCP	Accept
Fwr_2	192.168.1.0/24	All	All	22	TCP	Accept
Fwr_3	192.168.1.100/24	All	172.16.10.20	22	TCP	Deny

จากตารางที่ 3.3 สามารถนำมาสร้างเป็นไฟร์วอลล์ SDD ได้ดังนี้



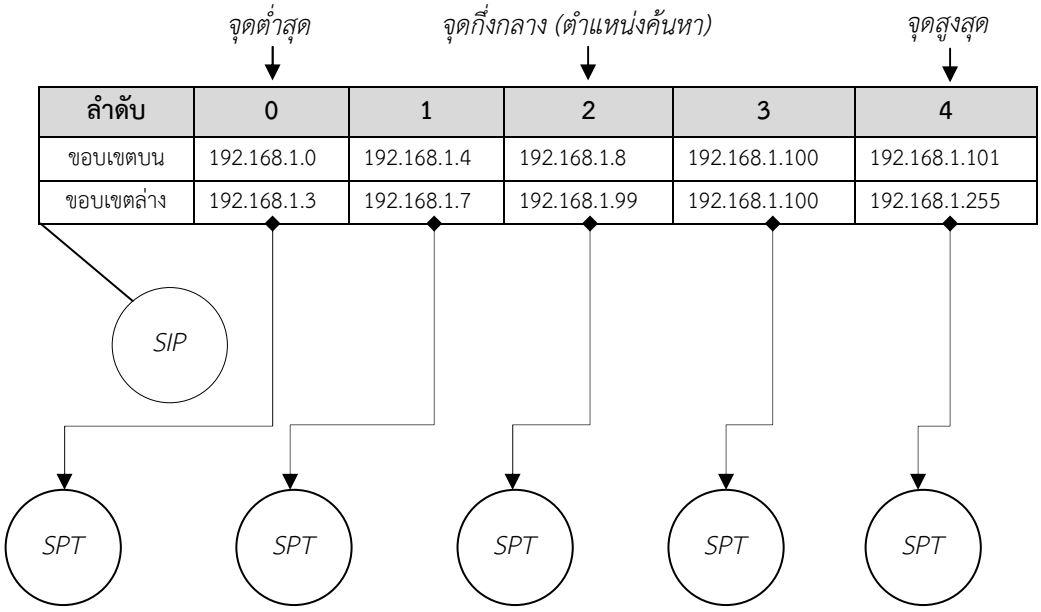
รูปที่ 3.22 การตรวจสอบกฎบนระบบไฟร์วอลล์ *SDD*

สมมุติให้แพ็กเก็ตที่ผ่านเข้าระบบไฟร์วอลล์มีข้อมูล TCP/IP Header เท่ากับ

$Packet_1 : 192.168.1.100 \in SIP,\ 8456 \in SPT,\ 172.16.10.21 \in DIP,\ 80 \in DPT,\ 0 \in SIP$

ดังนั้นเมื่อ $Packet_1$ เข้าสู่ระบบไฟร์วอลล์ *SDD* จากรูปที่ 3.21 จะมีวิธีการตรวจสอบดังต่อไปนี้

เริ่มตรวจสอบจากโหนดราก (*SIP*) โดยใช้การเปรียบเทียบข้อมูลแบบทวิภาค ซึ่งจะต้องคำนวณหาตำแหน่งกลางของเซตข้อมูล ดังรูปที่ 3.23



รูปที่ 3.23 การตรวจสอบกฎไฟร์วอลล์ระบบ SDD ระบบปิด (1)

วิธีการคำนวณหา *จุดกึ่งกลาง* จะต้องกำหนด *จุดต่ำสุด* กับ *จุดสูงสุด* โดยคำนวณจากจำนวนสมาชิกของข้อมูลภายในเซตของโหนดเงื่อนไข (SIP) จะได้ว่า

จุดต่ำสุด = สมาชิกตำแหน่งแรกของเซตข้อมูล (เท่ากับ 0 เมื่อค้นหาครั้งแรก)

จุดสูงสุด = สมาชิกตำแหน่งสุดท้ายของเซตข้อมูล (เท่ากับจำนวนข้อมูล -1)

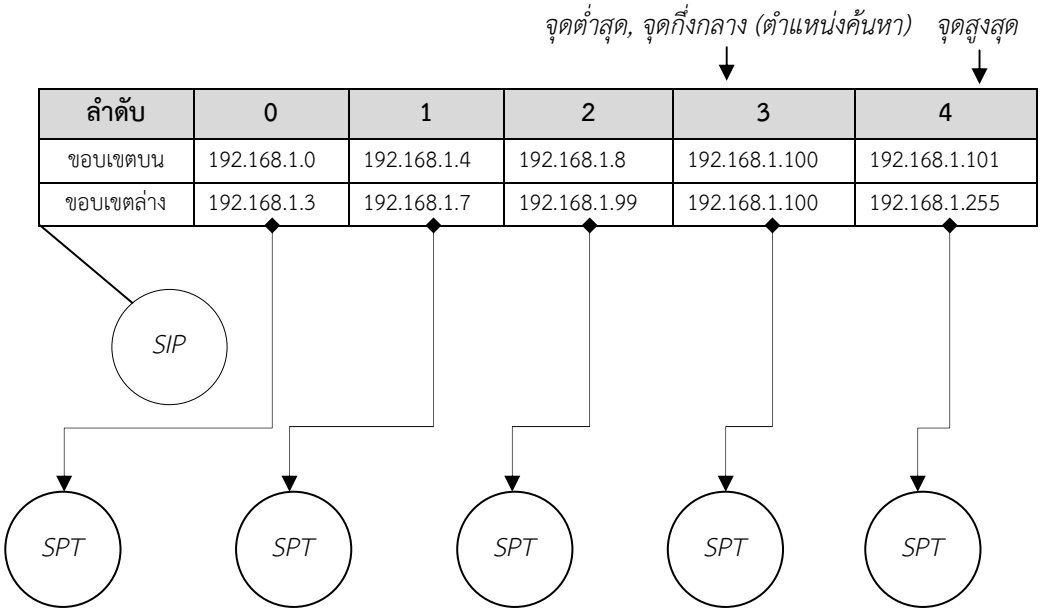
จุดกึ่งกลาง = *(จุดต่ำสุด + จุดสูงสุด) / 2* (กรณีที่หารได้ทศนิยมให้ตัดทศนิยม)

ดังนั้น จะได้ค่า *จุดกึ่งกลาง* เท่ากับ (0 + 4) / 2 = 2 ซึ่งเป็นตำแหน่งของกฎที่ใช้เปรียบเทียบ

เมื่อได้ค่าตำแหน่งของ *จุดกึ่งกลาง* แล้วให้ดึงข้อมูลที่อยู่ในตำแหน่งมาเปรียบเทียบกับข้อมูลของแพ็กเก็ต

$Packet_{1\ (SIP)}$ = 192.168.1.100 , $SDD_{[2]\ (SIP)}$ = 192.168.1.8 - 192.168.1.99

จะได้ว่า $Packet_{1\ (SIP)}$ ไม่เป็นสมาชิกของเซตข้อมูล $SDD_{[2]\ (SIP)}$ หรือไม่ตรงเงื่อนไข แล้วให้เปรียบเทียบค่ามากกว่าหรือน้อยกว่า ซึ่งจากตัวอย่าง $Packet_{1\ (SIP)}$ มากกว่า $SDD_{[2]\ (SIP)}$ กำหนดให้ *จุดกึ่งกลาง* เท่ากับ *จุดต่ำสุด+1* แล้วหาค่าจุดกึ่งกลางใหม่ จะได้เท่ากับ (3 + 4) / 2 = 3.5



รูปที่ 3.24 การตรวจสอบกฎไฟร์วอลล์ระบบ SDD ระบบปิด (2)

จากรูปที่ 3.24 จะได้เซตเงื่อนไข $SDD_{[3]\ (SIP)}$ = 192.168.1.100 - 192.168.1.100 เมื่อนำมาเปรียบเทียบกับ $Packet_{1\ (SIP)}$ พบว่า $Packet_{1\ (SIP)}$ เป็นสมาชิกของ $SDD_{[3]\ (SIP)}$ ดังนั้นจะทำการตรวจสอบกับโหนดเงื่อนไขในระดับที่ต่ำลงมา *(SPT)* ดังนี้

$Packet_{1\ (SPT)}$ เป็นสมาชิกของ $SDD_{[3]\ (SIP)\rightarrow[0]\ (SPT)}$

$Packet_{1\ (DIP)}$ เป็นสมาชิกของ $SDD_{[3]\ (SIP)\rightarrow[0]\ (SPT)\rightarrow[1]\ (DIP)}$

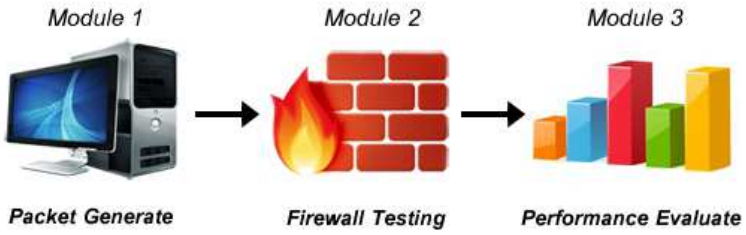
$Packet_{1\ (DPT)}$ ไม่เป็นสมาชิกของ $SDD_{[3]\ (SIP)\rightarrow[0]\ (SPT)\rightarrow[1]\ (DIP)\rightarrow\ (DPT)}$

สรุปได้ว่า $Packet_1$ ไม่สามารถผ่านไฟร์วอลล์ระบบ *SDD* ที่สร้างจากกฎในตารางที่ 3.3

ประสิทธิภาพในการผ่านกฎของไฟร์วอลล์โดยใช้คุณสมบัติการผ่านกฎแบบทวิภาค ซึ่งมีค่าเท่ากับ $O(\log n)$ ซึ่งในการเพิ่มกฎของไฟร์วอลล์แต่ละครั้งโหนดแต่ละโหนดจะมีการแบ่งเส้นเชื่อมมากที่สุดไม่เกิน $2n-1$ กำหนดให้ n คือจำนวนกฎที่นำเข้าไปยังระบบไฟร์วอลล์ การผ่านกฎของไฟร์วอลล์จึงมีประสิทธิภาพเท่ากับ $O(k \times d \times \log(2n-1))$ โดยที่ d คือจำนวนฟิลด์เงื่อนไข และ k คือขอบเขตการตรวจสอบของช่วงข้อมูลแต่ละครั้ง

3.5 การพัฒนาต้นแบบ

การพัฒนาต้นแบบของไฟร์วอลล์ *SDD* ได้มีการวางโครงสร้างในการทดลองดังต่อไปนี้



รูปที่ 3.25 โมเดลการวางตำแหน่งในการทดสอบไฟร์วอลล์

จาก รูปที่ 3.25 ไฟร์วอลล์จะวางไว้ส่วนกลางของการทดสอบซึ่งจะมีเครื่องไคลแอนท์จำลองแพ็คเกต (Packet Generator) เพื่อส่งไปยังไฟร์วอลล์ และไฟร์วอลล์จะทำการตรวจสอบแพ็คเกตที่วิ่งเข้ามาเพื่อส่งแพ็คเกตที่ถูกอนุมัติไปยังเครื่องเซิฟเวอร์ โมเดลนี้จะใช้ทดสอบไฟร์วอลล์ตัดสินใจเดียวและไฟร์วอลล์แบบดั้งเดิมโดยจำลองทั้งสองไฟร์วอลล์บนโปรแกรมจำลองที่พัฒนาด้วยภาษาจาวา (Java SE 7) เวอร์ชัน 7 บนระบบปฏิบัติการวินโดว์ 7 64บิต (Window 7 64bit) หน่วยประมวลผล 3.3 กิกะเฮิร์ต (Ghz) หน่วยความจำ 8 กิกะไบท์ (GB) โดยแบ่งหน้าทีการพัฒนาออกเป็น 2 ส่วนคือ

3.5.1 การสร้างแพ็กเก็ตจำลอง

การสร้างแพ็กเก็ตจำลอง (Packet Generator) คือ การสุ่มหรือจำลองแพ็กเก็ตข้อมูลเพื่อใช้ทดสอบประสิทธิภาพการทำงานของไฟร์วอลล์ ขั้นตอนในการจำลองแพ็กเก็ตในวิทยานิพนธ์นี้ได้ สุ่มกฎของไฟร์วอลล์โดยตั้งเงื่อนไขการทดสอบเพื่อวิเคราะห์รูปแบบการทำงานของไฟร์วอลล์ให้ครบถ้วนทุกกรณี โดยการสุ่มจะสุ่มแพ็คเกตทั้งหมด 100,000 แพ็คเกตเพื่อใช้ในการผ่านกฎไฟร์วอลล์ ส่วนประกอบในแพ็คเกตประกอบไปด้วย หมายเลขไอพีต้นทาง (SIP) หมายเลขพอร์ตต้นทาง (SPT) หมายเลขไอพีปลายทาง (DIP) หมายเลขพอร์ตปลายทาง (DPT) และโพรโทคอล (PRO)

ขอบเขตของการสุ่มแพ็กเก็ตที่จะอยู่ภายใต้ขอบเขตสมาชิกของกฎไฟร์วอลล์ เนื่องจากประสิทธิภาพของไฟร์วอลล์หากพบว่า มีการกระทบกับกฎของไฟร์วอลล์มากเพียงใดจะทำให้ไฟร์วอลล์ทำงานมากขึ้นดังนั้น หากสุ่มแพ็กเก็ตภายใต้ขอบเขตสมาชิกของกฎไฟร์วอลล์ แพ็กเก็ตที่สุ่มจะกระทบกับกฎของไฟร์วอลล์ทุกแพ็กเก็ต ซึ่งแต่ละฟิลด์ข้อมูลเงื่อนไขในการสุ่มจะมีด้วยกันดังนี้

Algorithm 4: การจำลองแพ็คเกตแบบสุ่ม จำนวน 100,000 แพ็คเกต

i = เริ่มต้นตั้งแต่ 1 เป็นแพ็คเกตแรก

Loop1 : i < 100,000 สุ่มแพ็คเกตตั้งแต่ 1 ถึง 100,000 แพ็คเกต

OctetSA1 = Random 0 - 255

OctetSA2 = Random 0 - 255

OctetSA3 = Random 0 - 255

OctetSA4 = Random 0 - 255

$\mathbf{SIP_i}$ = (OctetSA1 x 2^{24}) + (OctetSA2 x 2^{16}) + (OctetSA3 x 2^{8}) +OctetSA4

$\mathbf{SPT_i}$ = Random 0 - 65535

OctetDA1 = Random 0 - 255

OctetDA2 = Random 0 - 255

OctetDA3 = Random 0 - 255

OctetDA4 = Random 0 - 255

$\mathbf{DIP_i}$ =(OctetDA1 x 2^{24}) + (OctetDA2 x 2^{16}) + (OctetDA3 x 2^{8}) +OctetDA4

$\mathbf{DPT_i}$ = Random 0 - 65535

$\mathbf{PRO_i}$ = Random 0 - 256

END

เมื่อเริ่มทำงาน Algorithm 4 จะทำการสุ่มแพ็คเกตจำนวน 100,000 แพ็คเกตเก็บไว้ที่ตัวแปร Packet เป็นอาเรย์ 2 มิติโดยที่มีรายละเอียดของแพ็กเกตทั้งหมดดังนี้

SIP_i เก็บค่าของหมายเลขไอพีต้นทาง ตั้งแต่ 0.0.0.0 – 255.255.255.255 ตัวอย่างเช่น 172.168.9.5, 192.168.1.24 และหมายเลขไอพีต้นทางจะถูกเปลี่ยนให้เป็นเลขฐานสิบตั้งแต่ 0 ถึง 4,924,967,296 จากตัวอย่างจะเปลี่ยนได้เป็น 2,896,693,509 และ 3,232,235,800 ตามลำดับ

SPT_i เก็บค่าหมายเลขพอร์ตต้นทาง ตั้งแต่ 0 – 65535 ตัวอย่างเช่น 80, 21

DIP_i เก็บค่าของหมายเลขไอพีปลายทาง ตั้งแต่ 0.0.0.0 – 255.255.255.255 ตัวอย่างเช่น 202.28.34.234, 1.2.3.4 และหมายเลขไอพีปลายทางจะถูกเปลี่ยนให้เป็นเลขฐานสิบตั้งแต่ 0 – 4924967296 จากตัวอย่างจะเปลี่ยนได้เป็น 3,390,841,578 และ 16,909,060 ตามลำดับ

DPORTi เก็บค่าหมายเลขพอร์ตปลายทาง ตั้งแต่ 0 – 65535 ตัวอย่างเช่น 23, 52

PROi เก็บค่าโพรโทคอล ตั้งแต่ 0 – 1 โดยที่ 0 คือ TCP และ 1 คือ UDP

การสุ่มแพ็กเก็ตข้างต้นเป็นการสุ่มโดยจะได้สภาพแวดล้อมของแพ็กเก็ตทั่วไปที่มีการผ่านเข้าออกเครือข่ายหรือถูกปฏิเสธจากระบบ ดังนั้นการประเมินผลด้านการผ่านกฎของไฟร์วอลล์จะต้องมีการเปรียบเทียบประสิทธิภาพสัญกรโอใหญ่ควบคู่ไปด้วย

3.5.2 การสร้างกฎของไฟร์วอลล์ของ *SDD*

ได้มีการสร้างกฎโดยการสุ่มกฎเป็นจำนวนต่างๆ ได้แก่ 500, 1,000, 1,500, 2,000, 2,500, 3,000, 3,500 และ 4,000 ตามลำดับ ซึ่งวิธีการเพิ่มกฎเพื่อประเมินประสิทธิภาพให้ได้ผลที่สูงสุด กฎไฟร์วอลล์ที่เพิ่มเข้าระบบไฟร์วอลล์ *SDD* จะต้องเป็นกฎที่มีลักษณ์ความสัมพันธ์แบบกระทำซ้ำซ้อนดังสมการ

$$\sum_{i=0}^{n} \exists j : SDD_{[i][j]} \subset Fwr_{[j]} \text{ กำหนดให้ } j \in \{SIP, SPT, DIP, DPT, PRO, ACT\}$$

ซึ่งพบว่าเมื่อเพิ่มกฎไฟร์วอลล์ที่มีคุณสมบัติดังสมการข้างต้นจะทำให้กฎของไฟร์วอลล์มีการแบ่งตัวมากที่สุด ดังนั้นวิธีการสร้างกฎที่ใช้ในการทดลองนี้เป็นกรณีที่ใช้เวลามากที่สุดในการสร้างซึ่งหากผลการทดลองปรากฏว่า เวลาในการสร้างกฎยังสามารถยอมรับได้ จะแสดงว่าข้อเสนอของงานวิจัยนี้ยอมรับได้

3.6 การประเมิน

ในการประเมินผลการทำงานของไฟร์วอลล์ได้แบ่งออกเป็น 2 ส่วนหลักๆ คือ ด้านวิกลภาพของกฎไฟร์วอลล์ (Firewall Rule Anomaly) และด้านประสิทธิภาพการทำงานของไฟร์วอลล์ (Firewall Verification)

3.6.1 ด้านวิกลภาพของกฎไฟร์วอลล์

การประมวลผลด้านวิกลภาพของกฎไฟร์วอลล์ได้มีการจำลองกฎของไฟร์วอลล์ที่เกิดความสัมพันธ์ที่ขัดแย้ง 4 รูปแบบด้วยกันคือ กฎที่ถูกบัง (*Shadowing Anomaly*), กฎที่เกี่ยวพันกัน (Correlation Anomaly), กฎที่คลุมเครือ (Generalization Anomaly) และกฎที่กระทำซ้ำซ้อน (Redundancy Anomaly)

3.6.2 ด้านประสิทธิภาพการทำงานของไฟร์วอลล์

ในการประเมินผลได้เปรียบเทียบประสิทธิภาพการทำงานของไฟร์วอลล์ SDD กับไฟร์วอลล์แบบดั้งเดิมโดยแบ่งออกเป็น 2 ส่วนย่อยด้วยกันคือ

1. เวลาที่ใช้ในการประมวลผล

เวลาที่ใช้ในการประมวลผล (Time Complexity) คือเวลาทั้งหมดที่ใช้ในการประมวลผลอัลกอริทึม โดยวิทยานิพนธ์นี้ได้แบ่งแยกเวลาออกเป็น 2 ส่วนคือ

เวลาที่ใช้ในการสร้างไฟร์วอลล์ตัดสินใจเดียว คือ ระยะเวลาที่ใช้ในการสร้างกฎทั้งหมดได้จากการเวลาเพิ่มกฎของไฟร์วอลล์ (Insert Time) ทั้งหมดและเวลาในการเรียงจัดลำดับกฎ ในการเปรียบเทียบเวลาที่ใช้สร้างกฎของไฟร์วอลล์จะต้องนำเวลาทั้งสองมาบวกกัน ซึ่งมีค่าเท่ากับ $O(d\times(2n-1)\ +\ d\times(2n-1))$ และนำมาเปรียบเทียบกับไฟร์วอลล์แบบดั้งเดิมซึ่งจะเห็นได้ว่าไฟร์วอลล์แบบดั้งเดิมจะไม่มีระยะเวลาในการจัดเรียงกฎมีเพียงแต่ระยะเวลาในการเพิ่มกฎเท่านั้น

เวลาที่ใช้ในการผ่านกฎ คือ ระยะเวลาที่ใช้ในการเข้าถึงกฎหรือผ่านกฎของไฟร์วอลล์ ได้จากการจำลองข้อมูลโดยการสุ่มจำนวนมาก 100,000 ข้อมูล เปรียบเทียบกับกฎของไฟร์วอลล์และเปรียบเทียบระยะเวลาซึ่งประสิทธิภาพของไฟร์วอลล์ตัดสินใจเดียวจะเท่ากับ $O(k\times d\times\log(2n-1))$ ส่วนไฟร์วอลล์แบบดั้งเดิมจะเท่ากับ $O(d\times n)$ โดยที่ d จำนวนเงื่อนไขที่ใช้ในการผ่านกฎ และ n คือจำนวนกฎทั้งหมดของไฟร์วอลล์

2. หน่วยความจำที่ใช้ประมวลผล

หน่วยความจำที่ใช้ประมวลผล (Space Complexity) คือหน่วยความจำที่ใช้ในการสร้างกฎของไฟร์วอลล์ เนื่องจากการสร้างกฎของไฟร์วอลล์ตัดสินใจเดียวมีความยืดหยุ่นไม่คงที่ จึงได้ทำการคำนวณกรณีที่ใช้หน่วยความจำมากที่สุด ซึ่งสามารถคำนวนจากจำนวนกฎของไฟร์วอลล์ได้ ในกรณีที่กฎเกิดการแบ่งตัวในแต่ละฟิลด์เท่ากับ $2n-1$ โดยที่กฎหนึ่งกฎประกอบไปด้วยฟิลด์เงื่อนไข d ดังนั้นจะสามารถคำนวนกฎได้ทั้งหมด $d\times(2n-1)$ ซึ่งจะสามารถนำไปใช้ในการคำนวณหาค่าหน่วยความจำและเปรียบเทียบได้ ซึ่งกฎไฟร์วอลล์ใด หนึ่งกฎประกอบไปด้วย หมายเลขไอพีต้นทาง (Source IP Address, SIP) ขอบเขตบนและล่าง 64 บิต, หมายเลขพอร์ตต้นทาง(Source Port, SPT) ขอบเขตบนและล่าง 32 บิต, หมายเลขไอพีปลายทาง (Destination IP Address, DIP) ขอบเขตบนและล่าง 64 บิต, หมายเลขพอร์ตปลายทาง (Destination Port, DPT) ขอบเขตบนและล่าง 32 บิต และโพรโทคอล (Protocol, PRO) 8 บิต รวมทั้งหมด 200 บิตต่อ 1 กฎที่ใช้ในการคำนวน

3.7 สรุปผลการแก้ไขปัญหา

วิเคราะห์ผลที่ได้จากการประเมิน มีความถูกต้องแม่นยำ มีข้อดีข้อเสียอย่างไรและเหมาะสมกับการทำงานของไฟร์วอลล์หรือไม่ ระยะเวลาที่ใช้ในการทำงานมีส่วนเกินและยอมรับได้ การตรวจสอบความผิดพลาดต้องครอบคลุมกฎวิกลภาพทั้ง 4 กรณีคือ Shadowing Anomaly, Correlation Anomaly, Generalization Anomaly และ Redundancy Anomaly และประสิทธิภาพในการผ่านกฎเท่ากับ $O(k\times d\log(n))$ โดยที่ d คือจำนวนเงื่อนไขของกฎไฟร์วอลล์ n คือจำนวนกฎของไฟร์วอลล์ซึ่งแทนค่าด้วย $2n-1$ ในกรณีที่กฎมีการแบ่งกิ่งก้านมากที่สุด และ k คือค่าคงที่ในการตรวจสอบขอบเขตของช่วงข้อมูลในแต่ละครั้ง

บทที่ 4

ผลการวิจัยและการอภิปราย

จากบทที่ 3 ได้นำเสนอแนวคิดการออกแบบไฟร์วอลล์การตัดสินใจแบบโดเมนเดี่ยว (Single Decision Domain, *SDD*) ซึ่งประกอบไปด้วย แรงจูงใจในการออกแบบ ขั้นตอนออกแบบ การกำหนดองค์ประกอบสำหรับการทดลอง และรูปแบบที่ใช้สำหรับการทดลอง ในบทนี้จะวิเคราะห์ถึงผลที่ได้จากการทดลองว่า ผลที่ได้คืออะไร เกิดผลอย่างไร และผลที่ได้จะนำไปใช้ประโยชน์อะไรได้บ้าง โดยการทดลองแบ่งออกเป็น 2 ส่วนคือ ส่วนที่หนึ่งจะเป็นผลการแก้ไขปัญหาวิกลภาพของกฎไฟร์วอลล์ซึ่งได้เปรียบเทียบและวิเคราะห์กับงานวิจัยอื่นโดยละเอียด ส่วนที่สองจะเป็นการประเมินประสิทธิภาพในการทำงานของไฟร์วอลล์ *SDD* ซึ่งแบ่งย่อยออกเป็น 3 ส่วนด้วยกันคือ ผลการทดลองความเร็วในการสร้างโครงสร้างไฟร์วอลล์ *SDD* ผลการทดลองความเร็วในการผ่านกฎของไฟร์วอลล์ *SDD* และส่วนที่สามเป็นผลการทดลองหน่วยความจำที่ใช้ในการสร้างไฟร์วอลล์ *SDD* โดยผลการทดลองได้เปรียบเทียบระหว่างไฟร์วอลล์ *SDD* กับไฟร์วอลล์แบบดั้งเดิม (Traditional Firewall) ซึ่งรายละเอียดดังนี้

4.1 การเปรียบเทียบวิธีการแก้ไขวิกลภาพของกฎไฟร์วอลล์

วิกลภาพของกฎไฟร์วอลล์ที่นำมาใช้ในการตรวจสอบได้แบ่งออกเป็น 4 รูปแบบด้วยกัน โดยอ้างอิงจากงานวิจัยของ Al-Shaer และคณะ [1] จากรูปแบบวิกลภาพของกฎไฟร์วอลล์ทั้งหมด 6 รูปแบบด้วยกัน ซึ่งมี 2 รูปแบบของกฎวิกลภาพที่ไม่สามารถตรวจสอบได้โดยใช้ข้อมูลจากกฎของไฟร์วอลล์ทั้งหมดเพียงอย่างเดียว เพราะในการตรวจสอบกฎวิกลภาพของทั้ง 2 รูปแบบนั้นจำเป็นจะต้องใช้ข้อมูลของระบบเครือข่ายทั้งระบบในการวิเคราะห์และตรวจสอบกฎวิกลภาพ โดยรูปแบบกฎวิกลภาพที่นำมาตรวจสอบมีดังต่อไปนี้

4.1.1 เปรียบเทียบระหว่างไฟร์วอลล์ *SDD* กับ ไฟร์วอลล์แบบดั้งเดิม

ในการทดลองนี้เราได้เปรียบเทียบการแก้ไขปัญหากฎวิกลภาพระหว่างไฟร์วอลล์ *SDD* กับไฟร์วอลล์แบบดั้งเดิมซึ่งแน่นอนว่าการไฟร์วอลล์แบบดั้งเดิม (Traditional Firewall) กฎสามารถเกิดกฎวิกลภาพได้ด้วยกัน 4 รูปแบบดังต่อไปนี้

1. วิกลภาพของกฎไฟร์วอลล์แบบ Shadowing Anomaly กฎใดๆ จะเป็นกฎวิกลภาพกันแบบ Shadowing Rule เมื่อ

$$\forall i : Fwr_y[i] \subseteq Fwr_x[i] \wedge Fwr_x[\text{ACT}] \neq Fwr_y[\text{ACT}]$$

กำหนดให้ $i \in \{SIP, SPT, DIP, DPT, PRO\}$

ดังนั้นการหา Shadowing Anomaly ในโครงสร้างของไฟร์วอลล์จะต้องทำการทดสอบตั้งแต่ Fwr_2 ขึ้นไปจนกฎสุดท้าย (Fwr_n) ว่ามีกฎใดเป็น Shadowed Rule บ้าง ถ้าเกิดกฎใดเป็น Shadowed Rule ก็แสดงว่ากฎนั้นเกิด Shadowing Anomaly (Fwr_i ไม่มีโอกาสเป็น Shadowed Rule จึงไม่เป็นต้องทำการทดสอบ) ดังตารางที่ 4.1

ตารางที่ 4.1 กฎของไฟร์วอลล์ที่เกิดวิกลภาพแบบ Shadowing Anomaly

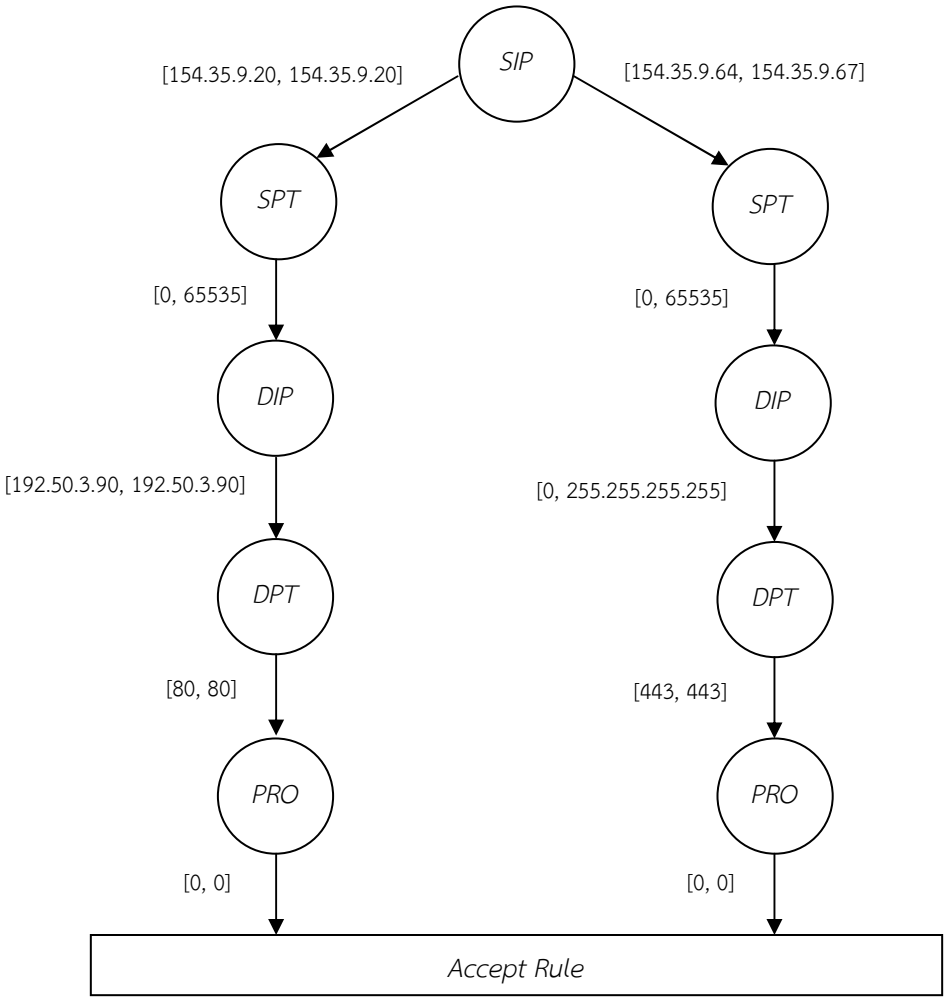
No	*SIP*	*SPT*	*DIP*	*DPT*	*PRO*	*ACT*
Fwr_1	154.35.9.0/24	all	all	80	TCP	Deny
Fwr_2	154.35.9.20	all	191.50.3.90	80	TCP	Accept
Fwr_3	154.35.9.64/30	all	all	443	TCP	Accept

จากตารางที่ 4.1 เป็นตัวอย่างของกฎไฟร์วอลล์ที่เกิดวิกลภาพแบบ Shadowing Anomaly ซึ่งเมื่อเกิดกฎวิกลภาพในระบบไฟร์วอลล์ จะทำให้กฎของไฟร์วอลล์ที่ถูกเพิ่มเข้ามาไม่มีทางที่จะถูกกระทำกับข้อมูลที่วิ่งเข้า - ออกระบบเครือข่ายได้ ซึ่งจะเห็นได้ว่า ทุกเงื่อนไขของ Fwr_1 ได้ครอบคลุมเงื่อนไข Fwr_2 หรือทุกเงื่อนไขของ Fwr_2 เป็นสมาชิกของ Fwr_1 และทั้งสองกฎมีผลของการกระทำที่แตกต่างกัน ซึ่งจะทำให้ Fwr_2 ไม่มีทางถูกกระทำเนื่องจากจะกระทบกับ Fwr_1 ก่อนเสมอ

สมมุติให้ P_2 เป็นแพ็กเก็ตที่ผ่านเข้าออกไฟร์วอลล์โดยมีข้อมูลดังนี้

$$P_2 : 154.35.9.20 \in SIP,\ 8457 \in SPT,\ 192.50.3.90 \in DIP,\ 80 \in DPT,\ 0 \in PRO$$

จาก P_2 เป็นแพ็กเก็ตที่ต้องการให้ผ่านเข้าระบบเครือข่ายซึ่งสอดคล้องกับกฎของไฟร์วอลล์ Fwr_2 อย่างไรก็ตาม P_2 จะไม่สามารถผ่านเข้าระบบเครือข่ายได้เนื่องจากติดเงื่อนไขของกฎไฟร์วอลล์ Fwr_1 ซึ่งอยู่ลำดับก่อนหน้า และ Fwr_2 เป็นกฎที่ถูกบัง จากกฎของไฟร์วอลล์ข้างต้นนี้สามารถสร้างเป็นโครงสร้างของไฟร์วอลล์แบบ *SDD* ไฟร์วอลล์ได้ ดังรูปที่ 4.1



รูปที่ 4.1 โครงสร้างไฟร์วอลล์ SDD เมื่อสร้างจากกฎที่เกิด Shadowing Anomaly

จากรูปที่ 4.1 เมื่อสร้างโครงสร้างจากกฎของไฟร์วอลล์ที่เกิดวิกลภาพแบบ Shadowing Anomaly จากตารางที่ 4.1 จะได้โครงสร้างของไฟร์วอลล์ที่ปราศจากกฎวิกลภาพ ซึ่งสามารถพิสูจน์ได้ว่า เมื่อเพิ่ม Fwr_1 ในตัวอย่างนี้มีผลของการกระทำคือ ปฏิเสธ ซึ่งกฎจะไม่ถูกสร้าง เนื่องจากไฟร์วอลล์ *SDD* มีลักษณะโครงสร้างเป็นไฟร์วอลล์แบบโดเมนเดี่ยวโดยเลือกใช้ในรูปแบบของระบบแบบปิด (Close System) ซึ่งจะไม่สนใจกฎไฟร์วอลล์ที่มีผลของการกระทำที่เป็นปฏิเสธ ดังนั้น เมื่อเพิ่มกฎของไฟร์วอลล์ Fwr_2 เข้ามาในระบบไฟร์วอลล์แบบปิดจึงนับกฎของไฟร์วอลล์กฎนี้เป็นกฎแรก ซึ่งจะทำให้ไฟร์วอลล์ไม่มีปราศจากกฎวิกลภาพในกรณีนี้

2. วิกลภาพของกฎไฟร์วอลล์แบบ Correlation Anomaly กฎใดๆ จะเป็นวิกลภาพกันแบบ Correlation Rule เมื่อ

$$\forall i : Fwr_x[i] \not\bowtie Fwr_y[i] \text{และ} \exists i : Fwr_x[i] \subset Fwr_y[i] \wedge \exists j : Fwr_x[j] \supset Fwr_y[j]$$

$$\wedge\ Fwr_x[\text{ACT}] \neq Fwr_y[\text{ACT}] \wedge i \neq j$$

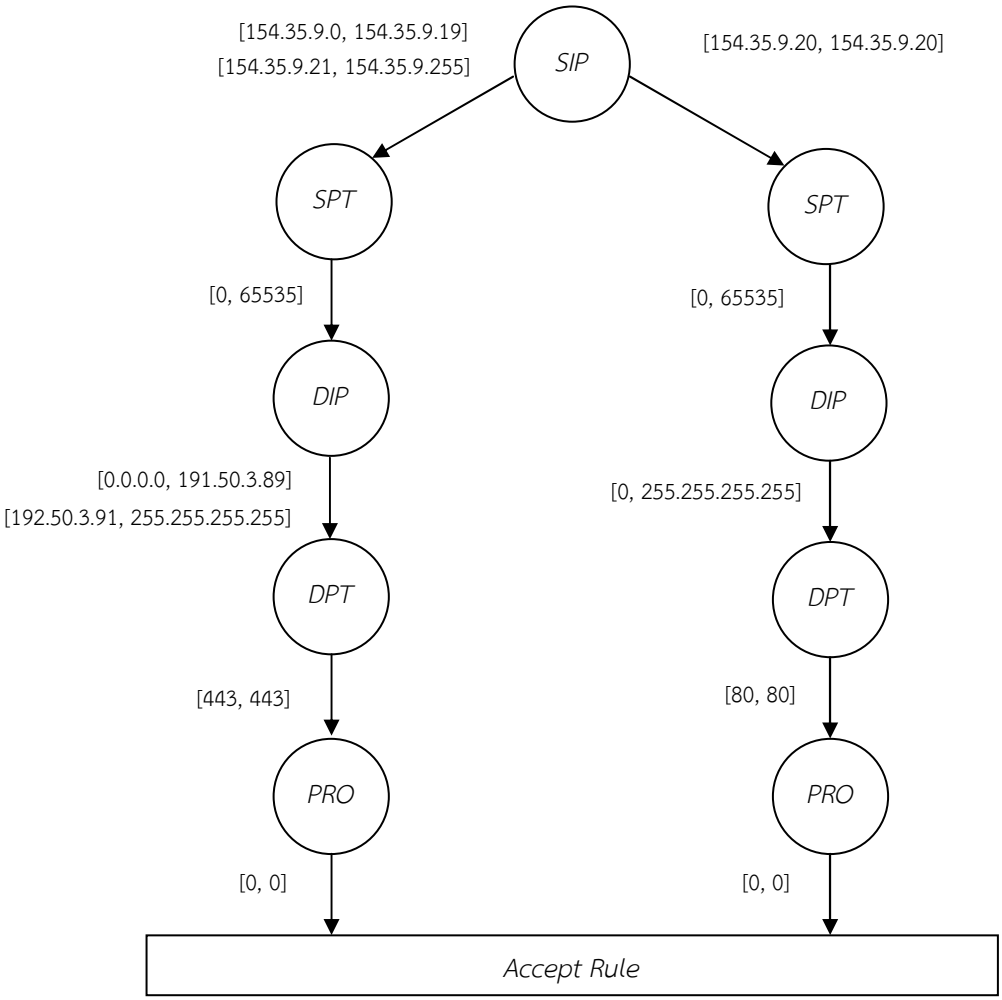
กำหนด $\not\bowtie \in \{\subset, \supset, =\}$, $i, j \in \{SIP, SPT, DIP, DPT, PRO\}$

การทดสอบว่ากฎใด เกิด Correlation Anomaly สามารถทำได้โดยทำการทดสอบว่ากฎทั้งสองเกิดการเกี่ยวข้องกันในบางฟิลด์ (Partially Correlation) เมื่อ Fwr_x และ Fwr_y เป็นกฎที่มีการเกี่ยวข้องกันในบางฟิลด์ เมื่อ Fwr_x ไม่เป็นสับเซตของ Fwr_y และ Fwr_y ไม่เป็นสับเซตของ Fwr_x และ Fwr_i ไม่เท่ากับ Fwr_y และ Fwr_x มีสมาชิกบางตัวเหมือนกับ Fwr_y แล้วไม่เป็นเซตว่างและผลของการกระทำของทั้งสองต่างกัน ดังตารางที่ 4.2

ตารางที่ 4.2 กฎของไฟร์วอลล์ที่เกิดวิกลภาพแบบ Correlation Anomaly

No	*SIP*	*SPT*	*DIP*	*DPT*	*PRO*	*ACT*
Fwr_1	154.35.9.0/24	all	all	443	TCP	Accept
Fwr_2	all	all	191.50.3.90	443	TCP	Deny
Fwr_3	154.35.9.20	all	all	80	TCP	Accept

จากตารางที่ 4.2 เป็นตัวอย่างของกฎไฟร์วอลล์ที่เกิดวิกลภาพแบบ Correlation Anomaly ซึ่งเมื่อเกิดกฎวิกลภาพ จะทำให้กฎของไฟร์วอลล์ที่ถูกเพิ่มเข้ามา มีบางส่วนที่ไม่มีทางถูกกระทำ ซึ่งจะได้ว่า Fwr_2 เป็นกฎที่มีความเกี่ยวพันธ์กับ Fwr_1 ซึ่งจะมีสมาชิกบางตัวไม่มีทางถูกกระทำนั้นคือช่วงของกฎที่มีหมายเลขไอพีต้นทาง (*SIP)* 154.35.9.0/24 ไปยังหมายเลขไอพีปลายทาง (*DIP*) 191.50.3.90 โดยใช้บริการหมายเลขพอร์ต 443 (HTTPS) ซึ่ง Fwr_2 ได้กำหนดผลของการกระทำเป็นปฏิเสธ แต่ว่าเงื่อนไขบางสมาชิกกฎนี้ไม่มีทางที่จะถูกกระทำ เนื่องจากเมื่อมีข้อมูลเข้ามาจะถูกกระทบกับ Fwr_1 ก่อน โดยจะอนุญาตให้ผ่านเข้าไปยังระบบเครือข่ายได้ ซึ่งจะทำให้เกิดกฎวิกลภาพและปัญหารูปแบบนี้ เป็นปัญหาวิกลภาพของกฎไฟร์วอลล์ที่แก้ปัญหาได้ยากที่สุด



รูปที่ 4.2 โครงสร้างไฟร์วอลล์ SDD เมื่อสร้างจากกฎที่เกิด Correlation Anomaly

จากรูปที่ 4.2 จะเห็นได้ว่าโครงสร้างของไฟร์วอลล์ *SDD* เมื่อสร้างจากกฎของไฟร์วอลล์ที่เป็นกฎวิกลภาพแบบ Correlation Anomaly จะทำให้กฎที่อยู่ภายในโครงสร้างของไฟร์วอลล์ *SDD* ปราศจากกฎวิกลภาพ จากตัวอย่างข้างบนที่กล่าวมา กฎของไฟร์วอลล์ในช่วง หมายเลขไอพีต้นทาง (*SIP*) 154.35.9.0/24 ไปยังหมายเลขไอพีปลายทาง (*DIP*) 191.50.3.90 ไม่ปรากฏบนโครงสร้างของไฟร์วอลล์แบบ *SDD* ซึ่งหลักการของไฟร์วอลล์ *SDD* เมื่อกฎใหม่มีความขัดแย้งกับกฎเดิม กฎเดิมจะถูกแก้ไขผลของการกระทำให้เท่ากับกฎใหม่ทันที ซึ่งหลักการเช่นนี้เป็นวิธีการหลักที่ทำให้กฎของไฟร์วอลล์ *SDD* ปราศจากกฎวิกลภาพ ซึ่งเชื่อการตัดสินใจองค์ความรู้ใหม่เช่นเดียวกับ Booth และคณะ [17] ที่ใช้แนวคิดการเปลี่ยนแปลงความเชื่อ (Belief Revision) อย่างไรก็ตาม ในการขจัดวิกลภาพของกฎไฟร์วอลล์โดยใช้โครงสร้าง *SDD* จะส่งผลให้กฎของไฟร์วอลล์จะมีการแตกกฎเพิ่มมากขึ้น ซึ่งทำให้ใช้เวลาในการผ่านกฎไฟร์วอลล์เพิ่มขึ้น

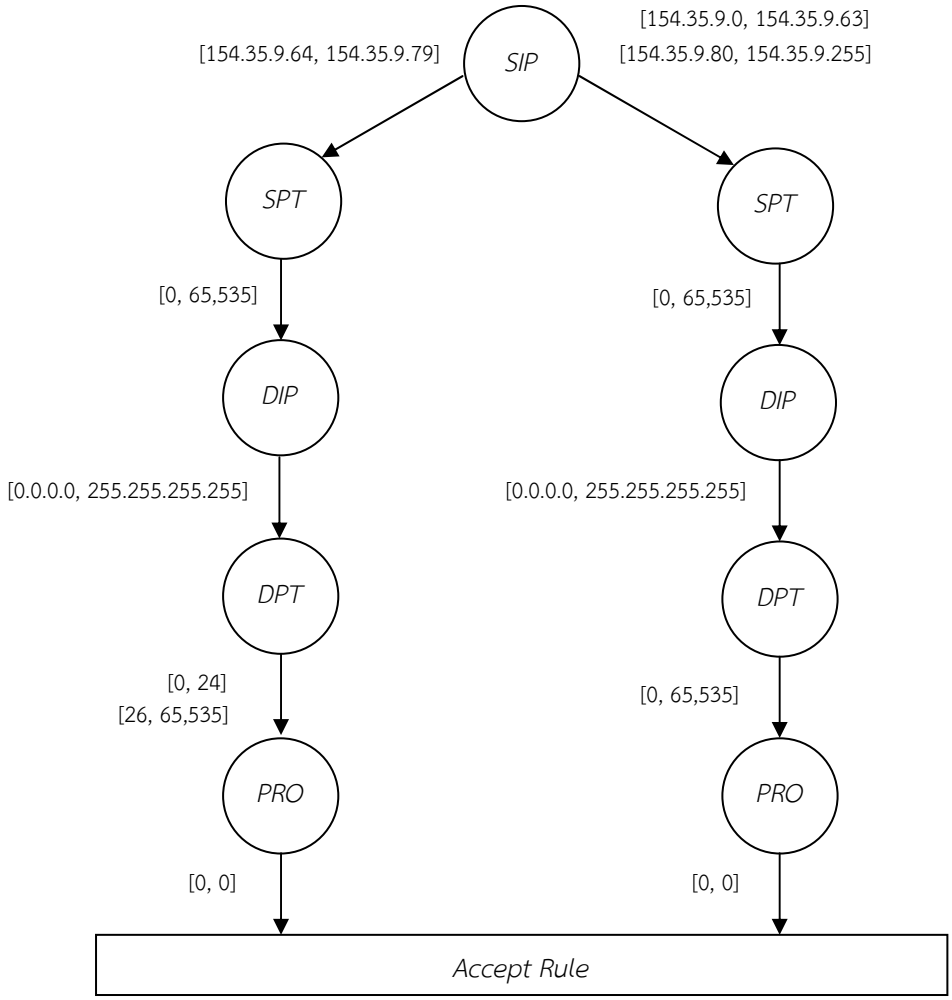
3. วิกลภาพของกฎไฟร์วอลล์แบบ Generalization Anomaly กฎใดๆ จะเป็นกฎวิกลภาพแบบ Generalization Rule เมื่อ ($Fwr_i \subset Fwr_{i+1}$) ^ (actions are different)

การค้นหากฎที่มีวิกลภาพแบบ Generalization มีการค้นหาเช่นเดียวกับ Correlation Anomaly ซึ่งต้องทดสอบกับกฎเป็นคู่ๆ ว่ามีความเกี่ยวข้องกันหรือไม่ โดยทดสอบจากการเกี่ยวข้องกันแบบบางฟิลด์ (Partially Correlation) ดังนั้นในทำนองเดียวกันการค้นหา Generalization Anomaly ก็ต้องทำการทดสอบกฎเป็นคู่ โดยทดสอบว่า กฎมีความเกี่ยวข้องกันแบบสมบูรณ์ (Complete Correlate) ดังตารางที่ 4.3

ตารางที่ 4.3 กฎของไฟร์วอลล์ที่เกิดวิกลภาพแบบ Generalization Anomaly

ลำดับ	*SIP*	*SPT*	*DIP*	*DPT*	*PRO*	*ACTION*
Fwr_1	154.35.9.64/28	all	all	80	TCP	Deny
Fwr_2	154.35.9.0/24	all	all	all	TCP	Accept
Fwr_3	154.35.9.64/28	all	all	25	TCP	Deny

จากตารางที่ 4.3 รูปแบบการเกิดกฎวิกลภาพ Generalization จะมีลักษณะคล้ายกับ Correlation ซึ่งจะแตกกันตรงที่สมาชิกทุกตัวของกฎที่เกิดการ Generalization เป็นสมาชิกของกฎที่เพิ่มเข้ามาทีหลังซึ่ง Correlation นั้นจะเป็นสมาชิกบางส่วนเท่านั้น ลักษณะตัวอย่างจะเห็นได้ว่า Fwr_1 เป็นกฎที่เกิดวิกลภาพกับ Fwr_2 ซึ่ง สมาชิกของ Fwr_2 ที่มีสมาชิกเช่นเดียวกับ Fwr_1 ($Fwr_2 \cap Fwr_1$) ไม่มีทางถูกกระทำ ตัวอย่างเช่น หมายเลขไอพีต้นทาง (*SIP*) 154.35.9.70 ไปยังหมายเลขไอพีปลายทาง (*DIP*) ทุกปลายทาง ผ่านการให้บริการพอร์ต 80 (HTTP) จากตารางที่ผู้ดูแลระบบได้เพิ่ม Fwr_2 เพื่อต้องการให้ หมายเลขไอพีต้นทาง 154.3.9.0/24 ใช้งาน ได้ทุกบริการ แต่ปรากฏว่ามี Fwr_1 ป้องกันการใช้งานพอร์ต 80 ซึ่งจะทำให้ผู้ใช้หมายเลขไอพี 154.35.9.70 ไม่สามารถใช้บริการพอร์ต 80 ได้เลย ซึ่งทำให้เกิดวิกลภาพของกฎไฟร์วอลล์เกิดขึ้น และจากกฎในตารางที่นี้สามารถสร้างไฟร์วอลล์ *SDD* เพื่อปราศจากวิกลภาพของกฎไฟร์วอลล์ได้ ดังรูปที่ 4.3



รูปที่ 4.3 โครงสร้างไฟร์วอลล์ *SDD* เมื่อสร้างจากกฎที่เกิด Generalization Anomaly

จากรูปที่ 4.3 โครงสร้างของไฟร์วอลล์ *SDD* ที่สร้างจากกฎที่มีเป็นกฎวิกลภาพแบบ Generalization จะมีลักษณะคล้ายกับ Correlation คือ กฎจะถูกแบ่งเพิ่มขึ้น เนื่องจากการที่กฎบางส่วนถูกแก้ไขเพื่อขจัดปัญหากฎวิกลภาพ เมื่อเปรียบเทียบกับรูปแบบของกฎไฟร์วอลล์แบบเดิมจะเห็นว่า Fwr_2 ของตารางที่ 4.3 ที่กำหนดให้ทุกหมายเลขไอพีต้นทาง (*SIP*) ของ 154.35.9.0/24 ต้องการใช้ทุกพอร์ตที่ให้บริการ ซึ่งในโครงสร้างของ *SDD* ได้ทำงานอย่างถูกต้องและ มีเพียงหมายเลขไอพีต้นทาง 154.35.9.64/28 ที่ไม่ให้บริการพอร์ต 25 ซึ่งเป็นกฎที่ถูกแตกเพิ่มออกไปเพื่อตรวจสอบอีกเงื่อนไข ดังรูปข้างต้นนี้

4. วิกลภาพของกฎไฟร์วอลล์แบบ Redundancy Anomaly กฎใดๆ จะเป็นกฎวิกลภาพแบบ Redundancy Rule เมื่อ $(Fwr_i \subset Fwr_{i+1})$ ^ (actions are similar)

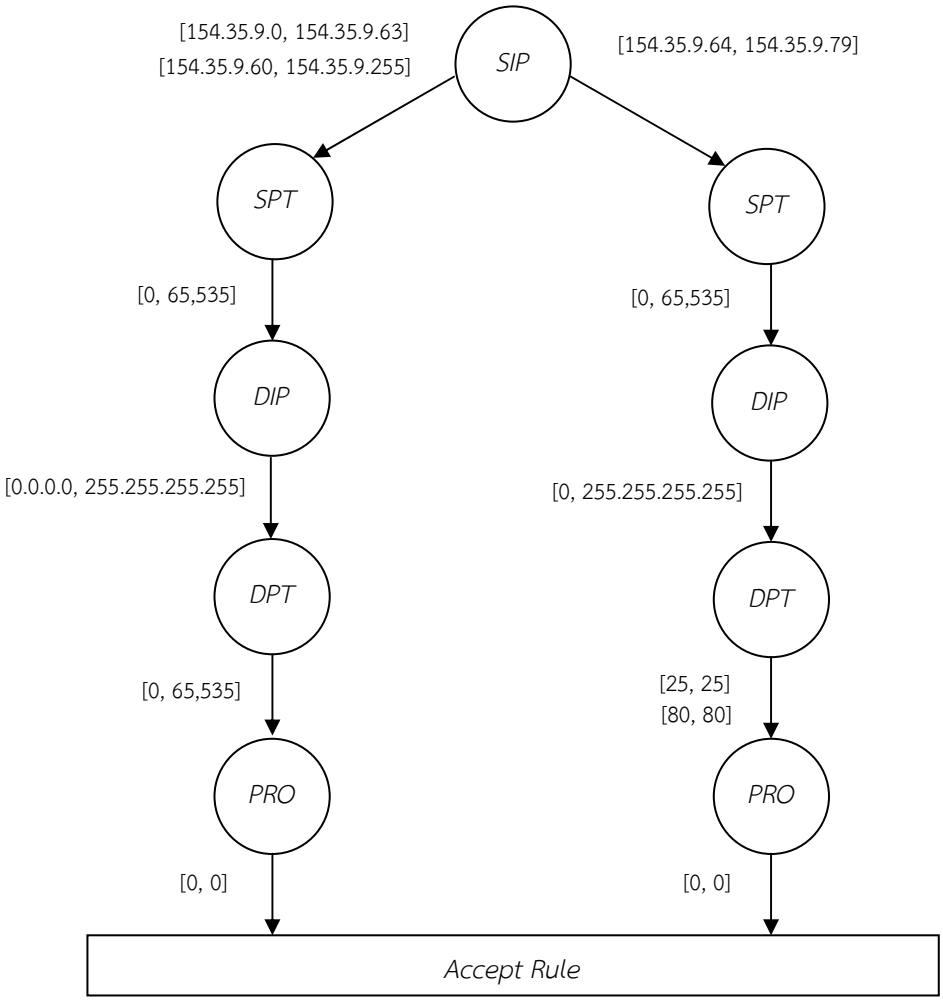
การค้นหากฎวิกลภาพแบบ Redundnacy ซึ่งเป็นการรูปแบบของกฎวิกลภาพที่มีผลกระทบต่อระบบเครือข่ายน้อยที่สุด มีลักษณะการค้นหาเช่นเดียวกับ Generalization Anomaly ซึ่งต้องทดสอบกับกฎเป็นคู่ๆ ว่ามีความเกี่ยวข้องกันหรือไม่ ในทำนองเดียวกันการหา Redundancy Anomaly ก็ต้องทดสอบกฎเป็นคู่ๆ เช่นกัน ว่ามีกฎใดที่ซ้ำซ้อน (Redundancy) ต่อกฎอื่นหรือไม่

Fwr_i จะเรียกว่า Redundant ต่อ Fwr_{i+1} เมื่อ Fwr_i เป็นสับเซตของ Fwr_{i+1} และผลของการกระทำของทั้งสองกฎเหมือนกัน ดังตารางที่ 4.4

ตารางที่ 4.4 กฎของไฟร์วอลล์ที่เกิดวิกลภาพแบบ Redundancy Anomaly

ลำดับ	***SIP***	***SPT***	***DIP***	***DPT***	***PRO***	***ACTION***
Fwr_1	154.35.9.64/28	all	all	80	TCP	Accept
Fwr_2	154.35.9.0/24	all	all	all	TCP	Accept
Fwr_3	154.35.9.64/28	all	all	25	TCP	Accept

จากตารางที่ 4.4 กฎของไฟร์วอลล์ที่เป็นวิกลภาพแบบซ้ำซ้อน (Redundancy Anomaly) กฎวิกลภาพรูปแบบนี้ส่งจะไม่ส่งผลเสียทางด้านความปลอดภัยของระบบเครือข่าย แต่จะส่งผลทางอ้อมต่อประสิทธิภาพการผ่านกฎของไฟร์วอลล์ อันเนื่องมาจากการที่กฎใดในไฟร์วอลล์มีผลของการกระทำที่ซ้ำซ้อนซึ่งเป็นการสร้างกฎอย่างฟุ่มเฟือย ทำให้ไฟร์วอลล์มีการผ่านกฎที่เพิ่มขึ้น ซึ่งจะเห็นว่า Fwr_2 เป็นเพียงกฎเดียวก็สามารถทดแทนการสร้างกฎของ Fwr_1 และ Fwr_3 ได้ซึ่งทำให้สามารถตัดกฎที่เหลือทิ้งไปได้ และความหมายของไฟร์วอลล์ไม่เปลี่ยนแปลงโดยที่ลักษณะของไฟร์กฎไฟร์วอลล์แบบซ้ำซ้อนมีรูปแบบคล้ายกับ Generalization เพียงแต่ผลของการกระทำจะต้องเหมือนกันเท่านั้นเอง ซึ่งในโครงสร้างของ *SDD* สามารถแก้ไขกฎวิกลภาพแบบซ้ำซ้อนได้ โดยสามารถสร้างโครงสร้างของไฟร์วอลล์ ดังรูปที่ 4.4



รูปที่ 4.4 โครงสร้างไฟร์วอลล์ *SDD* เมื่อสร้างจากกฎที่เกิด Redundancy Anomaly

จากรูปที่ 4.4 โครงสร้างของไฟร์วอลล์ *SDD* สามารถแก้ไขปัญหาความซ้ำซ้อนของกฎได้ดีในส่วนหนึ่งซึ่งในขั้นตอนการแก้ไขปัญหาความซ้ำซ้อน ถ้าหากกฎที่ถูกเพิ่มเป็นกฎที่มีขนาดของสมาชิกใหญ่ที่สุดก่อน จะทำให้กฎที่มาซ้อนซ้อนไม่ถูกเพิ่มเข้าไปในโครงสร้างไฟร์วอลล์นี้เลย แต่ถ้าหากกฎที่ถูกเพิ่มใหญ่เพิ่มขึ้นตามลำดับจะทำให้กฎของไฟร์วอลล์มีการแตกแยกเพิ่มขึ้น ซึ่งอาจจะทำให้กฎมีการแตกเพิ่มขึ้น อย่างไรก็ตามในการเข้าถึงข้อมูลภายในโครงสร้างของไฟร์วอลล์ *SDD* นี้มีการค้นหาแบบทวิภาค ซึ่งปัญหาการผ่านกฎของไฟร์วอลล์จะไม่ส่งผลกระทบแต่อย่างใด

4.1.2 เปรียบเทียบระหว่างไฟร์วอลล์ *SDD* กับไฟร์วอลล์แบบโครงสร้างต้นไม้

จากการแก้ไขปัญหากฎวิกลภาพในตัวอย่างที่ผ่านมาทั้งหมด โครงสร้างไฟร์วอลล์ *SDD* จะสามารถขจัดปัญหาวิกลภาพของไฟร์วอลล์ ได้ในทุกกรณี ซึ่งเมื่อเปรียบเทียบกับโครงสร้างของ

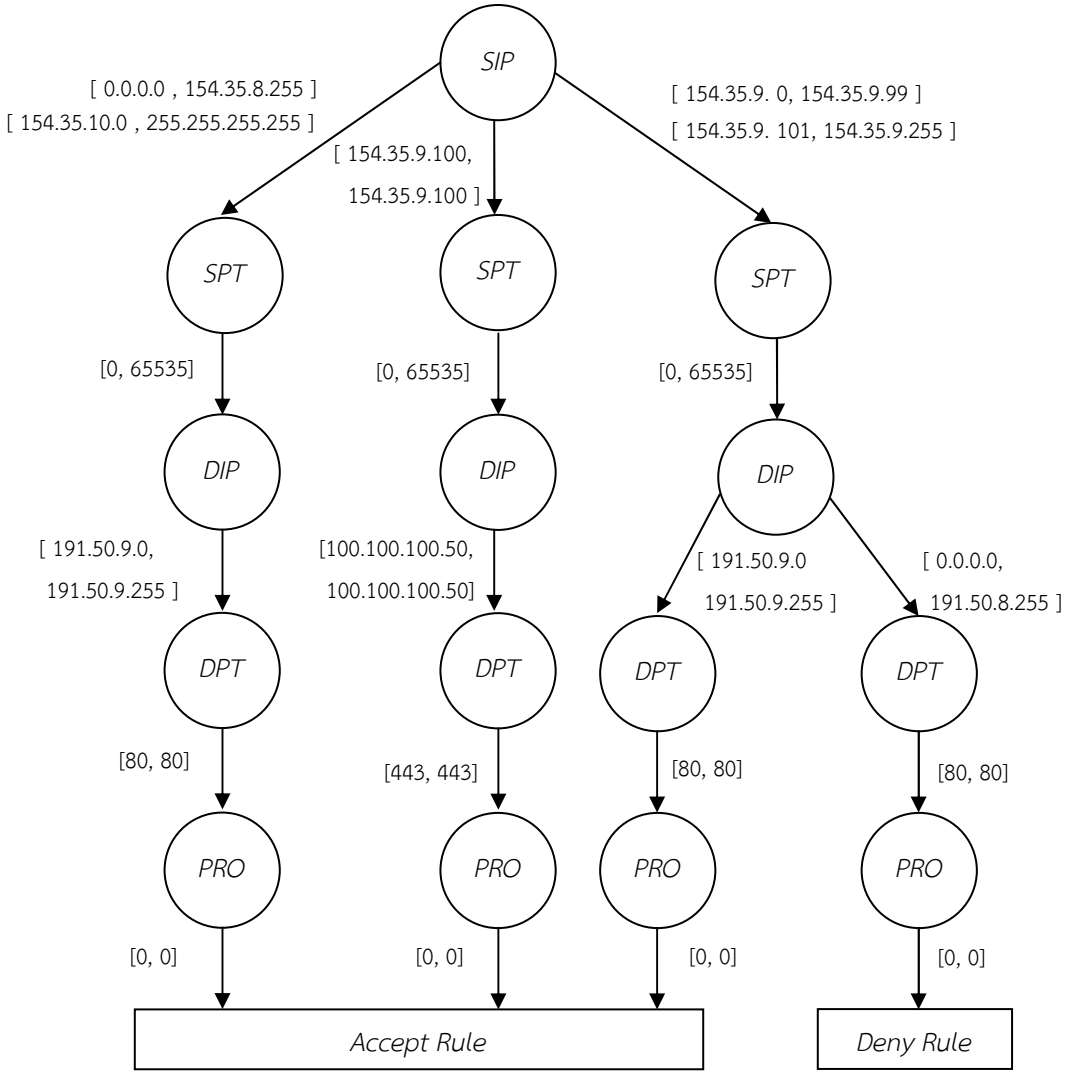
ไฟร์วอลล์แบบดั้งเดิม แต่จะเห็นว่าขั้นตอนในการแก้ไขปัญหาของไฟร์วอลล์ *SDD* จะเป็นการเชื่อกฎที่เพิ่มเข้ามาใหม่ แต่ถ้าหากในการจัดการกฎหรือการเพิ่มกฎที่เกิดจากความรู้เท่าไม่ถึงการณ์ของผู้ดูแลระบบ ก็ยังคงเป็นปัญหาที่สร้างความผิดพลาดให้กับองค์กร ดังนั้นในขั้นตอนต่อไปนี้ ก่อนที่จะนำเสนอวิธีการแก้ไขการเพิ่มกฎจากความรู้เท่าไม่ถึงการณ์ของผู้ดูแลระบบ ผู้วิจัยจะนำเสนอการเปรียบเทียบวิธีการแก้ไขปัญหาวิกลภาพของกฎไฟร์วอลล์ที่มีลักษณะคล้ายกับ *SDD* ซึ่งเป็นการประยุกต์งานวิจัยของ Lui [3] ที่ใช้คุณสมบัติโครงสร้างต้นไม้ (Tree based Structure) ร่วมกับแนวคิดการเปลี่ยนแปลงความเชื่อใหม่ (Belief Revision) [17] โดยผู้วิจัยเรียกว่า ไฟร์วอลล์แบบต้นไม้ประยุกต์ ซึ่งมีลักษณะคล้ายกับโครงสร้างไฟร์วอลล์ *SDD* แตกต่างตรงที่ *SDD* เป็นแนวคิดการสร้าง

ไฟร์วอลล์ที่มีการตัดสินใจแบบโดเมนเดียว ซึ่งมีการเปรียบเทียบโครงสร้างของไฟร์วอลล์ได้โดยใช้กฎของไฟร์วอลล์ ดังตารางที่ 4.5

ตารางที่ 4.5 กฎของไฟร์วอลล์ที่เกิดวิกลภาพในทั้ง 4 รูปแบบ

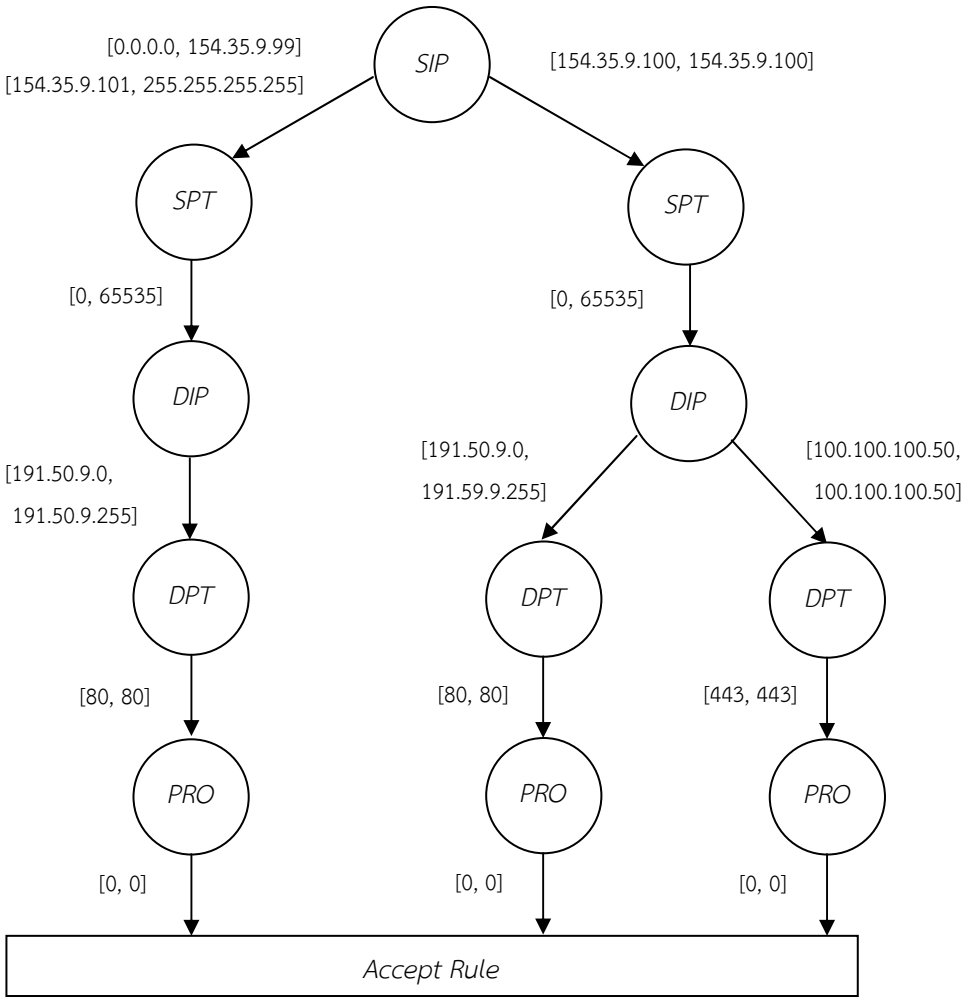
ลำดับ	*SIP*	*SPT*	*DIP*	*DPT*	*PRO*	*ACTION*
Fwr_1	154.35.9.64/28	all	all	80	TCP	Accept
Fwr_2	154.35.9.0/24	all	all	80	TCP	Deny
Fwr_3	all	all	191.50.9.0/24	80	TCP	Accept
Fwr_4	154.35.9.100	all	100.100.100.50	443	TCP	Accept

จากตารางที่ 4.5 จะพบว่ามีกฎวิกลภาพของไฟร์วอลล์ทั้ง 4 รูปแบบดังต่อไปนี้ ซึ่ง Fwr_1 จะมีกฎวิกลภาพแบบ Generalization Anomaly กับ $Fwr_{2\,โดย}$ที่ Fwr_1 มีสมาชิกทุกตัวอยู่ใน Fwr_2 โดยที่ทั้งสองกฎนี้มีผลของการกระทำที่แตกต่างกัน และ Fwr_2 ยังเป็นกฎที่มีความเกี่ยวพันกับ Fwr_3 โดยที่ทั้งสองกฎนี้มีผลของการกระทำที่แตกต่างกันซึ่งเรียกว่าเป็น Correlation Anomaly นอกจากนี้ยังมีกฎวิกลภาพแบบ Shadowing Anomaly ที่เกิดขึ้นกับ Fwr_2 และ Fwr_5 ในกรณี Redundancy Anomaly ที่เกิดกับ Fwr_2 และ Fwr_4 ซึ่งจะเห็นได้ว่ากฎของไฟร์วอลล์ทั้งหมดนี้จะมีกฎวิกลภาพเกิดขึ้นทั้ง 4 รูปแบบ ถ้านำไปใช้บนโครงสร้างของไฟร์วอลล์แบบดั้งเดิม แต่ในการทดลองนี้เราจะใช้นำมาเปรียบเทียบกับโครงสร้างของไฟร์วอลล์แบบต้นไม้ประยุกต์ที่ใช้โครงสร้างแผนผังของต้นไม้และการเปลี่ยนแปลงความเชื่อ ซึ่งสามารถสร้างเป็นโครงสร้างของไฟร์วอลล์ได้ดังรูปที่ 4.5



รูปที่ 4.5 โครงสร้างไฟร์วอลล์แบบต้นไม้ประยุกต์หลังจากขจัดวิกลภาพของกฎไฟร์วอลล์

จากรูปที่ 4.5 จะเห็นว่าโครงสร้างของไฟร์วอลล์แบบต้นไม้ประยุกต์จะมีลักษณะคล้ายคลึงกับโครงสร้างไฟร์วอลล์ *SDD* ซึ่งเมื่อเจอกฎที่มีวิกลภาพกฎของไฟร์วอลล์ก็จะแตกเพิ่มขึ้น แต่ผลการของกระทำของกฎไฟร์วอลล์นี้จะมีทั้งสองกรณีด้วยกันคือทั้ง อนุมัติ (Accept) กับ ปฏิเสธ (Deny) ซึ่งในกรณีไฟร์วอลล์โครงสร้างแบบ *SDD* จะสนใจแค่กรณีเดียวเท่านั้น สามารถสร้างเป็นโครงสร้างของไฟร์วอลล์ได้ ดังรูปที่ 4.6



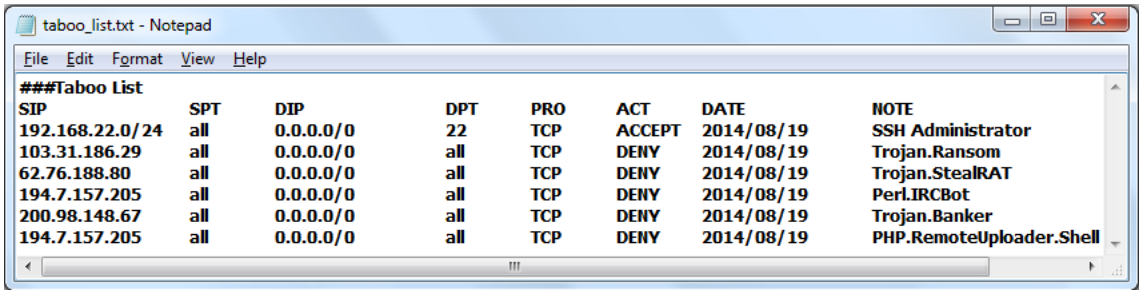
รูปที่ 4.6 โครงสร้างไฟร์วอลล์ *SDD* หลังจากขจัดวิกลภาพของกฎไฟร์วอลล์

จากรูปที่ 4.6 โครงสร้างของไฟร์วอลล์ *SDD* เมื่อเปรียบเทียบกับโครงสร้างไฟร์วอลล์แบบต้นไม้ประยุกต์ จะเห็นว่า ไฟร์วอลล์ *SDD* เป็นส่วนหนึ่งของไฟร์วอลล์แบบต้นไม้ประยุกต์ ที่สนใจเฉพาะผลการกระทำที่ อนุมัติเท่านั้น ซึ่งทำให้ประสิทธิภาพในการผ่านกฎของไฟร์วอลล์ *SDD* ทำงานได้ดีกว่าไฟร์วอลล์แบบต้นไม้ประยุกต์ โดยที่ประสิทธิภาพในความปลอดภัยเท่ากัน ดังนั้นไฟร์วอลล์ *SDD* จึงมีประสิทธิภาพโดยรวมที่ดีกว่าประสิทธิภาพของไฟร์วอลล์แบบต้นไม้ประยุกต์

4.1.3 ผลกระทบต่อการแก้ไขปัญหากฎวิกลภาพ

การแก้ไขปัญหาความผิดปรกติของกฎไฟร์วอลล์โดยใช้แนวความคิดการตัดสินใจแบบโดเมนเดี่ยวจะทำให้กฎของไฟร์วอลล์ปราศจากกฎปฏิเสธ (Deny Rule) ดังนั้นถึงแม้ว่าผู้ใช้ระบบจะเพิ่มกฎที่ไม่มีวิกลภาพบนระบบไฟร์วอลล์ แต่หากเพิ่มกฎที่ผิดพลาดอันเกิดจากความรู้เท่าไม่ถึงการณ์หรือกฎที่เพิ่มเป็นกฎที่ผิดซึ่งจะทำให้ไฟร์วอลล์ทำงานไม่สอดคล้องกับความขัดแย้งเช่นเดิม

เนื่องจากคุณสมบัติของกฎปฏิเสธแน่ชัด สร้างไว้เพื่อป้องกันผู้ดูแลระบบเปิดใช้ให้แพ็กเก็ตที่ไม่พึงประสงค์ผ่านเข้าออกเครือข่าย ในงานวิจัยนี้จึงได้นำเสนอวิธีการแก้ไขปัญหากฎวิกลภาพที่อาจเกิดจากความรู้เท่าไม่ถึงการณ์โดยจะต้องตรวจสอบกฎที่จะเพิ่มกับต้องห้าม (Taboo List) ก่อนที่จะเพิ่มกฎทุกครั้งดังรูปที่ 4.7



รูปที่ 4.7 ตัวอย่างกฎไฟร์วอลล์ต้องห้าม

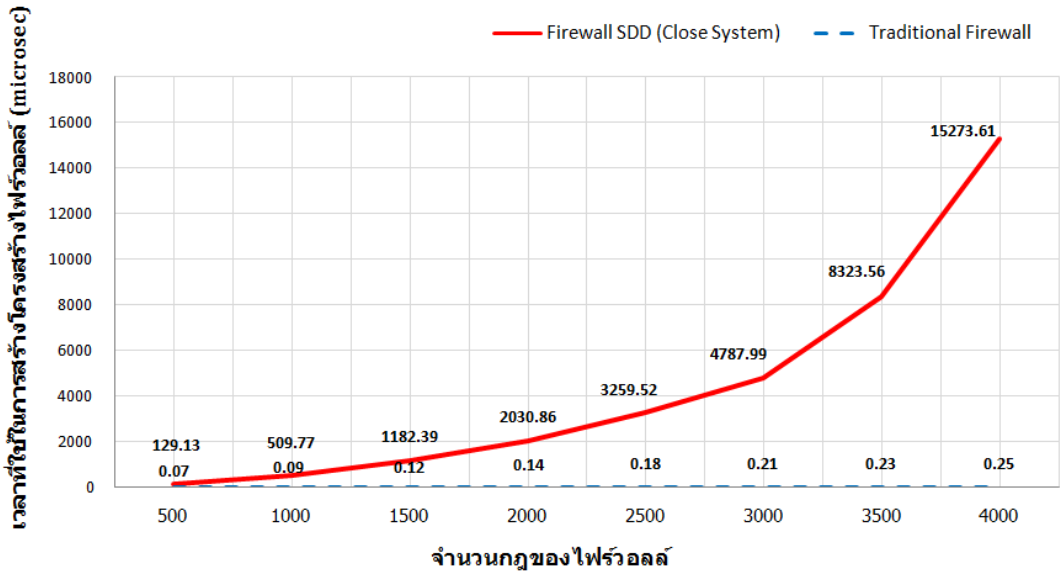
จากรูปที่ 4.7 กฎต้องห้ามจะมีการตรวจสอบทุกครั้งก่อนที่จะมีการเพิ่มกฎเข้าไปในระบบไฟร์วอลล์ เนื่องจากต้องการตรวจสอบว่ากฎที่เพิ่มเข้าไปในระบบ จะต้องไม่เป็นกฎที่ส่งผลกระทบต่อระบบเครือข่ายซึ่งกฎต้องห้ามนี้ อาจจะถูกกำหนดโดยผู้ดูแลระบบที่มีประสบการณ์หรือบุคคลที่มีความรับผิดชอบต่อระบบเครือข่ายได้โดยมีรูปแบบการทำงานดังตารางที่ 4.6

ตารางที่ 4.6 รายละเอียดข้อมูลของกฎต้องห้าม

ชื่อคอลัมน์	ความหมาย	รายละเอียด
SIP	หมายเลขไอพีต้นทาง	รูปแบบการกำหนด
SPT	หมายเลขพอร์ตต้นทาง	กำหนดหมายเลขพอร์ตตั้งแต่ 0 – 65535 หรือ all ทั้งหมด
DIP	หลายเลขไอพีปลายทาง	191.50.9.0/24
DPT	หมายเลขพอร์ตปลายทาง	กำหนดหมายเลขพอร์ตตั้งแต่ 0 – 65535 หรือ all ทั้งหมด
PRO	โพรโทคอล	กำหนดเป็นตัวย่อโพรโทคอลเช่น TCP หรือ UDP เป็นต้น
ACT	ผลของการกระทำ	กำหนดผลของการกระทำเป็น ACCEPT หรือ DENY
DATE	วันที่เพิ่มกฎต้องห้าม	วันที่เพิ่มกฎรูปแบบ yyyy/mm/dd เช่น 2014/09/01
DESCRIPTION	รายละเอียด	ใส่รายละเอียดของกฎต้องห้าม

4.2 ระยะเวลาที่ใช้ในการสร้างไฟร์วอลล์ SDD และไฟร์วอลล์แบบดั้งเดิม

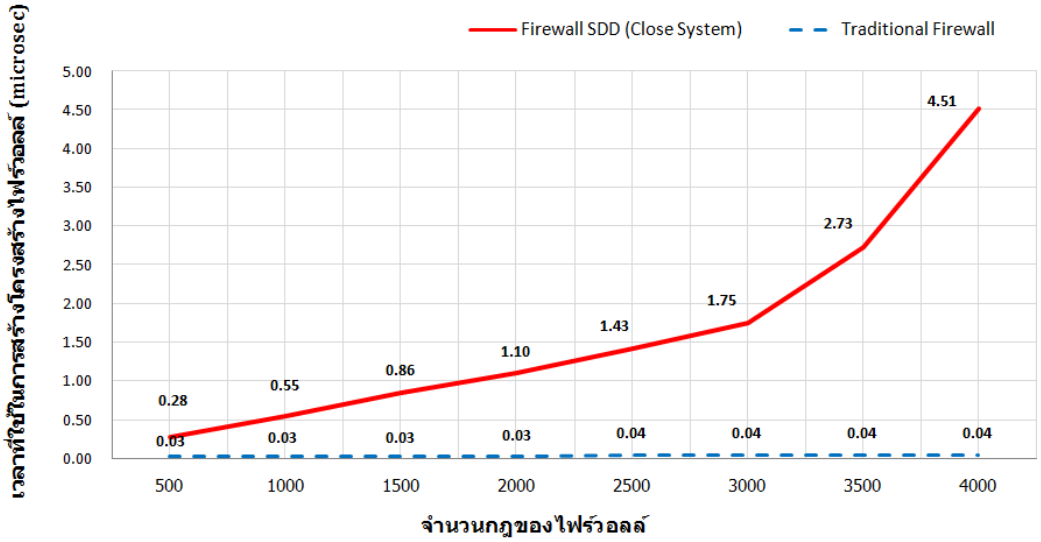
การปรับปรุงหรือแก้ไขกฎของไฟร์วอลล์ โดยมีการเปลี่ยนแปลงตามนโยบายของระบบเครือข่ายนั้นๆ ซึ่งจะทำให้โครงสร้างของไฟร์วอลล์มีการเปลี่ยนแปลง และในการเปลี่ยนแปลงโครงสร้างของไฟร์วอลล์นั้นจำเป็นต้องมีระยะเวลาในการสร้างกฎของไฟร์วอลล์ ซึ่งในการทดลองนี้ได้เปรียบเทียบความเร็วในการสร้างโครงสร้างของไฟร์วอลล์ *SDD* กับไฟร์วอลล์แบบดั้งเดิม (Traditional Firewall)



รูปที่ 4.8 ประสิทธิภาพในการสร้างกฎไฟร์วอลล์ครั้งละหลายกฎ

โครงสร้างของไฟร์วอลล์ *SDD* จะเป็นโครงสร้างของไฟร์วอลล์แบบไม่คงที่ (Dynamic Structure) การเพิ่มกฎของไฟร์วอลล์จะขึ้นอยู่กับความสัมพันธ์ของกฎในระบบที่จะส่งอาจมีการแบ่งเพิ่มขึ้น ดังนั้นแต่ละเงื่อนไขของกฎจะสามารถแบ่งเพิ่มขึ้นได้มากที่สุดเท่ากับ $2n-1$ โดยที่ n คือจำนวนของกฎไฟร์วอลล์ จากรูปที่ 4.8 เมื่อเปรียบเทียบระยะเวลาในการสร้างกฎของไฟร์วอลล์ *SDD* กับไฟร์วอลล์แบบดั้งเดิม (Traditional Firewall) โดยกำหนดให้เพิ่มกฎเข้าระบบตั้งแต่ 500 ไปจนถึง 4,000 กฎตามลำดับ จะเห็นว่าในระบบไฟร์วอลล์แบบดั้งเดิมจำนวนกฎไม่ส่งผลกระทบต่อระยะเวลาในการสร้างกฎ แต่ไฟร์วอลล์ *SDD* เมื่อเพิ่มกฎเข้าในระบบ ช่วงของกฎตั้งแต่ 3,000 ขึ้นไประยะเวลาในการสร้างกฎจะใช้เพิ่มขึ้นเป็นเท่าตัว และกรณีที่กฎไฟร์วอลล์มีจำนวน 4,000 กฎจะใช้เวลาในการสร้างกฎประมาณ 15 วินาที การทดลองนี้ได้นำกฎที่มีการเพิ่มเข้าระบบไฟร์วอลล์โดยกำหนดให้กฎที่ถูกเพิ่มเป็นกฎที่มีการแบ่งตัวมากที่สุด ดังนั้นจะสามารถสรุปได้ว่าในการสร้างกฎไฟร์วอลล์จำนวน 4,000 กฎในระบบเครือข่ายจริงจะใช้เวลาไม่เกิน 15 วินาที ซึ่งเวลาดังกล่าวเป็นเวลาที่ใช้ในการสร้างกฎทั้งหมดหรือ

เปรียบเสมือนการเพิ่มกฎเข้าใหม่ทีละหลายกฎพร้อม ซึ่งระยะเวลาดังกล่าวเป็นที่ยอมรับได้เนื่องจากในการปรับปรุงและแก้ไขกฎของไฟร์วอลล์ จะกระทำไม่บ่อยขึ้นอยู่กับการเปลี่ยนแปลงนโยบายการรักษาความปลอดภัยของระบบเครือข่ายขององค์กรนั้นๆ



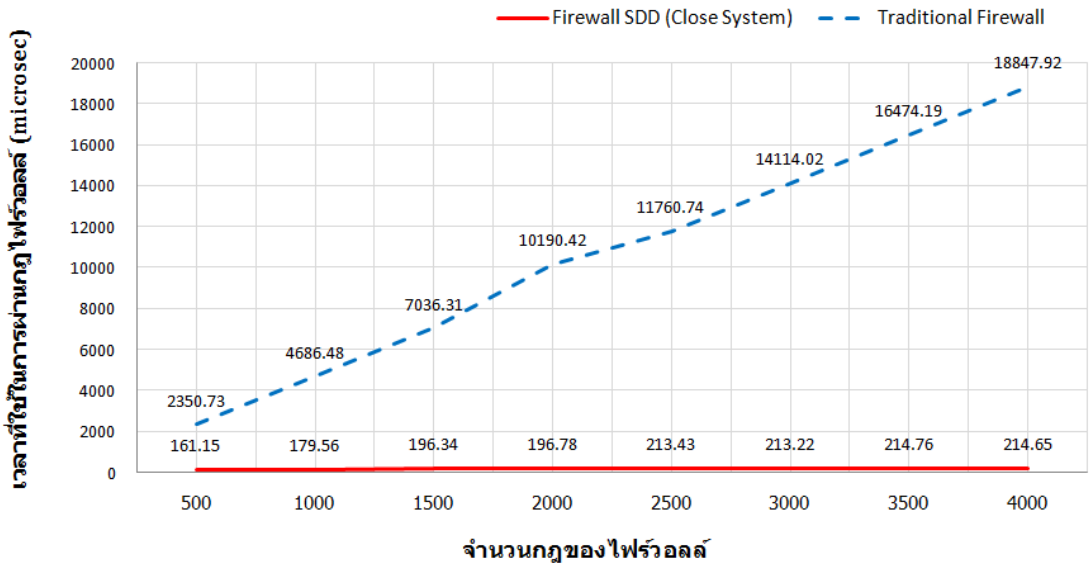
รูปที่ 4.9 เปรียบเทียบประสิทธิภาพในการสร้างกฎไฟร์วอลล์ในหนึ่งกฎ

จากรูปที่ 4.9 ในการแก้ไขหรือเพิ่มกฎในระบบไฟร์วอลล์โดยที่มีอยู่กฎแล้วในระบบซึ่งจะเป็นการเพิ่มกฎหนึ่งกฎดังนั้น ระยะเวลาที่ใช้จะแตกต่างจากรูปที่ 4.8 ซึ่งจะเห็นว่าในการเพิ่มกฎไฟร์วอลล์ซึ่งมีกฎอยู่แล้วในระบบและเพิ่มกฎในลำดับที่ 500 จะใช้เวลา 0.28 ไมโครวินาทีและเพิ่มกฎไฟร์วอลล์ลำดับที่ 4,000 จะใช้เวลาเพียง 4.51 ไมโครวินาที ดังนั้นถึงแม้ว่าระยะเวลาที่ใช้ในการสร้างกฎไฟร์วอลล์ *SDD* จะมากกว่าไฟร์วอลล์แบบดั้งเดิม (Traditional Firewall) แต่ระยะเวลาดังกล่าวเป็นที่ยอมรับได้ในทางปฏิบัติ เนื่องจากการปรับปรุงและแก้ไขกฎไฟร์วอลล์จะทำได้ก็ต่อเมื่อระบบเครือข่ายมีการเปลี่ยนแปลงนโยบายการรักษาความปลอดภัยขององค์กรซึ่งจะกระทำไม่บ่อย

4.3 ระยะเวลาที่ใช้ในการผ่านกฎของไฟร์วอลล์ *SDD* และไฟร์วอลล์แบบดั้งเดิม

การเข้าถึงข้อมูลของไฟร์วอลล์ *SDD* ใช้หลักการเข้าถึงข้อมูลแบบทวิภาค (Binary Search) ซึ่งมีประสิทธิภาพการเข้าถึงข้อมูลเท่ากับ $O(k \times d \log(2n-1))$ โดยที่ d คือจำนวนเงื่อนไขของกฎไฟร์วอลล์ n คือจำนวนกฎของไฟร์วอลล์และ k คือค่าคงที่ในการตรวจสอบขอบเขตของช่วงข้อมูลในแต่ละครั้ง และเมื่อเทียบกับไฟร์วอลล์แบบดั้งเดิมที่มีการเข้าถึงข้อมูลแบบตามลำดับ (Sequential

Search) มีประสิทธิภาพการเข้าถึงข้อมูลแบบ $O(d \times n)$ โดยที่ d เท่ากับจำนวนเงื่อนไขที่ใช้ในการตรวจสอบของแต่ละกฎ n คือ จำนวนกฎทั้งหมดในไฟร์วอลล์ และ k คือ หน่วยเวลาที่ใช้ในการตรวจสอบความสัมพันธ์แบบช่วง ซึ่งจะส่งผลให้ประสิทธิภาพของไฟร์วอลล์ *SDD* มีการเข้าถึงข้อมูลได้เร็วกว่าไฟร์วอลล์แบบดั้งเดิมดังรูปที่ 4.10



รูปที่ 4.10 ระยะเวลาในการผ่านกฎของไฟร์วอลล์ *SDD* กับไฟร์วอลล์แบบดั้งเดิม

จากรูปที่ 4.10 จะเห็นได้ว่า ค่าเวลาเฉลี่ยที่ใช้ในผ่านกฎระหว่าง SDDและไฟร์วอลล์แบบดั้งเดิมซึ่งแกนนอนของกราฟประกอบไปด้วยจำนวนของกฎตั้งแต่ 500 – 4,000 และแนวตั้งแสดงเวลาในการผ่านกฎโดยเฉลี่ยตั้งแต่ 0 – 20,000 (ไมโครวินาที) จากกราฟสังเกตว่าช่วงจำนวนของกฎที่ใช้ในการผ่านกฎของไฟร์วอลล์แบบดั้งเดิมจะเพิ่มสูงขึ้นเป็นเชิงเส้น ในทางตรงกันข้าม *SDD* ใช้ระยะเวลาเฉลี่ยในการผ่านกฎเพิ่มขึ้นจากเดิมเพียงเล็กน้อย

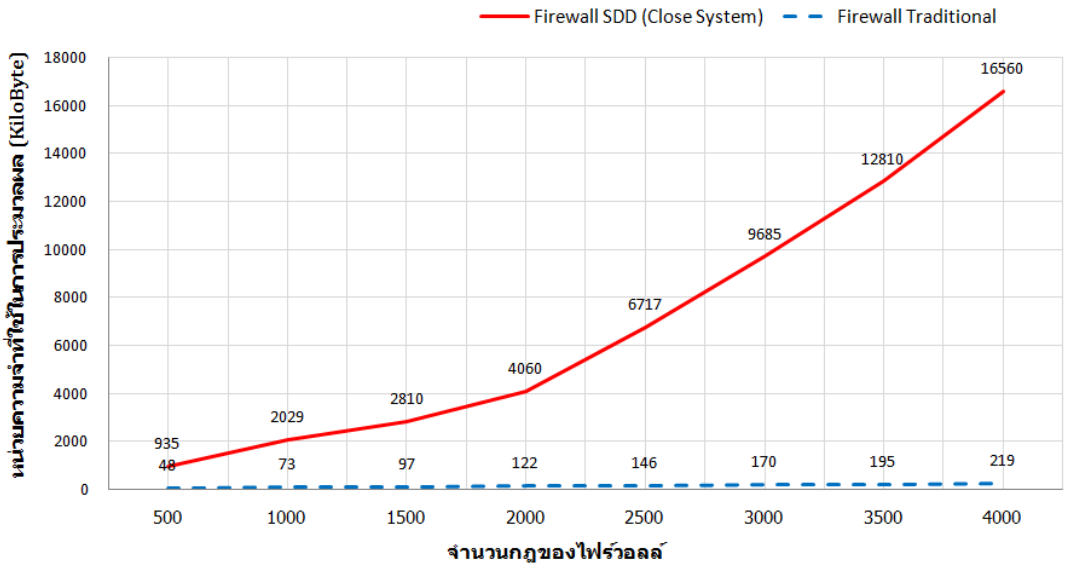
อย่างไรก็ตามประสิทธิภาพในการผ่านกฎของไฟร์วอลล์แบบดั้งเดิมจะมีประสิทธภาพที่ดีกว่าในกรณีที่แพ็กเก็ตกระทบกับกฎลำดับต้นของระบบไฟร์วอลล์ ซึ่งเป็นกรณีที่เกิดขึ้นไม่บอกนัก ในทางตรงกันข้ามหากแพ็กเก็ตที่กระทบกฎลำดับล่างของไฟร์วอลล์แบบดั้งเดิม จะทำให้ประสิทธิภาพการผ่านกฎของไฟร์วอลล์แบบดั้งเดิมลดลงอย่างมาก แต่จะไม่ส่งผลกระทบต่อการผ่านกฎไฟร์วอลล์ *SDD* ดังนั้นจะเห็นว่าประสิทธิภาพการผ่านกฎของไฟร์วอลล์ *SDD* กับแพ็กเก็ตที่สุ่มโดยทั่วไปสอดคล้องกับค่าสัญกรโอใหญ่

และถึงแม้ว่าเวลาที่ใช้ในการสร้างโครงสร้างของไฟร์วอลล์ *SDD* จะใช้เวลามากกว่าโครงสร้างของไฟร์วอลล์แบบดั้งเดิม (Traditional Firewall) อย่างไรก็ตาม สิ่งที่สำคัญของการทำงานไฟร์วอลล์คือการทำหน้าที่ในการเข้าถึงข้อมูลอย่างรวดเร็ว ซึ่งเป็นส่วนสำคัญที่สุดเพื่อให้ข้อมูลที่การผ่านเข้าออกของ

ระบบเครือข่ายมีประสิทธิภาพที่ดีขึ้น และขณะเดียวกัน ในการจัดการกฎของไฟร์วอลล์หรือการแก้ไขโครงสร้างไฟร์วอลล์ โดยส่วนใหญ่ผู้ดูแลระบบจะใช้ไม่มีการปรับเปลี่ยนกฎของไฟร์วอลล์บ่อยมากนัก เพราะการที่ปรับเปลี่ยนกฎของไฟร์วอลล์จะมีทำก็ต่อเมื่อระบบเครือข่ายมีการเปลี่ยนนโยบายในการให้บริการข้อมูลต่างเท่านั้น ซึ่งระยะเวลาที่ใช้ในการสร้างกฎของไฟร์วอลล์ *SDD* ในกรณีที่สร้างกฎจำนวน 4,000 กฎโดยใช้เวลาทั้งหมดประมาณ 15 วินาที ถือว่าเป็นที่ยอมรับได้ในการทำงาน

4.4 หน่วยความจำที่ใช้ในการประมวลผลไฟร์วอลล์ *SDD* และไฟร์วอลล์แบบดั้งเดิม

พื้นที่หน่วยความจำที่ใช้ในการประมวลผลโครงสร้างของไฟร์วอลล์ *SDD* ซึ่งไม่สามารถกำหนดขอบเขตได้อย่างชัดเจน เนื่องจากโครงสร้างของไฟร์วอลล์ *SDD* มีการเพิ่มและลดขนาดซึ่งไม่ได้ขึ้นอยู่กับจำนวนของกฎ แต่ขึ้นอยู่กับความสัมพันธ์ของกฎไฟร์วอลล์ ซึ่งเมื่อกฎมีความสัมพันธ์ที่ซับซ้อน จะทำให้โครงสร้างของไฟร์วอลล์ *SDD* มีการแบ่งตัวของกฎในแต่ละเงื่อนไขการตรวจสอบ มากที่สุดเท่ากับ $2n-1$ ในแต่ละฟิลด์เงื่อนไข โดยเราสามารถใช้ตัวเลขจำนวนนี้กำหนดของเขตที่มากที่สุดในการใช้พื้นที่หน่วยความจำได้ ซึ่งต่างจากโครงสร้างของไฟร์วอลล์แบบดั้งเดิมที่ใช้โครงสร้างข้อมูลแบบอาเรย์โดยใช้พื้นที่หน่วยความจำเท่ากับ $O(n)$ โดยที่ n คือ จำนวนกฎของไฟร์วอลล์ทั้งหมด ดังรูปที่ 4.10



รูปที่ 4.11 เปรียบเทียบหน่วยความจำของไฟร์วอลล์ *SDD* กับไฟร์วอลล์ดั้งเดิม

จากรูปที่ 4.11 สำหรับกรณีของประมาณการใช้พื้นที่หน่วยความจำระหว่างไฟร์วอลล์ทั้งสองจะแตกต่างกันมาก ซึ่งในโครงสร้างแบบดั้งเดิมจะมีการใช้หน่วยความจำที่เพิ่มมาเพียงเล็กน้อย แต่ในโครงสร้างของไฟร์วอลล์แบบ *SDD* มีแนวโน้มการใช้หน่วยความจำที่สูงขึ้นเมื่อกฎที่สร้างมีจำนวนมากขึ้นตามลำดับ และจะเห็นว่าช่วงกฎ 2,000 – 4,000 มีการใช้ช่วยความจำที่เพิ่มสูงอันเนื่องมาจากกฎที่เพิ่ม

เข้าในระบบไฟร์วอลล์เกิดกฎวิกลภาพและมีการแบ่งตัวเพิ่มสูงขึ้น อย่างไรก็ตามถึงแม้ว่าระบบไฟร์วอลล์ *SDD* มีการใช้หน่วยความจำมากกว่าระบบไฟร์วอลล์แบบดั้งเดิมหลายเท่าตัว แต่หน่วยความจำที่ใช้ของทั้งสองมีความเพียงพอต่ออุปกรณ์ที่ใช้อยู่ในปัจจุบัน

ในความเป็นจริงแล้วประสิทธิภาพของไฟร์วอลล์ที่ดีจะขึ้นอยู่กับระยะเวลาที่ใช้ในการผ่านกฎ ซึ่งจะมีความสำคัญเป็นอันดับแรก และตามปกติไฟร์วอลล์จะไม่ปรับปรุงกฎบ่อย หากไม่มีความจำเป็น

บทที่ 5

สรุปผล อภิปรายผล และข้อเสนอแนะ

5.1 สรุปผลและอภิปรายผล

ไฟร์วอลล์เป็นอุปกรณ์ที่ได้รับความนิยมอย่างแพร่หลายเพื่อใช้ในการป้องกันการรักษาความปลอดภัย การทำงานของไฟร์วอลล์ขึ้นอยู่กับกฎไฟร์วอลล์ซึ่งหากมีการออกแบบและจัดการกฎของไฟร์วอลล์ไม่ดี จะทำให้กฎของไฟร์วอลล์เกิดกฎวิกลภาพและส่งผลต่อประสิทธิภาพความปลอดภัยของระบบเครือข่าย เช่น อาจทำให้ระบบเครือข่ายขององค์กรมีช่องโหวเกิดขึ้นหรือมีการปิดกั้นการจราจรของข้อมูลที่ถูกต้อง ในงานวิจัยของ Al-Shaer และคณะ [1] ได้ให้คำนิยามและความหมายของวิกลภาพของกฎไฟร์วอลล์ไว้ 4 รูปแบบด้วยกัน ซึ่งต่อมาได้มีหลายงานวิจัยได้นำเสนอแนวคิดที่ใช้ในการแก้ไขปัญหาวิกลภาพของไฟร์วอลล์ โดยงานวิจัยของ Chomsiri [2] ได้นำเสนอวิธีการขั้นตอนการตรวจสอบวิกลภาพของกฎไฟร์วอลล์ด้วยใช้หลักพีชคณิตเชิงสัมพันธ์ (Relational Algebra) สามารถวิเคราะห์ความผิดปรกติที่เกิดขึ้นได้ครบถ้วน ในงานวิจัยของ Liu [3] ได้นำเสนอวิธีการตรวจสอบวิกลภาพของกฎโดยใช้โครงแผนภาพการตัดสินใจ (Decision Diagram) ซึ่งสามารถตรวจสอบความผิดปรกติของกฎไฟร์วอลล์ได้รวมเร็วขึ้น อย่างไรก็ตาม จากงานวิจัยที่กล่าวมาสามารถตรวจสอบหาวิกลภาพของกฎไฟร์วอลล์ได้ดีทั้งหมด แต่ยังไม่มีงานวิจัยใดที่ขจัดวิกลภาพของกฎได้อย่างสมบูรณ์เนื่องจากสาเหตุของปัญหาที่แท้จริงคือ การที่สมาชิกของกฎใดๆมากกว่าสองกฎ ที่เป็นสมาชิกซึ่งกันและกัน และมีผลของการกระทำที่แตกต่างกันในเวลาเดียวกัน

งานวิจัยนี้ได้จึงนำเสนอแนวความคิดในการแก้ปัญหาวิกลภาพของกฎไฟร์วอลล์ใหม่ เรียกว่า การตัดสินใจแบบโดเมนเดี่ยว (Single Decision Domain, *SDD*) แนวความคิดนี้สามารถขจัดวิกลภาพของกฎได้อย่างสมบูรณ์ เนื่องไฟร์วอลล์ *SDD* จะกำหนดให้กฎไฟร์วอลล์มีผลการตัดสินใจเพียงอย่างใดอย่างหนึ่งเท่านั้น ซึ่งได้กำหนดเป็น 2 ระบบด้วยกันคือ ระบบไฟร์วอลล์แบบเปิด (Open Firewall System, *OFS*) เป็นระบบไฟร์วอลล์ที่มีผลการกระทำเท่ากับปฏิเสธเท่านั้น (Deny) และระบบไฟร์วอลล์แบบปิด (Close Firewall System, *CFS*) เป็นระบบไฟร์วอลล์ที่มีผลการกระทำเท่ากับ ยอมรับเท่านั้น (Accept) และนำแผนภาพแบบต้นไม้ (Tree Diagram, *TD*) เป็นโครงสร้างข้อมูลที่แบ่งลำดับเงื่อนไขของกฎไฟร์วอลล์ให้มีขอบเขตที่แคบลงซึ่งจะเพิ่มประสิทธิภาพในการผ่านกฎให้เร็วขึ้น ในการประเมินผลการทำงานไฟร์วอลล์ได้นำระบบไฟร์วอลล์แบบปิด (*CFS*) มาประเมินผลเนื่องจากเป็นระบบไฟร์วอลล์ที่มีประสิทธิภาพในการรักษาความปลอดภัยมากกว่าไฟร์วอลล์ *OFS* ตามหลักการของการจำกัดสิทธิ (Principle of Least Privilege) เมื่อเกิดกฎวิกลภาพในระบบไฟร์วอลล์กฎเดิมที่ถูกขัดแย้งด้วยกฎใหม่จะถูกแทนที่ผลของการกระทำด้วยกฎใหม่ทันที โดยเชื่อว่ากฎใหม่ที่เพิ่มเข้ามาเป็นกฎที่ถูกต้องตาม

หลักการเปลี่ยนแปลงองค์ความรู้ [17] ซึ่งโครงสร้างไฟร์วอลล์ได้ใช้แผนผังแบบต้นไม้จะช่วยให้ประสิทธิภาพในการเข้าถึงกฎของไฟร์วอลล์ได้รวมเร็วขึ้น

ผลการประเมินของไฟร์วอลล์ได้แบ่งเป็น 2 ด้านด้วยกันคือ ด้านการแก้ไขปัญหาวิกลภาพของกฎไฟร์วอลล์ พบว่าไฟร์วอลล์ *SDD* สามารถแก้ไขวิกลภาพของกฎไฟร์วอลล์ได้อย่างสมบูรณ์แต่ผลกระทบจากส่งผลให้กฎไฟร์วอลล์มีการแบ่งจำนวนเงื่อนไขเพิ่มขึ้นซึ่งจะส่งผลต่อประสิทธิภาพในการทำงาน ในด้านประสิทธิภาพการทำงานของไฟร์วอลล์ได้แบ่งย่อยออกเป็น 3 ส่วนคือ ระยะเวลาที่ใช้ในการสร้างกฎไฟร์วอลล์ ระยะเวลาที่ใช้ในการผ่านกฎไฟร์วอลล์ และหน่วยความจำที่ใช้ในการสร้างกฎไฟร์วอลล์ จากผลการประเมินสรุปได้ว่า ไฟร์วอลล์ *SDD* ใช้เวลาในการสร้างกฎไฟร์วอลล์มากกว่า ไฟร์วอลล์แบบดั้งเดิม (Traditional Firewall) แต่ไฟร์วอลล์ *SDD* ใช้เวลาเข้าถึงกฎน้อยกว่าไฟร์วอลล์แบบดั้งเดิม ซึ่งหน้าที่หลักของไฟร์วอลล์จะทำการตรวจสอบข้อมูลเข้าออกระบบเครือข่ายดังนั้นระยะเวลาที่ใช้ในการตรวจสอบกฎไฟร์วอลล์จึงเป็นสิ่งที่สำคัญที่สุด และในขณะเดียวกันไฟร์วอลล์จะปรับปรุงกฎไฟร์วอลล์ก็ต่อเมื่อนโยบายการรรักษาความปลอดของเครือข่ายมีการเปลี่ยนแปลงซึ่งจะปรับปรุงกฎไม่บ่อยครั้ง และระยะเวลาที่ใช้ในการสร้างกฎของไฟร์วอลล์ *SDD* ถึงแม้จะช้ากว่าไฟร์วอลล์แบบดั้งเดิมแต่ถือว่าเป็นที่ยอมรับได้ และ หน่วยความจำที่ใช้ของระบบไฟร์วอลล์ SDD มีการใช้หน่วยความจำมากกว่าระบบไฟร์วอลล์แบบดั้งเดิมหลายเท่าตัว ผลการใช้หน่วยความจำที่ได้จากการประเมินนั้นในความเป็นจริงเพียงพอต่ออุปกรณ์ที่ใช้อยู่ในปัจจุบัน

5.2 ผลสัมฤทธิ์ของการวิจัย

5.2.1 ผลที่ได้จากการศึกษาและวิเคราะห์ปัญหาวิกลภาพของกฎไฟร์วอลล์

จากการศึกษาและวิเคราะห์ปัญหาวิกลภาพของกฎไฟร์วอลล์ พบว่า การสร้างกฎของไฟร์วอลล์ที่มีสมาชิกของกฎคาบเกี่ยวกันหรือซ้ำซ้อนกันเป็นจำนวนมากบนระบบไฟร์วอลล์ จะทำให้กฎของไฟร์วอลล์มีความซับซ้อนและความสัมพันธ์ที่ยากเกินกว่าจะมนุษย์เข้าใจ ซึ่งส่งผลให้ผู้ดูแลระบบไม่สามารถเข้าใจถึงความหมายการกระทำของกฎไฟร์วอลล์และยากต่อการปรับปรุงแก้ไข ดังนั้นในวิทยานิพนธ์นี้ จึงได้นำเสนอ แนวคิดการตัดสินใจแบบโดเมนเดี่ยว (Single Domain Decision, *SDD*) ซึ่งได้กำหนดให้กฎไฟร์วอลล์ใดๆมีผลการกระทำเพียงอย่างเดียว จะส่งผลให้กฎไฟร์วอลล์ปราศจากความสัมพันธ์ที่ซับซ้อน สามารถปรับปรุงและแก้ไขกฎของไฟร์วอลล์ได้อย่างรวดเร็วและไม่จำเป็นต้องคำนึงถึงความสัมพันธ์ของลำดับกฎไฟร์วอลล์ก่อนหน้า

5.2.2 ผลที่ได้จากประสิทธิภาพการผ่านกฎไฟร์วอลล์

จากวิทยานิพนธ์นี้ได้นำแผนภาพแบบต้นไม้ (Tree Diagram, *TD*) มาประยุกต์ใช้กับโครงสร้างข้อมูลของไฟร์วอลล์ ซึ่งพบว่า สามารถแบ่งลำดับการตรวจสอบเงื่อนไขของกฎไฟร์วอลล์ได้อย่างชัดเจนเป็นลำดับขั้น และเมื่อไฟร์วอลล์ปราศจากกฎวิกลภาพและไม่จำเป็นต้องคำนึงถึงลำดับความสัมพันธ์ของกฎไฟร์วอลล์แล้ว กฎสามารถเรียงลำดับข้อมูลของแต่ละเงื่อนไขจากน้อยไปมากเพื่อใช้คุณสมบัติการตรวจสอบกฎแบบทวิภาค ส่งผลให้ประสิทธิภาพในการผ่านกฎของไฟร์วอลล์เพิ่มมากขึ้น

5.3 ข้อเสนอแนะ

1. พัฒนาและต่อยอดไฟร์วอลล์โอเพนซอร์สโดยนำต้นแบบของไฟร์วอลล์ที่ได้จากวิทยานิพนธ์นี้ ซึ่งได้ไฟร์วอลล์ที่สามารถขจัดปัญหาวิกลภาพของกฎไฟร์วอลล์ได้อย่างสมบูรณ์และได้ไฟร์วอลล์ที่มีประสิทธิภาพในการผ่านกฎของไฟร์วอลล์ที่ดีขึ้น

2. แก้ไขปัญหาเรื่องความหมายของกฎที่มีผลต่อองค์ความรู้ขององค์กร เนื่องจากไฟร์วอลล์ *SDD* ระบบปิด (*CFS*) ที่ถูกนำมาใช้ในการนำเสนอและทดลอง การเพิ่มกฎที่มีผลการกระทำปฏิเสธจะถูกแทนด้วยกฎปฏิเสธโดยปริยาย ดังนั้นไฟร์วอลล์ *SDD* จะไม่ทราบถึงลำดับความสำคัญและจุดประสงค์ของการเพิ่มกฎเข้ามาในระบบ ซึ่งจะส่งผลผู้ดูแลระบบที่ไม่ทราบถึงองค์ความรู้ของกฎเดิมได้มีการเปลี่ยนกฎที่องค์กรอาจให้ความสำคัญเป็นอย่างมาก ยกตัวอย่าง

การเพิ่มกฎที่มีผลปฏิเสธ เข้าในระบบไฟร์วอลล์ *SDD* โดยที่ผู้ดูแลระบบคนที่หนึ่งมีจุดประสงค์ที่จะเพิ่มกฎเพื่อป้องกันการโจมตีจากแหล่งที่มาอันไม่พึงประสงค์ โดยไฟร์วอลล์ *SDD* จะไม่ทราบถึงลำดับความสำคัญแต่จะสามารถปฏิเสธการโจมตีจากแหล่งที่มาต่างได้ในขั้นแรก หากมีผู้ดูแลระบบคนใหม่เพิ่มกฎที่ส่งผลให้ยอมรับการโจมตีจากแหล่งที่มาอันไม่พึงประสงค์ โดยที่ไม่ทราบถึงองค์ความรู้หรือความหมายของการสร้างกฎเดิมก่อนหน้า จะส่งผลให้ระบบเครือข่ายเกิดความเสี่ยงที่อันตรายตามมาภายหลังได้

เนื่องจากในวิทยานิพนธ์นี้ได้มุ่งเน้นแก้ไขปัญหาวิกลภาพของกฎไฟร์วอลล์และประสิทธิภาพการผ่านกฎ จึงไม่ได้สนใจความหมายขององค์ความรู้ของกฎไฟร์วอลล์ ดังนั้นในการเพิ่มกฎแต่ละกฎควรจะต้องมีการตรวจสอบความสัมพันธ์ของกฎกับรายชื่อของกฎที่องค์กรให้ความสำคัญก่อนทุกครั้ง แล้วจึงนำกฎที่ผ่านการตรวจสอบความสำคัญแล้วเข้าระบบไฟร์วอลล์ SDD เพื่อประมวลผลต่อไป

3. โครงสร้างของไฟร์วอลล์แบบแผนภาพต้นไม้ (Tree Diagram, *TD*) และ โครงสร้างของไฟร์วอลล์แบบตารางแฮช (Hashing Firewall) เป็นโครงสร้างที่มีประสิทธิภาพในการผ่านกฎไฟร์วอลล์สูง โดยในวิทยานิพนธ์นี้เลือกใช้โครงสร้างแบบแผนภาพต้นไม้ และนำไปเปรียบเทียบกับ

โครงสร้างของไฟร์วอลล์แบบดั้งเดิม แต่ในปัจจุบันมีไฟร์วอลล์ที่พัฒนาบนโครงสร้างไฟร์วอลล์แบบตารางแฮช เช่น IPset, IPtables ซึ่งอาจจะมีประสิทธิภาพที่ดีกว่า ดังนั้นควรจะมีการวิเคราะห์และเปรียบเทียบประสิทธิภาพของไฟร์วอลล์ทั้งสองหรือมีการนำมาประยุกต์ใช้ร่วมกันในขั้นต่อไป เพื่อให้ได้ไฟร์วอลล์ที่มีประสิทธิภาพที่ดียิ่งขึ้น

เอกสารอ้างอิง

เอกสารอ้างอิง

[1] Al-Shaer E, Harmed H. "Firewall Policy Advisor for anomaly Detection and Rule Editing". Proceedings of IEEE Integrated Network Management; 24–38 March 2003; Colorado Springs, USA; 2003. pp. 17-30.

[2] Chomsiri T, Pornavalai C. “Firewall Policy Analyzing by Relational Algebra”. Proceedings of International Technical Conference on Circuits/Systems, Computers and Communications; 6-8 July 2004; Matsushima, Japan; 2004. pp. 559-562.

[3] Liu A. "Formal Verification of Firewall Policies". Proceedings of IEEE International Conference on Communications; 19-23 May 2008; Beijing, China; 2008. pp. 1494-1498.

[4] Oppliger R. "Internet security: firewalls and beyond". Communication of the ACM 1997; 4[5]: 92-102.

[5] Postel J. "Internet Protocol". IETF, RFC 791, September 1981.

[6] Reynolds J, Postel J. "Assigned Numbers". IETF, RFC 1700, October 1994.

[7] Postel J. "Transmission Control Protocol". IETF, RFC 793, September 1981.

[8] "Iptables". [computer program]. Version 1.4.5. University in Seville: Netfilter Project; 2009.

[9] "Ipset". [computer program]. Version 4.2. University in Seville: Netfilter Project; 2009.

[10] "nf-HiPAC". [computer program]. Version 0.9.1. University of Wisconsin: The HiPAC Project; 2005.

[11] Cherdantseva Y, Hilton J. “Information Security and Information Assurance: Discussion about the Meaning, Scope and Goals”. 1st edition. Pennsylvania: IGI Global Publishing; 2013.

[12] Cormen T. "Introduction to Algorithms". 3rd edition. Boston: Massachusetts Institute of Technology; 2009.

[13] Knuth D. “The Art of Computer Programming, Volume 1: Fundamental Algorithms”. 3rd edition. Boston: Addison-Wesley; 1997.

เอกสารอ้างอิง

[14] Liu A, Torng E, Meiners C. "Firewall Compressor: An Algorithm for Minimizing Firewall Policies". Proceedings of IEEE International Conference on Computer Communications; 15-19 March 2008; Phoenix, Arizon, USA; 2008. pp. 176-180.

[15] Liu A, Gouda M. “Firewall Policy Queries”. IEEE Transactions on Parallel and Distributed Systems 2009; 20[6]: 766-777.

[16] He X, Chomsiri T, Nanda P, Tan Z. "Improving cloud network security using the Tree-Rule firewall". Future Generation Computer Systems 2013, 30[1]: 116–126.

[17] Booth R, Noisanguan W. "An Axiomatic Approach to Firewall Rule Update". Proceedings of IEEE International Joint Conference on Computer Science and Software Engineering; 13-15 May 2009; Phuket, Thailand; 2009. pp. 70-75.

[18] Khummanee S, Khumseela A, Puangpronpitag S. "Towards a New Design of Firewall: Anomaly Elimination and Fast Verifying of Firewall Rules". Proceedings of International Joint Conference on Computer Sciences and Software Engineering; 29-31 May 2013; Khon Kean, Thailand; 2013. pp. 93-98.

[19] El-Alfy E. "A Heuristic Approach for Firewall Policy Optimization". IEEE/ACM Transactions on Advanced Communication Technology 2007; 3[1]: 1782-1787.

[20] ศุภพร รัฐอาจ, สุธน เจริญศิริ, สมนึก พ่วงพรพิทักษ์. “การปรับแต่งไฟร์วอลล์โอเพ่นซอร์สให้มีประสิทธิภาพสูงสุด”. เอกสารการประชุมทางวิชาการระดับชาติด้านคอมพิวเตอร์และเทคโนโลยีสารสนเทศ; 13 กุมภาพันธ์ 2555; เชียงใหม่, ประเทศไทย; 2555. หน้า 55-60.

ภาคผนวก

ภาคผนวก ก
โปรแกรมสุ่มกฎไฟร์วอลล์

โปรแกรมสุ่มกฎไฟร์วอลล์

การสุ่มกฎไฟร์วอลล์โดยใช้โปรแกรมช่วยสุ่มอย่างง่าย ได้มีการกำหนดขอบเขตของข้อมูลการสุ่มกฎโดยเริ่มต้นจาก 0.0.0.0 จนถึง 255.255.255.255 โดยวิทยานิพนธ์นี้จะยกตัวอย่างฟังก์ชันในการสุ่มกฎไฟร์วอลล์อย่างง่ายได้ดังนี้ตัวอย่างฟังก์ชัน 1

ฟังก์ชัน 1 สุ่มกฎไฟร์วอลล์

```
private String GenerateRule()
{
   String result = "";
   int nDefaultIP = 256;
   int nDefaultPort = 65536;
   int sipoct1 = (int)(Math.random()*nDefaultIP);
   int sipoct2 = (int)(Math.random()*nDefaultIP);
   int sipoct3 = (int)(Math.random()*nDefaultIP);
   int sipoct4 = (int)(Math.random()*nDefaultIP);
   int spt = (int)(Math.random()*nDefaultPort);
   int dipoct1 = (int)(Math.random()*nDefaultIP);
   int dipoct2 = (int)(Math.random()*nDefaultIP);
   int dipoct3 = (int)(Math.random()*nDefaultIP);
   int dipoct4 = (int)(Math.random()*nDefaultIP);
   int dpt = (int)(Math.random()*nDefaultPort);
   int pro = (int)(Math.random()*2);
   int act = (int)(Math.random()*2);
   result = sipoct1+"."+sipoct2+"."+sipoct3+"."+sipoct4+"\t\t";
   result += spt+"\t";
   result += dipoct1+"."+dipoct2+"."+sipoct3+"."+sipoct4+"\t\t";
   result += dpt+"\t";
   if(pro == 0)result += "TCP\t";
   else result += "UDP\t";
   if(act == 0)result += "ACCEPT";
   else result += "DENY";
   return result;
}
```

จากตัวอย่างข้างต้นจะเป็นฟังก์ GenerateRule() ซึ่งจะสุ่มกฎอย่างง่ายโดยมีรูปแบบการสุ่มกฎดังนี้

```
SIP                 SPT     DIP             DPT     PRO     ACT
-------------------------------------------------------------------
108.112.7.172       19682   14.59.7.172     45645   TCP     ACCEPT
73.31.83.186        10051   5.83.83.186     14901   UDP     DENY
34.57.210.205       63447   154.173.210.205 18357   UDP     DENY
109.179.55.90       54290   166.154.55.90   60674   TCP     DENY
185.9.31.182        27837   224.152.31.182  36152   UDP     DENY
105.196.131.217     8456    146.158.131.217 57180   TCP     DENY
187.213.242.33      56709   47.147.242.33   32388   TCP     DENY
```

จำนวนของกฎไฟร์วอลล์จะรับค่าจากพารามิเตอร์ของฟังก์ชัน `Generate_Rule(int nRule)` ดังตัวอย่างฟังก์ชัน 2

ฟังก์ชัน 2 สุ่มกฎตามจำนวน

```
public Generate_Rule(int nRule) {
   for(int i = 0; i < nRule ;i++){
          RuleSet.add(GenerateRule());
   }
}
```

จากฟังก์ชัน 2 จะทำการสุ่มกฎตามจำนวนที่รับค่าจากพารามิเตอร์และจะถูกเก็บค่าไว้ใน ArrayList ที่กำหนดชื่อตัวแปรว่า RuleSet ดังตัวอย่าง

```
ArrayList RuleSet = new ArrayList();
```

กฎที่สุ่มจะถูกเก็บไว้ในตัวแปร RuleSet ซึ่งจะสามารถสุ่มกฎเก็บไว้ได้หลายครั้งจนกว่าจะเรียกฟังก์ชันในการเคลียค่าตัวแปรของฟังก์ชันข้อมูลของกฎไฟร์วอลล์ถึงจะถูกลบออก สามารถเรียกฟังก์ชันเคลียข้อมูลได้ดังนี้

```
RuleSet.clear();
```

หลังจากเคลียข้อมูลกฎไฟร์วอลล์แล้ว จะสามารถสุ่มกฎได้ใหม่โดยการเรียกฟังก์ชัน `Generate_Rule(int nRule)` โดยข้อมูลกฎไฟร์วอลล์จะถูกเก็บโดยเริ่มจากกฎที่ 1 ใหม่เมื่อได้จำนวนกฎของไฟร์วอลล์ตามที่ต้องการแล้วสามารถเขียนไฟล์ .TXT เพื่อนำไปใช้ในการประมวลผลขั้นต่อไปได้ โดยเรียกฟังก์ชัน 3

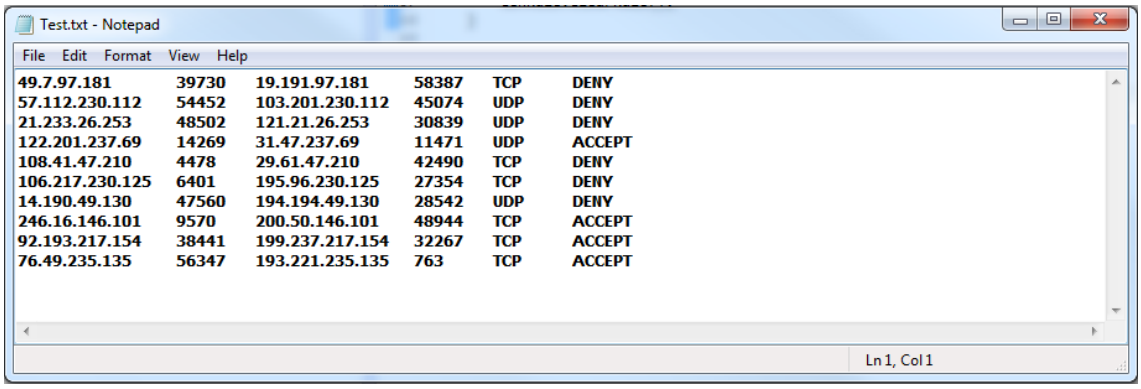
ฟังก์ชัน 3 บันทึกกฎไฟร์วอลล์

```
public void writeRule(String path) throws Exception{
   BufferedWriter bfw = new BufferedWriter(
   new OutputStreamWriter(
   new FileOutputStream(path)));
   for(int i = 0; i < RuleSet.size() ;i++){
      bfw.write(Rule, 0, Rule.length());
      bfw.newLine(); bfw.close();
   }
}
```

ในการเรียกใช้ฟังก์ชันจะต้องกำหนดที่อยู่ของไฟร์วอลล์ที่จะบันทึก เช่น

```
GenRule.writeRule("C:\\Test.txt");
```

ซึ่งจะได้ข้อมูลกฎที่ถูกสุ่มและเก็บไว้ในตัวแปร RuleSet ดังรูปตัวอย่างนี้



ประวัติย่อผู้วิจัย

ประวัติย่อผู้วิจัย

ชื่อ นามสกุล อธิพงศ์ คำสีลา

วัน เดือน ปีเกิด วันที่ 18 มิถุนายน พ.ศ. 2532

จังหวัด และประเทศที่เกิด อำเภอเมืองมหาสารคาม จังหวัดมหาสารคาม

ประวัติการศึกษา พ.ศ. 2547 มัธยมศึกษาตอนต้น โรงเรียนสารคามพิทยาคม
พ.ศ. 2550 มัธยมศึกษาตอนปลาย วิทยาศาสตร์-คณิตศาสตร์
โรงเรียนสารคามพิทยาคม
พ.ศ. 2554 ปริญญาวิทยาศาสตรบัณฑิต (วท.บ.)
สาขาวิชาวิทยาการคอมพิวเตอร์ มหาวิทยาลัยมหาสารคาม
พ.ศ. 2557 ปริญญาวิทยาศาสตรมหาบัณฑิต (วท.ม.)
สาขาวิชาวิทยาการคอมพิวเตอร์ มหาวิทยาลัยมหาสารคาม

ตำแหน่ง สถานที่ทำงาน วิศวกรซอฟแวร์ (Software Engineer)
บริษัท โกซอฟท์ (ประเทศไทย) จำกัด

ที่อยู่ที่สามารถติดต่อได้ ที่อยู่ 327 หมู่ที่ 12 ตำบลแวงน่าง อำเภอเมืองมหาสารคาม
จังหวัดมหาสารคาม รหัสไปรษณี 44000

รางวัลเรียนดี ทุนวิจัย และทุนการศึกษา

1. โครงการทุนสนับสนุนกสนวิจัย งบประมาณแผ่นดิน ประจำปีการศึกษา 2556
2. โครงการวิจัยงบประมาณรายได้ ประจำปีงบประมาณ 2557
3. โครงการทุนพัฒนาศักยภาพทางการวิจัย ประจำปีการศึกษา 2555 และปีการศึกษา 2556
4. โครงการทุนเรียนดี มีจิตอาสา ประจำปีการศึกษา 2555

ผลงานวิจัย

1. Khumseela A, Khummanee S, Puangpronpitag S. “Improving the Firewall Rule Management by a Single Domain Decision Concept”. In Proceedings of the 9th National Conference on Computing and Information Technology (NCCIT 2013); 9-10 May 2013; Bangkok, Thailand. 2013. pp. 403-410.

2. Khummanee S, Khumseela A, Puangpronpitag S. “Towards a New Design of Firewall: Anomaly Elimination and Fast Verifying of Firewall Rules”. In Proceedings of the 10th International Joint Conference on Computer Sciences and Software Engineering (JCSSE 2013); 29-31 May 2013; Khon Kean, Thailand. 2013. pp. 93-98.