# การจัดการกฎของไฟร์วอลล์แบบกึ่งอัตโนมัติ

สุชาติ คุ้มมะณี* และ จุรีภรณ์ ตั้งมั่นดี*

**บทคัดย่อ**

การกำหนดกฎของไฟร์วอลล์มีความซับซ้อนมาก และในปัจจุบันได้ทำการปรับปรุง และพัฒนาไฟร์วอลล์ให้มีการใช้งานที่ง่ายขึ้นกว่าเดิม แต่อย่างไรก็ตาม ก็ยังคงประสบปัญหาเกี่ยวกับการใช้งาน และความคลุมเครือในการตั้งกฎเสมอเมื่อมี Packet เข้ามากระทบ ไฟร์วอลล์จะทำการค้นหากฎที่ได้กำหนดไว้เปรียบเทียบกับ Packet ที่กระทบ ซึ่งในความเป็นจริงจำนวนกฎของไฟร์วอลล์จะมีเป็นจำนวนมาก หากกฎที่ตรงกับ Packet นั้นอยู่ลำดับท้าย ๆ ของ Rule List หรือไม่มีอยู่ใน Rule List เลย (ไฟร์วอลล์จะค้นหากฎเป็นลำดับ) จะทำให้การทำงานของไฟร์วอล์ทำงานได้ช้าผู้วิจัยจึงได้คิดค้นวิธีการจัดการกฎของไฟล์วอลล์แบบใหม่ซึ่งวิธีการดังกล่าวสามารถตัดสินใจแทนมนุษย์ได้ ในระดับหนึ่ง เพื่อลดปัญหา หรือข้อบกพร่องที่เกิดจากการทำงานของมนุษย์ลง โดยทำการค้นหากฎ ลดจำนวนกฎ และปรับความซับซ้อนของกฎให้มีความกระชับ และถูกต้องมากที่สุด โดยอาศัยกฎการกำจัดสิ่งผิดปกติในไฟร์วอลล์มาเป็นตัวแบบ (Engine) ในการทำงาน

ผลปรากฏว่าตัวแบบที่สร้างขึ้นสามารถตัดสินใจในการยุบรวม และกำจัดความกำกวมของกฎบนไฟร์วอลล์ได้ด้วยตัวเอง

**Keyword:** ไฟร์วอลล์ สิ่งผิดปกติ

## 1. บทนำ

การค้นหากฎบนไฟร์วอลล์จะเริ่มต้นค้นหาโดยเรียงลำดับจากแถวที่ 1 จนถึงแถวสุดท้าย เมื่อ packet ที่เข้ามากระทบและ match กับกฎใดๆ ไฟร์วอลล์ก็จะทำตาม action ที่กฎข้อนั้น ๆ กำหนดไว้ จากการวิจัยของ Chotipat and Thawatchai [1],[2] ได้แก้ไขปัญหากลุ่มกฎของไฟร์วอลล์ ที่มักจะมีสิ่งผิดปกติหรือมีความขัดแย้งแอบแฝงอยู่ ซึ่งเรียกว่า (สิ่งผิดปกติ) จากวิธีการที่ Ehab Al-Shaer [3] ได้เสนอไว้ แต่ยังไม่ครอบคลุมทุก ๆ ปัญหาที่เกิดขึ้น โดยอาศัยวิธีการแบบ Relational Algebra [4] จากผลการทดลองสามารถแก้ปัญหาได้ครบทุกกรณี แต่ปัญหาที่ประสบคือ การใช้งานจริงในทางปฏิบัติทำได้ไม่ง่ายเนื่องจากการแจกแจงกฎของไฟร์วอลล์ออกเป็น relation จะทำให้จำนวนของ relation เกิดขึ้นเป็นจำนวนมาก ยกตัวอย่างเช่น ในตารางที่ 1 ประกอบด้วย Source IP Address (src_ip) มีหมายเลข IP Address = 192.168.10.0 อยู่ใน class C subnet mask เท่ากับ 255.255.255.254 (/31) มีจำนวนหมายเลข IP สำหรับใช้งาน 2 หมายเลขคือ 192.168.10.0 และ 192.168.10.1 มี source port (src_port) คือ 23 และ Destination IP Address (des_ip) คือ 172.16.5.0 อยู่ใน class B subnet mask เท่ากับ 255.255.255.254 (/31) มีจำนวนหมายเลข IP สำหรับใช้งาน 2 หมายเลขคือ 172.16.5.0 และ 172.16.5.1 มี destinationport (des_port) คือ 8888-8889 คอลัมน์สุดท้ายจะเป็น action คือ Accept

***ตารางที่ 1*** *ตัวอย่างกฎบนไฟร์วอลล์*

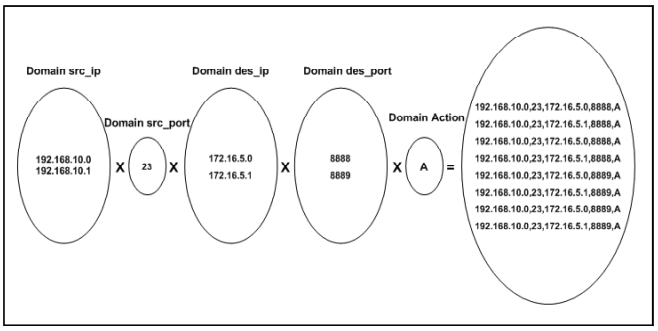
| Ordew | Src_IP | Src_Port | Des_IP | Dess_Port | Action |
|---|---|---|---|---|---|
| 1 | 192.168.10.3/32 | 23 | 172.16.5.0/32 | 8888- 8889 | Accept |

เมื่อทำ คาที-เซียนโปรดัค (Cartesian product) แจกแจงกฎของไฟร์วอลล์ในตารางที่ 1 ความสัมพันธ์ (Relation) จะได้ผลลัพท์ดังตารางที่ 2 ซึ่งมีจำนวนเท่ากับ src_ip x src_port x des_ip x des_port x Action (2 x 1 x 2 x 2 x 1 = 8 relations) ส่วนวิธีการของ Cartesian product แสดงไว้ดังภาพที่ 1

***ตารางที่ 2*** *ผลลัพธ์จาก Cartesian product เป็น ความสัมพันธ์ของกฎในตารางที่ 1*

| Ordew | Src_IP | Src_Port | Des_IP | Dess_Port | Action |
|---|---|---|---|---|---|
| 1 | 192.168.10.0 | 23 | 172.16.5.0 | 8888 | Accept |
| 2 | 192.168.10.0 | 23 | 172.16.5.1 | 8888 | Accept |
| 3 | 192.168.10.1 | 23 | 172.16.5.0 | 8888 | Accept |
| 4 | 192.168.10.1 | 23 | 172.16.5.1 | 8888 | Accept |
| 5 | 192.168.10.0 | 23 | 172.16.5.0 | 8889 | Accept |
| 6 | 192.168.10.0 | 23 | 172.16.5.1 | 8889 | Accept |
| 7 | 192.168.10.1 | 23 | 172.16.5.0 | 8889 | Accept |
| 8 | 192.168.10.1 | 23 | 172.16.5.1 | 8889 | Accept |

* *คณะวิทยาการสารสนเทศ มหาวิทยาลัยมหาสารคาม*




***ภาพที่ 1*** *วิธีการทำ Cartesian product บนกฎของไฟร์วอลล์*

ซึ่งในการใช้งานของไฟร์วอลล์บนระบบเครือข่ายจริง ๆ นั้น จำนวนของกฎบนไฟร์วอลล์จะมีขนาดของโดเมนที่กว้างมาก เช่น ตัวอย่างในตารางที่ 3 ลำดับที่ 1 src_ip คือ 192.168.15.0 subnet mask เท่ากับ 255.255.255.0 (/24) จะมีจำนวนของหมายเลขไอพีเท่ากับ 256 หมายเลข (192.168.15.0 – 192.168.15.255) src_port คือหมายเลข port ที่ต้องการใช้งานเป็น Any ซึ่งมีค่าอยู่ระหว่าง 0 – 65535 des_ip คือ 203.30.90.0/28 subnet mask เท่ากับ 255.255.255.240 (/28) มีจำนวนของหมายเลขไอพีเท่ากับ 16 หมายเลข (203.30.90.0 – 203.30.90.15) และสุดท้ายคือ des_port มี port 80 (protocol HTTP) เมื่อนำมาแจกแจงโดยใช้ Cartesian product และผลลัพธ์ที่ได้อยู่ในรูปของความสัมพันธ์ จะมีจำนวนเท่ากับ 255 x 65536 x 16x1 = 267,386,880 ความสัมพันธ์ สำหรับกฎข้ออื่นๆ มีจำนวนความสัมพันธ์ ดูได้จากตารางที่ 3

***ตารางที่ 3*** *ตัวอย่างกฎในไฟร์วอลล์*

| Ordew | Src_IP | Src_Port | Des_IP | Dess_Port | Relation Number |
|---|---|---|---|---|---|
| 1 | 192.168.15.0/24 | Any | 203.30.90.0/28 | 80 | 267. . |
| 2 | 10.100.0.0/22 | Any | Any | 443 | 288. . |
| 3 | Any | Any | Any | 22 | 120. . |
| … | … | … | … | … | … |
| N | Any | Any | Any | Any | 792. . |

หมายเหตุ : src_ip, des_ip = Any = $2^{32}$ = 4,294,967,296

src_port, des_port = Any = $2^{16}$ = 65,536

เมื่อนำ relations ที่เกิดขึ้นใส่ลงในฐานข้อมูล DBMS เพื่อใช้ฟังก์ชัน build-in ในตัว DBMS สำหรับการ QUERY, UNION, INTERSECT, DIFFERENCE ตามกฎของ relational algebra จะไม่มีฐานข้อมูลตัวใดที่สามารถทำได้ เนื่องจากจำนวน record มีขนาดใหญ่เกินไป และในตัวอย่างก็มีเพียงแค่ 3 - 4 กฎเท่านั้น

## 2. ทฤษฎีและงานวิจัยที่เกี่ยวข้อง

### 2.1 การทำงานของไฟร์วอลล์

ไฟร์วอลล์จะทำตามกลุ่มของกฎที่ผู้ดูแลระบบได้ตั้งไว้ [5] ดังตารางที่ 5 ซึ่งกลุ่มของกฎเหล่านี้ถูกเรียกว่า Policy, RuleList หรือ Access Control List ก็ได้ ในงานวิจัยฉบับนี้จะเรียกกลุ่มของกฎว่า RuleList และเรียกกฎแต่ละบรรทัดว่า Rule เช่นกฎข้อแรกก็เรียกว่า Rule 1 เป็นต้น เมื่อไฟร์วอลล์ได้รับ Packet มันจะอ่านข้อมูลที่อยู่ส่วนหัวของ Packet (ข้อมูลที่ได้รับเช่น IP Address ต้นทาง, IP Address ปลายทาง และหมายเลข Port ปลายทางเป็นต้น) จากนั้นมันจะนำมาตรวจสอบเทียบกับ RuleList ว่า ข้อกำหนดใน RuleList ใดตรงกับ Packet ที่เข้ามา ซึ่งจะมี 3 เงื่อนไข [6] คือ Accept Deny (Drop Packet นั้นทิ้งไป) หรือ Reject (Reject คล้ายกับ Deny เพียงแต่จะส่ง ICMP port unreachable กลับไปยังผู้ที่ส่ง packet มา) การเทียบข้อมูลในเฮดเดอร์ของ Packet กับ Rule List นั้นจะทำโดยการเทียบจาก Rule ที่อยู่บนสุดก่อนซึ่งก็คือ Rule1 ว่าตรงกันกับ Packet นั้นหรือไม่ ถ้าตรงกันก็จะทำตามค่าที่ระบุไว้ในช่อง Action ของ RuleList ถ้า Action เป็น ACCEPT (หรือ Permit, Allow) ก็จะส่งต่อ Packet แต่ถ้าเป็น DENY (หรือ reject หรือ drop) ก็จะไม่ส่งต่อ Packet นั้น และถ้าไม่ match ไฟร์วอลล์ก็จะทำการเทียบกับ Rule ถัดไป (Rule 2) ทำอย่างนี้ไปเรื่อย ๆ จนถึง Rule สุดท้ายซึ่งมักจะได้รับการตั้งค่าให้ match กับทุก Packet และตั้ง Action ให้เป็น DENY เพื่อที่จะทำการห้าม ไม่ให้ Packet ที่ไม่จำเป็นออกสู่ระบบเครือข่ายไป

***ตารางที่ 4*** *RuleList ของไฟร์วอลล์*

| Order | Protocol | Src_IP | Src_Port | Des_IP | Dess_Port | Action |
|---|---|---|---|---|---|---|
| 1 | tcp | 192.168.10.10 | Any | Any | 80 | deny |
| 2 | tcp | 192.168.15.0/24 | any | any | 80 | Accept |
| 3 | tcp | any | Any | 172.16.5.20 | 80 | deny |
| 4 | tcp | Any | Any | 172.16.5.0/22 | 80 | Accept |
| 5 | udp | 172.16.10.50 | Any | Any | 53 | deny |
| 6 | udp | 172.16.10.0/20 | Any | Any | 53 | Accept |
| 7 | … | … | … | … | … | … |
| 8 | Any | any | Any | Any | Any | deny |

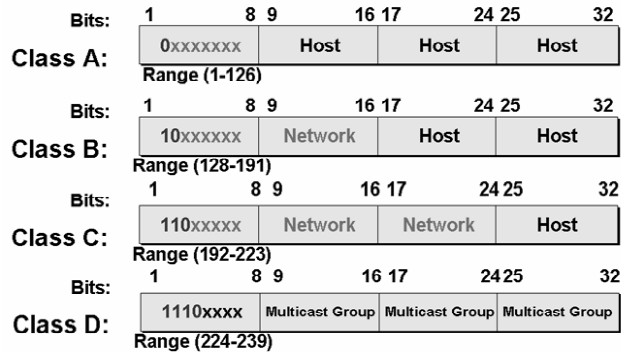
### 2.2 พีชคณิตเชิงความสัมพันธ์

ความสัมพันธ์ (Relation) คือ สับเซตของคาร์ทีเซียนโปรดัคของโดเมน [7] การดำเนินการระหว่างรีเลชันจะดำเนินการด้วยการกระทำเชิงความสัมพันธ์ (Relational Operation) ซึ่งตัวที่มักจะได้รับการใช้งานบ่อย ๆ ได้แก่ SELECT (เลือก rows), PROJECT (เลือก column), UNION, INTERSECTION และ DIFFERENCE เป็นต้น เมื่อความสัมพันธ์ตั้งแต่สองความสัมพันธ์มีการกระทำระหว่างกัน สามารถเขียนให้อยู่ใน

รูปพีชคณิต ได้ตัวอย่างเช่น R1 = R2 UNION R3 ซึ่งเรียกว่า พีชคณิตเชิงความสัมพันธ์ (Relational Algebra) กฎของไฟร์วอลล์สามารถที่จะทำให้อยู่ในรูปของรีเลชันได้ เช่น src_ip = 100.0.0.0/31, dst_ip = 200.0.0.1, dst_port = 80, action = deny, protocol = tcp สามารถที่จะทำเป็นรีเลชัน R1 ได้

**2.3 IP Address**

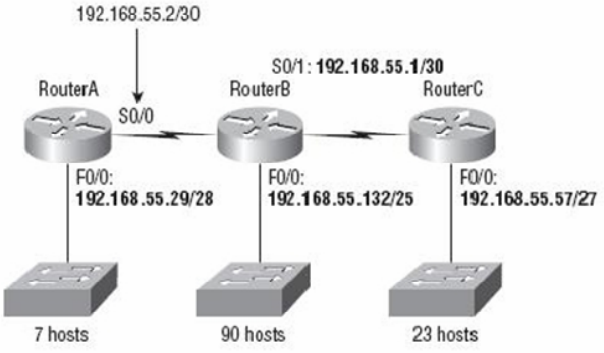
IP Address [8] มีทั้งหมด 32 บิต หรือ 4 ไบต์ โดยแต่ละไบต์ จะถูกคั่นด้วยเครื่องหมาย (.) ตัวอย่างเช่น 172.20.22.2 Subnet Mask [9-11] จะช่วยแยกแยะว่าส่วนใดภายในหมายเลข IP Address เป็น Network Address และส่วนใดเป็น Host Address ดังนั้นจะเห็นได้ว่าเวลาที่เรากำหนด IP Address เราจะต้องกำหนด Subnet Mask ด้วยเสมอ สำหรับการแบ่งคลาสในอินเทอร์เน็ตแบ่งออกเป็น 5 class ดังภาพที่ 2




***ภาพที่ 2*** *การแบ่งคลาสบนเครือข่ายอินเทอร์เน็ต*

**2.4 การทำ Subnet ซ้อน Subnet (Variable-Length Subnet Mask : VLSM)**

VLSM [12] คือ การทำ Subnet ภายใน Subnet เหตุผลที่ต้องทำ VLSM เพื่อต้องการให้เครือข่ายมีขนาดพอดีกับการใช้งานจริงๆ ซึ่ง VLSM จะยอมให้มีการแบ่ง subnet ได้มากกว่า 1 ครั้ง สำหรับแต่ละ IP เพื่อให้ได้ IP ตามต้องการอีกทั้งยังเป็นการใช้งาน IP ได้อย่างมีประสิทธิภาพ ยกตัวอย่าง ดังภาพที่ 3




***ภาพที่ 3*** *ตัวอย่างการทำ VLSM*

**2.5 การจัดหมวดหมู่ของ Anomaly**

2.5.1 Shadowing Anomaly [3]

Rule ที่ถูกบังคือ Rule ที่ไม่มีโอกาสที่จะถูก Execute เลย (ไม่มี Packet ใดที่จะมา Match กับ Rule นี้แล้ว Packet ถูกดำเนินการตาม Action ที่ระบุไว้ใน Rule นี้) เนื่องจาก Packet ทั้งหมดที่คาดว่าจะ Match กับ Rule นี้ จะ Match กับ Rule ใด Rule หนึ่งหรือหลาย ๆ Rule ที่อยู่ก่อนหน้า Rule-x จะเกิด Shadowing Anomaly เมื่อ Rule-x เป็น Shadowed Rule

2.5.2 Correlation Anomaly

Rule สอง Rule ที่มี Action ต่างกันจะเรียกว่า Correlate กันเมื่อ Rule ที่อยู่ลำดับก่อนหน้า Match กับบาง Packet ที่ Match กับ Rule ที่อยู่ลำดับหลัง และ Rule ที่อยู่ลำดับหลัง Match กับบาง Packet ที่ Match กับ Rule ที่อยู่ก่อนหน้า นิยามของ Correlation Anomaly คือ “Rule-x จะเกิด Correlation Anomaly กับ Rule-y เมื่อ Rule-x กับ Rule-y เป็น Partially Correlate ระหว่างกัน”

2.5.3 Generalization Anomaly

Rule ที่Generalize Rule อื่น หมายถึง Rule ที่ Match กับทุก Packet ที่จะ Match กับ Rule อื่น (ที่กล่าวถึง) ที่อยู่ในลำดับก่อนหน้าได้ Generalization Anomaly จะเกิดขึ้นถ้ามี Ruley ที่ Generalized Rule-x

2.5.4 Redundancy Anomaly

Rule ที่ redundant ต่อ Rule อื่น คือ Rule ที่กระทำการตาม Action เดียวกันบน Packet เดียวกันกับที่ Rule อื่น (ที่อยู่ลำดับหลัง) จะกระทำ Redundancy Anomaly จะเกิดขึ้นกับ Rule ที่เป็น Redundant Rule

## 3. เครื่องมือที่ใช้ในการดำเนินงาน

เครื่องมือประกอบด้วย

3.1 เครื่องคอมพิวเตอร์ Pentium M 1.6 GHz หน่วยความจำ 512 MB ฮาร์ดดิสก์ 80 GB

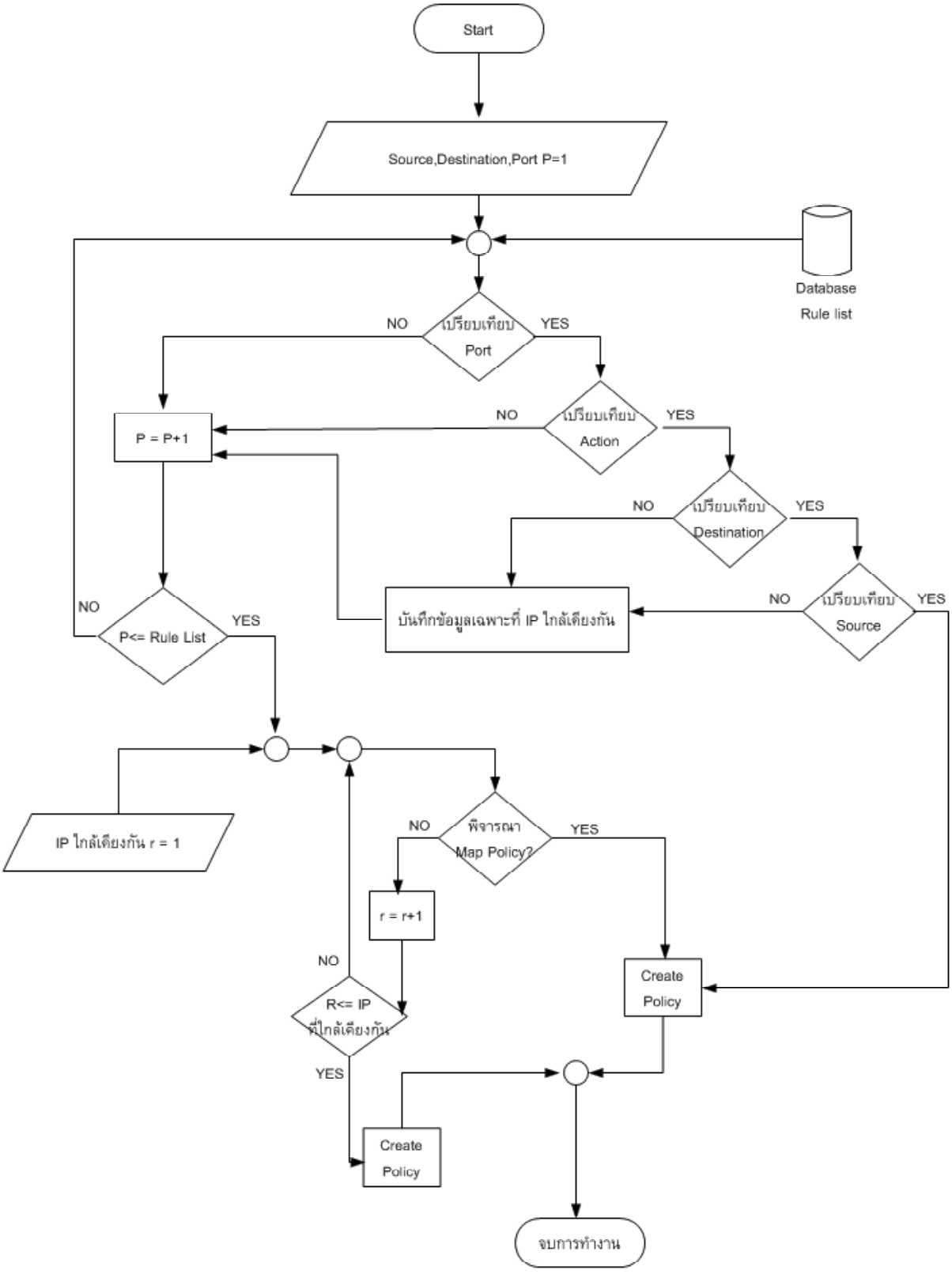
3.2 IP Table ไฟล์วอลล์บนลินุกซ์

3.3 PHP, MySQL, Shell Script สำหรับพัฒนา Interface ในการใช้งาน

## 4. หลักการในการออกแบบ

จากภาพที่ 4 เป็นการแสดงขั้นตอนการจัดการกฎของไฟร์วอลล์ หรือการ Optimize Rule ของโปรแกรม Semi-automatic Optimized Firewall Policy Management ตามลำดับขั้น โดย

อธิบายขั้นตอนการทำงานได้ดังนี้




***ภาพที่ 4*** *ขั้นตอนการทำงานของระบบ*

ลำดับแรก จะทำการดึงข้อมูลของ Packet ที่อยู่ใน Log File ของไฟล์วอลล์ทั้งหมดเมื่อ Packet ภายนอกมากระทบ จากนั้นทำการจัดฟอร์มให้อยู่ในรูปแบบคำสั่งของ IPTables และทำการกำหนด Action ได้จากสองกรณีคือ จากการกำหนดกฎโดยผู้ดูแลระบบ ซึ่งโดยปกติแล้วจะทำการกำหนด Action ด้วยเสมอ และอีกกรณีคือ ระบบทำการใส่ Action ให้โดยอัตโนมัติ โดยจะต้องมี Default Port เพื่อใช้ในการตรวจสอบ Action วิธีการทำคือว่าในการกำหนด Default Port จะทำการกำหนด 2 กรณีคือ Port ที่คาดว่ามีความปลอดภัย กำหนด Action ให้เป็น ACCEPT และ Port ที่คาดว่าอาจเป็นอันตรายกับระบบกำหนด Action เป็น DENY ส่วน Port ที่ไม่ได้อยู่ในสองกรณีให้ทำการแจ้งเตือนไปที่ผู้ดูแลระบบพร้อมกับคำแนะนำและขอคำยืนยัน Action จากผู้ดูแลระบบ เมื่อได้ Action จากผู้ดูแลระบบ คำสั่งก็ครบสมบูรณ์ และสามารถที่จะกำหนดกฎนั้นเป็นกฎของไฟร์วอลล์ได้ การทำงานของงานวิจัยนี้ จะทำการเปรียบเทียบกฎต่อไปเพื่อ Match หรือ กำจัด Anomaly ไปพร้อมกัน

ขั้นตอนถัดมาให้ทำการดึง Rule ที่อยู่ใน Rule List ขึ้นมาทำงานที่โปรแกรม หลังจากนั้นโปรแกรมจะทำการเปรียบที่ Service หรือ Port ว่าจาก Packet ที่ร้องขอมากับ rule ที่มีอยู่ใน Rule List ถ้าไม่ตรงกันจะเปรียบเทียบที่ละกฎ (Rule) ไปเรื่อย ๆ ถ้าเปรียบเทียบแล้วปรากฏว่า Port ของ Packet ที่ร้องขอเข้ามาใหม่ตรงกับ Port ของกฎ (Rule) ที่มีอยู่ใน Rule List นั้นแล้ว ในการเปรียบเทียบนี้จะทำให้ลดกฎ (Rule) ที่ซึ่งไม่เกี่ยวข้องลงไปได้จำนวนหนึ่ง ตัวอย่างเช่น
มีข้อมูล Packet เข้ามาดังนี้

```
Jan 9 03:49:03 liverpool kernel: ACCEPT_LOGIN=
OUT=eth0 SRC=192.168.0.13 DST=129.110.31.7    LEN=62
TOS=0x00 PREC=0x00 TTL=64 ID=10849 DF PROTO=UDP
SPT=32789 DPT=21 LEN=42
```

***ภาพที่ 5*** *ข้อมูลของ Packet*

และใน Rule List จะมีข้อมูล ดังนี้

***ตารางที่ 5*** *Rule List ของไฟร์วอลล์*

| Rule | Source IP | Dest. IP | Dest Port | Action |
|---|---|---|---|---|
| 1 | 192.168.0.1 | 10.114.0.0 | 21 | DENY |
| 2 | 192.168.0.1 | 10.114.0.0/30 | 21-22 | ACCEPT |
| 3 | 192.168.0.1 | 10.114.0.0/30 | 23-24 | ACCEPT |
| 4 | 192.168.0.1 | any | 80 | ACCEPT |
| 5 | any | any | any | DENY |

จากนั้นฟอร์มข้อมูลของ Packet จาก Log File ให้เป็นชุดคำสั่งของ IPTables ก่อน จะได้ดังนี้

```
iptables –A INPUT –s 192.168.0.13 –d 129.110.31.7 –p
UDP–dport 21 –j  ACCEPT
```

เนื่องจากโปรแกรม Auto-Optimized Firewall Policy ได้ทำการตรวจสอบจากฐานข้อมูลแล้วว่าเป็นพอร์ตที่ไม่มีความเสี่ยงต่อการบุกรุกจึงได้ทำการกำหนด Action เองโดยอัตโนมัติให้เป็น ACCEPT

จากนั้นก็จะทำการเปรียบเทียบกันดังภาพที่ 6

```
Rule Source IP    Dest. IP      Dest.Port     Action
_____________________________________________________
1    192.168.0.1  10.114.0.0      21          DENY
กับ
iptables -A INPUT -s 192.168.0.13 -d 129.110.31.7 -p UDP -dport 21 -j ACCEPT
```

***ภาพที่ 6*** *การเปรียบเทียบพอร์ตที่ไม่มีความเสี่ยงต่อการบุกรุก*

จากการเปรียบเทียบปรากฏว่าพอร์ตตรงกัน ขั้นต่อไปก็จะทำการนำ Action เปรียบเทียบของทั้งฟอร์มของ IPTables และ Rule ที่อยู่ใน Rule List ที่พอร์ตตรงกัน ดังภาพที่ 7

```
Rule Source IP    Dest. IP      Dest.Port     Action
_____________________________________________________
1    192.168.0.1  10.114.0.0      21          DENY
กับ
iptables -A INPUT -s 192.168.0.13 -d 129.110.31.7 -p UDP -dport 21 -j ACCEPT
```

***ภาพที่ 7*** *การนำ Action เปรียบเทียบของทั้งฟอร์มของ IPTables และ Rule ที่อยู่ใน Rule List ที่พอร์ตตรงกัน*

จากการเปรียบเทียบปรากฏว่า Action ของทั้งฟอร์มคำสั่งของ IPTables และ Rule เดิม ที่อยู่ใน Rule List ไม่ตรงกัน ซึ่งหมายความว่า กฎ (Rule) ที่ฟอร์มเป็นชุดคำสั่ง IPTables จะไม่สามารถจัดกลุ่มเข้ากับ Rule นี้ได้ จึงทำการบวก Rule อีก 1 เพื่อจะทำการตรวจสอบ Rule ถัดไปโดยใช้อัลกอริทึมแบบเดิม ดังภาพที่ 8

```
Rule Source IP    Dest. IP      Dest.Port     Action                    AND
_____________________________________________________
2    192.168.0.1  10.114.0.0/30   21-22       DENY
กับ
iptables -A INPUT -s 192.168.0.13 -d 129.110.31.7 -p UDP -dport 21 -j ACCEPT   XOR
```

***ภาพที่ 8*** *การตรวจสอบ Rule ถัดไปโดยใช้อัลกอริทึมแบบเดิม*

จากการเปรียบเทียบปรากฏว่าพอร์ตตรงกัน ขั้นต่อไปก็จะทำการนำ Action เปรียบเทียบของทั้งฟอร์มของ IPTables และ Rule ที่อยู่ใน Rule List ที่พอร์ตตรงกันนั้น ดังภาพที่ 9

| Rule | Source IP | Dest. IP | Dest.Port | Action |
|---|---|---|---|---|
| 2 | 192.168.0.1 | 10.114.0.0/30 | 21-22 | DENY |

กับ

iptables -A INPUT -s 192.168.0.13 -d 129.110.31.7 -p UDP -dport 21 -j ACCEPT

***ภาพที่ 9*** *การนำ Action เปรียบเทียบของทั้งฟอร์มของ IPTables และ Rule ที่อยู่ใน Rule List ที่พอร์ตตรงกัน*

จากการเปรียบเทียบปรากฏว่า Action ตรงกันขั้นต่อไปก็จะทำการนำ source เปรียบเทียบของทั้งฟอร์มของ IPTables และ Rule ที่อยู่ใน Rule List ที่ Action ตรงกันนั้น ดังภาพที่ 10

| Rule | Source IP | Dest. IP | Dest.Port | Action |
|---|---|---|---|---|
| 2 | 192.168.0.1 | 10.114.0.0/30 | 21-22 | DENY |

กับ

iptables -A INPUT -s 192.168.0.13 -d 129.110.31.7 -p UDP -dport 21 -j ACCEPT

***ภาพที่ 10*** *การนำ source เปรียบเทียบของทั้งฟอร์มของ IPTables และ Rule ที่อยู่ใน Rule List ที่ Action ตรงกัน*

ใน Rule หนึ่งๆ หรือว่าใน Group ของ Policy สามารถที่จะทำหาช่วงของ IP Address เริ่มต้น และ IP Address สิ้นสุดได้ก็จะทำให้รู้ว่า IP Address ของต้นทางหรือปลายทางของ Packet ที่ต้องการเชื่อมต่อกับเครือข่ายภายนอกที่ฟอร์มตัวอยู่ในรูปแบบคำสั่งของ IPTables อยู่ในช่วงนั้น ๆ หรือไม่ ซึ่งหลังจากที่เปรียบเทียบพอร์ตกับ Action แล้วจะทำให้ลดส่วนของกฎ (Rule) ที่ไม่เกี่ยวข้องไปได้มากที่เดียว โดยการพิจารณาจะทำดังนี้

ลำดับแรกจาก Source IP Address ของ Rule นั้นโปรแกรมจะทำการคำนวณเพื่อหาช่วงของ IP Address โดยมีวิธีการคำนวณดังนี้

Source IP Address of Rule 192.168.0.1

จะทำการแปลงเป็นเลขฐานสองก่อน โดยเมื่อแปลงออกมาแล้วจะได้ดังนี้

วิธีการคำนวณหา Subnet Address

| | |
|---|---|
| 11000000.10101000.00000000.00000001 | IP Address |
| 11111111.11111111.11111111.00000000 | Mark |
| 11000000.10101000.00000000.00000000 | Subnet Address |

จากนั้นก็คำนวณหา Broadcast ดังนี้

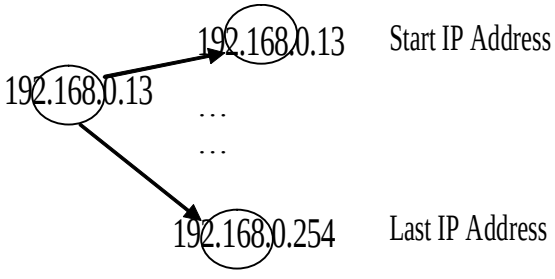
| | |
|---|---|
| 11000000.10101000.00000000.00000001 | IP Address |
| 00000000.00000000.00000000.11111111 | Mark Inverse |
| 11000000.10101000.00000000.11111110 | Broadcast |

เมื่อได้ทั้ง Subnet Address และ Broadcast ก็จะสามารถรู้ช่วงของ IP Address ของ กฎที่อยู่ใน Rule List ได้ ดังนั้นช่วงของ IP Address 192.168.0.0 ก็จะได้ Last IP Address เป็น 192.168.0.254

เมื่อได้ช่วงของ IP Address ของต้นทางที่จะพิจารณาเรียบร้อยแล้วจากนั้นก็นำ IP Address ต้นทางของ Packet ที่ต้องการเชื่อมต่อไปพิจารณาว่าอยู่ในช่วงของ IPAddress นั้น ๆ หรือไม่ ดังภาพที่ 11


***ภาพที่ 11*** *IP Address ต้นทางของ Packet*

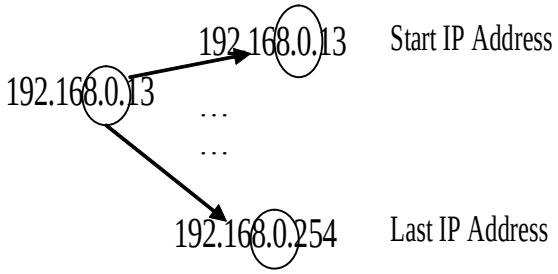
จากการเปรียบเทียบถ้าต่างกันก็ทำการตรวจสอบกับกฎ (Rule) ถัดไปได้ทันทีโดยไปเริ่มพิจารณาใหม่ตั้งแต่พอร์ตเหมือนเดิม และพิจารณามาโดยใช้อัลกอริทึมเดิม แต่ถ้าในกรณีที่ตรวจสอบแล้วตรงกันนั้นจะทำการพิจารณาในส่วนที่ย่อยลงไปอีกคือ จะตรวจสอบตำแหน่งถัดมา ดังภาพที่ 12




***ภาพที่ 12*** *การตรวจสอบกับกฎ และพิจารณาโดยใช้อัลกอริทึมเดิม*

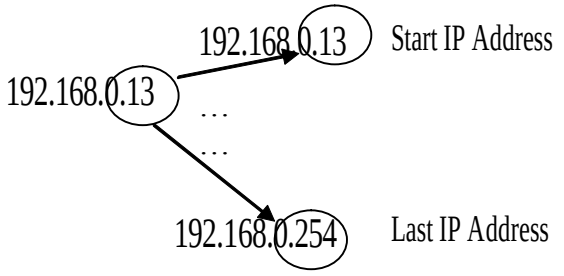
จากการเปรียบเทียบถ้าต่างกันก็จะทำการบันทึก IP Address ไว้เพื่อพิจารณาว่าสามารถที่จะทำการยุบรวมกฎ (Rule Combination) ได้หรือไม่หลังจากที่ตรวจสอบกับ Rule จนหมด

ทั้ง Rule List แล้ว หลังจากที่บันทึกเรียบร้อยแล้วก็จะทำการตรวจสอบกับกฎ (Rule) ลำดับต่อไป แต้ถ้าเปรียบเทียบแล้วปรากฏว่าตรงกันอีกก็จะทำการพิจารณาในส่วนที่ย่อยลงไปอีกคือจะตรวจสอบตำแหน่งถัดมา ดังภาพที่ 13




***ภาพที่ 13*** *การตรวจสอบกับกฎ โดยตรวจสอบตำแหน่งถัดมา*

จากการเปรียบเทียบถ้าต่างกันก็จะทำการบันทึก IP Address ไว้เพื่อพิจารณาว่าสามารถที่จะทำการยุบรวมกฎ (Rule Combination) ได้หรือไม่หลังจากที่ตรวจสอบกับ Rule จนหมดทั้ง Rule List แล้ว หลังจากที่บันทึกเรียบร้อยแล้วก็จะทำการตรวจสอบกับกฎ (Rule) ลำดับต่อไป แต้ถ้าเปรียบเทียบแล้วปรากฏว่าตรงกันอีกก็จะทำการพิจารณาในส่วนที่ย่อยลงไปอีกคือ จะตรวจสอบตำแหน่งถัดมา ดังภาพที่ 14




***ภาพที่ 14*** *การตรวจสอบกับกฎ โดยตรวจสอบตำแหน่งถัดมา*

จากการเปรียบเทียบถ้าต่างกันก็จะทำการบันทึก IP Address ไว้เพื่อพิจารณาว่าสามารถที่จะทำการยุบรวมกฎ (Rule Combination) ได้หรือไม่หลังจากที่ตรวจสอบกับ Rule จนหมดทั้ง Rule List แล้ว หลังจากที่บันทึกเรียบร้อยแล้วก็จะทำการตรวจสอบกับกฎ (Rule) ลำดับต่อไป แต้ถ้าเปรียบเทียบแล้วปรากฏว่าตรงกันอีกก็ทำการเปรียบเทียบที่ Destination IP Address โดยวิธีการพิจารณาจะกระทำเหมือนการตรวจสอบที่ Source IP Address โดยการพิจารณาจะทำการพิจารณาดังนี้

Destination IP Address of Rule 10.114.0.0/30

จะทำการแปลงเป็นเลขฐานสองก่อน โดยเมื่อแปลงออกมาแล้วจะได้ดังนี้

วิธีการคำนวณหา Subnet Address

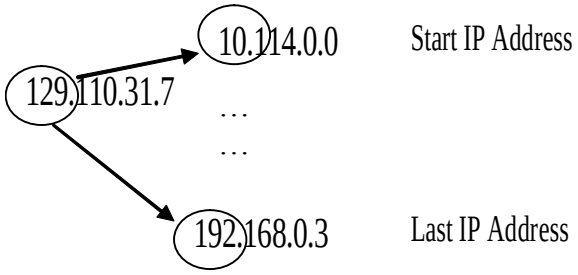
00001010.01110010.00000000.00000000 IP Address
AND 11111111.11111111.11111111.11111100 Mark
00001010.01110010.00000000.00000000 Address

จากนั้นก็คำนวณหา Broadcast ดังนี้

00001010.01110010.00000000.00000000 IP Address
XOR 00000000.00000000.00000000.00000011 Mark Inverse
00001010.01110010.00000000.00000011 Broadcast

เมื่อได้ทั้ง Subnet Address และ Broadcast ก็จะสามารถรู้ IP เริ่มต้น และ IP สุดท้าย ของ Destination ได้ นั่นหมายความว่าสามารถที่จะรู้ช่วงของ IP แล้ว

ดังนั้นช่วงของ IP Address of Destination จะได้ 10.114.0.0 ถึง 10.114.0.3 เมื่อได้ช่วงของ IP Address ของต้นทางที่เราจะพิจารณาเรียบร้อยแล้วจากนั้นก็นำ IP Address ต้นทางของ Packet ที่ต้องการเชื่อมต่อไปพิจารณาว่าอยู่ในช่วงของ IP Address นั้น ๆ หรือไม่ ดังภาพที่ 15




***ภาพที่ 15*** *IP Address ต้นทางของ Packet*

จากการเปรียบเทียบถ้าต่างกันก็ทำการตรวจสอบกับกฎ (Rule) ถัดไปได้ทันทีโดยไปเริ่มพิจารณาใหม่ตั้งแต่พอร์ตเหมือนเดิม และพิจารณามาโดยใช้อัลกอริทึมเดิม แต่ถ้าในกรณีที่ตรวจสอบแล้วตรงกันนั้นจะทำการพิจารณาในส่วนที่ย่อยลงไปอีกคือจะตรวจสอบตำแหน่งถัดมา ในที่นี้เมื่อทำการตรวจสอบแล้วปรากฏว่าไม่ตรงกันโปรแกรม Auto-Optimized Firewall Policy Management ก็จะไม่สนใจลำดับต่อไปจะทำการตรวจสอบที่กฎ (Rule) ถัดไปทันทีเลยเพราะจะไม่สามารถยุบรวมกฎ (Rule Combination) เข้าไว้ในกลุ่มเดียวกันได้อยู่แล้วแต่ถ้าในกรณีที่ตรวจสอบแล้วตรงกันก็แสดงว่า IP Address ของ Packet ที่ต้องการเชื่อมต่อไปยังเครือข่ายภายนอกนั้นอยู่ในช่วง IP Address นี้ ก็สามารถที่จะทำการยุบรวมกฎ (Rule Combination) เข้ากับ กฎ (Rule) นี้ได้ จากตารางเป็นการแสดงให้เห็นว่าเมื่อมี Packet ต้องการเชื่อมต่อไปยังเครือข่ายภายนอกแล้วโปรแกรม Auto-Optimized Firewall Policy Management ก็ได้ทำการจัดการกฎ (Rule) ของ

ไฟร์วอลล์ตามที่กล่าวไปข้างต้นแล้ว จะไม่ทำให้กฎ (Rule) ของไฟร์วอลล์เพิ่มจำนวนขึ้นตามตารางที่ 6

***ตารางที่ 6*** *การยุบรวมกฎแล้วไม่ทำให้กฎของไฟร์วอลล์เพิ่ม*

| Rule | Source IP | Dest. IP | Dest Port | Action |
|---|---|---|---|---|
| 1 | 192.168.0.1 | 10.114.0.0 | 21 | DENY |
| 2 | 192.168.0.1 | 10.114.0.0/30 | 21-22 | ACCEPT |
| 3 | 192.168.0.1 | 10.114.0.0/30 | 23-24 | ACCEPT |
| 4 | 192.168.0.1 | any | 80 | ACCEPT |
| 5 | any | any | any | DENY |

## 5. ผลการดำเนินงาน

### 5.1 การกำจัด Shadowing Anomaly

การกำจัด Shadowing Anomaly นั้น สามารถทำได้โดยการลบ Shadowed Rule ทิ้งไป ซึ่งการลบ Shadowed Rule นั้นจะไม่ทำให้ Firewall Policy เปลี่ยน ในกรณีที่ใน Rule List มี Shadowed Rule มากกว่า 1 Shadowed Rule เราสามารถที่จะเลือกลบ Shadowed Rule ใดก่อนก็ได้

(A)

| | Rule. | Protocol | Source IP | Src_Port | Destination IP | Destination Port | Action |
|---|---|---|---|---|---|---|---|
| | 1 | TCP | 10.0.0.5 | ANY | 20.0.0.1 | 20-21 | ACCEPT |
| | 2 | TCP | 10.0.0.5 | ANY | 20.0.0.1 | 21-22 | ACCEPT |
| | 3 | TCP | 10.0.0.5 | ANY | 20.0.0.1 | 23-25 | DENY |
| Shadowed | 4 | TCP | 10.0.0.5 | ANY | 20.0.0.1 | 22-23 | ACCEPT |
| | 5 | TCP | 10.0.0.0/30 | ANY | 20.0.0.1 | 80 | DENY |
| | 6 | TCP | 10.0.0.1 | ANY | 20.0.0.0/30 | 80 | ACCEPT |
| | 7 | TCP | 10.0.0.0/30 | ANY | 20.0.0.0/30 | 80 | DENY |
| | 8 | TCP | 10.0.0.1 | ANY | 20.0.0.1 | 80-143 | ACCEPT |
| | 9 | TCP | 10.0.0.1,10.0.0.4 | ANY | 20.0.0.1 | 80 | DENY |
| | 10 | TCP | 10.0.0.0/24 | ANY | 20.0.0.0/24 | ANY | ACCEPT |
| Shadowed | 11 | TCP | 10.0.0.0/24 | ANY | 20.0.0.0/24 | 80 | DENY |
| | 12 | ANY | ANY | ANY | ANY | ANY | DENY |

(B)

| | Rule. | Protocol | Source IP | Src_Port | Destination IP | Destination Port | Action |
|---|---|---|---|---|---|---|---|
| | 1 | TCP | 10.0.0.5 | ANY | 20.0.0.1 | 20-21 | ACCEPT |
| | 2 | TCP | 10.0.0.5 | ANY | 20.0.0.1 | 21-22 | ACCEPT |
| | 3 | TCP | 10.0.0.5 | ANY | 20.0.0.1 | 23-25 | DENY |
| | 4 | TCP | 10.0.0.0/30 | ANY | 20.0.0.1 | 80 | DENY |
| | 5 | TCP | 10.0.0.1 | ANY | 20.0.0.0/30 | 80 | ACCEPT |
| | 6 | TCP | 10.0.0.0/30 | ANY | 20.0.0.0/30 | 80 | DENY |
| | 7 | TCP | 10.0.0.1 | ANY | 20.0.0.1 | 80-143 | ACCEPT |
| | 8 | TCP | 10.0.0.1,10.0.0.4 | ANY | 20.0.0.1 | 80 | DENY |
| | 9 | TCP | 10.0.0.0/24 | ANY | 20.0.0.0/24 | ANY | ACCEPT |
| | 10 | ANY | ANY | ANY | ANY | ANY | DENY |

***ภาพที่ 16*** *แสดง Rule List ก่อน และหลังกำจัด Shadowing Anomaly*

### 5.2 การกำจัด Correlation Anomaly

การกำจัด Correlation Anomaly นั้น ทำได้โดยกำจัดส่วนที่เกิดการ Correlate กันออกไป ซึ่งอาจจะทำโดยการปรับปรุง (Modify) Rule ที่อยู่ลำดับก่อน หรือปรับปรุง Rule ที่อยู่ลำดับหลังก็ได้ หรือในบางครั้งอาจจะต้องปรับปรุงทั้งสอง Rule

***ตารางที่ 7*** *แสดง Rule List ก่อนกำจัด Correlation Anomaly*

| Order | Source IP | Destination. IP | Destination Port | Action |
|---|---|---|---|---|
| 1 | 10.0.0.3 | 20.0.0.1 | any | DENY |
| 2 | 10.0.0.3/30 | 20.0.0.1 | 80 | ACCEPT |

***ตารางที่ 8*** *แสดง Rule List หลังกำจัด Correlation Anomaly*

| Order | Source IP | Destination. IP | Destination Port | Action |
|---|---|---|---|---|
| 1 | 10.0.0.3 | 20.0.0.1 | any | DENY |
| 2 | **10.0.0.2** | 20.0.0.1 | 80 | ACCEPT |
| 3 | **10.0.0.0/31** | 20.0.0.1 | 80 | ACCEPT |

(A)

| | Rule. | Protocol | Source IP | Src_Port | Destination IP | Destination Port | Action |
|---|---|---|---|---|---|---|---|
| Redundant | 1 | TCP | 10.0.0.5 | ANY | 20.0.0.1 | 20-21 | ACCEPT |
| Redundant | 2 | TCP | 10.0.0.5 | ANY | 20.0.0.1 | 21-22 | ACCEPT |
| | 3 | TCP | 10.0.0.5 | ANY | 20.0.0.1 | 23-25 | DENY |
| Redundant | 4 | TCP | 10.0.0.5 | ANY | 20.0.0.1 | 22-23 | ACCEPT |
| | 5 | TCP | 10.0.0.0/30 | ANY | 20.0.0.1 | 80 | DENY |
| | 6 | TCP | 10.0.0.1 | ANY | 20.0.0.0/30 | 80 | ACCEPT |
| | 7 | TCP | 10.0.0.0/30 | ANY | 20.0.0.0/30 | 80 | DENY |
| | 8 | TCP | 10.0.0.1 | ANY | 20.0.0.1 | 80-143 | ACCEPT |
| | 9 | TCP | 10.0.0.1,10.0.0.4 | ANY | 20.0.0.1 | 80 | DENY |
| | 10 | TCP | 10.0.0.0/24 | ANY | 20.0.0.0/24 | ANY | ACCEPT |
| Redundant | 11 | TCP | 10.0.0.0/24 | ANY | 20.0.0.0/24 | 80 | DENY |
| | 12 | ANY | ANY | ANY | ANY | ANY | DENY |

(B)

| | Rule. | Protocol | Source IP | Src_Port | Destination IP | Destination Port | Action |
|---|---|---|---|---|---|---|---|
| | 1 | TCP | 10.0.0.5 | ANY | 20.0.0.1 | 23-25 | DENY |
| | 2 | TCP | 10.0.0.0/30 | ANY | 20.0.0.1 | 80 | DENY |
| | 3 | TCP | 10.0.0.1 | ANY | 20.0.0.0/30 | 80 | ACCEPT |
| | 4 | TCP | 10.0.0.0/30 | ANY | 20.0.0.0/30 | 80 | DENY |
| | 5 | TCP | 10.0.0.1 | ANY | 20.0.0.1 | 80-143 | ACCEPT |
| | 6 | TCP | 10.0.0.1,10.0.0.4 | ANY | 20.0.0.1 | 80 | DENY |
| | 7 | TCP | 10.0.0.0/24 | ANY | 20.0.0.0/24 | ANY | ACCEPT |
| | 8 | ANY | ANY | ANY | ANY | ANY | DENY |

***ภาพที่ 17*** *แสดงก่อนและหลังกำจัด Redundancy Anomaly*

### 5.3 การกำจัด Generalization Anomaly

Generalization Anomaly ถือว่าเป็น Anomaly ที่สามารถเกิดขึ้นได้เสมอ และเป็นเรื่องธรรมดา สำหรับการออกแบบกฎของไฟร์วอลล์ ซึ่งการออกแบบกฎโดยทั่วไปแล้วปกติจะต้องมี Generalization Anomaly แทบทั้งสิ้น Generalization Anomaly ไม่จำเป็นที่จะต้องมีการกำจัดออกจากกฎของไฟร์วอลล์

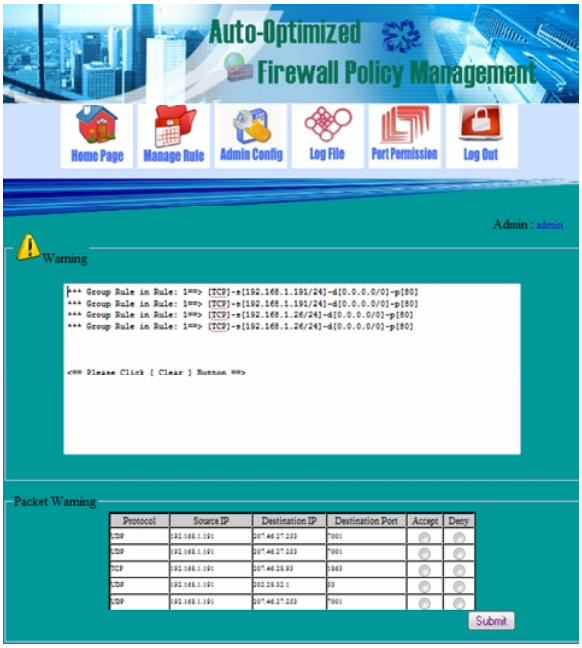
เป็นเพียงสิ่งที่ช่วยให้ผู้ออกแบบไฟร์วอลล์ทราบว่า Rule คู่ใดที่เกิด Generalization Anomaly ขึ้นแล้วเมื่อทำการสลับตำแหน่งจะทำให้ Policy เปลี่ยน ดังนั้น Generalization Anomaly จึงไม่จำเป็นที่จะต้องกำจัด

**5.4 การกำจัด Redundancy Anomaly**

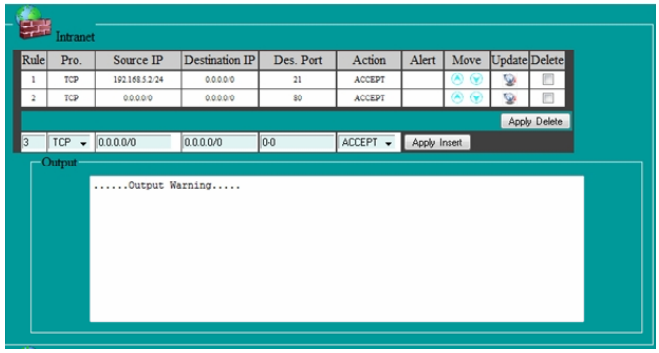
การกำจัด Redundancy Anomaly ทำได้โดยการลบ Redundant Rule ทิ้งไป ซึ่งจะไม่มีผลทำให้ Policy เปลี่ยน (ยกเว้นบางกรณี) ในกรณีที่ Rule List มี Redundant Rule กว่าหนึ่ง Rule เราสามารถที่จะเลือกลบ Redundant Rule ใดออกไปก่อนก็ได้ ซึ่งจะไม่มีผลต่อ Policy

**5.5 แสดงโปรแกรมต้นแบบของ Auto-Optimized Firewall Policy Management**

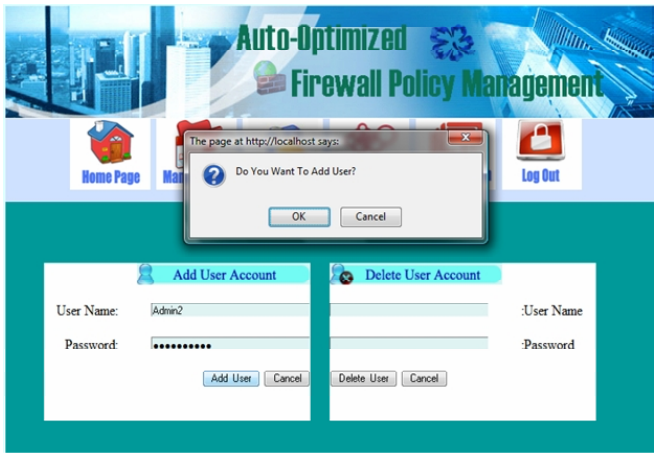

*ภาพที่ 18 หน้า Login ของระบบ*




*ภาพที่ 19 เมื่อมี Packet กระทบไฟร์วอลล์สร้างกฎอัตโนมัติ*




*ภาพที่ 20 ผู้ดูแลระบบสามารถเพิ่มกฎเองได้*




*ภาพที่ 21 ฟังก์ชันการเพิ่ม-ลบ ผู้ใช้งาน*

**6. สรุปผล**

ในงานวิจัยนี้ได้เสนอวิธีการจัดการกฎของไฟร์วอลล์แบบกึ่งอัตโนมัติ โดยมีความสามารถคือ สามารถยุบรวมกฎ ขจัดความซับซ้อน และจัดกลุ่มของกฎใหม่ตามความเหมาะสมกับการใช้งานในแต่ละแอพพลิเคชัน ซึ่งตัวแบบดังกล่าวสามารถตัดสินใจสร้างกฎ และปรับแต่งกฎโดยอาศัยทฤษฎีการกำจัดแบบ anomaly detection [3] จุดประสงค์หลักในการสร้างวิธีการดังกล่าวเพื่อลดความผิดพลาดที่เกิดจากมนุษย์ในการกำหนดกฎของไฟร์วอลล์ และเพื่อเพิ่มขีดความสามารถของไฟร์วอลล์ให้มีความฉลาดในการตัดสินใจกับข้อมูลที่ไหลผ่านเข้า และออกตัวมันเองได้เพิ่มมากขึ้น ในงานวิจัยนี้ได้ทำการสร้างตัวต้นแบบไฟร์วอลล์ที่ทำงานกึ่งอัตโนมัติดังกล่าวด้วยโดยอาศัยซอฟต์แวร์ที่ทำหน้าที่เป็นไฟร์วอลล์บนระบบปฏิบัติการลินุกซ์คือ IPTABLE ทำงานร่วมกับกลไกการตรวจสอบข้อมูลที่กำกวม (Packet Anomaly Detection [3]) ผลจากการทดสอบโปรแกรมสามารถขจัดความกำกวม และสามารถขจัดความซับซ้อนของกฎได้ตรงตามที่ออกแบบไว้ได้เป็นอย่างดี แต่มีขีดจำกัดในเรื่องของความเร็วในการลดกฎเนื่องจาก Packet ที่เข้ามาในระบบจะสูงตามขนาดของระบบเครือข่าย ดังนั้น

เวลาในการยุบรวมกฎจะต้องคำนวณให้เหมาะสมกับข้อมูลที่สื่อสารอยู่บนระบบเครือข่าย

**7. เอกสารอ้างอิง**


[1] Chotipat Pornavalai and T. Chomsiri. **Firewall Policy Analyzing by Relational Algebra**. in The 2004 International Technical Conference on Circuits/ Systems, Computers and Communications (ITC-CSCC 2004). JULY 2004.

[2] Chomsiri T, P.C. Firewall Rules Analysis. **in The 2006 World Congress in Computer Science Computer Engineering, and Applied Computing**. 2006 June 26-29. Las Vegas, USA.

[3] Ehab Al-Shaer and Hazem Hamed. **Firewall Policy Advisor for anomaly Detection and Rule Editing**. in IEEE/IFIP Integrated Management IM'2003. March 2003.

[4] Abraham Silberschatz, H.F.K., Sudharsan S.,. **Database System Concepts**, 3rd Edition: Tata McGraw-Hill, 1997.

[5] ศูนย์ประสานงานการรักษาความปลอดภัยคอมพิวเตอร์ประเทศไทย, **ความรู้พื้นฐานเกี่ยวกับไฟร์วอลล์**, http://thaicert.or.th/paper/firewall/fwbasics.php.

[6] GWirken.nl, **Firewall**, http://www.gwirken.nl/content/veiliginternet/firewall.

[7] Abraham Silberschatz, H.F.K., Sudharsan S.,, **Database System Concepts**. 3rd Edition ed. 1997: Tata McGraw-Hill.

[8] Forouzan, B.A., **Data Communications and Networking**. 3 ed. 2003: McGraw-Hill Professional.

[9] Shay, W.A., **Understanding Data Communications and Networks**. 3 ed. October 30, 2003.

[10] Inc., C.S., **Cisco Networking Academy (CCAI)**. 1992 - 2007.

[11] คุ้มมะณี, ส. and ธ. ชมศิริ, **เรียนรู้เครือข่ายและอุปกรณ์ Cisso ด้วยโปรแกรม Simulation**. 2550: โปรวิชั่น

[12] Cisco Systems, I., **CCNP 1: Advanced IP Addressing Management**. Aug 27, 2004: Cisco Press.