บทที่ 2
ทฤษฎีและงานวิจัยที่เกี่ยวข้อง

2.1 ทฤษฎีที่เกี่ยวข้อง

2.1.1 ไฟร์วอลล์ (Firewall)

ความหมายของไฟร์วอลล์ (Definition of a firewall) ไฟร์วอลล์ เป็นเครื่องมือหรือระบบของระบบคอมพิวเตอร์ที่ทำหน้าที่ป้องกันเครือข่ายจากผู้ที่ไม่ได้รับ อนุญาต ไม่ให้สามารถมาใช้หรือมองเห็นข้อมูลหรือเครือข่ายคอมพิวเตอร์ได้ ถือว่าเป็นระบบที่ทำหน้าที่รักษา ความปลอดภัยสำหรับเครือข่ายคอมพิวเตอร์ที่มีความสำคัญยิ่ง เพราะไฟร์วอลล์สามารถตรวจสอบและป้องกัน การบุกรุกหรือการโจมตีที่เป็นอันตรายต่อระบบเครือข่าย โดยขั้นตอนในการทำงานของระบบไฟร์วอลล์จะทำ หน้าที่ตรวจสอบการบุกรุกหรือการโจมตีโดยการค้นหาความผิดปกติ โดยการเปรียบเทียบกฎที่ถูกกำหนดไว้ เมื่อตรวจสอบพบความผิดปกติของข้อมูล ไฟร์วอลล์ก็จะทำการปิดกั้นไม่ให้ข้อมูลเหล่านั้นผ่านไปได้ ในทาง กลับกัน ถ้าข้อมูลที่ถูกตรวจสอบไม่มีสิ่งผิดปกติใด ๆ ไฟร์วอลล์ก็จะอนุญาตให้ข้อมูลเหล่านั้นผ่านไปทำงานที่ ต้องการได้

2.1.1.1 IPTables

Linux สามารถใช้งานเป็นไฟร์วอลล์ได้ตั้งแต่เคอร์เนล 1.1 ซึ่งเป็นเวอร์ชันแรก โดย Alan Cox ใช้ชื่อว่า ipfw (จาก BSD) ต่อมา Linux 2.0 ได้ถูกพัฒนาและปรับปรุงได้เครื่องมือที่มีชื่อว่า ipfwadm โดยเครื่องมือชิ้นนี้อนุญาตให้ผู้ใช้สามารถควบคุม filtering rule ได้ และต่อมา Linux 2.2 ก็ได้สร้างเครื่องมือตัวใหม่ที่มีชื่อว่า ipchains ซึ่งเผยแพร่ในปี ค.ศ.1998 โดย Rusty Russell และทีมงาน[7] ทั้งนี้ ipchains นี้ถือได้ว่าเป็นพัฒนาการขั้นที่สามของ Linux Firewall จวบจนกระทั่งในปัจจุบัน ก็มีการพัฒนา Netfilter และ IPTables ซึ่งถือได้ว่าเป็นพัฒนาการชั้นที่สี่ของ Linux Firewall

รูปแบบการใช้งาน IPTables เบื้องต้น

IPTables จะมีรูปแบบการใช้งานดังนี้ [7]

Iptables [table] <command><match><target/jump>

[table] หมายถึง ตารางที่ต้องการระบุ

<command> จะเป็นตัวสั่งให้ iptables ทำในสิ่งที่ต้องการ

<match> เป็นส่วนที่ใช้ตรวจสอบว่า Packet มีข้อมูลตรง (Match) กับที่ระบุไว้หรือไม่

<target/jump> เป็นตัวระบุเมื่อเจอ Packet ที่ match ก็จะกระทำ (Action) ตามที่ระบุไว้

Table

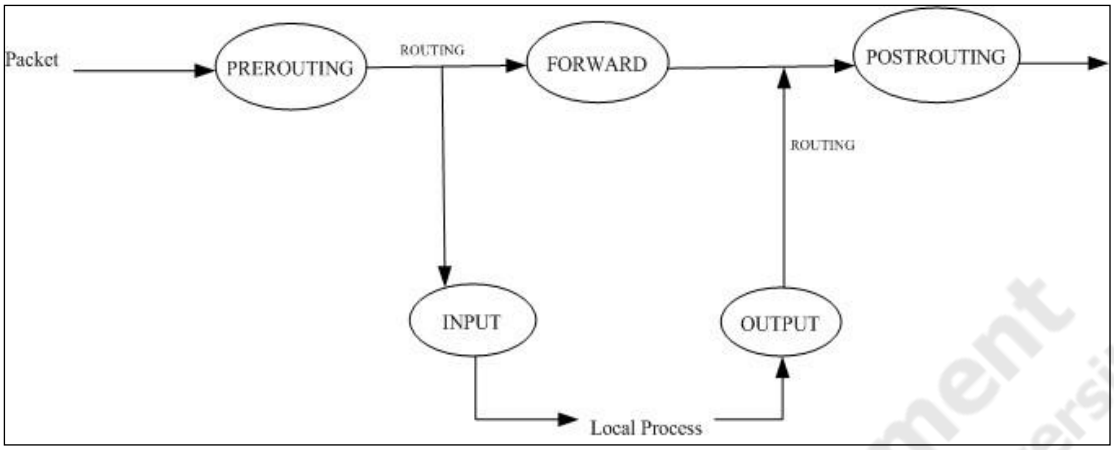
IPtables สามารถทำงานได้กับตาราง (Table) 3 ตารางหลัก สามารถระบุตารางได้โดยใช้ option –t ตามด้วยชื่อ table ดังนี้

1. Filter Table เป็นตารางที่ใช้งานบ่อยที่สุดเป็นจุดที่ใช้ในการตรวจสอบและควบคุมการผ่านเข้าออกของ packet packet มี 3 built-in chain คือ INPUT, OUTPUT, FORWARD โดยเมื่อ Packet เข้ามาในระบบ จะทำการเข้าไปยัง Routing Decision เพื่อตัดสินใจว่า Packet จะถูกส่งไปที่ใด

- ในกรณีที่ Packet ถูกส่งผ่านไปยังเครื่องอื่น Packet นั้นจะต้องถูกตรวจสอบโดย Rule ใน FORWARD chain

- ถ้า Packet นั้นมีเป้าหมายเป็นเครื่องปัจจุบัน (เครื่องที่ทำการรัน IPTables อยู่นี้ เรียกอีกอย่างว่า Linux Box) ตัว Packet จะถูกตรวจสอบโดย Rule ใน INPUT Chain

- และในกรณีที่ Packet ถูกสร้างจากเครื่องปัจจุบัน (Linux Box) ตัว Packet จะถูกตรวจสอบจาก Rule ใน OUTPUT Chain ก่อนที่จะส่งออกไป ดังภาพประกอบที่ 2.4



ภาพประกอบที่ 2.1 แสดงเส้นทางการเดินของ Packet เมื่อเข้ามาในระบบ (Filter Table)

2. **Mangle Table** เป็นตารางที่ใช้สำหรับแก้ไขข้อมูล TOS, TTL, MARK ของ Packet ซึ่งโดยปกติแล้วแทบจะไม่ได้ใช้งาน และไม่ควรทำ Packet Filtering หรือกรอง Packet ที่ตารางนี้

3. Nat Table เป็นตารางที่ใช้สำหรับทำ Network Address Translation เช่น เปลี่ยนค่า Source IP Address, Destination IP Address จุดสำคัญอีกอย่างหนึ่งที่ต้องรู้คือ มีเพียง Packet แรกเท่านั้นที่เข้ามาที่ Chain นี้ส่วน Packet ถัดไปนั้นจะถูกกระทำเหมือนที่ Packet แรกได้รับ จึงไม่ควรทำ Packet Filtering ที่ Chain นี้เช่นกัน

Command

IPtables มี Command ที่ใช้งานต่างๆ ดังนี้

-**A** เพิ่ม Rule ใหม่ต่อท้าย Chain

เช่น iptables –A INPUT –p ALL –I eth0 –j ACCEPT

-**D** ลบ Rule (Delete Rule)

เช่น iptables –D INPUT –dport 80 –j DROP

-**I** เพิ่ม Rule ใหม่ใน Chain (Insert Rule)

เช่น iptables –I OUTPUT –p All –s 127.0.0.1/32 –j ACCEPT

-R แทนที่ Rule เดิม ด้วย Rule ใหม่

-**L** แสดง Rule ทั้งหมดใน Chain (ถ้าไม่ระบุ Chain จะแสดง Rule ทั้งหมด

-**F** ลบ Rule ทั้งหมดใน Chain ทิ้ง เช่น iptables –F INPUT

-**N** ใช้สร้าง Chain ใหม่

เช่น iptables –N mychain

-**X** ลบ Chain ที่ไม่มี Rule ซึ่งสามารถลบ user-defined chain ที่ไม่มี Ruleได้ แต่ไม่สามารถลบ built-in chain ได้ เช่น iptables –X emptychaim

-**P** เปลี่ยน Default Policy ของ Chain ค่าที่ได้คือ ACCEPT, DROP ทั้งนี้ค่านี้มีความสำคัญอย่างมากเพราะหาก Packet ถูกส่งเข้ามาใน Chain แล้วและไม่ Match กับ Rule ใด Packet นั้นก็จะถูกตัดสินใจโดย Policy ของ Chain นั้นๆ

-E ใช้เปลี่ยนชื่อ Chain ใหม่และยังมี Command ที่ใช้ใน IPTables เป็นจำนวนมาก จากข้างต้นเป็นเพียงการยกตัวอย่างเท่านั้น

Match

เป็นส่วนที่ใช้ตรวจสอบว่า Packet มีข้อมูลตรงกัน (Match) กับที่ระบุไว้หรือไม่

การระบุ Source, Destination IP Address

สามารถระบุ Source IP Addressของ Packet โดยใช้ **–S** หรือ **–source** หรือ **–src** และสำหรับ Destination IP Address ก็ใช้ **–d** หรือ **–destination** หรือ **–dst** การระบุไอพีแอดเดรสนั้นสามารถกระทำได้ 4 แบบด้วยกัน คือ

1) ใช้ชื่อเต็มแทน เช่น localhost หรือ www.www.com

2) ระบุไอพีแอดเดรสโดยตรง เช่น 127.0.0.1 หรือ 202.44.204.33

3) ระบุเป็น group ของไอพีแอดเดรส เช่น 202.44.204.0/24 ซึ่งหมายถึงไอพีแอดเดรสตั้งแต่ 202.44.204.0 – 202.44.204.255

4) หรืออาจจะใช้ 202.44.204.0/255.255.255.0 แทน 202.44.204.0/24 ได้

การระบุโพรโทคอล

สามารถระบุโพรโตคอลที่ต้องการได้ดังนี้คือ TCP, UDP, ICMP และสามารถใช้ได้ทั้งอักษรเล็กหรือใหญ่ (ใช้ได้ทั้ง tcp หรือ TCP) เช่น –p TCP หรือ –p tcp

Target เมื่อมี Packet ที่ Match กับ Rule แล้ว ต้องกำหนด Target สำหรับ Packet ไว้ด้วย โดยปกติแล้วจะใช้ 2 Target คือ DROP และ ACCEPT นอกจากนี้ยังมี Target แบบอื่นได้อีก คือ

-user-defined chain เนื่องจาก IPTables อนุญาตให้ผู้ใช้สามารถสร้าง Chain ขึ้นมาใหม่ได้ นอกเหนือจาก built-in chainทั้งสาม ตัว ทั้งนี้จะต้องใช้อักษรตัวเล็กทั้งหมด

-new target

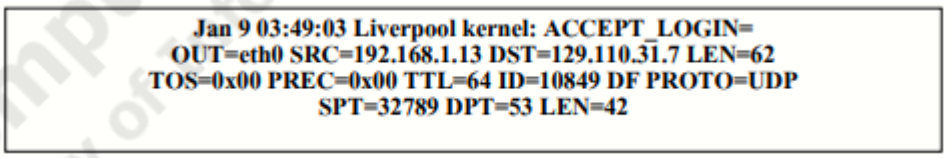
เป็น Target ที่สร้างเพิ่มเติมขึ้นมา

LOG เป็นโมดูลที่มีความสามารถในการเก็บข้อมูลลง Log สำหรับ Packet ที่ Match กับ Rule ที่ระบุ Target เป็น LOG มี option ที่ให้เลือกใช้งานดังนี้

--log-level เป็นการระบุ priority level ของ log

--log-prefix ตามด้วยชุดของตัวอักษรยาวไม่เกิน 29 ตัว ซึ่งชุดของตัวอักษรดังกล่าวจะไปปรากฏอยู่บน Log File

ภายใน Log File นั้นจะมีข้อมูลต่างๆของ Packet ที่ทำการร้องขอที่จะผ่านไฟร์วอลล์ ทำให้สามารถที่จะนำข้อมูลใน Log File มาเป็นข้อมูลในการจัดทำโครงงานนี้ ดังรูป



ภาพประกอบที่ 2.2 ตัวอย่างของ Linux Firewall Log

REJECT คล้ายกับ DROP เพียงแต่จะส่ง ICMP port unreachable กลับไปยังผู้ที่ส่ง Packet มา

-special built-in target RETURN กรณีที่ Packet Match กับ Rule ที่มี Target เป็น RETURN นั้นเสมือนกับเป็นคำสั่งให้ออกไปจาก Chain ปัจจุบัน QUEUE เป็น Chain พิเศษ ใช้สำหรับส่งต่อ Packet ไปยัง Application ที่เขียนขึ้นมารองรับโดยเฉพาะ นอกจากทฤษฎีที่เกี่ยวข้องกับ IPTables แล้วยังต้องศึกษาทฤษฎีที่เกี่ยวกับกฎของไฟร์วอลล์ด้วยเพราะกฎของ ไฟร์วอลล์เป็นสิ่งสำคัญในโครงงานครั้งนี้

ดังทฤษฎีต่อไปเป็นทฤษฎีการวิเคราะห์กฎของไฟร์วอลล์โดยใช้รีเล ชันแนลอัลจีบรา
ซึ่งทฤษฎีนี้จะช่วยสนับสนุนให้โครงงานครั้งนี้บรรลุวัตถุประสงค์ไปได้

2.1.1.2 การวิเคราะห์กฎของไฟร์วอลล์โดยใช้รีเลชันแนลอัลจีบรา

ในโครงงานนี้เป็นการจัดการกฎของไฟร์วอลล์ทฤษฎีการวิเคราะห์กฎของไฟร์วอลล์โดยใช้รีเลชันแนลอัลจีบราก็จะทำให้ประสิทธิภาพของไฟร์วอลล์ที่จะสร้างขึ้นใหม่ในโครงงานนี้มีประสิทธิภาพมากขึ้นรีเลชันแนลอัลจีบราเป็นทฤษฎีที่ใช้ในการสร้างกลไกภายในของ Relational Database Management System รีเลชัน (Relation) คือ สับเซตของคาร์ทีเซียนโปรดักของโดเมน

การดำเนินการระหว่างรีเลชันจะดำเนินการด้วยรีเลชันแนลโอเปอเรชัน (Relational Operation) เมื่อรีเลชันมีมากกว่าสองรีเลชันขึ้นไปมีโอเปอร์เรชันระหว่างกัน นักคณิตศาสตร์สามารถเขียนให้อยู่ในรูปพีชคณิต (Algebra) ซึ่งเรียกว่า รีเลชันแนลอัลจีบรา (Relational Algebra) [8] กฎของไฟร์วอลล์สามารถที่จะแม็ปให้อยู่ในรูปของรีเลชันโดยจะมีทฤษฎีดังนี้

ทฤษฎีที่ 1 Policy ของไฟร์วอลล์จะไม่เปลี่ยนถ้าทำการลบ Rule-x เมื่อ Rule-x เป็น Shadowed Rule

ทฤษฎีที่ 2 Policy ของไฟร์วอลล์จะไม่เปลี่ยนถ้าทำการสลับตำแหน่งกันระหว่าง Rule-x กับ Rule-y เมื่อ Rule-x กับ Rule-y เป็น consecutively non-correlated ระหว่างกัน

ทฤษฎีที่ 3 Policy ของไฟร์วอลล์จะไม่เปลี่ยนถ้าทำการสลับตำแหน่งกันระหว่าง Rule-x กับ Rule-y เมื่อ Rule-x อยู่ลำดับก่อน Rule-y และ Rule-x เป็น consecutively non-correlated downward ไปถึง Rule-y และ Rule-y เป็น consecutively non-correlated upward ขึ้นไปถึง Rule-x

ทฤษฎีที่ 4 Policy ของไฟร์วอลล์จะไม่เปลี่ยนถ้าทำการสลับกันระหว่าง Rule-x กับ Rule-y เมื่อ Rule-x กับ Rule-y เกิดการ correlate กัน และ Rule-x-y ถูกบัง และ Rule-x เป็น consecutively noncorrelated downward ลงไปถึง Rule-(y-1) เป็น consecutively non-correlated upward ขึ้นไปถึง Rule-(x-1)

ทฤษฎีที่ 5 Policy ของไฟร์วอลล์จะไม่เปลี่ยนถ้าทำการลบ Rule-x ออกไปจาก Rule list เมื่อ Rulex เป็น consecutively redundant ต่อ Rule-y

ทฤษฎีที่ 6 Policy ของไฟร์วอลล์จะไม่เปลี่ยนถ้าทำการลบ Rule-x ออกไปจาก Rule list เมื่อ Rulex ซ้ำซ้อนกับ Rule-y และ Rule-x เป็น consecutively non-correlated downward ไปยัง Rule-(x-1)

ทฤษฎีที่ 7 Rule-x กับ Rule-y สามารถยุบรวมกันเป็น Rule-z ได้โดยไม่ทำให้ Policy ของไฟร์วอลล์เปลี่ยน เมื่อ Rz คือ RxURy และ Action ของทั้งสองเหมือนกัน และ Rule-y อยู่ลำดับที่สูงกว่าและอยู่ลำดับที่ติดกันกับ Rule-x ทฤษฎีที่ 8 Policy ของไฟร์วอลล์จะไม่เปลี่ยนถ้าทำการแทรก consecutively redundant rule เข้าไป

การยุบรวมกฎ (Rule Combination)

เป็นการยุบรวมกฎหลายๆข้อรวมกันเพื่อลดกฎให้เหลือเป็นกฎข้อเดียว และไม่ทำให้ Policy เปลี่ยนแปลง

before

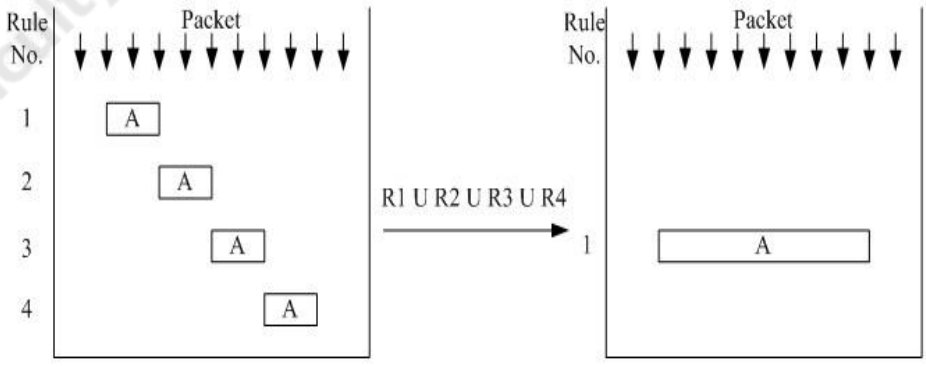
order	src_ip	dst_ip	dst_port	action
1	50.0.0.0/26	60.0.0.0/24	80	accept
2	50.0.0.64/26	60.0.0.0/24	80	accept
3	50.0.0.128/26	60.0.0.0/24	80	accept
4	50.0.0.192/26	60.0.0.0/24	80	accept

after

Order	src_ip	dst_ip	dst_port	action
1	50.0.0.0/24	60.0.0.0/24	80	accept

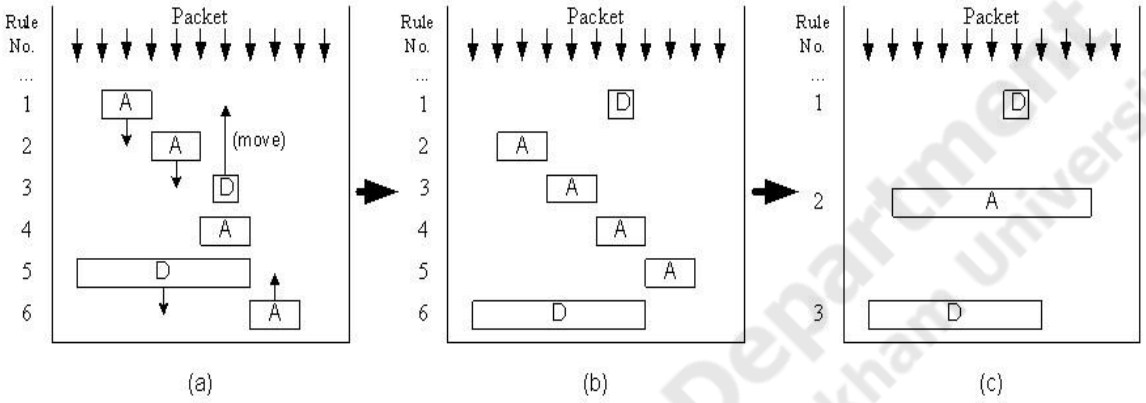
ภาพประกอบที่ 2.3 การรวมกฎ

หลักการคือจะทำการแม็ปจาก Rule ให้เป็น Relation แล้วนำ Relation ของกฎมาพิจารณา โดยจะพิจารณา Action และ Service ที่เหมือนกัน มาทำการ UNION กัน ดังภาพ



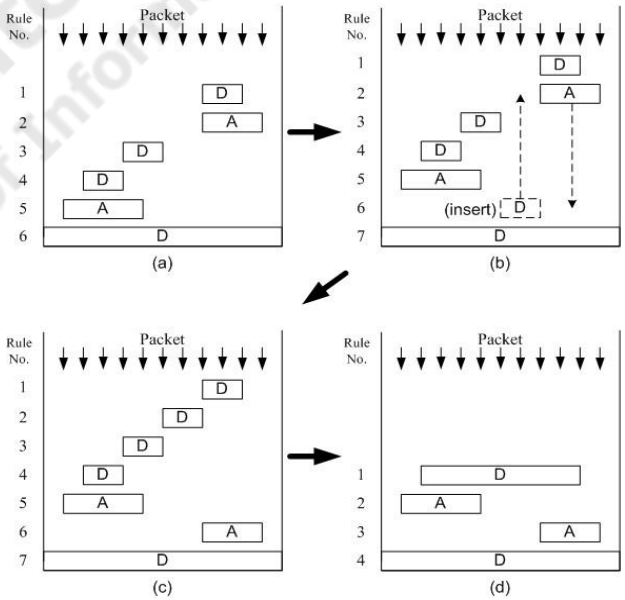
ภาพประกอบที่ 2.4 การยุบรวมกฎในรูปแบบของรีเลชัน

ในบางครั้งยังมี Rule ที่สามารถที่จะยุบรวมกันได้แต่ Rule ดังกล่าวไม่ได้อยู่ติดกัน สามารถที่จะทำให้ Rule ทั้งสองมาอยู่ชิดกันได้ [9] โดยการย้ายตำแหน่ง (ซึ่งจะทำได้ก็ต่อเมื่อไม่เกิดการ Correlate กับ Rule ข้างเคียงที่จะสลับตำแหน่งกัน) จากนั้นทำการยุบรวม ดังแสดงในภาพ 2.4



ภาพประกอบที่ 2.5 การยุบรวมโดยการย้าย Rule เข้ามาให้ชิดกันเสียก่อน [10]

และมีอีกเทคนิคหนึ่งในการยุบรวมกฎ ซึ่งก็คือ การแทรก Consecutive Redundant Rule การแทรก Consecutive Redundant Rule จะไม่ทำให้ Policy เปลี่ยน



ภาพประกอบที่ 2.6 การยุบรวมโดยการแทรก Consecutive redundant rule [10]

2.1.1.3 การวิเคราะห์ไฟร์วอลล์ของ Ehab Al-Shaer

Ehab Al-Shaer [3] ได้นำเสนอการวิเคราะห์ไฟร์วอลล์โดยใช้ทฤษฎีของเซตในงานวิจัย FIREWALL POLICY ADVISOR FOR ANOMALY DISCOVERY AND RULE EDITING (ซึ่งได้รับรางวัล Best Paper Award, IEEE/IFIP IM'2003) เพื่อที่จะค้นหา Anomaly ที่มีอยู่ใน Rule List โดยแบ่ง Anomaly ออกเป็น 4 อย่างด้วยกันและอธิบายโดยใช้ทฤษฎีเซตว่า Anomaly ต่าง ๆ นั้นเกิดขึ้นจากอะไร ซึ่งนิยามของ Anomaly ทั้ง 4 มีดังต่อไปนี้

1. Shadowing Anomaly คือ Rule ที่ถูกบังโดย Rule ใน Rule หนึ่งที่อยู่ก่อนหน้า

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

ตัวอย่างของ Shadowing Anomaly แสดงในรูปที่ 10 ใน Rule-List A นั้น Rule-3 ถูกบังโดย Rule2 เป็น Shadowing Anomaly ที่สามารถเกิดได้กับไฟร์วอลล์มาตรฐานเกือบทุกชนิดไม่ว่าจะเป็น ACL: Access Control List บน Cisco Router, Check Point Firewall-1 หรือ IPTABLES ส่วนใน Rule-List B นั้น Rule-3 ถูกบังโดยสอง Rule ร่วมกันซึ่งก็คือ Rule-1 และ Rule-2 ซึ่งกรณีสามารถเกิดได้บน Check Point Firewall-1 เนื่องจากสามารถที่จะป้อน Rule แบบ Multi-Address ได้ ส่วนใน Rule-List C นั้น Rule3 ถูกบังโดยสอง Rule ร่วมกันเช่นเคย (ซึ่งก็คือ Rule-1 และ Rule-2) กรณีนี้สามารถเกิดได้บนทั้ง Check Point Firewall-1 และ IPTABLES ที่ใช้คุณสมบัติ Multi-Port ส่วน Rule-List D เกิดได้บน IPTABLES ซึ่ง Rule-3 จะถูกบังโดย Rule-1 ร่วมกับ Rule-2 เนื่องจากมีการใช้คุณสมบัติ Port-Range

เมื่อเกิด Shadowing Anomaly แล้ว Shadowed Rule (Rule ที่ถูกบัง) จะไม่มีโอกาสที่จะถูกใช้งานเลย เพราะไม่มี Packet ใดที่จะเล็ดลอดมาถึง Rule นี้ได้ ดังนั้นการลบ Shadowed Rule ทิ้งไปจึงไม่มีผลกระทบใด ๆ ต่อ Policy ของไฟร์วอลล์ การลบ Shadowed Rule ทิ้งไปนั้นยังช่วยลดขนาดของ Rule List ให้สั้นลงด้วย ทำให้ประสิทธิภาพการทำงานของไฟร์วอลล์ดีขึ้น และยังทำให้ผู้ออกแบบกฎสามารถที่จะอ่าน Rule List ได้ง่ายขึ้นด้วย

Rule-List A (Rule-3 shadowed by Rule-2)

order	Source IP	Dest. IP	Dest. Port	Action
1	10.0.0.1	20.0.0.1	80	ACCEPT
2	10.0.0.4 /30	20.0.0.5	23	ACCEPT
3	10.0.0.5	20.0.0.5	23	ACCEPT
4	any	any	any	DENY

Rule-List B (Rule-3 shadowed by Rule-1 and Rule-2)

order	Source IP	Dest. IP	Dest. Port	Action
1	10.0.0.1, 10.0.0.3	20.0.0.5	23	ACCEPT
2	10.0.0.5, 10.0.0.6	20.0.0.5	23	DENY
3	10.0.0.1, 10.0.0.6	20.0.0.5	23	ACCEPT
4	any	any	any	DENY

Rule-List C (Rule-3 shadowed by Rule-1 and Rule-2)

order	Source IP	Dest. IP	Dest. Port	Action
1	10.0.0.5	20.0.0.5	21, 25	ACCEPT
2	10.0.0.5	20.0.0.5	23 ,80	DENY
3	10.0.0.5	20.0.0.5	23, 25	ACCEPT
4	any	any	any	DENY

Rule-List D (Rule-3 shadowed by Rule-1 and Rule-2)

order	Source IP	Dest. IP	Dest. Port	Action
1	10.0.0.5	20.0.0.5	21-22	DENY
2	10.0.0.5	20.0.0.5	23-24	ACCEPT
3	10.0.0.5	20.0.0.5	22-23	ACCEPT
4	any	any	any	DENY

ภาพประกอบที่ 2.7 ตัวอย่างของ Rule-List ที่มี Shadowing Anomaly [11]

2. Correlation Anomaly คือ Rule ที่อยู่ในลำดับก่อนสามารถที่จะ Match กับบาง Packet ที่จะ Match กับ Rule ที่อยู่ลำดับหลังได้

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”

Rule-List E (Rule-2 correlated with Rule-3)

order	Source IP	Dest. IP	Dest. Port	Action
1	10.0.0.1	20.0.0.1	80	ACCEPT
2	10.0.0.0 /24	20.0.0.5	23	DENY
3	10.0.0.3	20.0.0.0 /24	23	ACCEPT
4	any	any	any	DENY

Rule-List F (Rule-2 correlated with Rule-3)

order	Source IP	Dest. IP	Dest. Port	Action
1	10.0.0.1	20.0.0.1	80	ACCEPT
2	10.0.0.1, 10.0.0.3	20.0.0.5	23	DENY
3	10.0.0.3	20.0.0.5, 20.0.0.7	23	ACCEPT
4	any	any	any	DENY

Rule-List G (Rule-2 correlated with Rule-3)

order	Source IP	Dest. IP	Dest. Port	Action
1	10.0.0.1	20.0.0.1	80	ACCEPT
2	10.0.0.1, 10.0.0.3	20.0.0.5	23	DENY
3	10.0.0.3, 10.0.0.5	20.0.0.5	23	ACCEPT
4	any	any	any	DENY

Rule-List H (Rule-2 correlated with Rule-3)

order	Source IP	Dest. IP	Dest. Port	Action
1	10.0.0.1	20.0.0.1	80	ACCEPT
2	10.0.0.5	20.0.0.5	23, 25	DENY
3	10.0.0.5	20.0.0.5	25, 110	ACCEPT
4	any	any	any	DENY

Rule-List I (Rule-2 correlated with Rule-3)

order	Source IP	Dest. IP	Dest. Port	Action
1	10.0.0.1	20.0.0.1	80	ACCEPT
2	10.0.0.5	20.0.0.5	23-24	DENY
3	10.0.0.5	20.0.0.5	22-23	ACCEPT
4	any	any	any	DENY

ภาพประกอบที่ 2.8 ตัวอย่างของ Rule-List ที่มี Correlation Anomaly [11]

ตัวอย่างของ Correlation Anomaly ได้แสดงไว้ในรูปที่ 2.12 ใน Rule-List E, F, G, H และ I นั้นเป็นการ Correlate กันระหว่าง Rule-2 กับ Rule-3 ซึ่ง กรณีของ Rule-List E เป็น Correlation Anomaly ที่สามารถเกิดได้กับไฟร์วอลล์มาตรฐานเกือบทุกชนิดไม่ว่าจะเป็น ACL: Access Control-List บน Cisco Router, Check Point Firewall-1 หรือ IPTABLES ส่วนใน Rule-List F และ G นั้นเกิดได้กับ Check Point Firewall-1 (ใช้คุณสมบัติ Multi-Address) ส่วนใน Rule-List H และ I นั้นสามารถเกิดขึ้นได้กับ IPTABLES โดยที่ Rule-List H ใช้คุณสมบัติ Multi-Port ในขณะที่ Rule-List I ใช้คุณสมบัติ Port-Range

การเกิด Correlation Anomaly นั้น Rule คู่ใดที่เกิด Correlation ต่อกัน เมื่อทำการสลับตำแหน่งกันจะทำให้ Policy ของไฟร์วอลล์เปลี่ยน เช่นเมื่อเราสลับตำแหน่งกันระหว่าง Rule-2 กับ Rule-3 ใน RuleList E จะทำให้ไฟร์วอลล์รับ (ACCEPT) Packet ที่ต้นทางมาจาก 10.0.0.3 และมีปลายทางไปยัง 20.0.0.5 พอร์ตปลายทาง 23 ได้ ซึ่งจากเดิมก่อนทำการสลับตำแหน่งกันนั้น Packet นี้จะถูก DENY ดังนั้นเมื่อตรวจพบ Correlation Anomaly ก็จะเป็นสัญญาณเตือนผู้ออกแบบไฟร์วอลล์ว่า (1)

อาจจะมีการออกแบบกฎผิดพลาด (2) ให้พิจารณาดูให้ดีว่าควรจะให้ Rule ใดอยู่ลำดับก่อนหรือหลัง และ (3) การสลับตำแหน่งของสอง Rule ดังกล่าวจะมีผลต่อ Policy ของไฟร์วอลล์

3. Generalization Anomaly คือ Rule ที่พูดคลุม (Generalize) Rule อื่นที่อยู่ลำดับก่อน Rule ที่พูดคลุม (Generalize) Rule อื่น หมายถึง Rule ที่ Match กับทุก Packet ที่จะ Match กับ Rule อื่น (ที่กล่าวถึง) ที่อยู่ในลำดับก่อนหน้าได้ การเกิด Generalization Anomaly นั้น Rule คู่ใดที่เกิด Generalization Anomaly ต่อกันแล้ว เมื่อทำการ สลับตำแหน่งกันจะทำให้ Policy ของไฟร์วอลล์เปลี่ยน เช่นเมื่อเราสลับตำแหน่งกันระหว่าง Rule-2 กับ Rule3 ใน Rule-List L จะทำให้ไฟร์วอลล์รับ (ACCEPT) Packet ที่ต้นทางมาจาก 10.0.0.3 และมีปลายทางไปยัง 20.0.0.5 พอร์ตปลายทาง 23 ได้ ซึ่งจากเดิมก่อนทำการสลับตำแหน่งกันนั้น Packet จะถูก DENY ดังนั้นเมื่อ ตรวจพบ Generalization Anomaly ก็จะเป็นสัญญาณเตือนผู้ออกแบบไฟร์วอลล์ว่า การสลับตำแหน่งของ สอง Rule ดังกล่าวจะมีผลต่อ Policy ของไฟร์วอลล์

Rule-List J (Rule-3 generalized Rule-2)

order	Source IP	Dest. IP	Dest. Port	Action
1	10.0.0.1	20.0.0.1	80	ACCEPT
2	10.0.0.3	20.0.0.5	23	DENY
3	10.0.0.0 /24	20.0.0.5	23	ACCEPT
4	any	any	any	DENY

Rule-List K (Rule-3 generalized Rule-2)

order	Source IP	Dest. IP	Dest. Port	Action
1	10.0.0.1	20.0.0.1	80	ACCEPT
2	10.0.0.3	20.0.0.5	23	DENY
3	10.0.0.0 /24	20.0.0.5, 20.0.0.7	23	ACCEPT
4	any	any	any	DENY

Rule-List L (Rule-3 generalized Rule-2)

order	Source IP	Dest. IP	Dest. Port	Action
1	10.0.0.1	20.0.0.1	80	ACCEPT
2	10.0.0.3	20.0.0.5	23	DENY
3	10.0.0.0 /24	20.0.0.5, 20.0.0.7	22-25	ACCEPT
4	any	any	any	DENY

ภาพประกอบที่ 2.9 ตัวอย่างของ Rule-List ที่มี Generalization Anomaly [11]

จากการศึกษาการวิเคราะห์กฎของไฟร์วอลล์โดยใช้รีเลชันแนลอัลจีบรา พบว่าเมื่อเกิด Generalization Anomaly จะมีบางกรณีที่การสลับตำแหน่งของสอง Rule จะไม่มีผลต่อ Policy ของไฟร์วอลล์

4. Redundancy Anomaly คือ Rule ที่ซ้ำซ้อนกับ Rule อื่น

Rule ที่ Redundant (ซ้ำซ้อน) ต่อ Rule อื่น คือ Rule ที่กระทำการตาม Action เดียวกันบน Packet เดียวกันกับที่ Rule อื่น (ที่อยู่ลำดับหลัง) จะกระทำ

Redundancy Anomaly จะเกิดขึ้นกับ Rule ที่เป็น Redundant Rule (Rule ที่ซ้ำซ้อนต่อ Rule อื่น) เมื่อเกิด Redundancy Anomaly แล้ว Redundant Rule (Rule ที่ซ้ำซ้อน) จะ Match กับ Rule ที่อยู่ลำดับหลัง ซึ่ง Rule ที่อยู่ลำดับหลังดังกล่าวก็ครอบคลุมกลุ่มของ Packet มากกว่า Redundant Rule อยู่แล้ว ดังนั้นการลบ Redundant Rule ทิ้งไปจึงไม่มีผลกระทบใด ๆ ต่อ Policy ของไฟร์วอลล์

Rule-List M (Rule-2 redundant to Rule-3)

order	Source IP	Dest. IP	Dest. Port	Action
1	10.0.0.1	20.0.0.1	80	ACCEPT
2	10.0.0.3	20.0.0.5	23	ACCEPT
3	10.0.0.0 /24	20.0.0.5	23	ACCEPT
4	any	any	any	DENY

Rule-List N (Rule-2 redundant to Rule-3)

order	Source IP	Dest. IP	Dest. Port	Action
1	10.0.0.1	20.0.0.1	80	ACCEPT
2	10.0.0.3	20.0.0.5	23	ACCEPT
3	10.0.0.0 /24	20.0.0.5, 20.0.0.7	23	ACCEPT
4	any	any	any	DENY

Rule-List O (Rule-1 redundant to Rule-3)

order	Source IP	Dest. IP	Dest. Port	Action
1	10.0.0.3	20.0.0.5	23	ACCEPT
2	10.0.0.1	20.0.0.1	80	ACCEPT
3	10.0.0.0 /24	20.0.0.5, 20.0.0.7	22-25	ACCEPT
4	any	any	any	DENY

ภาพประกอบที่ 2.10 ตัวอย่างของ Rule-List ที่มี Redundancy Anomaly [11]

การลบ Redundant Rule ทิ้งไปนั้นยังช่วยลดขนาดของ Rule List ให้สั้นลงด้วย ทำให้ประสิทธิภาพการทำงานของไฟร์วอลล์ดีขึ้น และยังทำให้ การออกแบบไฟร์วอลล์ Rule List อ่านได้ง่ายขึ้นด้วย จากการศึกษาการวิเคราะห์กฎของไฟร์วอลล์โดยใช้รีเลชันแนลอัลจีบรา พบว่าเมื่อเกิด Redundancy Anomaly จะมีบางกรณีที่การลบ Redundant Rule ทิ้งไปจะมีผลต่อ Policy ของไฟร์วอลล์

2.1.1.4 การสร้าง rule สำหรับไฟร์วอลล์

โดยทั่วไปแล้วหน้าที่ของไฟร์วอลล์จะทำการกรอง (Filter) ข้อมูลเฉพาะส่วนที่ได้รับอนุญาตเท่านั้น ดังนั้นการเขียนกฎหรือการ สร้าง Rule สำหรับไฟร์วอลล์จึงเป็นเรื่องที่สำคัญมาก หากการสร้างกฎของไฟร์วอลล์ผิดพลาดไปจะทำให้ไฟร์วอลล์ไม่สามารถที่จะช่วยป้องกันเครือข่ายให้รอดพ้นจากการถูกบุกรุกหรือโจมตีได้อย่างแน่นอน แต่อย่างไรก็ตามผู้ดูแลระบบจะต้องมั่นใจก่อนว่าไฟร์วอลล์มีความปลอดภัยในระดับโฮสต์ (host based security) อยู่แล้ว ต่อไปจะขอกล่าวถึงรายละเอียดดังนี้ [10]

2.1.1.5 Firewall Host Based Security

ผู้ดูแลระบบจะต้องสร้างความปลอดภัยในระดับโฮสต์โดยจะมีคำแนะนำเบื้องต้นดังต่อไปนี้

1) ปิด TCP/UDP service ที่ไม่ได้ใช้งาน เช่น bootps, finger ยิ่งเปิด Service น้อยก็ยิ่งลดโอกาสใน การโจมตีของผู้บุกรุก และยังเป็นการลดการใช้งาน CPU และหน่วยความจำ ของระบบอีกด้วย

2) ในกรณีที่จำเป็นต้องเปิด Service บนเครื่องไฟร์วอลล์ จะต้องจำกัดการเข้าถึงให้ใช้งานได้เฉพาะผู้ดูแล ระบบเท่านั้น

3) ปิด Service ที่ไม่จำเป็นอื่นๆ บนเครื่องไฟร์วอลล์ เช่น การทำ Remote Configuration ยกเลิก Interface ที่ไม่ได้ใช้งานในเครื่องไฟร์วอลล์ (หรือ Router)

4) ในกรณีที่ใช้ฮาร์ดแวร์ไฟร์วอลล์หรือ Router จะต้องป้องกันการเข้าถึง Port ที่ใช้ในการควบคุม เช่น Console Port

5) แก้ไขค่า Default Password โดยให้มีความยาวอย่างต่ำ 8 ตัวอักษร, ไม่เป็นคำที่อยู่ในดิกชันนารี, ต้องไม่ขึ้นต้นด้วยตัวเลข และมีตัวเลขรวมทั้งตัวอักษรพิเศษรวมอยู่ด้วย (เช่น ,./<>;':"[]{}\|~!@#$%^&*()_+-

=) และควรใช้รหัสผ่านที่แตกต่างกันในแต่ละเครื่อง ทั้งนี้ควรเปลี่ยนรหัสผ่านทุกๆ 90 วัน

2.1.1.6 Building Firewall Rulebase

การสร้างกฎ (Rule) จะต้องอ่านง่าย ได้ใจความ ไฟร์วอลล์ที่ดีไม่ควรมีกฎ (Rule) มากกว่า 30 กฎ [2] (Rule) เพราะถ้ามากกว่านี้จะทำให้เกิดความสับสนได้ง่าย และอาจจะทำให้เกิดความผิดพลาดโดยไม่รู้ตัวขึ้น นอกจากนี้ยังมีข้อดีในส่วนที่ทำให้เครื่องทำงานน้อยลงอีกด้วย

การสร้างกฎ (Rule) ของไฟร์วอลล์ถือได้ว่าเป็นการนำ Security Policy ขององค์กรมาบังคับใช้งานในทาง เทคนิค โดยใช้ไฟร์วอลล์เป็นเครื่องมือให้เกิดผลตามที่ต้องการ นอกจากนี้ยังมี Rule บางส่วนที่ถือได้ว่า ผู้ดูแล ระบบควรเพิ่มเข้าไปในกฎ (Rule) ของไฟร์วอลล์ เช่น การป้องกัน IP Spoofing, ป้องกันการโจมตีแบบ Land Attack

2.1.1.7 Rule Order การเรียงลำดับของกฎ (Rule)

ก็เป็นอย่างหนึ่งที่ต้องคำนึงเพราะมีความสำคัญมากเช่นเดียวกัน ไฟร์วอลล์ส่วนใหญ่จะทำงานแบบ Sequence คือตรวจสอบ Packet กับกฎ (Rule) ตามลำดับกฎ (Rule) ที่สร้างไว้โดยจะมีคำแนะนำในการวาง ลำดับของกฎ (Rule) คือ วางกฎ (Rule) ที่เป็นกฎ (Rule) ทั่วไปไว้ด้านล่าง และให้นำกฎ (Rule) ที่มีความเฉพาะเจาะจงไว้ด้านบน เพื่อป้องกันไม่ให้ Packet Match กับกฎ (Rule) ที่เป็น กฎ (Rule) ทั่วไปก่อน

2.1.1.8 TCP/IP Filter

ในการกรองนั้นผู้ดูแลระบบจะกำหนด default policy ได้ 2 รูปแบบ คือ

1) Default ACCEPT ผู้ดูแลระบบจะต้องสร้างกฎ (Rule) เพื่อกำหนดว่าจะปิด service และโฮสต์ใดบ้าง และโฮสต์อื่นๆที่ไม่ถูกกำหนดไว้จะมีค่าเป็นเปิด

2) Default DROP ACCEPT ผู้ดูแลระบบจะต้องสร้างกฎ (Rule) เพื่อกำหนดว่าจะเปิด Service และโฮสต์ใดบ้าง และโฮสต์อื่นๆที่ไม่ถูกกำหนดไว้จะมีค่าเป็นปิด

ต่อไปนี้ผู้ดูแลระบบควรจะทราบ TCP/IP Service ที่เป็นจุดอ่อนต่างๆ ในระบบ โดยจะแสดงออกมาเป็นตารางดังนี้

ตารางที่ 2.1 แสดง TCP/IP Service ที่ควรปิดกั้นที่ไฟร์วอลล์

Port(s) (Transport)	Server	Port(s) (Transport)	Server
1 (TCP & UDP)	tcpmux	1981 (TCP)	Shockrave
7 (TCP & UDP)	echo	1999 (TCP)	BackDoor
9 (TCP & UDP)	discard	2001 (TCP)	Trojan Cow
11 (TCP & UDP)	systat	2023 (TCP)	Ripper
13 (TCP & UDP)	daytime	2049 (TCP & UDP)	nfs
15 (TCP & UDP)	netstat	2115 (TCP)	Bugs
17 (TCP & UDP)	qotd	2140 (TCP)	Deep Throat
19 (TCP & UDP)	chargen	2222 (TCP)	Subseven21
37 (TCP & UDP)	time	2301 (TCP & UDP)	compaqdiag
43 (TCP & UDP)	whois	2565 (TCP)	Striker
67 (TCP & UDP)	bootps	2583 (TCP)	WinCrash
68 (TCP & UDP)	bootpc	2701 (TCP & UDP)	sms-rcinfo
69 (UDP)	tftp	2702 (TCP & UDP)	sms-remctrl
93 (TCP)	supdup	2703 (TCP & UDP)	sms-chat
111 (TCP & UDP)	sunrpc	2704 (TCP & UDP)	sms-xfer
135 (TCP & UDP)	loc-srv	2801 (TCP)	Phineas P.
137 (TCP & UDP)	netbios-ns	4045 (TCP)	lockd
138 (TCP & UDP)	netbios-dgm	5800 - 5899 (TCP)	winvnc web server

ตารางที่ 2.1 แสดง TCP/IP Service ที่ควรปิดกั้นที่ไฟร์วอลล์ (ต่อ)

Port(s) (Transport)	Server	Port(s) (Transport)	Server
139 (TCP & UDP)	netbios-ssn	5900 - 5999 (TCP)	winvnc
177 (TCP & UDP)	xdmcp	6000 - 6063 (TCP)	X11 Window System
1024 (TCP)	NetSpy	31337 -31338 (TCP & UDP)	Back Orifice
1045 (TCP)	Rasmin	32700 - 32900 (TCP & UDP)	RPC services
1090 (TCP)	Xtreme	32720 (TCP)	Trinity V3
1170 (TCP)	Psyber S.S	39168 (TCP)	Trinity V3
1234 (TCP)	Ultors Trojan	65000 (TCP)	Stacheldraht
1243 (TCP)	Backdoor-G		
1245 (TCP)	VooDoo Doll		
1349 (UCP)	Back Orifice DLL		
1492 (TCP)	FTP99CMP		
1600 (TCP)	Shivka-Burka		
1761 – 1764 (TCP & UDP)	sms-helpdesk		
1807 (TCP)	SpySender		

ตารางที่ 2.2 แสดง TCP/UDP Service ที่ควรปิดกั้นไม่ให้เข้ามาจากภายนอก

Port(s) (Transport)	Server
79 (TCP)	finger
161 (TCP & UDP)	snmp

ตารางที่ 2.2 แสดง TCP/UDP Service ที่ควรปิดกั้นไม่ให้เข้ามาจากภายนอก

Port(s) (Transport)	Server
162 (TCP & UDP)	snmp trap
514 (UDP)	syslog
550 (TCP & UDP)	new who

ตารางที่ 2.3 แสดง TCP/UDP Service ที่อาจเปิดให้บริการใน DMZ

Port(s) (Transport)	Server
20 (TCP)	ftpdata
21 (TCP)	ftp
22 (TCP)	ssh
23 (TCP)	telnet
25 (TCP)	smtp
53 (TCP & UDP)	domain
80 (TCP)	http
110 (TCP)	pop3
119 (TCP)	nntp
123 (TCP)	ntp
143 (TCP)	imap
179 (TCP)	bgp
389 (TCP & UDP)	ldap
443 (TCP)	ssl
1080 (TCP)	socks

ตารางที่ 2.3 แสดง TCP/UDP Service ที่อาจเปิดให้บริการใน DMZ (ต่อ)

Port(s) (Transport)	Server
3128 (TCP)	squid
8000 (TCP)	http (alternate)
8080 (TCP)	http-alt
8888 (TCP)	http (alternate)

และยังมีคำแนะนำอีกสำหรับการสร้างกฎของไฟร์วอลล์[12] โดยจะแสดงเป็นข้อๆดังนี้

1) ควรมีการบันทึกข้อมูลลง Log สำหรับกฎ (Rule) ที่ใช้ Block การเข้าถึง ซึ่งข้อมูลนี้จะเป็นประโยชน์ ในการตรวจสอบการบุกรุก

2) ป้องกันการปลอมไอพี (IP Spoof) สำหรับข้อมูลขาเข้ามาจากอินเทอร์เน็ต โดยป้องกันไม่ให้ Packet ที่ มีไอพีดังต่อไปนี้เข้ามายังเครือข่ายภายใน

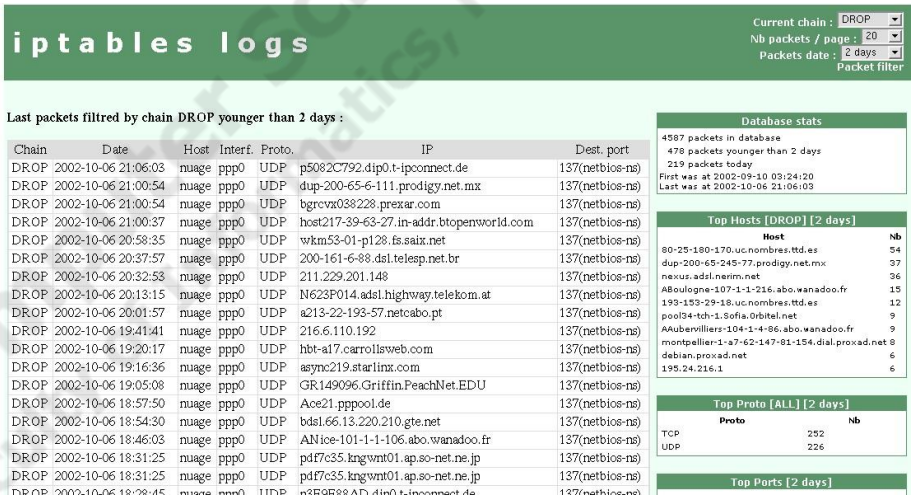
- 127.0.0.0 - 127.255.255.255 : Local Host Address
- 10.0.0.0 - 10.255.255.255 : Reserved Address
- 172.16.0.0 - 172.31.255.255 : Reserved Address
- 192.168.0.0 - 192.168.255.255 : Reserved Address
- 224.0.0.0 - 239.255.255.255 : Multicast Address

3) ป้องกันเครื่องไฟร์วอลล์จากการโจมตีแบบ Land Attack ซึ่งการโจมตีแบบนี้จะใช้วิธีส่ง Packet ที่มี Source IP Address ตรงกันกับ Destination IP Address รวมทั้งค่า Source Port และ Destination Port ที่ตรงกัน ซึ่งก่อให้เกิดการโจมตีแบบ Denial of Service ซึ่งป้องกันได้โดย Block ไม่ให้ข้อมูลขาเข้าที่มี Source IP Address ตรงกันกับไอพีของเครือข่ายภายในเข้ามาในระบบ

4) ป้องกันการโจมตีแบบ SYN Flood ที่เครื่องไฟร์วอลล์ ซึ่งผู้บุกรุกจะส่ง SYN Packet จำนวนมากมายัง เครื่องปลายทาง ทำให้คิวของการรับ Connection ใน Service ดังกล่าวเต็ม ทำให้ไม่สามารถให้บริการแก่ เครื่องอื่น ๆ ได้

5) ป้องกันไฟร์วอลล์และเครื่องอื่น ๆ ภายในเครือข่ายจาก traceroute เพราะ traceroute เป็นโปรแกรม ที่ช่วยให้ทราบถึงไอพีแอดเดรสของ Router ที่รับส่งต่อ Packet ไปทีละ hop จนกระทั่งถึงปลายทางที่ต้องการ โดยใช้คุณสมบัติของ IP Time To Live (TTL) ในการทำงาน โดยมันจะกำหนดค่า TTL counter ที่ทำให้ Router ที่ Packet ผ่านไปนั้นต้องสร้าง ICMP message กลับมาเสมอ สำหรับคำสั่ง tracert ใน Windows นั้น จะใช้ ping (ICMP Echo) เป็นตัวส่ง Packet ออกไป ในขณะที่ traceroute ใน Unix นั้น จะใช้ UDP datagram เป็นตัวส่งข้อมูลออกไป datagram ที่ถูกส่งออกไปนั้นจะถูกส่งไปยัง port 33434 โดยดีฟอลต์ และ ค่าหมายเลข port นี้จะถูกเพิ่มขึ้นเมื่อได้รับ packet ที่ตอบกลับมาอย่างถูกต้อง โดยปกติแล้ว traceroute มักจะส่ง datagram ออกไปจำนวน 3 datagram เพื่อป้องกันการสูญหายระหว่างทาง

6) ถึงแม้ว่าจะมีการป้องกันการใช้งาน traceroute จากทั้ง Unix และ Windows แล้วก็ตาม ผู้บุกรุกก็ยัง สามารถใช้วิธีอื่นในการ trace เข้ามายังเครือข่ายภายใน เช่น การใช้โปรแกรม Firewalk ดังนั้นหากต้องการ หยุดยั้งการใช้ traceroute รวมทั้ง Firewalk แล้ว จะต้องใช้วิธี drop TTL Exceeded in Transit packet ที่ ขาออกไปสู่อินเทอร์เน็ต



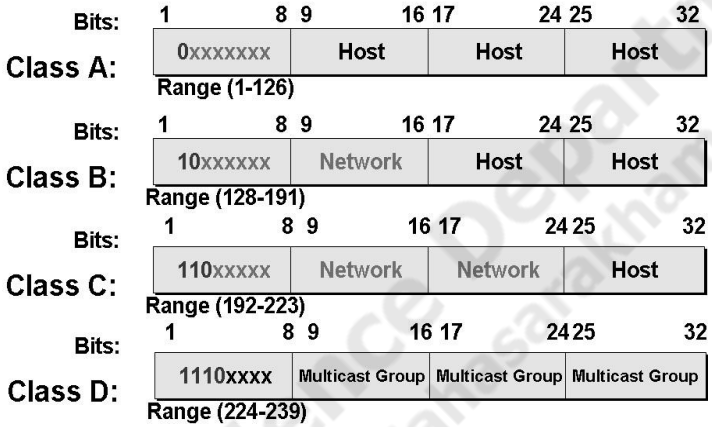
ภาพประกอบที่ 2.11 แสดงโปรแกรมวิเคราะห์ Log File [13]

2.1.1.9 IP Address

IP Address [14] มีทั้งหมด 32 บิต หรือ 4 ไบต์ โดยแต่ละไบต์ จะถูกคั่นด้วยเครื่องหมาย (.) ตัวอย่างเช่น 172.20.22.2 และยังสามารถแบ่งออกเป็น 2 ส่วนใหญ่ๆ คือ

- ส่วนแรกเรียกว่า หมายเลย Network Address หรือ Subnet Address
- ส่วนที่สองเรียกว่า หมายเลข Host Address

Subnet Mask [7-9] จะช่วยแยกแยะว่าส่วนใดภายในหมายเลข IP Address เป็น Network Address และส่วนใดเป็น Host Address ดังนั้นจะเห็นได้ว่าเวลาที่เรากำหนด IP Address เราจะต้องกำหนด Subnet Mask ด้วยเสมอ สำหรับการแบ่งคลาสในอินเทอร์เน็ตแบ่งออกเป็น 5 class ดังรูปที่ 2.11



ภาพประกอบที่ 2.12 แสดงการแบ่งคลาสบนเครือข่ายอินเทอร์เน็ต

Class A มีหมายเลข Network Address ตั้งแต่ 1 – 126 และ Host Address มีจำนวนเท่ากับ 2^{24}

Class B มีหมายเลข Network Address ตั้งแต่ 128 – 191 และ Host Address มีจำนวนเท่ากับ 2^{16}

Class C มีหมายเลข Network Address ตั้งแต่ 192 – 223 และ Host Address มีจำนวนเท่ากับ 2^{8}
Class D มีหมายเลข Network Address ตั้งแต่ 224 – 239 และ IP Address ที่เหลืออีก 2^{16} ใช้สำหรับ multicast group

0 ถูกนำไปใช้แทน default route

127 ถูกนำไปใช้เป็น Loopback Address สำหรับจำลองการส่งข้อมูลภายในเครื่อง

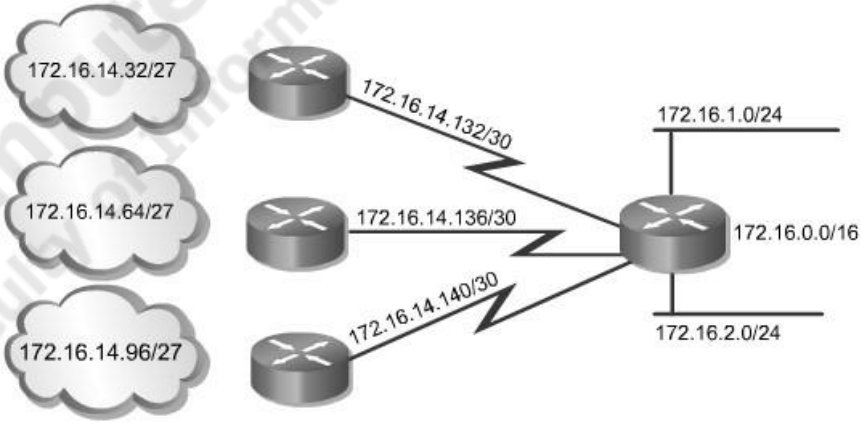
Class C ค่าของอพีแอดเดรสมีค่าต่ำกว่า 192 ถึง 233

Class D ค่าของอพีแอดเดรสมีตั้งแต่ 224 ขึ้นไป

Subnetting เป็นการนำเอา Network Address ที่มีอยู่มาซอยย่อยออกเป็นหลายๆ Subnet Addressง โดยให้จำนวนของ Subnet Address มากกว่าหรือว่าเท่ากับจำนวนเน็ตเวิร์กเซกเมนต์ที่มีอยู่ ตัวอย่างเช่น เราได้ Pubic IP Address 203.166.16.30/28 (16 IP Address) มาจาก ISP เพื่อจัดสรรให้แก่อุปกรณ์เน็ตเวิร์กและ Server ต่างๆ เราสามารถใช้วิธีการทำ Subnetting มาแบ่งซอยย่อยออกมาให้เป็นเครือข่ายย่อยๆ เช่น แบ่งเป็น 2 subnet (subnet ละ 8 IP Address) เป็นต้น

2.1.1.10 การทำ Subnet ซ้อน Subnet (Variable-Length Subnet Mask : VLSM)

VLSM [15] คือการทำ Subnet ภายใน Subnet เหตุผลที่สำคัญที่สุดในการทำ VLSM คือ ต้องการให้เครือข่ายมีขนาดพอดีกับการใช้งานจริงๆ เช่น interface ของ WAN Link ต้องการไอพีเอดเดรสเพียงแค่ 4 ไอพี ในการเชื่อมต่อ โดยใช้ 2 ไอพีสำหรับเป็น Network Address และบรอสคลาส ที่เหลืออีก 2 ไอพีเพื่อใช้กำหนดให้ขาอินเตอร์เฟสของเราเตอร์ที่เชื่อมต่อกัน ดังรูปที่ 5 จากรูป WAN Link ที่เชื่อมต่อไปที่เน็ตเวิร์ค 172.16.14.132/30 จะใช้ Network Address เป็น 172.16.14.132 ส่วนบรอดคลาสคือหมายเลข 172.16.14.135 ส่วนไอพี 172.16.14.133 และ 172.16.14.134 จะใช้เป็นหมายเลขประจำขาอินเตอร์เฟสของเราเตอร์ จะเห็นได้ว่าไอพีที่ต้องการในกรณีนี้ใช้เพียง 4 ไอพี ถ้าไม่มีการทำ VLSM จะทำให้ต้องเสียไอพีไปเป็นจำนวนมากและไม่คุ้มค่ากับการใช้งาน



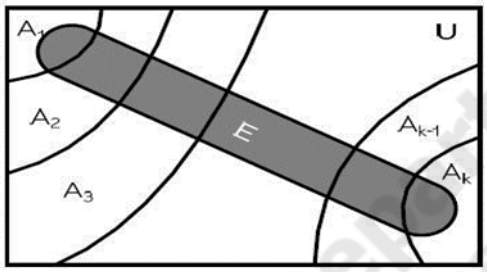
ภาพประกอบที่ 2.13 การใช้ VLSM ในการจัดสรรหมายเลข 4 IP Address ให้กับ WAN Link

2.1.2 ทฤษฎีของเบย์ (Bayes' theorem) [14]

ทฤษฎีของเบย์(Bayes' Theorem) ถูกพัฒนาขึ้นโดย Thomas Bayes โดยใช้

หลักการของ ความน่าจะเป็นแบบมีเงื่อนไขมาพัฒนาทฤษฎีบทดังกล่าว โดยมีรายละเอียด ดังต่อไปนี้

ถ้าให้เอกภพสัมพัทธ์ U ประกอบด้วยเหตุการณ์ที่ไม่สามารถเกิดขึ้นได้พร้อมกันk จำนวน เหตุการณ์คือ A_1, A_2, A_3, ..., A_k ดังรูปที่ 7.1 และให้ E เป็นเหตุการณ์หนึ่งในปริภูมิตัวอย่างที่เกิดจาก การทดลองเดียวกันนี้และต้องเป็นส่วนหนึ่งของ A_i(i=1, 2, 3, ..., k) จะสามารถคำนวณความน่าจะ เป็นแบบมีเงื่อนไขของเหตุการณ์หนึ่งใน A_iเมื่อเหตุการณ์ E เกิดขึ้นแล้วได้ดังสมการที่ (2.1)



ภาพประกอบที่ 2.14 เหตุการณ์ E บนเหตุการณ์ k เหตุการณ์ที่เกิดพร้อมกันไม่ได้

$$p(A_i/E) = \frac{P(E/A_i)*P(A_i)}{\sum_{i=1}^{k} P(E/A_i)*P(A_i)} = \frac{P(E/A_i)*P(A_i)}{P(E)} \qquad (2.1)$$

2.1.2.1 การเรียนรู้แบบเบย์ (Bayesian Learning)

การเรียนรู้แบบเบย์ เป็นเทคนิคที่ใช้ทฤษฎีความน่าจะเป็นตามกฎของเบย์ (Bayes' Theorem) เพื่อหาว่าสมมติฐานใดน่าจะถูกต้องที่สุด โดยใช้ความรู้ก่อนหน้า (Prior Knowledge) ได้แก่ ความน่าจะเป็นก่อนหน้าสำหรับสมมติฐานหนึ่ง ๆ ร่วมกับข้อมูล เช่น ความน่าจะเป็นที่สังเกตได้สำหรับสมมติ หนึ่ง ๆ เพื่อหาสมมติฐานที่ดีที่สุด

การเรียนรู้แบบเบย์อาศัยหลักการของการคำนวณความน่าจะเป็นของแต่ละสมมติฐาน (ในที่นี่คือคลาสเป้าหมายหรือผลลัพธ์การทำนาย) โดยการเรียนรู้แบบเบย์เป็นการเรียนรู้เพิ่มได้เนื่องจาก ตัวอย่างใหม่ที่ได้มาถูกนำมาปรับเปลี่ยนการแจกแจงซึ่งมีผลต่อการเพิ่มหรือลดความน่าจะเป็น ทำให้มีการเรียนรู้ที่เปลี่ยนไป วิธีการนี้ตัวแบบจะถูกปรับเปลี่ยนไปตามตัวอย่างใหม่ที่ได้โดยผนวกกับความรู้เดิมที่มีซึ่งการทำนายค่าคลาสเป้าหมายของตัวอย่างใช้ความน่าจะเป็นมากที่สุดของทุกสมมติฐาน

จากทฤษฎีของเบย์เราสามารถคำนวณความน่าจะเป็นของสมมติฐานต่าง ๆ โดยใช้สมการที่ 2.2

$$P(h|D) = \frac{P(D|h)*P(h)}{P(D)} \quad (2.2)$$

โดย

- D แทนข้อมูลที่นำมาใช้ในการคำนวณการแจกแจงความน่าจะเป็นposteriori probability ของสมมติฐาน h คือ P(h|D) ตามทฤษฎี
- P(h) คือ ความน่าจะเป็นก่อนหน้าของสมมติฐาน h
- P(D) คือ ความน่าจะเป็นก่อนหน้าของชุดข้อมูลตัวอย่าง D
- P(h|D) คือ ความน่าจะเป็นของ h เมื่อรู้ D
- P(D|h) คือ ความน่าจะเป็นของ D เมื่อรู้ h

ตัวอย่างการคำนวณเพื่อเลือกกสมมติฐานโดยกฎของเบย์

คนไข้คนหนึ่งไปตรวจหาโรค มะเร็ง ผลการตรวจเป็นบวก (+) อยากทราบว่า เราควรวินิจฉัยโรคคนไข้คนนี้ว่าเป็นโรคมะเร็งจริงหรือไม่โดยมีข้อมูลความเป็นจริงดังนี้

- ผลการตรวจเมื่อเป็นบวกจะให้ความถูกต้อง 98% กรณีที่มีโรคนั้นอยู่จริง
- ผลการตรวจเมื่อเป็นลบจะให้ความถูกต้อง 97% กรณีที่ไม่มีโรคนั้น
- 0.008 ของประชากรทั้งหมดเป็นโรคมะเร็ง

จากความน่าจะเป็นข้างต้นเราจะทราบว่าความน่าจะเป็นต่อไปนี้

P(cancer) =0.008	P(~cancer) = 0.992
P(+ | cancer) =0.98	P(- | cancer) =0.02
P(+ |~ cancer) =0.03	P(- |~ cancer) =0.97

<u>สมมติฐานที่1</u> คนไข้เป็นโรคมะเร็งจริงเมื่อมีผลการตรวจเป็นบวก เขียนแทนด้วย P(cancer |+) แทนค่าในสูตร

$$\mathrm{P(cancer|+)} = \frac{\mathrm{P(+|cancer)P(cancer)}}{\mathrm{P(+)}}$$

= 0.98*0.008

= 0.0078

<u>สมมติฐานที่2</u> คนไข้เป็นไม่เป็นโรคมะเร็งจริงเมื่อมีผลการตรวจเป็นบวก เขียนแทนด้วย P(~cancer |+) แทนค่าในสูตร

$$P(\sim cencer|+) = \frac{P(+|\sim cencer)P(\sim cencer)}{P(+)}$$

= 0.03*0.992

= 0.0298

เนื่องจากผลรวมของ P(cancer |+) กับ P(~cancer |+) เท่ากับ1 เราสามารถ Normalize ค่าของ P(cancer |+) = 0.0078/(0.0078+0.0298) = 0.21 และ P(~cancer |+)

= 0.0298/(0.0078+0.0298) = 0.79

สรุปว่า <u>สมมติฐานที่1</u> มีค่าความน่าจะเป็นเท่ากับ 0.21 และ<u>สมมติฐานที่2</u> มีความน่าจะเป็นเท่ากับ 0.79 ในการเลือกตอบสมมติฐานเนื่องจาก สมมติฐานที่2 มีค่ามากกว่า สมมติฐานว่าคนไข้ไม่เป็นโรคมะเร็งเมื่อทราบผลตรวจเป็นบวกด้วยความน่าจะเป็น 0.79 จึงถูกเลือก

ตัวจำแนกประเภทที่ดีที่สุดแบบเบย์ (Bayes Optimal Classification)

ในการจำแนกประเภทตัวอย่าง X ใด ๆ ที่น่าจะเป็นที่สุดของกรณีที่ผลการจำแนกประเภทตัวอย่างมีความแตกต่างจากสมมติฐานที่ต่างกัน เราจะใช้ตัวจำแนกประเภทที่ดีที่สุดแบบเบย์ เพื่อ จำแนกประเภทตัวอย่าง X ที่น่าจะเป็นที่สุด

กำหนดให้

h_{MAP} (Maximum A Posterior Hypothesis) แทน สมมติฐานที่น่าจะเป็นที่สุด

ถ้า h_{MAP} เป็นสมมติฐานที่น่าจะเป็นที่สุด h_{MAP} อาจไม่เป็นการจำแนกประเภทตัวอย่างที่น่าจะ เป็นที่สุด (most probable classification)

นิยามตัวจำแนกประเภทที่ดีที่สุดแบบเบย์

$$h_{MAP}(x) = \underset{v_j \in V}{\operatorname{argmax}} \sum_{h_i \in H} P(v_j|h_i)P(h_i|D) \qquad (2.3)$$

อธิบายได้โดยพิจารณาสมมติฐานทั้งสามต่อไปนี้ P(h1|D) = 0.4 P(h2|D) = 0.3 P(h3|D) = 0.3

เมื่อให้ตัวอย่าง X ผลการจำแนกประเภทของสมมติฐานเป็นดังนี้ h1 (X) = + h2 (X) = - h3 (X) = -

จากตัวอย่างข้างต้น เราทราบว่า

P(h1|D) = 0.4 P(-|h1) = 0 P(+|h1) = 1

P(h2|D) = 0.3 P(-|h2) = 1 P(+|h2) = 0

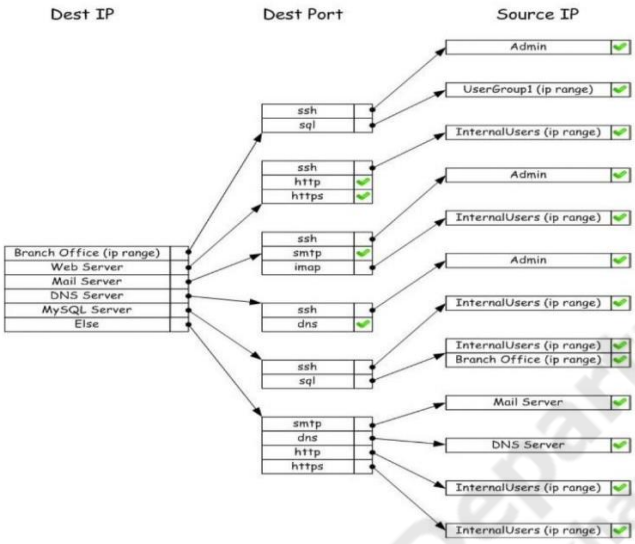
P(h3|D) = 0.3 P(-|h3) = 1 P(+|h3) = 0

จะได้ว่า $\sum$ P(+|hi)P(hi|D) = 0.4 Hi $\in$ H

$\sum$ P(|hi)P(hi|D) = 0.6 hi $\in$ H

ดังนั้น การจำแนกประเภทตัวอย่าง X ที่มี MAP Class คือ คลาสที่น่าจะเป็นมากที่สุด คือ –

2.2 งานวิจัยที่เกี่ยวข้อง



ภาพประกอบที่ 2.15 Using an IP address & port name [15]

An Improvement of Tree-Rule Firewall for a Large Network: Supporting Large Rule Size and Low Delay [15]

ไฟร์วอลล์เป็นอุปกรณ์เน็ตเวิร์คที่มีความสำคัญซึ่งจะเตรียมหรือสนับสนุนเรื่องของภัยคุกคามบนเครือข่าย มาตราการของการป้องกันขึ้นอยู่กับกฎไฟร์วอลล์ ไฟร์วอลล์แบบดั้งเดิม ได้แก่ i.n. Cisco ACL,IPTABLES, Checkpoint และ Juniper NetScreen firewall จะตรวจสอบแพ็คเกจจากกฎ และกฎก็จะถูกออกแบบในลักษณะของลิสต์ ซึงจะตรวจสอบแพ็คเกจตามทิศทางการไหลของแพ็คเกจ อย่างไรก็ตามไฟร์วอลล์ที่มีลักษณะการทำงานด้วยกฎแบบนี้ อาจจะนำไปสู่การขัดแย้ง ซึ่งอาจจะทำให้ไฟร์วอลล์มีความปลอดภัยน้อยลงหรือทำให้ประสิทธิภาพในการทำงานลดลง โดยพื้นฐานงานวิจัยก่อนหน้านี้นักวิจัยได้ทำการนำเสนอไฟร์วอลล์ที่มีลักษณะของโครงสร้างแบบ Tree ซึงไม่พบปัญหาการขัดแย้งของกฎ และทำงานได้เร็วกว่าไฟร์วอลล์แบบพื้นฐาน