โปรแกรมควบคุมระบบความไฟล์วอลิบนลีนุกซ์

Linux Firewall Management Software

โดย

วิชิตฤกษ์ บริบูรณ์

รหัส 45066058

อาจารย์ที่ปรึกษา

ผศ.ดร.โชติพัชร์ ภรณวลัย

รายงานนี้เป็นส่วนหนึ่งของวิชาโครงการพัฒนาระบบงาน

หลักสูตรวิทยาศาสตรมหาบัณฑิต สาขาวิชาเทคโนโลยีสารสนเทศ

ภาคเรียนที่ 2 ปีการศึกษา 2547

คณะเทคโนโลยีสารสนเทศ

สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

ชื่อหัวข้อ	โปรแกรมควบคุมระบบความปลอดภัยบนลีนุก
นักศึกษา	นายวิชิตฤกษ์ บริบูรณ์
อาจารย์ที่ปรึกษา	ผศ.ดร.โชติพัชร์ ภรณวลัย
ระดับการศึกษา	วิทยาศาสตรมหาบัณฑิต
แขนงวิชา	วิทยาการสารสนเทศ
ปีการศึกษา	2547

บทคัดย่อ

ระบบเครืออินเตอร์เน็ตในปัจจุบัน จะต้องมีระบบรักษาความปลอยภัย เพื่อป้องกันการโจมตีจากผู้ไม่หวังดี ซึ่งมีโปรแกรมที่สามารถป้องกันได้คือ iptables ซึ่ง Iptables เป็นโปรแกรมไฟล์วอลตัวหนึ่งบนลีนุกซ์ ซึ่งการใช้งานค่อนข้างจะยุ่งยาก จึงได้พัฒนาหน้าจอของผู้ใช้ขึ้นมา สั่งงาน iptables ผ่านทางเว็บเพจ แทนที่จะให้ผู้ใช้สั่งงานเอง จึงทำให้สามารถที่จะควบคุม iptables ได้ง่าย โดยที่ระบบจะทำงานอยู่บนระบบปฏิบัติการลีนุกซ์ และใช้ภาษา PHP และ Shell Script ในการพัฒนา ซึ่งจะทำให้ผู้ใช้สร้างระบบความปลอดภัยได้โดยง่าย โดยไม่จำเป็นต้องศึกษา Iptables

Title	Linux Security Management Software
Student	Mr. Wichitrerg Boriboon
Advisor	Asst. Prof. Dr. Chotipat Pornavalai
Level of study	Master degree of science
Major	Information science
Academic year	2004

ABSTRACT

Since all Internet network systems require protection against various attacks, firewall application can be used to minimize such attacks . Iptables is a firewall software on linux operating system but it is difficult to configure .This project was developed for help security administrator to manage and insert, update and delete policy . It was developed on Linux environment using PHP, Shell script, and HTML .

กิตติกรรมประกาศ

ขอขอบคุณบิดา-มารดา ที่ให้กำลังใจในการศึกษาตลอดมา รวมถึงการให้โอกาสในการศึกษาที่ทำให้มาจนถึงวันนี้ ขอขอบคุณ ผศ.ดร.โชติพัชร์ ภรณวลัย ผู้ให้คำปรึกษาและข้อเสนอแนะในพัฒนาโครงงานในครั้งนี้ รวมถึงเพื่อนๆ พี่ๆ น้องๆ ทุกท่านที่คอยกระตุ้นและให้กำลังใจตลอดมา

วิชิตฤกษ์ บริบูรณ์

14 กุมภาพันธุ์ 2548

สารบัญ

หน้า

สารบัญภาพ

รูปที่ หน้าที่

สารบัญภาพ(ต่อ)

รูปที่ หน้าที่

สารบัญตาราง

ตาราง หน้าที่

บทที่ 1
บทนำ

การเติบโตอย่างต่อเนื่องของอินเตอร์เน็ตในโลกปัจจุบันนี้ การติดต่อสื่อสารทางอินเตอร์เน็ตมีการโปรแกรมจำนวนมากในการติดต่อกัน ทั้งการสื่อสารหรือการรับบริการต่าง ๆ มากมาย เนื่องจากผู้ใช้จำนวนมากมายที่เข้ามาใช้อินเตอร์เน็ตในแต่ละวันและทุกคนเชื่อมต่อถึงกันหมดแล้ว จึงจำเป็นจะต้องมีระบบรักษาความปลอดภัยของทรัพยากรในระบบ ในโครงการนี้จึงมีการพัฒนาโปรแกรมจัดการไอพีเทเบิลบนระบบปฏิบัติการลีนุกเรดเฮด ซึ่งเป็นระบบความปลอดภัยอย่างหนึ่งที่ระบบปฏิบัติการลีนุกเรดเฮดเตรียมไว้ให้กับผู้ใช้ สำหรับป้องกันการรุกรานทรัพยากรในเครื่อขายของตน ดั้งนั้นรายงานนี้จะกล่าวถึงความเป็นมาและความสำคัญของปัญหา วัตถุประสงค์ของการพัฒนาระบบงาน เป้าหมายของการพัฒนาระบบงาน ประโยชน์ที่คาดว่าจะได้รับ ขอบเขตของการพัฒนาระบบงาน ขั้นตอนการทำงาน ทฤษฎีที่นำมาใช้ในการพัฒนาระบบงาน และรายละเอียดในบทต่าง ๆ

1.1 ความเป็นมาและความสำคัญของปัญหา

ปัจจุบันอินเตอร์เน็ตมีบทบาทสำคัญต่อการดำเนินกิจกรรมต่างๆ เป็นอย่างมาก ไม่ว่าจะเป็นด้านการติดต่อสื่อสาร ธุรกิจ การศึกษา หรือว่าเพื่อความบันเทิง องค์กรต่างๆ ทั้งภาครัฐและเอกชน ต่างก็นำเอาเน็ตเวิร์กของตนเชื่อมต่อเข้ากับอินเตอร์เน็ตเพื่อที่จะได้รับประโยชน์เหล่านี้ แต่เราต้องไม่ลืมว่าการนำเอาเน็ตเวิร์กไปเชื่อมต่อกับอินเตอร์เน็ตนั้น ทำให้ใครก็ได้บนอินเตอร์เน็ตสามารถเข้ามายังเน็ตเวิร์กนั้นๆ ได้ ปัญหาที่ตามมาก็คือความปลอดภัยของระบบเน็ตเวิร์ก เช่น ทำให้เกิดความเสี่ยงต่อการถูกเจาะระบบ และ ขโมยข้อมูล เป็นต้น

จากปัญหาดังกล่าวทำให้เราต้องมีวิธีการในการรักษาความปลอดภัย สิ่งที่สามารถช่วยลดความเสี่ยงนี้ได้ก็คือ ไฟร์วอลล์ โดยไฟร์วอลล์นั้นจะทำหน้าที่ป้องกันอันตรายต่างๆ จากภายนอกที่จะเข้ามายังเน็ตเวิร์กของเรา

โดยที่ไฟล์วอลใน ลีนุกเรดเฮด(Redhat 9) ซึ่งจะเรียกว่า “ไอพีเทเบิล” (IPTABLES) โดยที่จะให้มีการตั้งกฎต่างๆ ที่ใช้ในการกรอง packet ได้ โดยที่การจะยอมรับหรือยกเลิกนั้นก็จะขึ้นอยู่กับจะคิดอยู่กับคำสำคัญ (keyword) หรือฟิลด์ต่าง ๆ ที่อยู่ในแฟลม อย่างเช่น แอดเดรสต้นทาง แอดแดรสปลายยทาง พอร์ดต้นทาง และ ฯลฯ โดยที่ผู้ใช้สามารถที่จะแก้ไขค่าหรือคำสั่งตามรูปที่กำหนดไว้ จึงสามารถจะทำให้ไฟล์วอลทำงานได้ถูกต้อง ซึ่งค่อนข้างที่จะยุ่งยากและเป็นคำสั่งเฉพาะในการทำงาน

ร่วมถึงเครื่องมือต่าง ๆ ที่จะช่วยในการวิเคราะห์และบริหารจัดการไฟล์วอล อย่างเช่น การออกรายงาน การแสดง packet ที่เข้ามา หรือการตั้งเวลาการทำงาน

1.2 วัตถุประสงค์ของการพัฒนาระบบงาน

- เพื่อศึกษาการทำงานของไอพีเทเบิล
- เพื่อวิเคราะห์และออกแบบรวมถึงการพัฒนาโปรแกรมที่ควบคุมไอพีเทเบิลบนระบบปฏิบัติการลีนุกผ่านทางเว็บ
- เพื่อลดความยุ่งยากของคำสั่งจัดการไอพีเทเบิล
- เพื่อให้ผู้ใช้สามารถใช้งานได้ง่ายและจากที่ไหนก็ได้ภายในเครือข่าย

1.3 เป้าหมายของการพัฒนาระบบงาน

- โปรแกรมจัดการไอพีเทเบิลที่สร้างขึ้นจะเป็น Shell Script ที่เกิดจากการสร้างกฎจากเว็บที่ทำงานอยู่บนเว็บเซิร์ฟเวอร์จะเป็นตัวที่ติดต่อกับลีนุกที่เป็นระบบเดียวกัน
- ไอพีเทเบิลสามารถที่จะกรองและตรวจจับแพ็คเก็ตที่เข้ามาในระบบได้
- มี GUI ที่ทำงานได้ง่าย
- สามารถสร้างและลบกฎ ซึ่งจะมีผลทันที

1.4 ประโยชน์ที่คาดว่าจะได้รับ

ผู้ใช้สามารถที่จะได้ใช้งานได้สะดวกสบายมากขึ้น เนื่องจากสามารถที่จะทำงานที่เครื่องไหนก็ได้ และสามารถที่จะจัดการดูแลเครือข่ายได้ดี รวมถึงมีฟังชั่นการทำงานเพิ่มเติมเพื่อที่จะทำให้การทำงานมีประสิทธิภาพดีขึ้น สามารถที่ช่วยองค์กรประหยัดค่าใช้จ่าย ของระบบความปลอดภัย

1.5 ขอบเขตของการพัฒนาระบบงาน

1.5.1) ฟังชั่นการทำงานทั่วไป

- การเพิ่มกฎเข้าไปใหม่
- การลบกฎที่ไม่ต้องการออก
- การลบกฎทั้งหมดออก
- การแสดงกฎทั้งหมด

1.5.2) ฟังชั่นการทำงานเพิ่มเติม

- สามารถทำรายงานได้
- สามารถตรวจสอบการโจมตีแบบต่าง ๆ ได้

1.6 ทฤษฎีที่ใช้ในการพัฒนาระบบ

- หลักการของโปรโตคอลทีซีพี/ไอพี(TCP/IP) โดยเน้นที่โปรโตคอลไอพี ทีซีพี และยูดีพี
- การทำงานของไฟล์วอลทั่ว ๆ ไป
- ลักษณะของระบบปฏิบัติการลีนุกทะเล
- การทำงานของไอพีเทเบิล โดยที่ไอพีเทเบิลเป็นฟังชั่นหนึ่งของลีนุกทะเล ที่จัดไว้สำหรับระบบรักษาความปลอดภัย

1.7 ขั้นตอนการพัฒนาระบบงาน

- ศึกษาการทำงานของไอพีเทเบิล
- มองถึงปัญหาและข้อจำกัดต่าง ๆ ของไอพีเทเบิลที่เกิดขึ้นของฟังชั่นการทำงานของไอพี่เทเบิล
- ศึกษาการทำงานของไฟล์วอล
- หาแนวทางปัญหาที่จะเกิดขึ้น
- การออกแบบรูปแบบการจัดการไอพีเทเบิลเพื่อให้มีความปลอดภัยตามที่เราต้องการ
- ศึกษาถึงเครื่องมือที่จะนำมาใช้
- พัฒนาโปรแกรมจัดการไฟล์วอล พร้อมทั้งเอกสารการพัฒนา
- ทดสอบการทำงานของโปรแกรมที่พัฒนาแล้ว
- สรุปผลการทำงาน

1.8 เครื่องมือที่ใช้ในการพัฒนาระบบ

- Hardware : Intel Pentium 4 1.4 Ghz
- Software :
  - OS : Linux ,Windows XP
  - Application : VMware
  - Database : Mysql

บทที่ 2
ไอพีเทเบิล

ในการที่จะสร้างระบบงานขึ้นมานั้น สิ่งแรกที่จะต้องทำความเข้าใจอันดับแรก คือทำการศึกษาให้เข้าถึงการทำงาน และทฤษฎีต่าง ๆที่เกี่ยวข้องของระบบที่พัมนา เพื่อศึกษาข้อดีและข้อเสียขององค์ประกอบต่าง ๆให้เข้าใจและจะนำสิ่งต่าง ๆที่ได้ศึกษามารวบรวมออกแบบและทำการสร้างระบบงานให้ตรงความต้องการ ในโครงการนี้จะได้นำหลักและทฤษฎีต่างๆ ที่มีอยู่แล้ว เพื่อจะได้เหมาะสมกับโครงการที่พัฒนาได้แก่ การทำงานของโปรโตคอลทีซีพี/ไอพี่ โดยจะแน้นที่ ไอพี ทีซีพี ยูดีพี ในส่วนของไฟล์วอลจะเน้นในระดับพื้นฐาน

2.1 โปรโตคอลทีซีพี/ไอพี่

ในเครือข่ายอินเทอร์เน็ตนั้น Protocol ใน Transport layer ที่ให้บริการกับ Application layer โดยทั่ว ไปจะมีอยู่ 2 ตัว คือ ยูดีพี (User Datagram Protocol) ซึ่งให้บริการแบบ Unreliable และ Connectionless กับแอพพลิเคชั่นส่วนอีก protocol หนึ่งก็คือ ทีซีพี (Transmission Control Protocol)ซึ่งให้บริการแบบ Reliable และ Connection-Oriented กับแอพพลิเคชั่น ดังนั้นในขณะที่สร้างโปรแกรมผู้พัฒนาโปรแกรมจะต้องทำการระบุให้แน่ชัดว่าจะให้โปรแกรมนั้นใช้ protocol ใดใน protocol 2 อย่างนี้ เพื่อให้ง่ายต่อการทำความเข้าใจ ในขอบเขตของเครือข่ายอินเทอร์เน็ตทั้งหมด เราจะเรียก 4-PDU ว่าเป็น segment เพราะโดยปกติแล้วเรามักจะทำการเรียก PDU ของ ทีซีพี ว่าเป็น segment ส่วน PDU ของ ยูดีพี จะเรียกว่า Datagram แต่ก็ใช้คำว่า datagram เพื่อเรียก PDU ของ network layer ด้วยเช่นกัน สำหรับ ในเอกสารฉบับนี้ เราจะเรียกว่า segment แทน PDU ของทั้ง ยูดีพี และ ทีซีพี ใน Network layer ของ อินเทอร์เน็ต (ซึ่งจะอธิบายในบทที่ 4) Protocol ของ Network layer มีชื่อว่า IP (อินเตอร์เน็ต Protocol) ซึ่ง IP เป็นตัวกำหนดการสื่อสารแบบ logical ระหว่าง host โดยมีรูปแบบการบริการ เป็นแบบ Best Effort Delivery Service ซึ่งก็คือ IP จะพยายามอย่างดีที่สุดในการส่งข้อมูล หมายความว่า IP จะทำให้ดีที่สุดในการส่ง segments ระหว่าง host แต่ไม่มีการรับประกันใดๆทั้งสิ้น เช่น จะไม่รับประกันถึงการส่ง segment ว่าจะไม่สูญหาย, ลำดับของ segment และ ความสมบูรณ์ของข้อมูลใน segments ด้วยเหตุผลนี้ IP จึงเป็นการให้บริการแบบ unreliable หลักการพื้นฐานของ ยูดีพี และ ทีซีพี ก็คือการเพิ่มเติมความสามารถของบริการ IP จากบริการการ แจกจ่ายข้อมูลของ IP ระหว่าง end systems ให้สามารถบริการถึงระดับ process ของ end systems ได้ การ เพิ่มเติมความสามารถจากระดับ host-to-host ไปถึงระดับ process-to-process ในลักษณะดัง กล่าวคือการทำ multiplexing และ demultiplexing นอกจากนี้

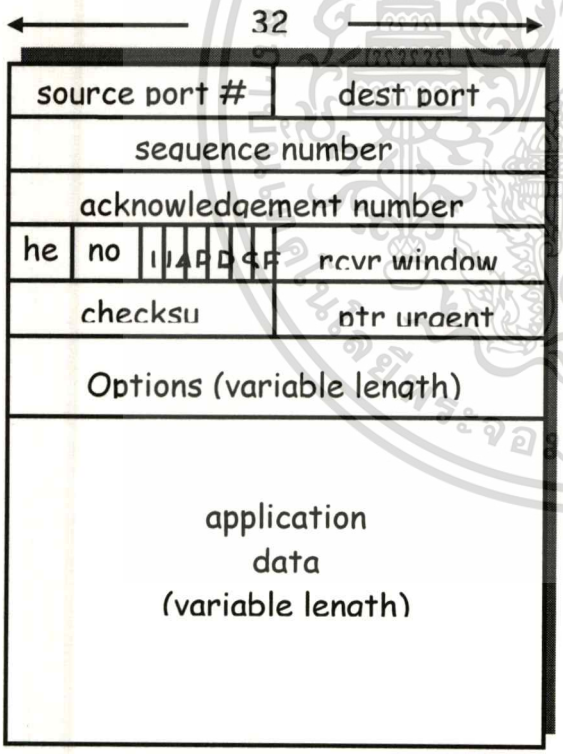
ทั้ง ยูดีพี และ ทีซีพี ยังมีวิธีการตรวจสอบความถูกต้องโดยวิธีการเพิ่ม field สำหรับตรวจสอบ error เข้าไปในส่วนของ header อีกด้วย ส่วน ทีซีพี ยังมีการให้บริการที่นอกเหนือจากนี้อยู่อีกหลายอย่าง โดยบริการที่สำคัญคือการส่งข้อมูล แบบ reliable โดยการใช้ flow control, sequence numbers, acknowledgements และ timers ทำให้แน่ใจได้ ว่าข้อมูลจะถูกส่งถึง process ฝั่งรับอย่างถูกต้องและตรงตามลำดับโดย ทีซีพี จะเปลี่ยนการนำส่งข้อมูลแบบ unreliable ระหว่าง host ของ IP ไปเป็นบริการนำส่งข้อมูลแบบ reliable ระหว่าง processes นอกจาก ทีซีพี ยังมีวิธีการควบคุมความคับคั่งของข้อมูล (Congestion control) ด้วย ซึ่งการควบคุมความคับคั่งของข้อมูลจะ ช่วยป้องกันไม่ให้การสื่อสารข้อมูลนั้นมีปริมาณมากกว่าขนาดของ bandwidth ที่จะสามารถให้บริการได้ โดย จะทำการควบคุมอัตราการส่งของฝั่งส่งให้เหมาะสมเพื่อเป็นการ share bandwidth ให้เท่าๆ กัน ซึ่งจะต่างจาก ยูดีพี ที่เป็นแบบ unregulates คือไม่มีการควบคุมอัตราการส่งข้อมูล ทำให้ข้อมูลที่ถูกส่งโดยใช้ ยูดีพี จะถูกส่ง เข้าไปในเครือข่ายตามอัตราการส่งที่ต้องการจะใช้ในช่วงเวลานั้นๆ นั่นเอง

2.2 โปรโตคอล ทีซีพี

ทีซีพี จะเหมือนกับ ยูดีพี ตรงที่สามารถทำการ multiplexing, demultiplexing และตรวจจับความผิดพลาดได้เหมือนกัน แต่สิ่งที่ต่างกันก็คือ ยูดีพี เป็น Connectionless ส่วน ทีซีพี เป็น Connection-Oriented ซึ่งจะต้องมีการ Handshake กันก่อนที่จะมีการติดต่อสื่อสารกัน เพื่อทำ การตั้งค่า parameters และ state variables ก่อนที่จะส่งข้อมูลจริง คำว่า Connection ใน ทีซีพี ไม่ได้หมายถึงการติดต่อแบบ end-to-end TDM หรือ FDM ใน circuit switched network และก็ไม่ได้เป็น virtual circuit ด้วย แต่จะหมายถึงการมี connection state และ มีการทำงานอยู่ที่ end-system (ไม่ได้ทำงานที่ intermediate network elements) ทีซีพี connection มี ลักษณะการส่งข้อมูลที่เป็นแบบ full duplex คือสามารถส่งข้อมูลได้ทั้งสองทิศทางในเวลาเดียวกัน และเป็น connection แบบ point-to-point คือ จะผู้ส่งหนึ่งเพียงหนึ่งคนต่อผู้รับเพียงหนึ่งคนเช่นกัน เราเรียก host ที่เป็นตัวสร้าง connection ว่าเป็น client ส่วนอีกด้านที่ client ไปติดต่อด้วยนั้นเรียก ว่า server โดย client จะส่ง segment แรกไปยัง server และทาง server จะตอบกลับมาเป็น segment ที่สอง และสุดท้าย client ตอบกลับไปยัง server อีกครั้งด้วย segment ที่สาม สำหรับ 2 segment แรกจะไม่มี payload คือไม่มี application-layer data ส่วน segment ที่สามอาจจะมีหรือไม่มี payload ก็ได้ (คืออาจจะมี การส่งข้อมูลไปด้วยก็ได้) ขบวนการดังกล่าวเรียกว่า “three way handshake” เมื่อสร้าง connection เสร็จ แล้ว เราก็สามารถส่ง data ระหว่าง process ทีซีพี send buffer ของ connection นั้นๆ ซึ่งมีจองไว้อยู่แล้วในระหว่างที่ทำ three-way handshake และ ทีซีพี จะนำ data จาก send buffer มาใส่ใน segment โดย data ที่ใส่นำมาใส่นั้นจะต้องมีขนาดไม่เกิน Maximum Segment

Size (MSS) ซึ่งมีขนาดแล้วแต่การ implement ของระบบหรือกำหนดเอง ค่าทั่วไปก็คือ 1500 bytes, 536 bytes และ 512 bytes โดย MSS เป็นขนาดสูงสุดของ application-level data ที่ใส่ใน segment ได้ ไม่ใช่ขนาดสูงสุดของ segment ทีซีพี จะทำการเพิ่มส่วนของ ทีซีพี header เข้า ไปใน data เพื่อทำให้เป็น ทีซีพี segment แล้วส่ง segment ที่ได้นี้ไปยัง network layer ต่อไป ในทางกลับกันเมื่อ ทีซีพี ได้รับ segment ที่ส่งมาจาก Network layer ก็จะนำเอา data ใน segment นั้น ใส่ใน receive buffer จากนั้น แอพพลิเคชั่นก็อ่าน data จาก buffer นี้ไปใช้งานต่อไป เราจะเห็นว่า ทีซีพี connection ประกอบด้วยชุดของ buffers, variables และ socket ในคู่ของ host ที่ติดต่อกัน และจะไม่มีปรากฏใน intermediate network element ระหว่าง host (routers, bridges และ repeaters)

ทีซีพี segment ประกอบด้วยสองส่วนคือส่วนที่เป็น header และส่วนที่เป็น data ซึ่งส่วนที่เป็น data ก็คือกลุ่มของ application data ซึ่งมีขนาดไม่เกิน MSS ส่วนใหญ่จะส่งที่ขนาดของ MSS ยกเว้นกลุ่มสุดท้าย (last chunk) แต่ในบางครั้งก็มีการส่งที่มีขนาดน้อยกว่า MSS เช่น Telnet ที่มี data 1 byte เมื่อรวมกับheader ของ ทีซีพี (ปกติ 20 bytes มากกว่า header ของ ยูดีพี อยู่ 12 bytes) แล้ว segment ก็มีขนาดเพียง 21 bytesเท่านั้น จากรูป 2.1 จะแสดงโครงสร้างของ ทีซีพี segment ซึ่งมี header ประกอบด้วย field ต่างๆ ดังนี้

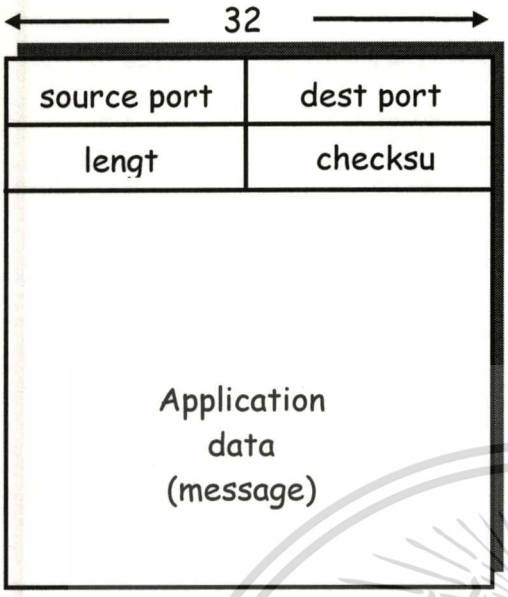


รูปที่ 2.1 โครงสร้างของ โปรโตคอล ทีซีพี

- Sequence number field และ acknowledgment number field ขนาด 32 บิต ซึ่งจะถูกใช้โดย ทีซีพีsender และ ทีซีพี receiver เพื่อตรวจสอบความถูกต้องของข้อมูล
- Windows size ขนาด 16 บิต ซึ่งจะใช้ ในการ Flow control และบอกจำนวน byte ที่ฝั่ง รับ สามารถรับได้
- Length field ขนาด 4 บิต บอกขนาดของ ทีซีพี header field ทั้งหมด
- Option field จะถูกใช้เมื่อฝ่ายรับ และฝ่ายส่ง ทำการ negotiate ขนาดของ segment มากท สุด (MSS)และ ในกรณีอื่นๆ
- Flag field มี 6 บิต คือ ACK บิต บอกว่าใช้ acknowledge field หรือไม่; RST, SYN และ FIN บิต ใช้ setup และ teardown connection ซึ่งจะอธิบายต่อไป; PSH บิต เมื่อ set เป็ น การบอกให ผ รับ ส่ง data ไปยัง upper layer ทันที; สุดท าย URG บิต ใช้บอกว่า data ใน segment นี้ upper layer ของผัส่งmarked ว่าเป็น "urgent" คือด่วนที่สุด เมื่อ ทีซีพี ฝั่งรับได้ รับการ จะบอก upper layer ทันที ว่ามีข้อมูลด่วนเข้ามาและส่งค่า pointer ที่ชี้ไปยัง ข้อมูลด่วนนั้นให้ด้วย (16-บิต urgent data pointer) ในทางปฏิบัติPSH, URG และ pointer ของ urgent data นั้นไม่ค่อยได้ใช้

2.3 ยูดีพี (Connectionless transport: ยูดีพี)

ใน Transport layer มี protocol สำหรับให้บริการแก่แอพพลิเคชั่น 2 ตัวด้วยกันคือ ยูดีพี และ ทีซีพี ตามที่ได้กล่าวมาแล้ว โดย ยูดีพี ถูกกำหนดอยู่ใน RFC768 ซึ่ง ยูดีพี นั้นมีการทำงานอยู่ใน Transport layer โดยมีหน้าที่ในการทำ multiplexing/demultiplexing และ การตรวจสอบข้อผิดพลาดของข้อมูลอย่างง่าย ๆ ถ้า เลือกใช้ ยูดีพี ในการพัฒนาแอพพลิเคชั่นจะหมายถึงแอพพลิเคชั่นนั้นสามารถทำการสื่อสารกับ IP ได้โดยตรง โดย ยูดีพี จะรับ message มาจาก process ของ แอพพลิเคชั่นและเพิ่มเติมหมายเลข port ของต้นทางและ ปลายทางในระหว่างการทำ Multiplexing นอกจากนั้นยังมี field อื่นๆ อีกเล็กน้อย ก่อนจะส่ง segments นั้น ต่อไปยัง Network layer ซึ่งเป็นชั้นที่จะนำ segment ไปบรรจุไว้ใน IP datagram แล้วหาเส้นทางที่ดีที่สุดเพื่อ ส่ง segment นั้นไปยัง host ปลายทาง เมื่อ segment เดินทางไปถึงปลายทางแล้ว ยูดีพี จะใช้หมายเลข port ที่บรรจุมากับ segment เพื่อส่งต่อข้อมูลที่ได้รับไปยัง process ที่ถูกต้องอีกต่อหนึ่ง เนื่องจาก ยูดีพี ไม่ได้ กระทำ handshaking ระหว่างตัวส่งและตัวรับก่อนทำการส่งมัน จึงถูกเรียกว่า connectionless นั่นเอง จากรูป 2.2 จะแสดงโครงสร้างของ ยูดีพี segment ซึ่งมี header ประกอบด้วย field ต่างๆ ดังนี้



รูปที่ 2.2 โครงสร้างของ โปรโตคอล ยูดีพี

เหตุผลสำคัญในการเลือกใช้ ยูดีพี เมื่อเทียบกับ ทีซีพี ในแง่ของผู้พัฒนาแอพพลิเคชั่น คือ

1. ยูดีพี ไม่ต้องทำการสร้าง connection เพื่อเริ่มการส่งข้อมูล แต่ในขณะที่ ทีซีพี จะใช้ Three-way handshake ก่อนจะเริ่มส่งข้อมูล ทำให้ ยูดีพี สามารถส่งข้อมูลไปได้ โดยไม่ต้องเตรียมการ จึงไม่ต้องมีการล่าช้าเนื่องจากการสร้าง connection ซึ่งมันอาจเป็นเหตุผลหนึ่งที่ DNS เลือกใช้ ยูดีพี เนื่องจากมันมีการ ทำงานที่เร็วกว่าทีซีพี มาก แต่สำหรับ HTTP ใช้ ทีซีพี เพราะต้องการความถูกต้องของ การ ส่งข้อมูล แต่การสร้าง connection ก็เป็นเหตุผลสำคัญอย่างหนึ่งที่ทำให้ HTTP เกิดความล่าช้า นั่นเอง

2. ยูดีพี ไม่มีการเก็บ connection state ไม่ว่าจะเป็น buffers ของตัวรับและตัวส่ง, congestion control parameters และหมายเลข sequence และ acknowledgement เหมือนกับ ทีซีพี ซึ่งใน ทีซีพี จะมีการเก็บ connection state ต่างๆเอาไว้ เพราะข้อมูลเหล่านี้จะมีจำเป็นในการส่งข้อมูลแบบ reliable และ congestion control สำหรับ ทีซีพี ทำให้ ยูดีพี ไม่มีความจำเป็นต้องใช้ parameters เหล่านี้ ด้วยเหตุนี้ server จึงสามารถให้บริการแก่ clients ได้ในจำนวนมากกว่าเมื่อมีการทำงานแบบ ยูดีพี

3. ยูดีพี มีขนาดของ header น้อยกว่าของ ทีซีพี คือมีแค่ 8 bytes เท่านั้นในขณะที่ header ของ ทีซีพี มีขนาดถึง 20 bytes

4. ยูดีพี ไม่มีการควบคุมอัตราการส่งข้อมูล ในขณะที่ ทีซีพี นั้นจะมี วิธีการในการควบคุมความคับคั่ง ของข้อมูล โดยจะทำการควบคุมอัตราการส่งทางฝั่งส่งให้ทำการลดอัตราการส่งเมื่อเห็นว่ามี link

ระหว่างฝั่งส่งและฝั่งรับเริ่มเกิดความคับคั่งขึ้น ซึ่งการควบคุมอัตราการรับส่งข้อมูลนี้จะมีผลกระทบ ต่อแอพพลิเคชั่น ประเภท real time multimedia ที่ยอมให้มีการสูญหายของข้อมูลได้แต่ต้องการ อัตราส่งที่แน่นอน ในทางตรงกันข้าม ความเร็วของการส่งข้อมูลของ ยูดีพี จะเพียงถูกบังคับโดย อัตราที่แอพพลิเคชั่นส่งมา, ความสามารถของตัวส่ง (CPU, ความถี่ Clock, ฯลฯ) และ bandwidth ของการส่งเข้าสู่ อินเตอร์เน็ต แต่เมื่อเครือข่ายเกิดความคับคั่งข้อมูลอาจมีการสูญหายได้เนื่องจาก buffer ของ router เกิด overflow อัตราการรับจึงถูกจำกัดโดยความคับคั่งของเครือข่ายในขณะที่ อัตราการส่งไม่ถูกจำกัดเลย

2.4 อินเทอร์เน็ตโพรโตคอล

เนื่องจาก network layer บนเครือข่าย อินเตอร์เน็ต ให้บริการแบบ connectionless datagram serviceคือเมื่อ network layer ฝั่งผู้ส่งได้รับข้อมูลจาก transport layer ก็จะใส่ข้อมูลนั้นเข้าไปใน IP datagramเขียน Address ของ host ผู้รับลงบน datagram และส่ง datagram นั้นไปตาม network เหมือนกับทีการเขียนจดหมาย ใส่จดหมายเข้าซองจดหมาย ระบุที่อยู่ปลายทางลงบนซองจดหมาย แล้วนำซองจดหมายนั้นไปไว้ในตู้ไปรษณีย์ โดยที่ อินเตอร์เน็ต's network layer และที่ทำการไปรษณีย์ ไม่ได้ติดต่อกับจุดหมายปลายทางไว้ก่อน ล่วงหน้าว่าจะส่งของไปให้ อย่างไรก็ตาม การให้บริการของ Network Layer จะเป็นแบบ best effort service (คือ การพยายามที่ จะรับส่งข้อมูลให้ดีที่สุด) ไม่มีการรับประกันว่า datagram นั้นจะถึงตามเวลาที่กำหนด และจะไม่รับประกันว่า datagram นั้นจะถึงในรูปแบบเดิมอย่างที่มาถึง ณ จุดเริ่มต้น ความจริงแล้วไม่มีการคำประกันแม้ แต่ datagram จะมาถึงจุดหมายปลายทางหรือไม่ด้วย Network Protocol มีอยู่ 2 ส่วนด้วยกันคือ network protocol ส่วนซึ่งมีหน้าที่กำหนด network-layer addressing และ action ที่ end systems and routers ต้องทำกับ datagram ที่ได้รับ และส่วนในการเลือกเส้น ทาง (path determination component) ซึ่งมีหน้าที่คอยตัดสินเส้นทางให้ datagram จากจุดเริ่มต้น จนถึง ปลายทาง network protocol ใน อินเตอร์เน็ต ซึ่งเรียกกันว่า อินเตอร์เน็ต Protocol นั้น โดยธรรมดาแล้วจะรู้จักกันใน IP Protocol ปัจจุบันมี 2 แบบที่ใช้กัน อย่างแพร่หลาย ตัวที่ได้รับความนิยมมากกว่าคือ อินเตอร์เน็ต protocol version 4 (RFC 791) หรือ IPv4 แต่ก็กำลังจะถูกแทนที่ด้วย IPv6 ในอนาคต

ก่อนที่จะพูดถึงเรื่อง IP Addressing เราต้องเข้าใจในเรื่อง hosts และ routers ก่อน โดย host (หรือที่เรียกกันว่า end system) ปกติมีเพียง 1 link ที่ทำงานกับ network เมื่อ IP ใน host ต้องการส่ง datagram มันจะผ่าน datagram ไปที่ link ซึ่งทำหน้าที่เป็นประตูระหว่าง host และ link เรียกว่า interface router มีหลักการที่ต่างจาก host ก็คือมี 2 links หรือมากกว่านั้น ไว้สำหรับการส่งผ่าน datagram โดย

จะส่งผ่านไปที่ link ใด link หนึ่งเพียง link เดียว โดยมี interface เป็นตัวกั้นระหว่าง router ซึ่งแสดงว่า router นั้นมีหลาย interfaces โดยจะมี 1 interface ต่อ 1 link เพราะทุก interface (ทั้งสำหรับ host หรือสำหรับ router) สามารถที่จะทั้งส่งและรับ IP datagrams, IP จึงต้องกำหนด IP address ให้ interface แต่ละ อันโดยที่แต่ละ IP address มีความยาว 32 บิต (4 bytes) IP address โดยทั่วไป จะเขียนแบบ "dotted-decimal notation" นั่นก็คือแต่ละ byte ของ address จะเขียนในรูปเลขฐาน 10 และแบ่งด้วย "จุด" เช่น IP address = 193.32.216.9 193 คือ ฐาน 10 ของ 8 บิต แรก ของ address 32 คือ ฐาน 10 (decimal equivalent) ของ 8 บิต ที่สอง ของ address ส่วนค่า address 193.32.216.9. ใน binary notation ก็คือ 11000001 00100000 11011000 00001001

IP address สามารถแบ่งได้ ออกเป็น คลาส ตามรูป สำหรับ address คลาส A 8 บิตแรก บอก network และ 24 บิต หลัง บอกตำแหน่งของ interface ใน network ดังนั้น ใน คลาส A สามารถมี networks ได้ถึง 27–1 networks หรือ 127 networks (บิต ที่ 1 ของ 8 บิต แรก จะเท่ากับ 0) และในแต่ละ network (ที่ เป็ น คลาส A) จ ะส าม า รถ มี interfaces ได้ 224 interfaces หรือ 16,777,216 interfaces คลาส B address ใช้ 16 บิต แรกเป็น network address (โดยที่ 2 บิต แรกต้องมีค่าเท่ากับ 10 เสมอ) และ 16 บิต ที่เหลือบอก interface address ดังนั้นใน คลาส B มี 214 Networks หรือ 16,384 และใน แต่ละ network สามารถมี interfaces ได้ 216 interfaces หรือ 65,536 interfaces. คลาส C address ใช้ 24 บิต แรกบอก network address (โดยที่ 3 บิต แรกต้องมีค่าเท่ากับ 110 เสมอ) และอีก 8 บิต ที่เหลือบอก interface address ดังนั้นใน คลาส C มี 221 Networks หรือ 2,097,152 networks และในแต่ละ network สามารถมี interfaces ได้ 28 interfaces หรือ 256 interfaces. คลาส D addresses ถูกสำรองไว้เป็น multicast addresses ซึ่งจะอธิบายต่อไปภายหลังว่า addresses เหล่านี้ ไม่ได้ระบุ interface แต่เป็นกระบวนการที่ทำให้ผู้ส่งสามารถส่งข้อมูลให้หลายๆ host ได้ โดยใช้ แพ็กเก็ต ชุดเดียว

2.5ไอพีเทเบิล

จากความรู้พื้นฐานเรื่องไฟร์วอลล์ ก็พอจะทราบว่า ไฟร์วอลล์ ในปัจจุบันนั้น แบ่งเป็นสองชนิดหลักๆ คือ state full packet filtering firewall และ proxy server ซึ่งจะมีการตรวจสอบข้อมูลที่ไหลผ่านเข้าออกที่คนละเลเยอร์กัน โดย packet filtering นั้นจะตรวจสอบข้อมูลที่ network layer และ session layer ในขณะที่ proxy นั้น สามารถตรวจสอบข้อมูลที่ application layer ได้ด้วย 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 ก็ได้สร้างเครื่องมือ

ตัวใหม่ชื่อ ipchains ซึ่งเผยแพร่ในปี 1998 โดย Rusty Russel และทีมงาน ทั้งนี้ ipchains นี้ถือได้ว่าเป็นพัฒนาการขั้นที่สามของ Linux Firewall จวบจนกระทั่งในปัจจุบัน ก็มี netfilter และ iptables ซึ่งถือว่าเป็นพัฒนาการขั้นที่สี่ของ Linux Firewall Netfilter นั้นเป็นชื่อใหม่ของโค้ดที่ทำหน้าที่เป็น packet handler(stateful inspection) ใน Linux kernel 2.4 (จริงคือเวอร์ชัน 2.3.15 และเวอร์ชันต่อๆ มา) ซึ่งได้ถูกออกแบบและปรับปรุงใหม่จากเวอร์ชันก่อนหน้านี้ เป็นเรื่องที่น่ายินดีคือ netfilter นั้นสามารถทำงานย้อนหลังร่วมกับ ipchains และ ipfwadm ได้ และคำสั่งในการเรียกใช้งานคือ iptables

2.5.1 ความแตกต่างระหว่าง iptables และ ipchains

- ชื่อของ built-in chain (ประกอบไปด้วย INPUT, OUTPUT, FORWARD) เปลี่ยนจากตัวอักษรเล็ก (lowercase) เป็นตัวอักษรใหญ่ (uppercase)
- การใช้งานที่ต้องระบุ port ทั้ง ทีซีพี และ ยูดีพี นั้น ต้องใช้คำว่า --source-port หรือ --sport (--destination-port หรือ --dport) และต้องใช้ตามหลังจาก -p ทีซีพี หรือ -p ยูดีพี
- ทีซีพี -y flag เปลี่ยนเป็น --syn และต้องใช้ร่วมกับ -p ทีซีพี
- target จาก DENY เปลี่ยนเป็น DROP
- chain ที่ไม่มี rule ใดๆ เลยก็สามารถทำงานได้
- การทำ zeroing built-in chain จะทำให้ byte counter ถูกล้างค่าไปด้วย
- ชื่อของ chain ยาวสูงสุดได้ 31 ตัวอักษร
- MASQ เปลี่ยนเป็น MASQUERADE และมีรูปแบบการใช้งานเปลี่ยนไป รวมทั้งREDIRECT ก็มีการเปลี่ยนแปลงรูปแบบใหม่

2.5.2 รายละเอียดโปรดศึกษาในส่วนของ NAT รูปแบบการใช้งาน iptables

iptables จะมีรูปแบบการใช้งานดังนี้คือ iptables [table] <command> <match> <target/jump> โดย rule ที่เขียนขึ้นจะเป็นเป็นตัวบอกเดอร์เนลว่าให้กระทำ action อย่างไร ในกรณีที่พบ แพ็กเก็ต ตรงตามที่ระบุไว้

- [table] หมายถึง ตารางหรือ table ที่ต้องการระบุ เช่น iptables -t nat หมายถึงให้ทำงานกับ nat table ในกรณีที่ไม่ได้ระบุตาราง iptables จะถือว่าคำสั่งดังกล่าวระบุถึง filter table โดยอัตโนมัติ
- <command> จะเป็นตัวสั่งให้ iptables ทำในสิ่งที่ต้องการ เช่น iptables -A INPUT ซึ่งหมายถึงให้สร้าง rule ต่อท้าย INPUT chain ใน filter table

- <match> เป็นส่วนที่ใช้ตรวจสอบว่า แพ็กเก็ต มีข้อมูลตรง (match) กับที่ระบุไว้หรือไม่ เช่น มี source ip address เป็น 1.2.3.4
- <target/jump> เป็นตัวระบุว่าเมื่อเจอ แพ็กเก็ต ที่ match ก็จะกระทำ (action) ตามที่ระบุไว้ เช่น ถ้า แพ็กเก็ต ใดมี source ip address เป็น 1.2.3.4 ให้ DROP แพ็กเก็ต นั้นทิ้งไป Table iptables สามารถทำงานได้กับตาราง(table) 3 ตารางหลัก สามารถระบุตารางได้โดยใช้ออปชัน -t ตามด้วยชื่อ table คือ

1. Filter table ใช้สำหรับกรอง แพ็กเก็ต มี 3 built-in chain คือ INPUT, OUTPUT, FORWARD ซึ่งจะได้อธิบายรายละเอียดต่อไป
2. Nat table ใช้สำหรับการแปลงแอดเดรส (Network Address Translation) มี 3 built-in chain คือ PREROUTING, POSTROUTING, OUTPUT ซึ่งรายละเอียดจะได้อธิบายต่อไป
3. Mangle table เป็นตารางที่ใช้ เปลี่ยนแปลงหรือแก้ไข แพ็กเก็ต เช่น เปลี่ยนค่า TTL, MARK ซึ่งปกติจะใช้ในการทำ routing ที่มีความซับซ้อนสูง มี 2 built-in chain คือ PREROUTING chain (ใช้แก้ไข แพ็กเก็ต ก่อนที่จะเข้าสู่ไฟร์วอลล์ และก่อนเข้าสู่ routing decision) และ OUTPUT chain (ใช้แก้ไข แพ็กเก็ต ที่ถูกสร้างโดยไฟร์วอลล์ก่อนที่มันจะถูกส่งไปยัง routing decision) ทั้งนี้ไม่สามารถทำ network address translation หรือ masquerading ที่ table นี้ได้ และในเอกสารฉบับนี้จะไม่กล่าวถึง mangle อีก เนื่องจากเป็นส่วนที่ไม่นิยมนำไปใช้งาน

2.5.3 คำสั่งต่าง ๆ ของการทำ NAT

- -A เพิ่ม rule ใหม่ต่อท้าย chain (Append rule) 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 ใหม่ ใน chain (Insert rule) เช่น

  # iptables -I OUTPUT -p ALL -s 127.0.0.1/32 -j ACCEPT
- -R แทนที่ rule เดิม ด้วย rule ใหม่ (Replace rule)
- -L แสดง rule ทั้งหมดใน chain (ถ้าไม่ระบุ chain จะแสดง rule ทั้งหมดใน filter table ทั้งสาม built-in chain) เช่น

  # iptables -L

iptables -L -t nat

iptables -L INPUT

- -F ลบ rule ทั้งหมดใน chain ทิ้ง เช่น

  # iptables -F INPUT

  # iptables -F mychain

- -Z ใช้ reset byte counter สำหรับทุก rule ใน chain ที่กำหนด เช่น

  # iptables -Z INPUT

- -N ใช้สร้าง chain ใหม่ เช่น

  # iptables -N mychain

- -X ลบ chain ที่ไม่มี rule ซึ่งสามารถลบ user-defined chain ที่ไม่มี rule ได้ แต่ไม่สามารถลบ built-in chain ได้ เช่น

  # iptables -X emptychain

- -P เปลี่ยน default policy ของ chain ค่าที่ใช้ได้คือ ACCEPT, DROP ทั้งนี้ค่านี้มีความสำคัญอย่างมากเพราะหาก แพ็กเก็ต ถูกส่งเข้ามาใน chain แล้ว และไม่ match กับ rule ใดๆ เลย แพ็กเก็ต นั้นก็ต้องถูกตัดสินใจโดย policy ของ chain นั้นๆ เช่น

  # iptables -P FORWARD DROP

  ซึ่งหาก แพ็กเก็ต ถูกส่งเข้ามายัง FORWARD chain และไม่ match กับ rule ใดๆ ใน FORWARD chain นี้เลย มันก็จะถูก DROP ทันที

- -E ใช้เปลี่ยนชื่อ chain ใหม่ เช่น

  # iptables -E myoldchain mynewchain

การใช้ command ด้านบนนั้นสามารถใช้ร่วมกับออปชันบางอย่างได้ คือ

- -V, --verbose ใช้ร่วมกับ -L, -A, -I, -D, -R เพื่อให้แสดงจำนวน byte ที่ match กับ rule ออกมาด้วย (หน่วยเป็นได้ทั้ง K(x1,000),M(x1,000,000),G(x1,000,000,000)) เช่น

  # iptables -L -v

- -x, --exact ใช้ร่วมกับ -L และ -v เพื่อให้แสดงจำนวน แพ็กเก็ต และจำนวนของ byte ข้อมูลที่ match โดยไม่ให้แสดงผลในหน่วยของ K,M,G เช่น

  # iptables -L OUTPUT -v -x

- -n, --numeric ใช้ร่วมกับ -L เพื่อสั่งให้ iptables แสดงข้อมูลไอพีแอดเดรส และ port เป็นตัวเลขเท่านั้น เช่น

  # iptables -L OUTPUT -n
- --line-numbers ใช้ร่วมกับ -L เพื่อแสดงเลขบรรทัดของ rule ซึ่งตัวเลขที่แสดงนี้จะสามารถใช้ได้กับคำสั่ง insert rule ที่ระบุเป็นลำดับที่ของ rule เช่น

  # iptables -L --line-numbers
- --modprobe=command เพื่อโหลด module ที่เกี่ยวข้อง

2.5.4 การตั้งเงื่อนไขของการ match

การตั้งเงื่อนไขของการ match นั้นจะต้องอาศัยความเข้าใจในเรื่อง IP, ทีซีพี, ยูดีพี, และ ICMP มาบ้างพอสมควร จึงจะสามารถตั้งเงื่อนไขที่เหมาะสมและตรงตามความต้องการได้ ซึ่งมีรายละเอียดดังนี้

- การระบุ source, destination IP addressสามารถระบุ source ip address ของ แพ็กเก็ต โดยใช้ -s หรือ --source หรือ --src และสำหรับ destination ip address ก็ใช้ -d หรือ --destination หรือ --dst การระบุไอพีแอดเดรสนั้นสามารถทำได้ 4 แบบด้วยกันคือ
  1. ใช้ชื่อเต็มแทน เช่น localhost หรือ www.nectec.or.th
  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
- การทำ Inversion ในบางกรณีนั้นหากต้องการระบุเป็น inverse เช่น อนุญาตให้ทุกไอพียกเว้นไอพีที่ระบุไว้ ซึ่งการใช้คำสั่งดังกล่าวสามารถทำได้โดยใช้เครื่องหมาย ! นำหน้า argument ที่ต้องการ (เครื่องหมาย ! หมายถึง NOT) เช่น -p ! ทีซีพี ซึ่งจะ match กับโพรโตคอลทุกๆ ตัวที่ไม่ใช่ ทีซีพี หรือ -s ! localhost ซึ่งหมายถึง แพ็กเก็ต ที่มี source ip address อื่นๆ ยกเว้น localhost (127.0.0.1)
- การระบุโพรโตคอล สามารถระบุโพรโตคอลที่ต้องการได้ดังนี้คือ ทีซีพี, ยูดีพี, ICMP หรือสามารถใช้ตัวเลขแทนได้ (สำหรับ *NIX อ้างอิงได้จาก /etc/protocols) และยังสามารถใช้ได้ทั้งตัวอักษรเล็กหรือใหญ่ (ใช้ได้ทั้ง ทีซีพี และ ทีซีพี) เช่น -p ทีซีพี หรือ -p ! ทีซีพี

- การระบุ interface -i หรือ --in-interface ตามด้วยชื่อ interface ใช้เพื่อระบุ incoming interface ซึ่งหมายถึงว่า แพ็กเก็ต ที่จะ match กับ rule นี้ต้องเข้ามาจาก interface ที่กำหนด เช่น -i eth0 หมายความว่า ทุก แพ็กเก็ต ที่เข้ามาทาง eth0 จะ match กับ rule นี้ ทั้งนี้ชื่อ interface ที่สามารถใช้ได้นั้น สามารถตรวจสอบได้โดยใช้คำสั่ง ifconfig และ -o หรือ --out-interface ตามด้วยชื่อของ interface ใช้เพื่อระบุ outgoing interface ซึ่งหมายถึงว่า แพ็กเก็ต ที่จะ match กับ rule นี้ กำลังจะเดินทางผ่าน interface ที่ระบุไว้ เช่น -o eth1 หรือ -o ! eth
  ตัวอย่างการใช้ --limit และ --limit-burst ร่วมกัน เช่น

  # iptables -A INPUT -m limit --limit 3/minute --limit-burst 3 -j LOG

โดยส่วนใหญ่นิยมวาง rule นี้ไว้เป็น rule สุดท้ายใน chain โดยเฉพาะ chain ที่มี default policy เป็น DROP เพื่อเป็นตัวเก็บหลักฐานว่ามี แพ็กเก็ต ใดที่ถูกส่งมาและไม่ผ่านการตรวจสอบจาก rule และกำลังจะถูก DROP โดย default policy โดย rule ด้านบนนี้กำหนดจำนวน match สูงสุดไว้ 3 ครั้งต่อนาที ซึ่งแสดงว่าใน 1 นาทีจะมีการบันทึกล็อกได้สูงสุด 3 ครั้งเท่านั้น และมีค่า burst เท่ากับ 3 ซึ่งอธิบายได้ว่า ถ้าสมมุติมี แพ็กเก็ต ที่ match กับ rule นี้ 3 ครั้งภายใน 2 วินาที และถึงแม้ว่าจะมี แพ็กเก็ต ที่ match ส่งมาอีกก็จะไม่มีการบันทึกลงล็อกแต่อย่างใด และจะต้องรอไปอีก 1 นาทีจึงจะมีการเริ่มการบันทึกล็อกใหม่อีกครั้ง ซึ่งมีประโยชน์ในกรณีที่มีคนต้องการส่ง แพ็กเก็ต เพื่อ flood log หรือทำให้ล็อกในเครื่องเต็ม ทั้งนี้นิยมใช้ร่วมกับ --log-level (อ้างอิงค่าจาก level ใน syslogd เพื่อกำหนดค่า level สำหรับ syslog) และ --log-prefix เพื่อใช้อธิบายเพิ่มเติม เช่น

iptables -A INPUT -m limit --limit 3/minute --limit-burst 3 -j LOG --log-level DEBUG --log-prefix "แพ็กเก็ต died: "

- นอกจากนี้ยังใช้ป้องกันการโจมตีแบบ Denial of Service เช่น SYN flood ได้ด้วย เช่น
  # iptables -A FORWARD -p ทีซีพี --syn -m limit --limit 1/s -j ACCEPT
- ใช้ป้องกันการโจมตีแบบ Ping of Death
  #iptables -A FORWARD -p icmp --icmp-type echo-request -m limit --limit 1/s -j ACCEPT
- โดยปกติมักใช้วิธี drop icmp packet ทิ้งทั้งหมด เพราะถือว่า ICMP แพ็กเก็ต เป็นข้อมูลที่มีอันตรายยิ่งและสามารถปลอมแปลงได้ง่าย หรือใช้ป้องกันการถูก scan
  # iptables -A FORWARD -p ทีซีพี --ทีซีพี-flags SYN,ACK,FIN,RST RST -m limit --limit 1/s -j ACCEPT

โดยปกติไม่นิยมใช้วิธีนี้นัก เพราะมีทางเลือกที่ดีกว่า คือการตรวจสอบสอบจาก state match ว่าเป็นการเชื่อมต่อใหม่หรือไม่ (state = new) ถ้าใช่และ SYN บิต ไม่ถูก set ตัว แพ็กเก็ต นั้นก็จะถูก DROP ทิ้งไป

2.5.5 การระบุ target

เมื่อมี แพ็กเก็ต ที่ match กับ rule แล้ว ต้องกำหนด target สำหรับ แพ็กเก็ต ไว้ด้วย โดยปกติจะใช้กัน 2 target คือ DROP และ ACCEPT นอกจากนี้ยังมี target แบบอื่นได้คือ

- user-definedchain

  เนื่องจาก iptables อนุญาตให้ผู้ใช้สามารถสร้าง chain ขึ้นมาได้ใหม่นอกเหนือจาก built-in chain ทั้งสามตัว (INPUT, OUTPUT, FORWARD) ทั้งนี้จะต้องใช้ตัวอักษรตัวเล็กทั้งหมดสำหรับ chain ที่ผู้ใช้สร้างขึ้นเอง เมื่อ แพ็กเก็ต match กับ rule ที่เป็น user-defined chain ตัว แพ็กเก็ต จะถูกนำไปตรวจสอบใหม่โดย user-defined chain นั้นๆ และถ้าใน chain นั้นๆ ไม่มีการตัดสินใจใดๆ ตัว แพ็กเก็ต ก็สามารถย้อนกลับมายัง rule ถัดไปใน chain ที่เริ่มต้นเดินทางได้ เช่น ถ้า ทีซีพี แพ็กเก็ต เดินทางจาก 192.168.1.1 ไปยัง 1.2.3.4 ดังนั้น แพ็กเก็ต จะเข้าสู่ INPUT chain และไม่ match กับ rule1 แต่ match กับ rule2 ซึ่งมี target เป็น test ดังนั้น แพ็กเก็ต จะเข้าสู่ test chain และ match กับ rule1 แต่เนื่องจาก rule1 ของ test ไม่ได้ระบุ target ดังนั้น แพ็กเก็ต จึงผ่านไปยัง rule2 ซึ่งไม่ match จากนั้น แพ็กเก็ต จึงจะเดินทางกลับไปยัง rule3 ของ INPUT chain อีกครั้ง ซึ่งก็ไม่ match เช่นกัน ในกรณีที่ผ่าน rule ทั้งหมดแล้วแต่ไม่ match หรือ match แต่ไม่มี target นั้น แพ็กเก็ต จะถูก DROP หรือ ACCEPT ก็ขึ้นอยู่กับ default policy ของ chain นั้นๆ ซึ่งสามารถตั้งค่าได้ง่ายๆ เช่น # iptables -P INPUT DROP หรือ # iptables -P FORWARD ACCEPT

- new target

  เป็น target ที่สร้างเพิ่มเติมขึ้นมาคือ

  - LOG

    เป็นโมดูลที่มีความสามารถในการเก็บข้อมูลลงล็อก (มี syslog facility เป็น kernel) สำหรับ แพ็กเก็ต ที่ match กับ rule ที่ระบุ target เป็น LOG มีออปชันให้เลือกใช้งานดังนี้คือ

    - --log-level

      เป็นการระบุ priority level ของ log ซึ่งกำหนดได้ตั้งแต่ debug, info ,notice, warning, crit, alert, emerg รายละเอียดเกี่ยวกับ syslog สามารถ

- log-prefix
  ตามด้วยชุดของตัวอักษรยาวไม่เกิน 29 ตัว ซึ่งชุดของตัวอักษรดังกล่าวจะปรากฏอยู่บนล็อกไฟล์

- REJECT
  คล้ายกับ DROP เพียงแต่จะส่ง ICMP port unreachable กลับไปยังผู้ที่ส่ง แพ็กเก็ต มา (ข้อยกเว้นคือ ICMP error message ไม่ respons กับ ICMP error message ด้วยกันเอง เพราะอาจจะทำให้เกิดลูปที่ไม่รู้จบ) ทั้งนี้สามารถใช้ร่วมกับ --reject-with ตามด้วย argument ที่ต้องการได้ รายละเอียดโปรดศึกษาจากคู่มือการใช้งาน iptables ที่มาพร้อมตัวโปรแกรม (#man iptables)

- special built-in target
  - RETURN
    กรณีที่ แพ็กเก็ต match กับ rule ที่มี target เป็น RETURN นั้นเสมือนกับเป็นคำสั่งให้ออกไปจาก chain ปัจจุบัน เช่น หาก match กับ rule ที่อยู่ใน built-in chain (INPUT, FORWARD, OUTPUT) แพ็กเก็ต ดังกล่าวจะถูกโยนไปยัง default policy ของ chain นั้นๆ และหาก แพ็กเก็ต match กับ rule ที่เป็น user-defined chain ตัว แพ็กเก็ต จะถูกโยนออกมา chain ก่อนหน้านั้น
  - QUEUE
    เป็น chain พิเศษ ใช้สำหรับส่งต่อ แพ็กเก็ต ไปยัง application ที่เขียนขึ้นมารองรับโดยเฉพาะ โดยจะต้องมี queue handler และ application เป็นส่วนประกอบที่จะทำงานร่วมกัน

2.5.6 การทำงาน แพ็กเก็ต ในระบบ

เมื่อ แพ็กเก็ต เข้ามาถึงไฟร์วอลล์ มันจะผ่านฮาร์ดแวร์เข้ามายัง device ที่เหมาะสมในเคอร์เนล จากนั้น แพ็กเก็ต จะเดินทางไปเป็นทอดๆ ก่อนที่จะถูกส่งไปยังปลายทางที่แท้จริง เช่น แอพลิเคชันในเครื่องไฟร์วอลล์ หรือ forward ต่อไปยังเครื่องอื่น ซึ่งจะยกตัวอย่างเพื่อให้เห็นภาพอย่างชัดเจนดังนี้

ตารางที่ 2.1 Forwarded packet

Step	Table	Chain	Comment
1			ข้อมูลอยู่ในระหว่างการเดินทาง เช่น กำลังมาจากอินเทอร์เน็ต
2			ข้อมูลเข้ามายังเครื่องไฟร์วอลล์ผ่านทาง incoming interface (เช่น eth0)
3	mangle	PREROUTING	ใช้สำหรับการทำ mangling packet เท่านั้น เช่น เปลี่ยนค่า TOS ของแพ็กเก็ต ซึ่งในกรณีปกติแล้วแทบไม่ได้ใช้งาน
4	Nat	PREROUTING	chain นี้ใช้สำหรับทำ Destination Network Address Translation ไม่ควรสร้าง rule เพื่อกรอง แพ็กเก็ต ที่ chain นี้ เพราะอาจจะมีบาง แพ็กเก็ต ที่ไม่เข้าสู่ chain นี้ (มีแค่ แพ็กเก็ต แรกเท่านั้นที่ผ่านเข้าสู่ chain ส่วน แพ็กเก็ต ถัดไปใน connection เดียวกันนั้น จะถูกกระทำเหมือนกับที่ แพ็กเก็ต แรกได้รับ)
5			เข้าสู่ Routing decision เพื่อตัดสินใจว่า แพ็กเก็ต จะถูกส่งไปที่ใด
6	Filter	FORWARD	เนื่องจากในตัวอย่างนี้ แพ็กเก็ต จะถูกส่งไปยังเครื่องอื่นในเครือข่าย ดังนั้น แพ็กเก็ต จึงต้องเข้า FORWARD chain ของ filter table ซึ่งสามารถเขียน rule สำหรับควบคุมการผ่านเข้าออกของ แพ็กเก็ต สำหรับ forwarded แพ็กเก็ต ได้ที่นี่
7	nat	POSTROUTING	และก่อนที่ แพ็กเก็ต จะออกไปจากเครื่องไฟร์วอลล์ โดยส่วนใหญ่ (ไม่ใช่ทั้งหมด) จะผ่าน chain นี้ ซึ่งใช้ทำ Source Network Address Translation ไม่ควรสร้าง rule เพื่อกรอง แพ็กเก็ต ที่ chain นี้ เพราะอาจจะมีบาง แพ็กเก็ต ที่ไม่เข้าสู่ chain นี้ (มีแค่ แพ็กเก็ต แรกเท่านั้นที่ผ่านเข้าสู่ chain ส่วน แพ็กเก็ต ถัดไปใน connection เดียวกันนั้น จะถูกกระทำเหมือนกับที่ แพ็กเก็ต แรกได้รับ)
8			แพ็กเก็ต ออกไปทาง outgoing interface (เช่น eth1)
9			แพ็กเก็ต เดินทางไปสู่เป้าหมาย (เช่น ผ่านทาง LAN)

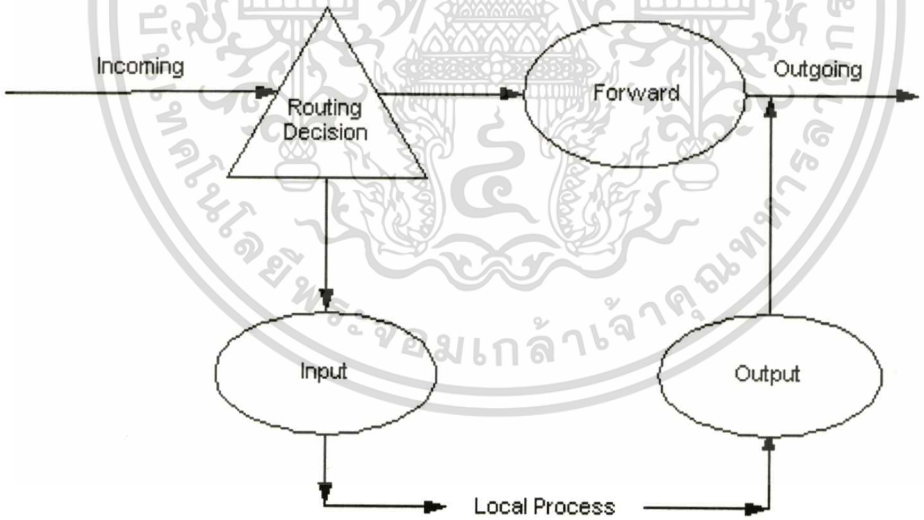
ตารางที่ 2.2 Destination localhost

Step	Table	Chain	Comment
1			ข้อมูลอยู่ในระหว่างการเดินทาง เช่น กำลังมาจากอินเทอร์เน็ต
2			ข้อมูลเข้ามายังเครื่องไฟร์วอลล์ผ่านทาง incoming interface (เช่น eth0)
3	mangle	PREROUTING	ใช้สำหรับการทำ mangling packet เท่านั้น เช่น เปลี่ยนค่า TOS ของแพ็กเก็ต ซึ่งในกรณีปกติแล้วแทบไม่ได้ใช้งาน
4	Nat	PREROUTING	chain นี้ใช้สำหรับทำ Destination Network Address Translation ไม่ควรสร้าง rule เพื่อกรอง แพ็กเก็ต ที่ chain นี้ เพราะอาจจะมีบาง แพ็กเก็ต ที่ไม่เข้าสู่ chain นี้ (มีแค่ แพ็กเก็ต แรกเท่านั้นที่ผ่านเข้าสู่ chain ส่วน แพ็กเก็ต ถัดไปใน connection เดียวกันนั้น จะถูกกระทำเหมือนกับที่ แพ็กเก็ต แรกได้รับ)
5			เข้าสู่ Routing decision เพื่อตัดสินใจว่า แพ็กเก็ต จะถูกส่งไปที่ใด
6	Filter	INPUT	ทุก แพ็กเก็ต ที่มีเป้าหมายเป็นเครื่องไฟร์วอลล์จะต้องเข้าสู่ chain นี้เสมอ ไม่ว่าจะมาจาก interface ใดก็ตาม
7			Local process/application (เช่น server/client program)
แพ็กเก็ต ที่มีปลายทางเป็นเครื่องไฟร์วอลล์นั้น จะต้องผ่าน INPUT chain เสมอ			

ตารางที่ 2.3 Source localhost

Step	Table	Chain	Comment
1			Local process/application (เช่น server/client program)
2	Mangle	OUTPUT	ใช้สำหรับการทำ mangling packet เท่านั้น การกรอง แพ็กเก็ต ที่ chain

Step	Table	Chain	Comment
			นี้จะไม่มีผลใดๆ ต่อ แพ็กเก็ต
3	Nat	OUTPUT	ไม่ได้ใช้งาน
4	Filter	OUTPUT	ใช้สำหรับกรอง แพ็กเก็ต ที่ออกมาจาก localhost หรือเครื่องไฟร์วอลล์เอง
5			เข้าสู่ Routing decision เพื่อตัดสินใจว่า แพ็กเก็ต จะถูกส่งไปที่ใด
6	Nat	POSTROUTING	และก่อนที่ แพ็กเก็ต จะออกไปจากเครื่องไฟร์วอลล์ โดยส่วนใหญ่ (ไม่ใช่ทั้งหมด) จะผ่าน chain นี้ ซึ่งใช้ทำ Source Network Address Translation ไม่ควรสร้าง rule เพื่อกรอง แพ็กเก็ต ที่ chain นี้ เพราะอาจจะมีบาง แพ็กเก็ต ที่ไม่เข้าสู่ chain นี้ (มีแค่ แพ็กเก็ต แรกเท่านั้นที่ผ่านเข้าสู่ chain ส่วน แพ็กเก็ต ถัดไปใน connection เดียวกันนั้น จะถูกกระทำเหมือนกับที่ แพ็กเก็ต แรกได้รับ)
7			แพ็กเก็ต ออกไปทาง outgoing interface (เช่น eth1)
8			Local process/application (เช่น server/client program)



รูปที่ 2.4 แสดงให้เห็นว่า แพ็กเก็ต มีเส้นทางการเดินทางอย่างไรเมื่อเข้ามาในระบบ (filter table)

ดังภาพ iptables ประกอบไปด้วย built-in chain จำนวน 3 chain ซึ่งไม่สามารถลบได้คือ INPUT, OUTPUT, FORWARD เมื่อเครื่องคอมพิวเตอร์เริ่มทำงานในครั้งแรก ทั้งสาม chain จะมี

default policy เป็น ACCEPT ซึ่งหมายความว่าอนุญาตให้ทุกอย่างผ่านเข้าออกได้หมด และสำหรับ FORWARD chain นั้น ถึงแม้จะได้กำหนดให้ policy เป็น ACCEPT แล้ว แพ็กเก็ต ก็จะยังไม่สามารถถูก forward ไปยังจุดหมายที่ต้องการได้ ตราบใดที่ยังไม่ได้เซ็ตให้ enable IP forwarding ทั้งนี้โดย default แล้ว forward=0 สามารถกำหนด ให้ enable IP forwarding (forward=1) ได้ โดย ใช้คำสั่ง echo "1" > /proc/sys/net/ip_forward เพื่อกำหนดให้ IP forwarding เป็น enable เพื่อให้ Linux box สามารถ forward ip แพ็กเก็ต ได้ ในบางครั้งนั้นการใช้คำสั่งดังกล่าวทุกครั้งอาจจะไม่สะดวก สามารถแก้ไขไฟล์ configuration ที่ /etc/sysctl.conf แล้ว set ให้ net.ipv4.ip_forward=1 เพื่อเป็นการแก้ไขแบบถาวร ในกรณีที่ต้องการให้สนับสนุนการทำงานกับ dynamic IP ด้วย เช่น PPP, SLIP, DHCP ก็สามารถทำได้โดยใช้คำสั่ง echo "1" > /proc/sys/net/ipv4/ip_dynaddr ได้เช่นเดียวกัน

บทที่ 3

การออกแบบระบบงาน

บทนี้จะกล่าวถึงขั้นตอนการทำงานของระบบและการออกแบบการทำงานของโปรแกรมทั้งหมด โดยจะแสดงในรูปของการไหลของข้อมูล

3.1 ความต้องการของระบบ

ความต้องการของระบบควบคุมไฟร์วอลผ่านผ่านทางเว็บที่จะพัฒนาขึ้น มีดังต่อไปนี้

- ระบบดังกล่างทำงานบนระบบปฏิบัติการลีนุกซ์ ซึ่งจะต้องติดตั้ง Apacheและ Mysql แล้วซึ่งในการพัฒนาใช้ภาษา PHP ในการพัฒนาระบบ
- ระบบจะทำการรับกฎผ่านทางเว็บแล้วนำไปเก็บในฐานข้อมูล แล้วจึงแปลงเป็คำสั่งของ iptables ในการสร้างกฎ
- ระบบจะสร้าง กฎเบื้องต้นให้ก่อน เพื่อจะได้ทำงานกับกฎที่เพิ่มมาได้ถูกต้อง
- ระบบจะต้องมีการสร้าง script ของ iptables เสมอเพื่อที่จะได้ทำการสร้างกฎให้เหมือนเดิมเมื่อมีการเริ่มระบบใหม่
- ระบบจะอ่าน log จาก syslog แล้วจึงจะนำไปใส่ในฐานข้อมูลเพื่อใช้ในการเรียกดู
- สามารถทำ NAT ได้ทั้ง Dynamic Nat,Static NAT , Port NAT
- เมื่อมีการลบกฎจะต้องมีผลทันที
- ในระบบผู้ใช้จะต้องสร้างกลุ่มของ Network object ก่อนจึงจะเลือกไปใส่กฎได้
- ผู้ใช้สามารถกำหนดได้ว่า Interface อันไหนจะชื่อว่าอะไร และสามรถที่จะแก้ไข IP ของแต่ละ interface

3.2 การออกแบบการทำงานของระบบ

การศึกษาการทำงานของ Iptables แล้วจึงได้แบ่งการทำงานของระบบออกเป็น ส่วนต่าง ๆ ดังนี้

- **การออกแบบเว็บเพจและการจัดเก็บลงฐานข้อมูล**

ในส่วนนี้จะเป็นการออกแบบเว็บเพจเพื่อให้ผู้ใช้ทำงานโดย ระบบจะมีหน้าเว็บดังนี้

1) หน้าหลักของระบบ และจะเป็นหน้าเว็บที่ login ด้วย
2) หน้าเว็บการสร้างกฎใหม่ให้กับ Iptables
3) หน้าเว็บการแสดงกฎที่กำหนดให้กับระบบไปแล้ว
4) หน้าเว็บการแสดง Nat ที่ได้กำหนดไปแล้ว
5) หน้าเว็บการกำหนด Nat ใหม่
6) หน้าเว็บการกำหนดobject ต่างๆ ของระบบ
7) หน้าเว็บการแสดงและตั้งค่าของระบบ
8) หน้าเว็บการแสดง Log ของระบบ
9) การออกจากระบบ

การจัดเก็บข้อมูลนั้นจะประกอบด้วยส่วนต่าง ๆ ดังนี้

1) จัดเก็บกฎของระบบฐานข้อมูล
2) จัดเก็บค่าต่าง ๆ ของระบบ
3) จัดเก็บObject ที่ได้กำหนดไว้
4) จัดเก็บ Log ของระบบ

- **การสร้างกฎให้กับ iptables**

โดยการสร้างกฎนั้นจะดึงมาจากกฎมาจาก ฐานข้อมูลแล้วจึงได้ทำการแปลงเป็นคำสั่งของ Iptables ซึ่งกฎจะอยู่บน forward Chain ของ Iptables โดยจะมีรูปแบบของคำสั่งดังนี้

- การสร้างกฎแบบ allow

iptables -A FORWARD -p <Protocol> -i <Incoming Interface> -o <Outgoing Interface > -s <Source Address> -d <Destination Address> --dport <Destination Port > -j ACCEPT

- การสร้างกฎแบบ Drop

iptables -A FORWARD -p <Protocol> -i <Incoming Interface> -o <Outgoing Interface > -s <Source Address> -d <Destination Address> --dport <Destination Port > -j DROP

- การสร้างกฎที่ต้องการ log

iptables -A WATCH -m limit -j LOG --log-level cri --log-prefix " ACCEPT "

- การสร้างกฎของการ NAT

iptables -t nat -A POSTROUTING -o <Outgoing Interface > -s <Source Address> -j MASQUERADE

- **การสร้างรูปแบบการอ้างอิงระหว่าง Iptables และ ฐานข้อมูล**

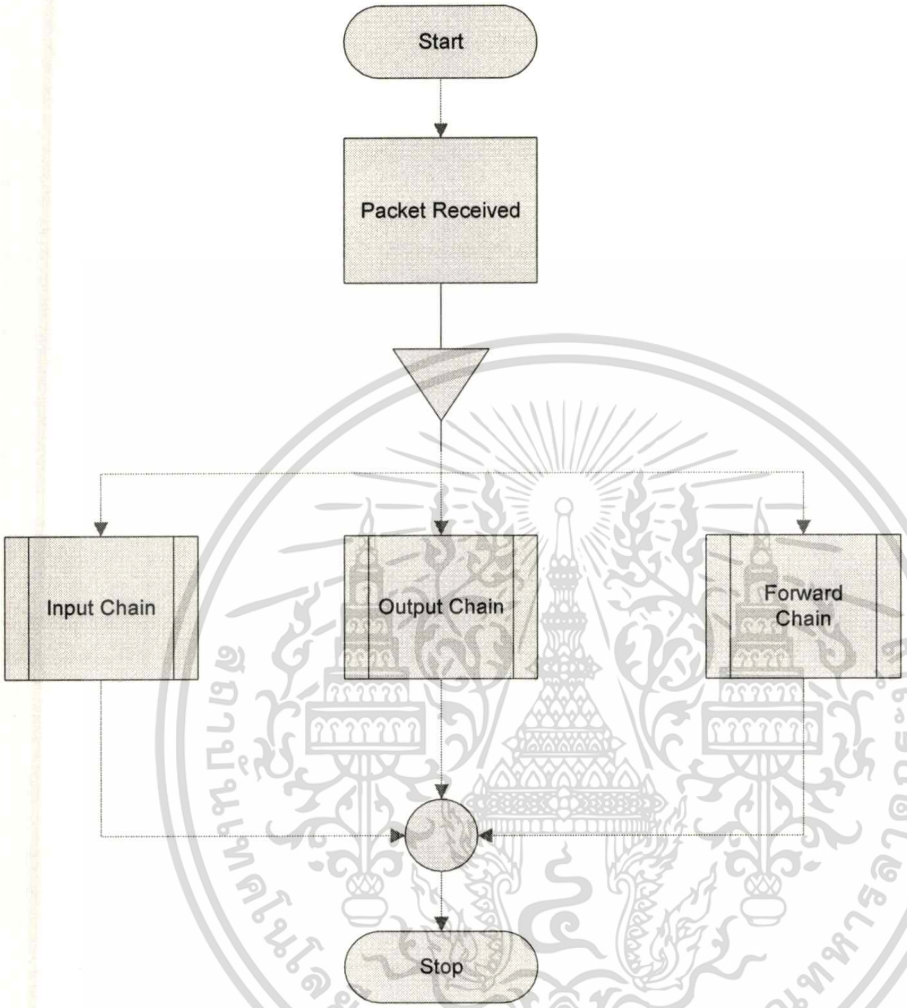
การทำงานของระบบจะต้องมีการลบและแก้ไขตลอดเวลา ซึ่งจะต้องการเก็บการอ้างอิงระหว่างระบบ และ iptables ซึ่งจะได้ให้มีความถูกต้องของการสร้างหรือลบกฎของ iptables ซึ่งได้ใช้หมายเลขของกฎใน iptables ในการอ้างอิงกฎ ดังตัวอย่าง รูปที่ 3.1

```
Chain FORWARD (policy DROP)
num  target     prot opt source               destination
1    ACCEPT     all  --  anywhere             anywhere            state RELATED,ESTABLISHED
2    CHECK_FLAGS  tcp  --  anywhere             anywhere
3    ACCEPT     tcp  --  anywhere             anywhere            tcp spt:ftp-data dpts:1024:65535
4    WATCH      tcp  --  192.168.0.0/24       192.168.1.0/24      tcp dpt:ftp
5    WATCH      tcp  --  169.254.23.0/24      192.168.0.2         tcp dpt:ssh
6    ALLOW_ICMP  icmp --  169.254.23.0/24      192.168.0.0/24
```

รูปที่ 3.1 แสดงเลขของกฎใน iptables

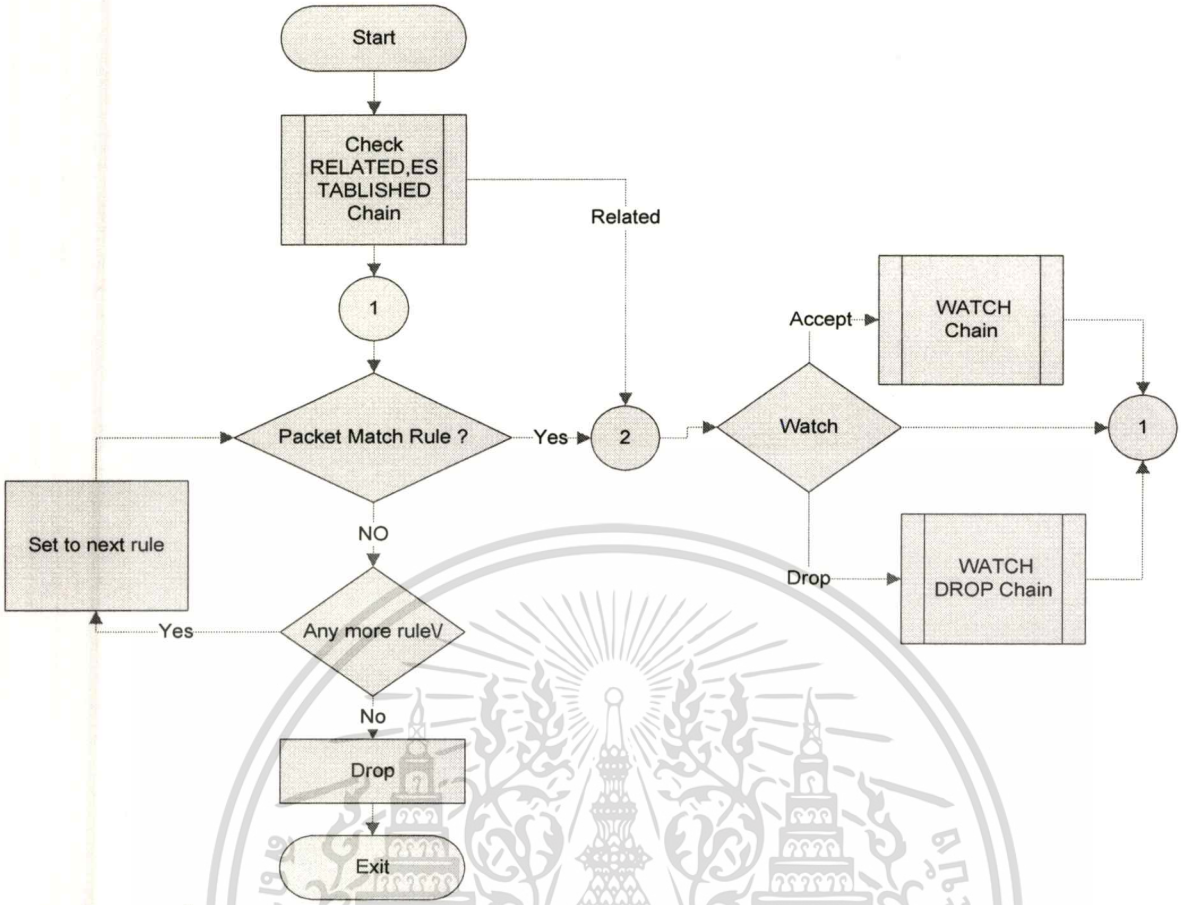
ซึ่งจะต้องเก็บในฐานข้อมูลเพื่อใช้ในการอ้างอิงในการสร้างหรือว่าลบกฎต่าง ๆ

3.3 การทำงานของระบบในโครงงานนี้



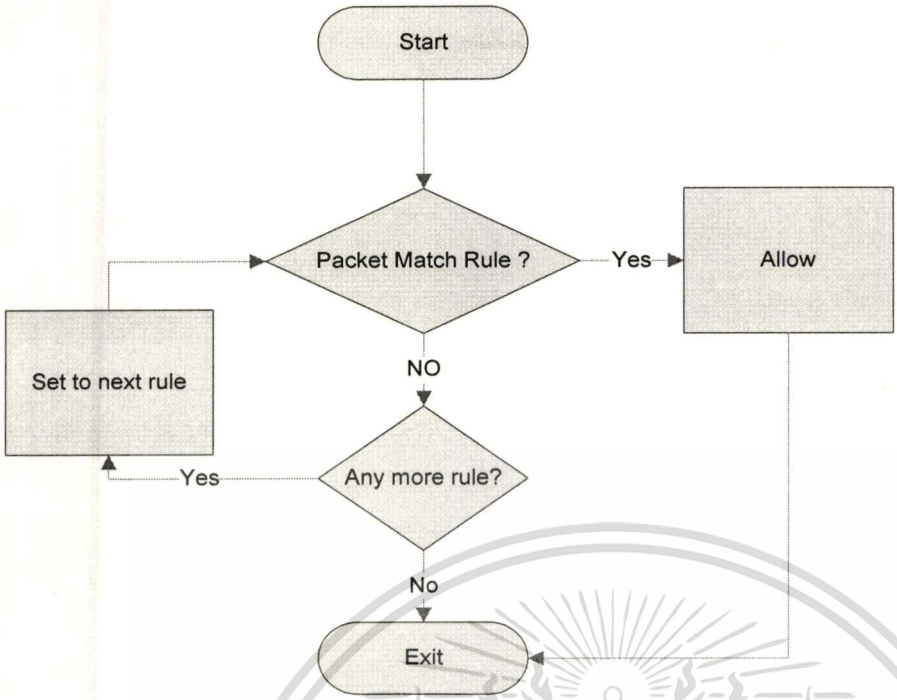
รูปที่ 3.2 โฟล์ชาร์ทการกรองเพ็คแก็ตโดยผ่านตามโซ่ต่างๆ

จากรูป 3.2 แสดงถึงการไหลของแพ็คแก็ตของ iptables โดยที่แพ็คแก็ตที่เข้ามานั้นจะแบ่งออกเป็นสามส่วนคือ ถ้าเป็นแพ็คแก็ตที่ต้องการผ่านตัวระบบไปจะใช้ Forward Chain ,ถ้าแพ็คแก็ตนั้นต้องการเข้าหาตัวเครื่องเองจะต้องผ่าน Input Chain แต่ถ้าเป็นแพ็คแก็ตที่ออกจากเองก็จะผ่าน Output Chain



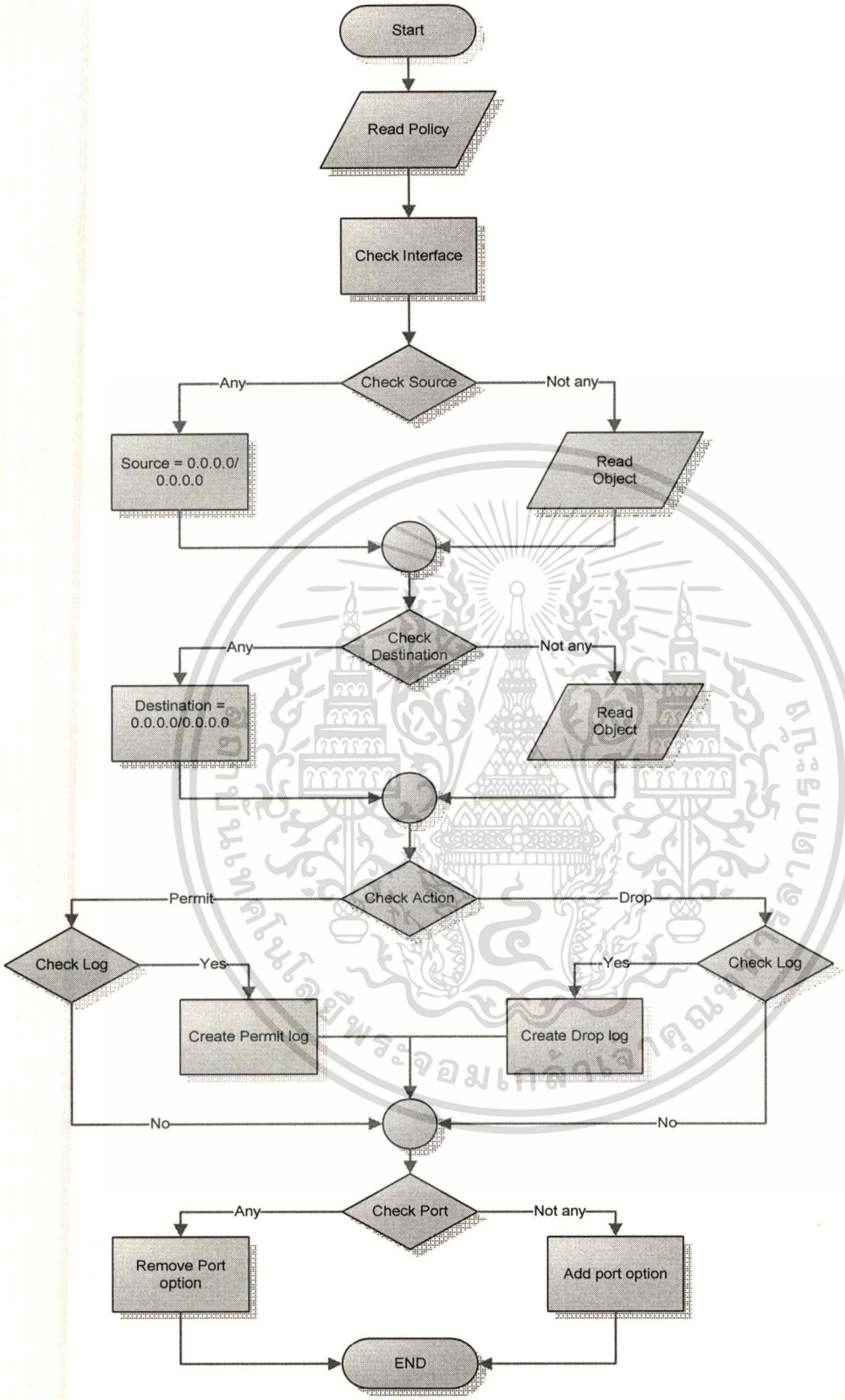
รูปที่ 3.3 แสดงการทำงานของแต่ละโซ่ของไอพีเทเบิล(Chain,forward Chain)

จากรูป 3.3 นี้จะแสดงถึงโซ่ต่าง ๆ ในระบบ คือทุกๆแพ็คแก็ตที่เข้ามาจะต้องทำการตรวจสอบว่าเคยได้รับการอนุญาตรึยังโดยใช้ RELATED และ ESTABISHED ถ้ามีแล้วก็จะให้ผ่านไป แต่ถ้ายังจะต้องไปตรวจสอบตามกฎต่าง ๆ ก่อนโดย ถ้าอนุญาตแล้วก็จะส่งไปยังโซ่ที่ทำการเก็บ Log ของระบบแล้วจึงจะให้ผ่านไปได้ แต่ถ้าไม่เจอกฎใด ๆ เลยก็จะโดนทิ้งไปเลย



รูปที่ 3.4 แสดงการทำงานของโซ่ย่อยต่าง ๆของไอพีเทเบิล

จากรูป 3.4 แสดงถึงการทำงานของ โซ่ย่อยต่าง ๆ ของระบบ ซึ่งทุก ๆ โซ่ย่อยก็จะมีการทำงานที่คล้ายกันจึงมากล่าวรวมกันโดยที่ทุกๆ แพ็คเก็ตจะต้องเข้าไปตรวจสอบตามกฎต่าง ๆ ถ้าไม่พบว่าเข้ากับกฎใดเลยก็จะโดนทิ้งไป ถ้าผ่านก็จะอนุญาตให้ผ่านไปได้

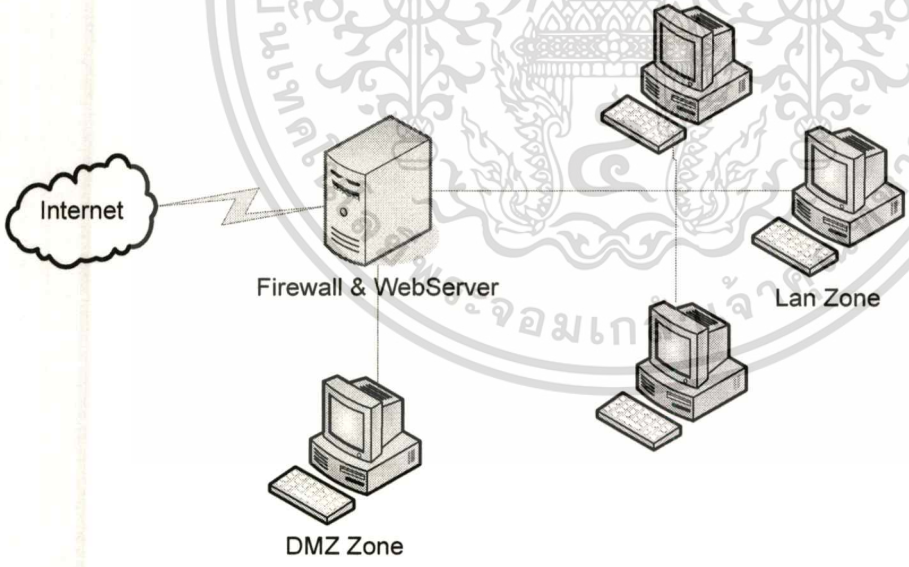


รูปที่ 3.5 แสดงการสร้างกฎจากฐานข้อมูลของระบบ

จากรูป 3.5 แสดงถึงการสร้างกฎของระบบให้กับ iptables โดยมีขั้นตอนดังนี้

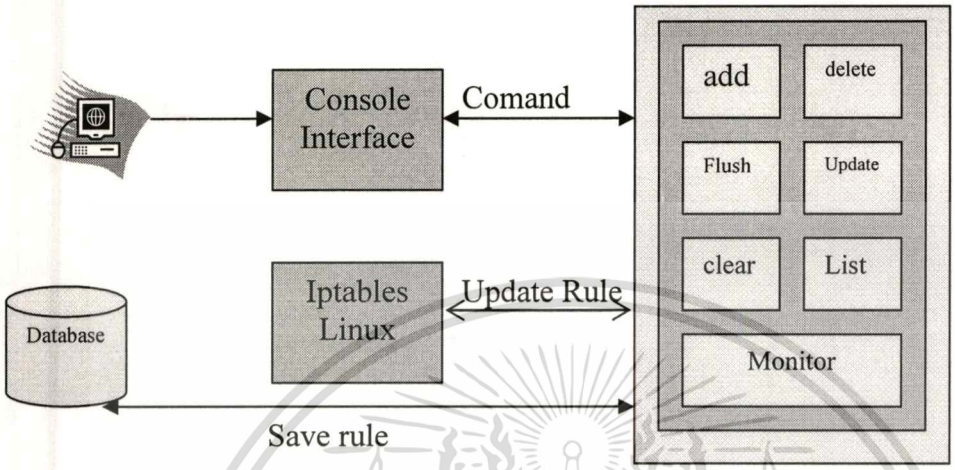
0) อ่านค่าต่าง ๆ ของกฎจากฐานข้อมูล
1) ตรวจสอบว่ากฎนี้จะต้องอยู่บน interface ใด
2) ตรวจสอบค่าของต้นทางโดยถ้าเป็น any กำหนดเป็น 0.0.0.0/0.0.0.0 ถ้าไม่ใช่ให้ไปอ่านค่าจาก ฐานข้อมูล ในตาราง Network Object
3) ตรวจสอบค่าของต้นปลายทางโดยถ้าเป็น any กำหนดเป็น 0.0.0.0/0.0.0.0 ถ้าไม่ใช่ให้ไปอ่านค่าจาก ฐานข้อมูล ในตาราง Network Object
4) ตรวจค่าว่า Action เป็น drop หรือ allow
5) ตรวจค่าว่าจะมีการเก็บ log หรือไม่
6) ตรวจค่าเบอร์พอร์ต ถ้าไม่มีก็จะเอาส่วนของเบอร์พอร์ตออก ถ้ามีก็จะใส่ด้วย
7) สร้างคำสั่งของ iptables

จากการทำงานตามที่กล่าวมา การทำงานของไอพีเทเบิลนั้น จะกล่าวในภาพรวมของโครงงาน ดังรูป 3.6



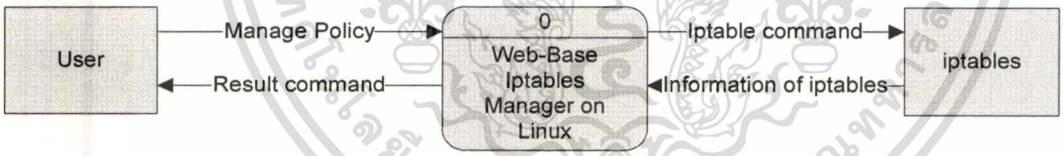
รูปที่ 3.6 ภาพรวมของโครงงาน

จะเห็นได้ว่าผู้ใช้สามารถที่จะควบคุมการทำงานผ่านเว็บบราวเซอร์ โดยที่จะสามรถควบคุมการไหลเข้ามาของเพ็กเก็ด โดยดูจากกฎที่ผู้ใช้ได้ตั้งขึ้นมา ซึงจะสามารถที่จะขวัดขวางหรืออนุญาติโดยที่ทำตามกฎ ซึ่งจะเก็บกฎไว้ที่หนึ่งเพื่อที่จะใช้ในการทำให้ไอพีเทเบิลอ่าน และทำงานตามกฎโดยที่รูปแบบจะเป็นดังรูป 3.7



รูปที่ 3.7 โครงสร้างภายในโครงงาน

3.4 ผังการไหลของข้อมูลของโครงงาน

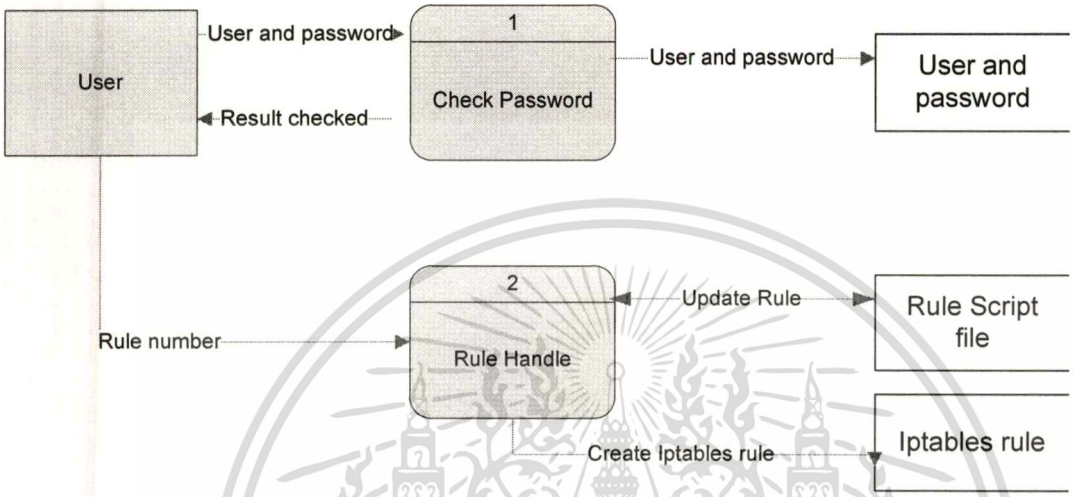


รูปที่ 3.8 แสดง Context Diagram ของโครงงาน

จากรูปที่ 3.6 แสดงถึง Context diagram ของระบบ จะเห็นว่า ส่วนต่างๆจะทำงานดังนี้

ระบบจะทำงานร่วมกันได้โดย

- ผู้ใช้จะสั่งงานผ่านทางเว็บและตัวเว็บจะแปลงคำสั่งเป็นคำสั่งของ iptables



รูปที่ 3.9 แสดงผังการไหลของข้อมูลระดับที่ 0

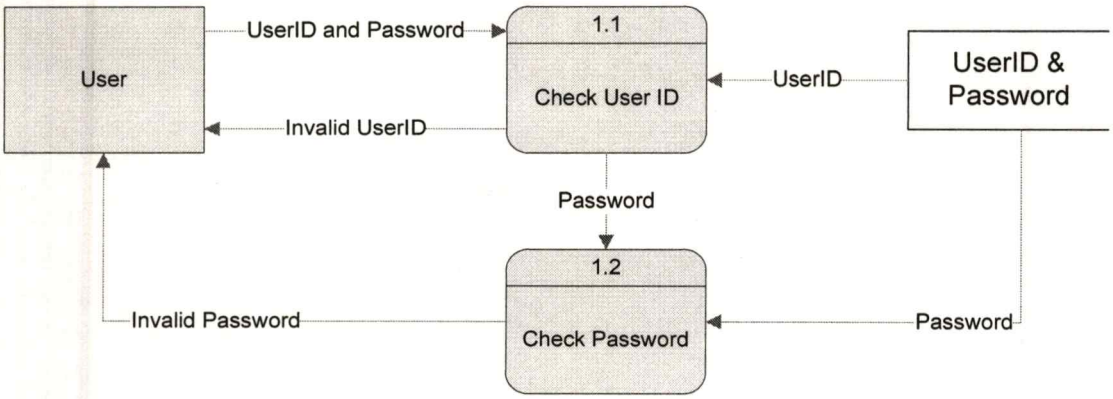
จากรูป 3.9 แสดงถึง Data flow Diagram Level 1

- **Check Password (1)**

เป็นส่วนเริ่มต้น โดยผู้ใช้จะต้อง login เข้าระบบก่อน โดยที่ผู้ใช้จะต้อง user และ password แล้วระบบเอาไปตรวจสอบกับระบบ เมื่อถูกต้องก็จะอนุญาติให้ผู้ใช้เข้าสู่ระบบได้

- **Rule Handle(2)**

เมื่อผู้ใช้จะทำงานกับ กฎจะต้องทำงานจะส่งกฎที่ต้องการจะดูและแก้ไขเข้าสู่ระบบ โดยที่ตัวระบบจะทำการสร้างกฎหรือแก้ไขกฎเอง



รูปที่ 3.10 แสดงผังการไหลของข้อมูลระดับที่ 1 กระบวนการที่ 1

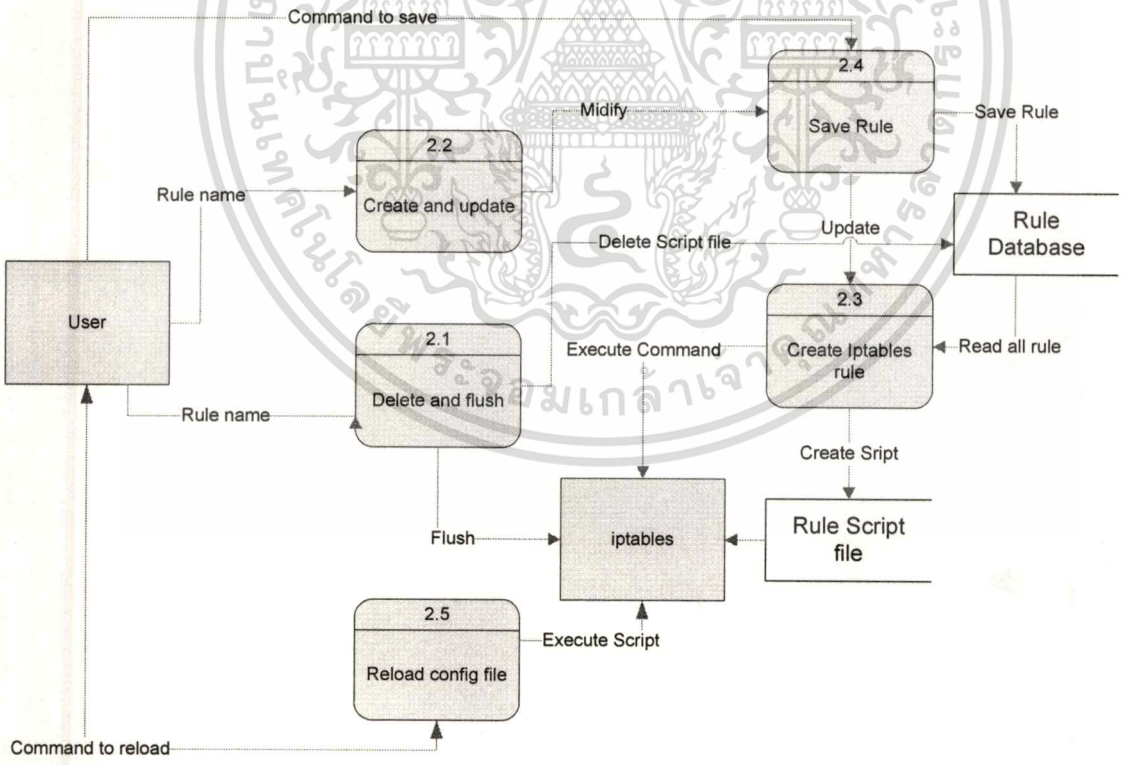
จากรูป 3.10 แสดงผังการไหลของข้อมูลระดับที่ 1 กระบวนการที่ 1

- Check User ID (1.1)

  เป็นการตรวจสอบว่ามี user id ในระบบหรือไม่

- Check Password (1.2)

  เป็นการตรวจสอบว่า password ถูกต้องหรือไม่



รูปที่ 3.11 แสดงผังการไหลของข้อมูลระดับที่ 1 กระบวนการที่ 2

จากรูป 3.11 แสดงผังการไหลของข้อมูลระดับที่ 1 กระบวนการที่ 2

- Create and update (2.2)

เป็นการรับข้อมูลกฎเข้ามาทำการสร้างกฎให้อยู่ในรูปแบบที่ต้องการและส่งต่อไปให้ทำการบันทึกต่อไป

- Delete and flush (2.1)

เป็นการรับกฎที่ต้องการจะลบออกจากระบบ พร้อมทั้งสามารถจะลบกฎทั้งหมดได้ด้วย แต่เมื่อมีการลบก็จะส่งค่าไปแก้ไขยังฐานข้อมูลด้วย

- Create iptable rule (2.3)

เป็นการดึงข้อมูลของกฎขึ้นมาทำการสร้างคำสั่งของ iptables

- Save rule (2.4)

เป็นการรับข้อมูลของกฎมาบันทึกลงฐานข้อมูล พร้อมสั่งให้มีการสร้างคำสั่งของ iptables ด้วย

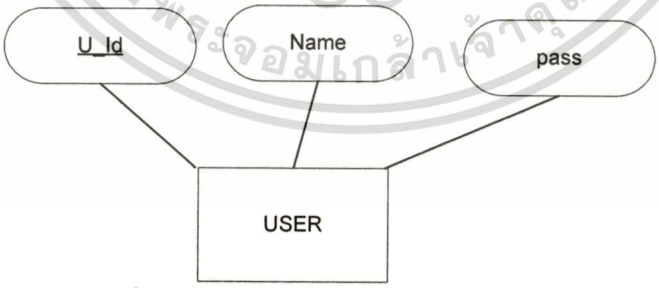
- Reload Config file (2.5)

เป็นส่วนที่ใช้งานเมื่อมีการ restart ระบบ โดยที่จะเอา script มาทำงานกับ iptables

3.5 การออกแบบฐานข้อมูล

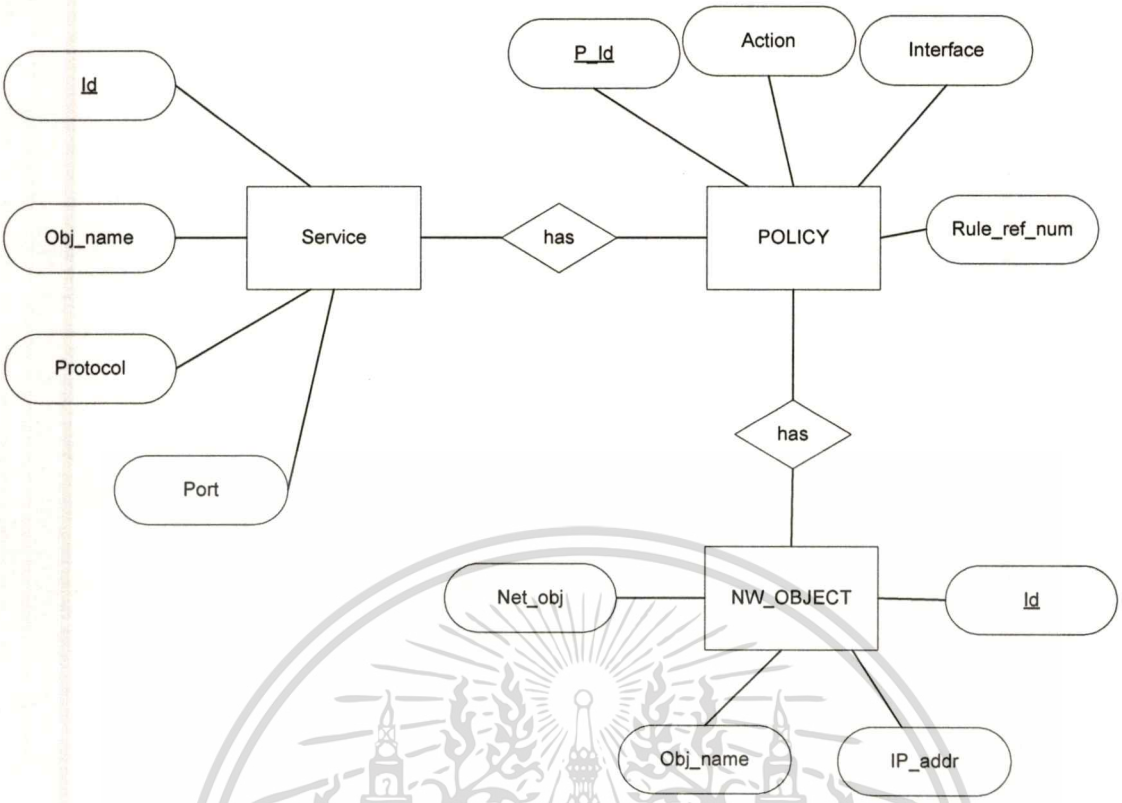
ในการพัฒนาระบบนี้นั้นจะเก็บข้อมูลเฉพาะส่วนของข้อมูลผู้ใช้ และ Policy ที่จะใช้ในระบบเท่านั้น เพื่อที่ช่วยในการทำ Script และง่ายต่อการเข้าใจและแก้ไข

โดยจะแบ่งออกเป็นสามส่วนใหญ่ คือตารางที่เก็บข้อมูลของผู้ใช้ที่จะมีสิทธิในการเข้ามราใช้ระบบเท่านั้น ตารางที่เก็บรายละเอียดของ Policy สุดท้ายจะเก็บค่าในการ setting ของระบบต่าง ๆ



รูปที่ 3.12 ER-Diagram ของ Entity User

จากรูป 3.2 เป็น ER-Diagram ของ ข้อมูลผู้ใช้โดยที่จะเก็บชื่อผู้ใช้ และ พาสเวิร์ดเอาไว้ โดยที่มี U_Id เป็น Primary Key ซึ่งในระบบจะรองปัจจุบันรองรับผู้ใช้ได้เพียงคนเดียว

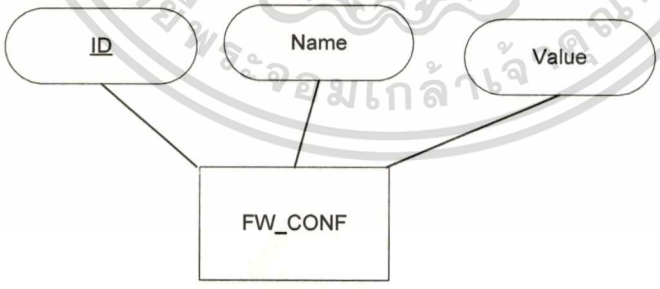


รูปที่ 3.13 ER-Diagram ของระบบ Linux Firewall Management software

จากรูป 3.13 แสดงถึงความสัมพันธ์ของของกฎกับการสร้าง object ต่าง ๆ ระบบ

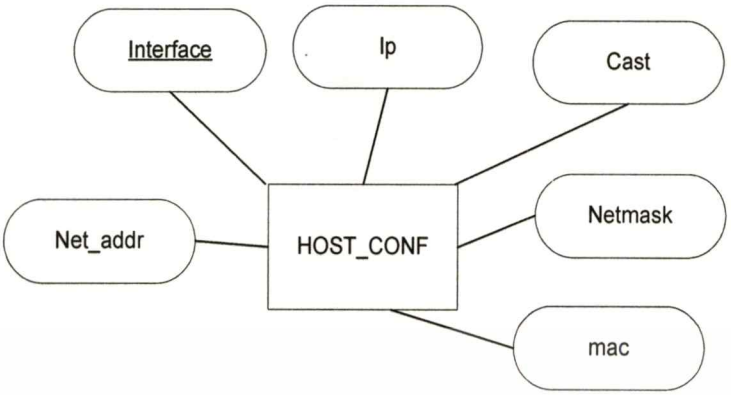
- ทุก ๆ กฎจะต้องมีการอ้าง object ไปยังส่วนของ service และ network จึงต้องมีการกำหนดค่าไว้ก่อน

- การสร้างกฎของระบบนี้จะต้องประกอบด้วย Service และ Object เสมอ



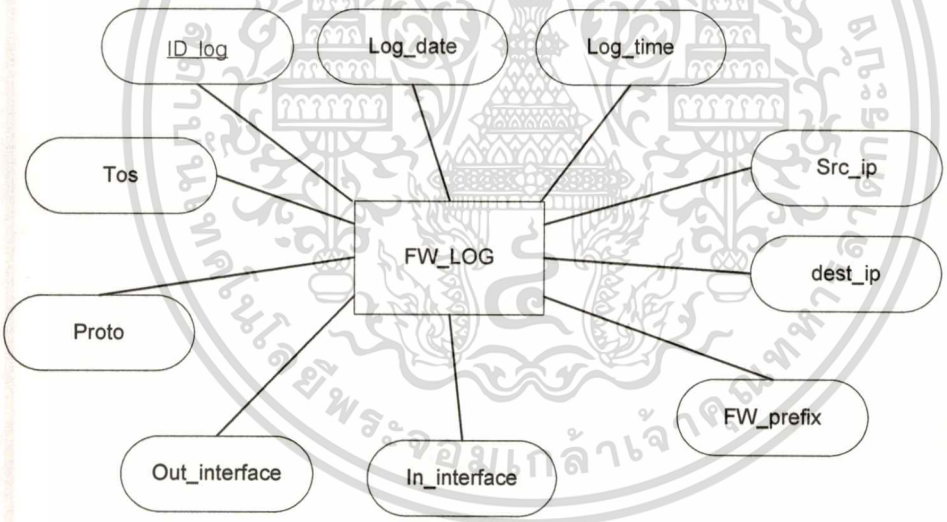
รูปที่ 3.14 ER-Diagram ของระบบ Fw_conf

จากรูป 3.14 เป็นส่วนที่เก็บค่าต่าง ๆ ของระบบ โดยที่ใช้ Name ในการบอกชื่อของสิ่งที่ต้องการจะเก็บ และ value จะเก็บค่าเอาไว้อย่างการ การกำหนดชื่อให้กับ interface ต่าง ๆ



รูปที่ 3.15 ER-Diagram ของ Host Config

จากรูป 3.15 เป็นตารางที่เก็บค่าของไอพีของ Server โดยที่จะอ่านมาใหม่ทุกครั้งที่มีการเริ่มต้นระบบใหม่



รูปที่ 3.16 ER-Diagram ของ Firewall Log

จากรูป 3.16 เป็นตารางที่เก็บ log ของระบบโดยค่าต่าง ๆ ที่เก็บจะโดนเพิ่มเข้ามาโดยระบบเอง

3.6 Data dictionary

ตารางที่ 3.1 user

Table Name : User Description : ข้อมูลผู้ใช้				
Name	Description	Type	Length	Key
U_ID	User ID	Number	9	PK
Name	User name	Varchar	10	
Password	Password	Varchar	10	

ตารางที่3.2 Policy

Table Name Policy Description : เก็บกฎของระบบ				
Name	Description	Type	Length	Key
P_Id	เลขกฎ	Number	9	PK
Soure	เป็น key อ้างมาจาก ตาราง mw_object	int	6	
Destination	เป็น key อ้างมาจาก ตาราง mw_object	int	6	
Service	เป็น key อ้างมาจาก ตาราง service_obj	int	6	
Action	บอกถึงทำงานของ นโยบายนี้	Varchar	9	
Interface	บอกว่านโยบายทำงาน ที่ interface ไหน	varchar	10	
Rule_ref_num	บอกลำดับที่ iptables	int	5	

ตารางที่ 3.3 mw_object

Table Name : nw_object Description : ข้อมูลของ object ในระบบ				
Name	Description	Type	Length	Key
Id	เป็น Primarykey	Number	9	PK
Obj_name	ชื่อของ object	Varchar	30	
Ip_addr	ไอพี	Varchar	20	
Net_obj	เน็ตมาส	Varchar	20	

ตารางที่ 3.4 Service_obj

Table Name : Service_obj Description : ข้อมูลของ service object ในระบบ				
Name	Description	Type	Length	Key
Id	เป็น Primarykey	Number	9	PK
Obj_name	ชื่อของ object	Varchar	30	
Protocol	ประเภทของ protocol	Varchar	20	
Port_num	เบอร์port	Varchar	20	

ตารางที่ 3.5 Fw_conf

Table Name :Fw_Conf Description : ข้อมูลของ parameter ในระบบ				
Name	Description	Type	Length	Key
Id	เป็น Primarykey	Number	3	PK
name	ชื่อของ parameter	Varchar	50	
value	ค่าของ parameter	Varchar	50	

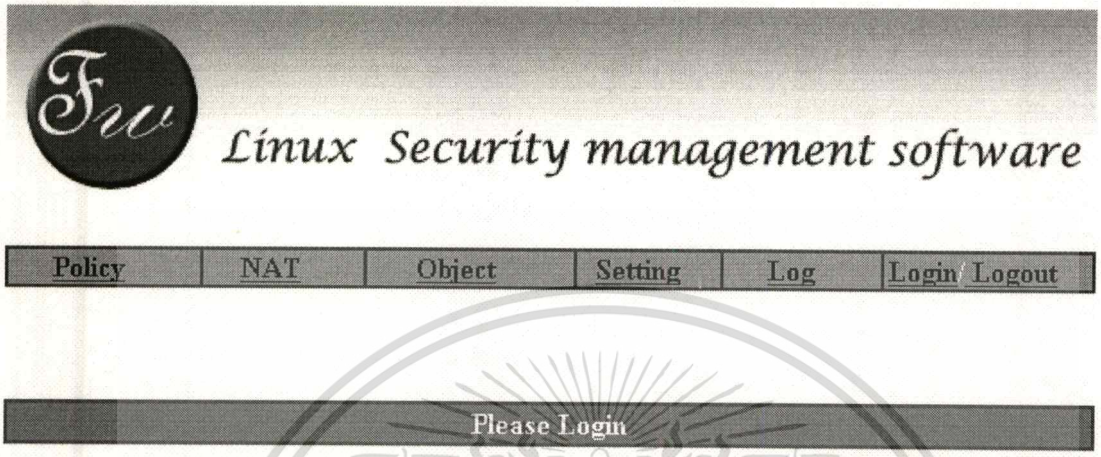
ตารางที่ 3.6 Host_conf

Table Name : Host_conf Description : ข้อมูลของ Server ของระบบ				
Name	Description	Type	Length	Key
Interface	ชื่อของ Interface	Number	9	PK
Ip	ไอพี	Varchar	20	
Cast	บรอดคาส	Varchar	20	
Mac	ค่า Mac address	Varchar	20	
Net_addr	ค่า network	Varchar	20	

ตารางที่ 3.7 FW_Log

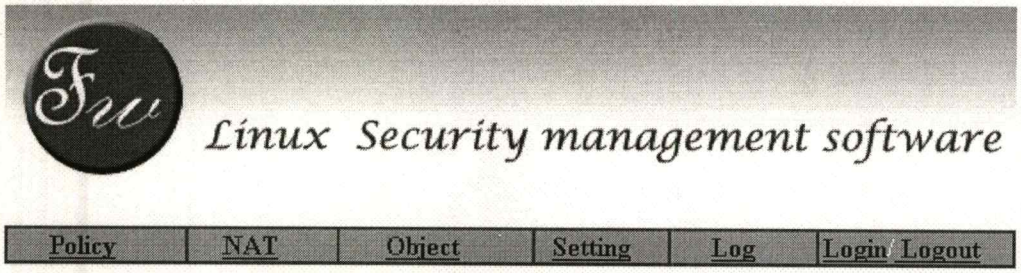
Table Name : FW_Log Description : เก็บ log ของระบบ				
Name	Description	Type	Length	Key
ID_log	เป็น Primarykey	Number	20	PK
Log_date	วันที่เก็บ log	Date		
Log_time	เวลาที่เก็บ log	Time		
Fw_prefix	เก็บ description	Varchar	20	
In_interface	ชื่อ Interface ขาเข้า	Varchar	9	
Out_interface	ชื่อ interface ขาออก	Varchar	9	
Src_ip	Ip ต้นทาง	Varchar	15	
Dest_ip	Ip ปลายทาง	Varchar	15	
Proto	โปรโตคอล	Varchar	10	
Extra_detail	รายละเอียดอื่น ๆ	Varchar	30	

3.7 การออกแบบส่วนติดต่อของผู้ใช้



รูปที่ 3.17 หน้าจอเข้าสู่ระบบ

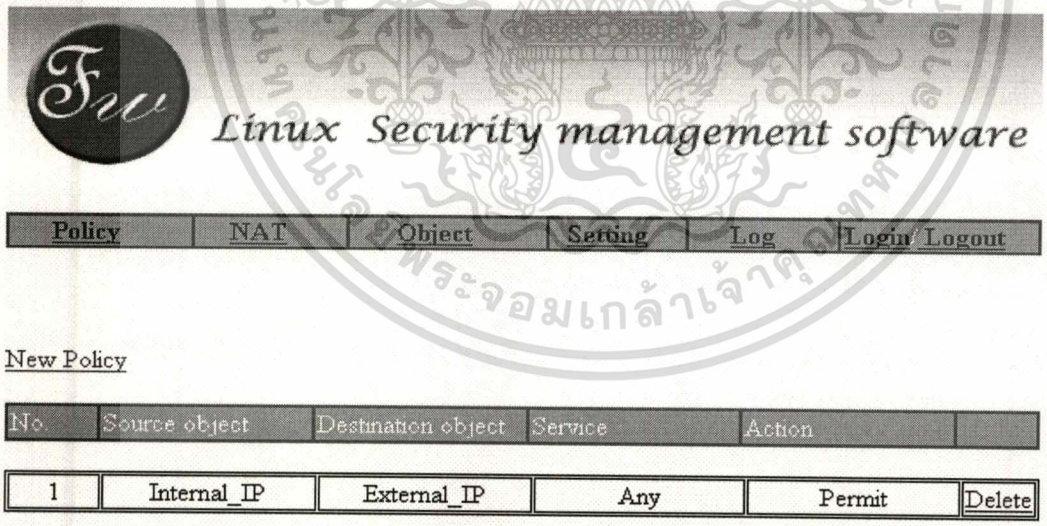
จากรูป 3.17 เป็นหน้าจอหลักที่ผู้ใช้จะต้องใส่ User name และ Password เพื่อที่จะเข้าสู่ระบบ โดยที่ทุกครั้งจะต้องเจอหน้าจอนี้ก่อน และเข้าสู่ระบบไม่งั้นจะไม่สามารถกดเมนูใด ๆ ได้



Interface	
Internal --> External	Edit
Internal --> Dmz	Edit
External --> Internal	Edit
External --> Dmz	Edit
Dmz --> Internal	Edit
Dmz --> External	Edit

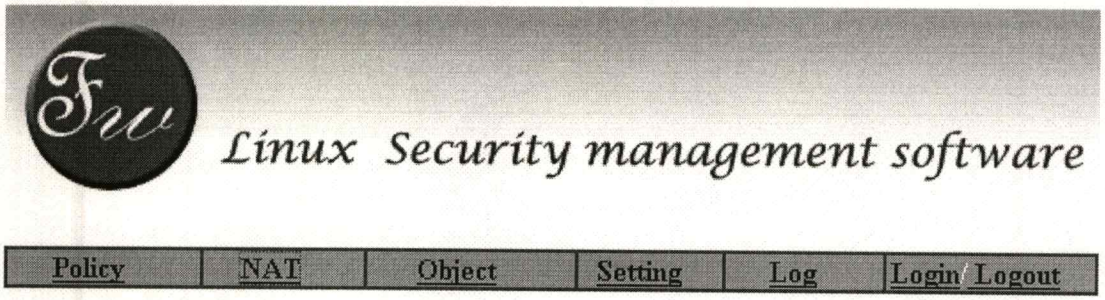
รูปที่ 3.18 หน้าจอเลือก interface เพื่อแสดง Policy

จากรูป 3.18 เป็นหน้าจอที่ให้ผู้ใช้เลือก interface ที่จะเข้าไปแก้ไข Policy ซึ่งการที่จะมาหน้าจอนี้ได้จะต้องเลือกที่เมนู Policy ก่อน



รูปที่ 3.19 หน้าจอแสดง Policy

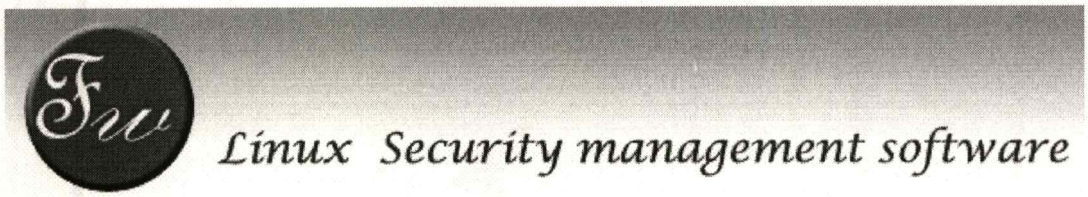
จากรูป 3.19 เป็นหน้าจอที่ให้ผู้ใช้ดู Policy ที่กำหนดไปแล้วในแต่ละ interface ซึ่งผู้ใช้จะต้องกดที่เมนู edit ของแต่ละ interface และสามารถจะเพิ่มหรือลบกฎของแต่ละ Interface ได้



รูปที่ 3.20 หน้าจอแสดงการตั้งค่า NAT

จากรูป 3.20 เป็นหน้าจอที่ใช้ในการตั้งค่า NAT ซึ่งจะต้องเลือกมาจากเมนู NAT ซึ่งจะสามารถกำหนดได้ 3 รูปแบบ คือ

1) Dynamic NAT คือการทำเนทจากเครื่องข่ายภายในไปยังภายนอกโดยใช้ไอพี ของ Interface ภายนอก
2) Destination port NAT คือการทำเนทขาเข้าโดยที่จะเอาเบอร์พอร์ตในการกำหนดว่าจะเนทไปหา Server ที่อยู่เครื่องข่ายภายใน
3) Static NAT คือการทำเนทแบบ 1:1 ระหว่างไอพีภายในและภายนอก



Policy	NAT	Object	Setting	Log	Login/ Logout

New Network object

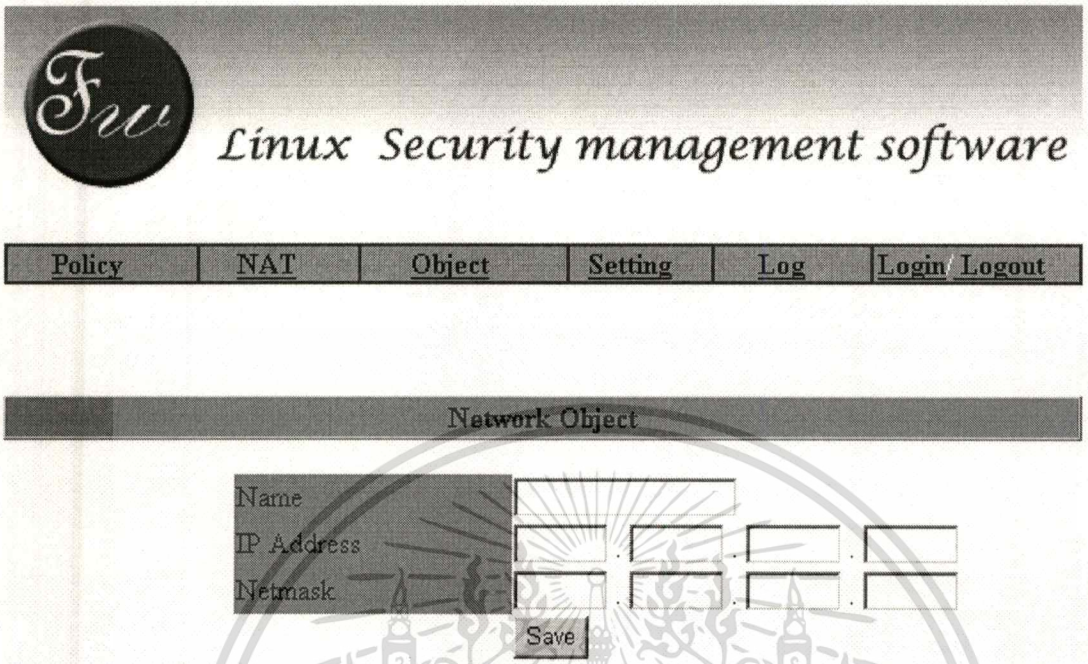
Network object	Name	IP Address	Netmask	
1	Internal_IP	192.168.0.0	255.255.255.0	Delete
2	External_IP	169.254.23.0	255.255.255.0	Delete
3	DMZ_IP	192.168.1.0	255.255.255.0	Delete
4	Http_Server	192.168.1.2	255.255.255.255	Delete
5	test_server	169.254.23.1	255.255.255.255	Delete

New Service object

Service object	Name	Protocol	Port	
2	Ssh	TCP	23	Delete
3	POP3	TCP	110	Delete
4	POP3(SSL)	TCP	995	Delete
5	SMTP	TCP	25	Delete
6	Telnet	TCP	23	Delete

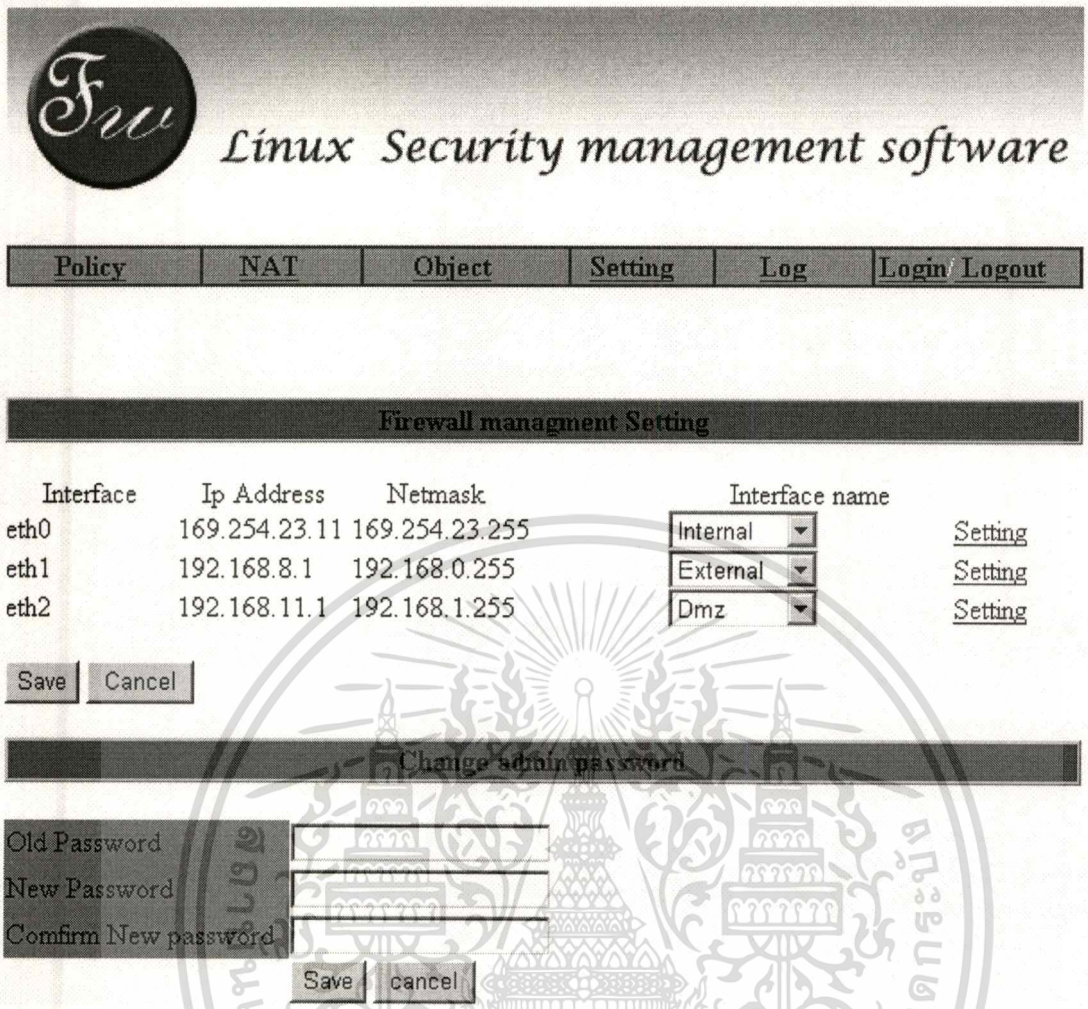
รูปที่ 3.21 หน้าจอแสดงการกำหนดค่าของ object ต่าง ๆ

จากรูป 3.21 เป็นหน้าจอที่มาจากการกดเมนู Object ซึ่งจะแสดงการกำหนดค่าของ network object และ Service object ที่ได้กำหนดไปแล้ว ซึ่งผู้ใช้สามารถที่จะลบ และเพิ่มใหม่ได้โดยกดที่เมนู New Network Object และ New Service object



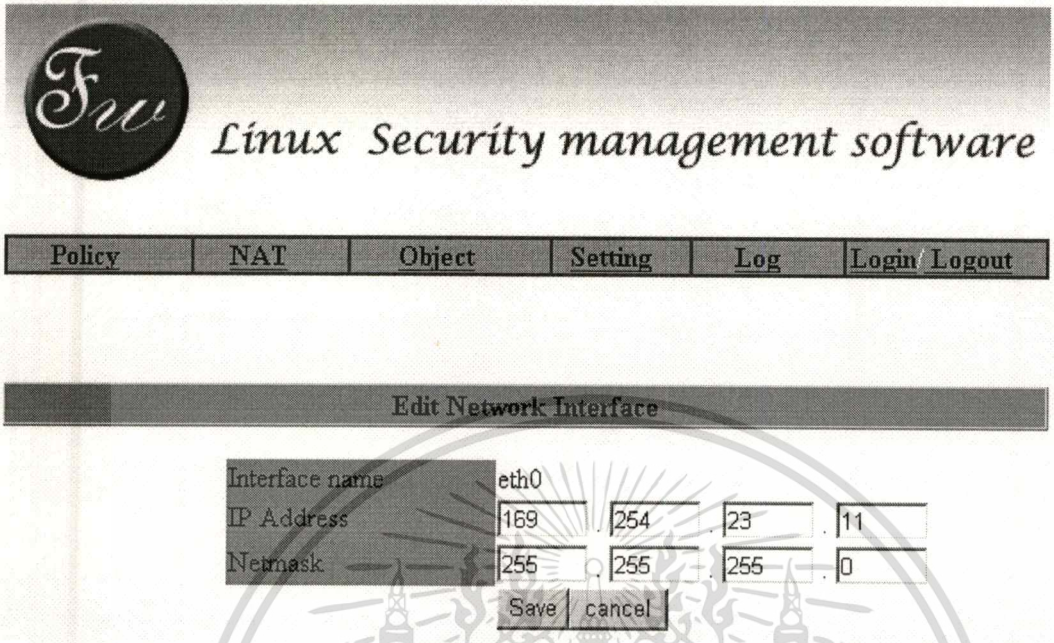
รูปที่ 3.22 หน้าจอแสดงการสร้าง object ใหม่

จากรูป 3.22 เป็นหน้าจอที่ใช้ในการสร้าง network object ใหม่ หลังจากการกดเมนู add network object ในเมนู Object ซึ่งจะให้ผู้ใช้ป้อนชื่อและ ไอพีที่จะต้องการไว้ และถ้าต้องการเพิ่มเป็นทั้งคลาสก็ให้ใช้ sub netmask ให้ถูกต้อง



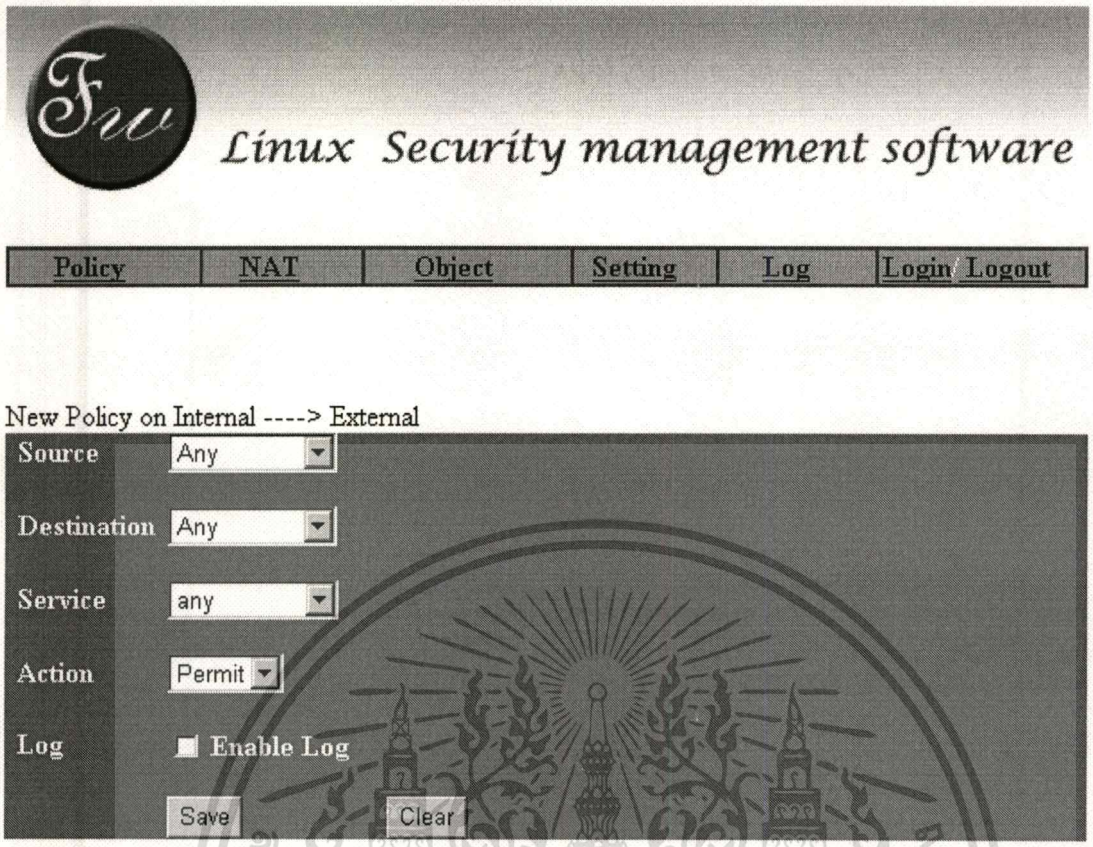
รูปที่ 3.23 หน้าจอแสดงการกำหนดค่าต่าง ๆ ของระบบ

จากรูป 3.23 เป็นหน้าจอที่จะแสดงไอพีของระบบและจะผู้ใช้จะต้องกำหนดค่าต่าง ๆ เช่น interface ที่จะเป็น internal , External หรือ DMZ โดยที่ผู้ใช้สามารถจะกำหนดไอพีได้เลือกที่ Setting ของแต่ละ Interface และในหน้าจอนี้ยังให้ผู้ใช้สามารถที่จะแก้ไขพาสเวิร์ดของ admin ได้



รูปที่ 3.24 หน้าจอเพื่อแก้ไข ไอพี ของ Interface

จากรูป 3.24 เป็นเมนูที่ใช้ในการให้ผู้ใช้แก้ไขไอพีของแต่ละ interface โดยที่จะมีผลกับ Server ทันที



รูปที่ 3.25 หน้าจอแสดงการสร้าง Policy ใหม่

จากรูป 3.25 เห็นหน้าจอที่ผู้ใช้จะเพิ่มกฎเข้าใหม่โดยจะมาจากการที่ผู้ใช้กด add ในแต่ละ interface ของระบบ และเมื่อมีการบันทึกแล้วระบบก็จะสร้างกฎให้กับ iptables ทันที

บทที่ 4

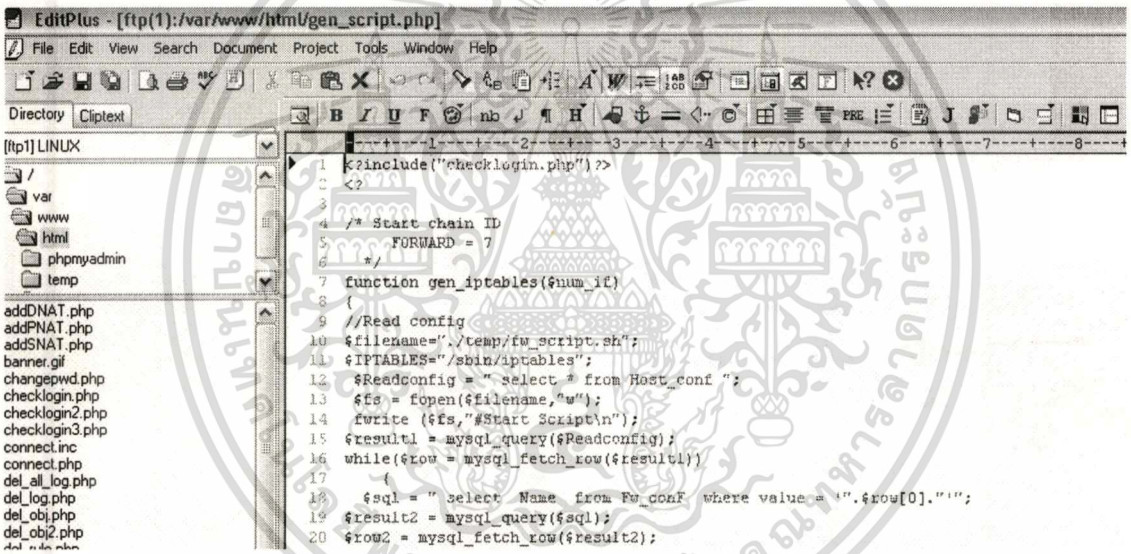
การพัฒนาระบบ

ระบบที่จะพัฒนาขึ้นนั้นมีจุดมุ่งหมายที่ทำงานบนระบบปฏิบัติการลินุกซ์ ซึ่งในครั้งนี้ได้ใช้ Redhat เวอร์ชั่น 8 เป็นระบบปฏิบัติการในการพัฒนาโปรแกรม

4.1 เครื่องมือที่ใช้ในการพัฒนา

1) Edit plus 2.0

เป็นโปรแกรมที่ช่วยในการเขียนโปรแกรม ซึ่งจะช่วยในการเขียนภาษา HTML,PHP ,Java script และ FTP ไปยัง Server



รูปที่ 4.1 ตัวอย่างหน้าจอโปรแกรม Edit plus

2) ไฟร์วอล IPtables เวอร์ชั่น v1.2.7a

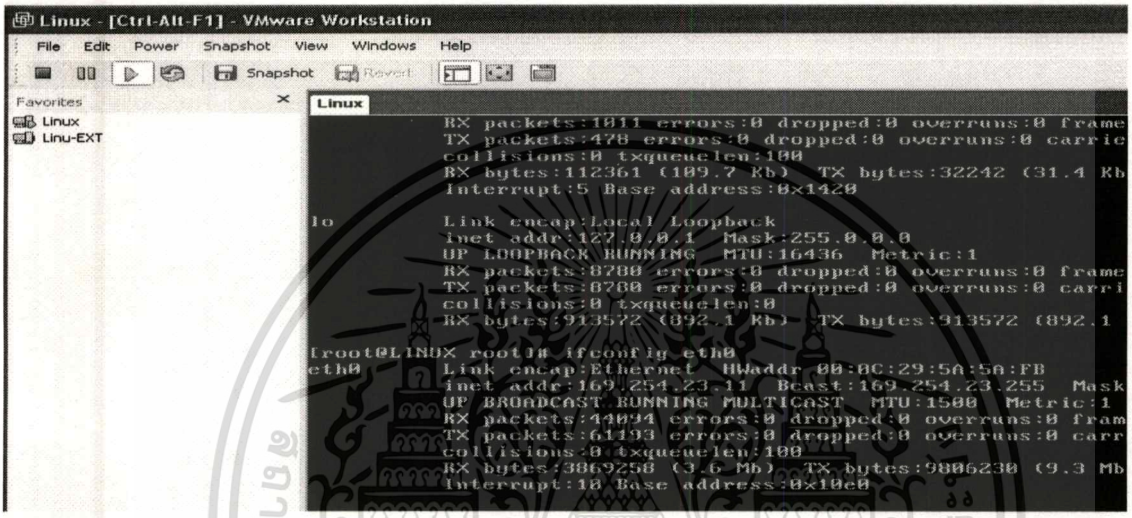
เป็นไฟร์วอลบนลินุกซ์ ซึ่งสามรถสร้างกฎ และ กรอง Packet ได้ส่วนรายละเอียดอื่นๆ ของ IPtables ได้อธิบายไว้แล้วในบทที่ 2

3)Mysql

เป็นโปรแกรมที่เก็บข้อมูลทั้งหมดของระบบ ซึ่งทำงานของอยู่บนเครื่องเดียวกัน

4)Vmware Version 4

เป็นโปรแกรมที่ใช้ในการจำลองเครื่อง PC ในเครื่องเดียวกัน โดยในการพัฒนาระบบ จะลง ลีนุกซ์เอาไว้พัฒนา และ ทดสอบ



รูปที่ 4.2 ตัวอย่างหน้าจอโปรแกรม vmware

4.2 ขั้นตอนในการพัฒนาระบบ

หลังจากการออกแบบระบบแล้ว ได้ทำการศึกษาส่วนต่าง ๆ ของระบบว่าจะทำงานร่วมกันได้อย่างไร อย่างเช่น การติดต่อระหว่าง PHP และ Mysql

ในการเขียนโปรแกรมนั้นได้เขียนฟังก์ชั่นหลักๆ ดังนี้

- ฟังก์ชั่นการเข้าสู่ระบบและการตรวจสอบของการเข้าระบบ

เป็นฟังก์ชั่นที่ทำการตรวจสอบผู้ใช้ที่จะเข้าสู่ระบบ โดยทำการสร้าง Session ของผู้ใช้แต่ละคนไว้ ซึ่งถ้ามีผู้ใช้คนใดมี session ไม่ถูกต้องก็จะถูกออกจากระบบไป

- ฟังก์ชั่นการเชื่อมต่อฐานข้อมูล

เป็นฟังก์ชั่นที่คอยที่จะเชื่อมต่อกับฐานข้อมูล Mysql ของระบบและ ส่งค่ากลับเป็นผลของการ query ต่าง ๆ

- ฟังก์ชั่นสร้างคำสั่งของ iptables

เป็นฟังก์ชั่นที่จะอ่านค่าจากฐานข้อมูลของระบบ และแปลงเป็น คำสั่งของ iptables โดยที่ประกอบด้วยสองส่วนหลัก คือการเพิ่มและลบกฎ ตัวอย่างเช่น

การเพิ่มกฎของระบบ โดยจะไปค่าต่าง ๆ มาจากฐานข้อมูล

โปรโตคอล : TCP

Interface ขาเข้า : eth0

Interface ขาออก : eth1

ไอพีต้นทาง : 169.254.23.0/255.255.255.0

ไอพีปลายทาง : 192.168.0.2/255.255.255.255

เบอร์พอร์ทปลายทาง : 22

เก็บ Log : เก็บ

ซึ่งจะได้เป็นคำสั่งของ iptables ได้ดังนี้

#/sbin/iptables -A FORWARD -p TCP -i eth0 -o eth1 -s 169.254.23.0/255.255.255.0 -d 192.168.0.2/255.255.255.255 --dport 22 -j WATCH

ส่วนถ้าเป็นการลบนั้น จะเป็นการลบโดยใช้ เบอร์ของกฎเป็นตัวลบ ซึ่งจะไปดึงมาจากฐานข้อมูล ซึ่งได้เก็บไว้ก่อนแล้ง ตัวอย่างเช่นลบกฎข้อ 5

#/sbin/iptables –delete –A FORWARD 5

ส่วนการสร้างกฎของ NAT นั้นแบ่งออกเป็น 3 แบบ โดยจะมีตัวอย่างการสร้างกฎดังนี้

1) Dynamic NAT ถ้ากำหนดค่ามาดังนี้

ไอพีต้นทาง: 169.254.23.0 / 255.255.255.0

Interface ขาออก : eth0

ซึ่งจะเป็นคำสั่งของ iptables ได้ดังนี้

#/sbin/iptables -t nat -A POSTROUTING -o eth0 -s 169.254.23.0/255.255.255.0 -j MASQUERADE

2) Destination Port NAT ถ้ากำหนดค่ามาดังนี้

โปรโตคอล : TCP

Interface ขาเข้า : eth0

ไอพีปลายทาง : 192.168.0.2/255.255.255.255

เบอร์พอร์ทปลายทาง : 22

เก็บ Log : เก็บ

ซึ่งจะเป็นคำสั่งของ iptables ได้ดังนี้

#/sbin/iptables -t nat -A PREROUTING -p TCP -i eth0 --dport 22 -j DNAT --to-destination 192.168.0.2

3) Static NAT ถ้ากำหนดค่ามาดังนี้

โปรโตคอล : TCP

Interface ขาเข้า : eth0

Interface ขาออก : eth1

ไอพีภายใน : 192.168.0.2/255.255.255.255

ไอพีภายนอก : 202.0.118.10/255.255.255.0

ซึ่งจะเป็นคำสั่งของ iptables ได้ดังนี้

#/sbin/iptables -t nat -t nat -A PREROUTING -p TCP -d 202.0.118.10 -j DNAT --to-destination 192.168.0.2

แต่ในกรณีนี้จะต้องสร้าง virtual interface เพิ่มโดยใช้คำสั่ง

ifconfig eth0:1 202.0.118.10 netmask 255.255.255.0

- ฟังก์ชั่นสร้างคำสั่งทั้งหมดของ iptables

เนื่องจาก iptables ไม่สามารถจำกฎไว้ได้เมื่อมีการ restart เครื่องจึงต้องสร้าง Script ไว้ด้วย เพื่อใช้ในการทำงานเมื่อมีการ restart โดยแบ่งออกเป็น 2 ไฟล์

1) fw_script_default.sh เป็นไฟล์ที่เก็บกฎเบื้องต้นของระบบ

```
#!/bin/sh
/sbin/iptables -F
/sbin/iptables -N CHECK_FLAGS
/sbin/iptables -F CHECK_FLAGS
```

```
/sbin/iptables -A CHECK_FLAGS -p tcp --tcp-flags ALL FIN,URG,PSH -m limit  --limit 5/minute -j
LOG --log-level cri  --log-prefix "NMAP-XMAS: "
/sbin/iptables -A CHECK_FLAGS -p tcp --tcp-flags ALL FIN,URG,PSH -j DROP
/sbin/iptables -A CHECK_FLAGS -p tcp --tcp-flags SYN,RST SYN,RST -m limit --limit 5/minute -j
LOG --log-level cri  --log-prefix "SYN/RST: "
/sbin/iptables -A CHECK_FLAGS -p tcp --tcp-flags SYN,RST SYN,RST -j DROP
/sbin/iptables -A CHECK_FLAGS -p tcp --tcp-flags SYN,FIN SYN,FIN -m limit  --limit 5/minute -j
LOG --log-level cri  --log-prefix "SYN/FIN: "
/sbin/iptables -A CHECK_FLAGS -p tcp --tcp-flags SYN,FIN SYN,FIN -j DROP
/sbin/iptables -N ALLOW_ICMP
/sbin/iptables -F ALLOW_ICMP
/sbin/iptables -A ALLOW_ICMP -p icmp -j LOG --log-level cri --log-prefix " ACCEPT "
/sbin/iptables -A ALLOW_ICMP -p icmp --icmp-type echo-reply -j ACCEPT
/sbin/iptables -A ALLOW_ICMP -p icmp --icmp-type destination-unreachable -j ACCEPT
/sbin/iptables -A ALLOW_ICMP -p icmp --icmp-type echo-request -j ACCEPT
/sbin/iptables -A ALLOW_ICMP -p icmp --icmp-type time-exceeded -j ACCEPT
/sbin/iptables -N WATCH
/sbin/iptables -A WATCH -m limit -j LOG --log-level cri --log-prefix " ACCEPT "
/sbin/iptables -A WATCH -j ACCEPT
/sbin/iptables -N WATCH_DROP
/sbin/iptables -A WATCH_DROP -m limit -j LOG --log-level cri --log-prefix " DROP "
/sbin/iptables -A WATCH_DROP -j DROP
/sbin/iptables -A INPUT -m state --state RELATED,ESTABLISHED -j ACCEPT
/sbin/iptables -A OUTPUT -m state --state RELATED,ESTABLISHED -j ACCEPT
/sbin/iptables -P FORWARD DROP
/sbin/iptables -A FORWARD -m state --state RELATED,ESTABLISHED -j ACCEPT
/sbin/iptables -A FORWARD -p tcp -j CHECK_FLAGS
/sbin/iptables -A FORWARD -p TCP --source-port 20 --destination-port 1024:65535 -j ACCEPT
```

โดยที่โซ่ต่าง ๆ ที่กำหนดมาจะมีจุดประสงค์ดังนี้

- CHECK_FLAGS เป็นโซ่ที่ใช้ในการป้องกันการสแกนพอร์ต การโจมตีแบบ sync flood
- ALLOW_ICMP เป็นโซ่ที่ใช้เมื่อมีการต้องการอนุญาตให้ใช้ ICMP
- WATCH เป็นโซ่ที่ใช้เมื่อเก็บ Log ของแต่ละกฎที่มีการอนุญาต
- WATCH_DROP เป็นโซ่ที่ใช้เมื่อเก็บ Log ของแต่ละกฎที่มีการไม่อนุญาต

2) fw_script.sh เป็นที่เก็บกฎต่าง ๆ ที่มีการเพิ่มเติม ซึ่งจะถูกสร้างใหม่ทุกครั้งที่มีการแก้ไขกฎต่าง ๆ ในระบบ ซึ่งจะเพิ่มเข้าไปที่ Forward Chain เท่านั้น

```
/sbin/iptables -A FORWARD -p TCP -i eth1  -o eth2  -s 192.168.0.0/255.255.255.0 -d
192.168.1.0/255.255.255.0 --dport 21 -j WATCH
```

4.3 การทดสอบการทำงานระบบ

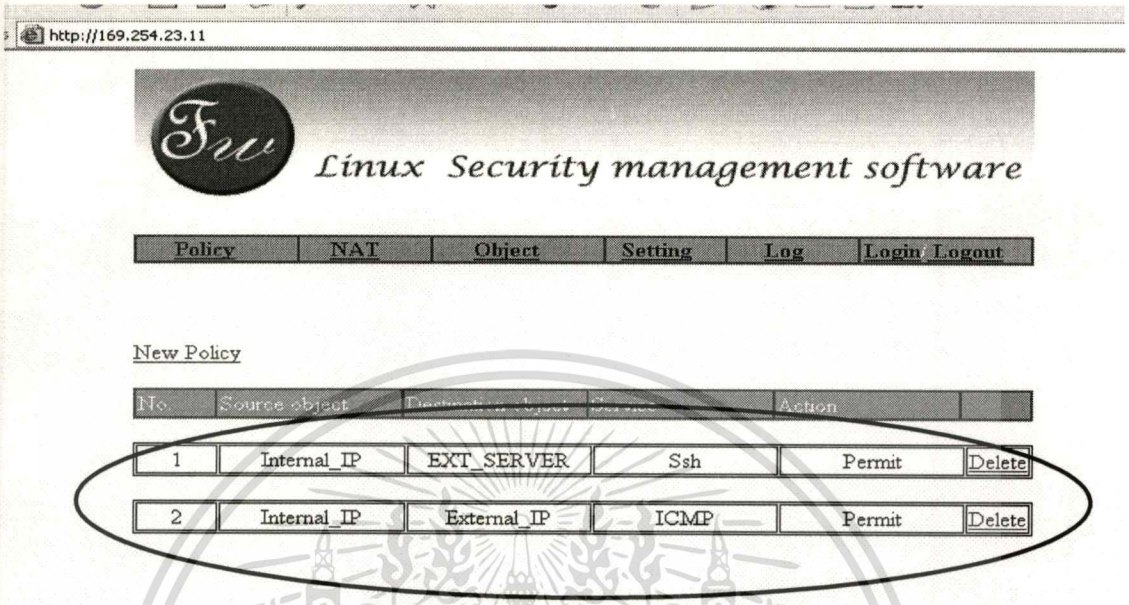
เนื่องจากการทดสอบนั้นต้องใช้เครื่อง อย่างน้อย 3 เครื่อง คือเครื่องหลักที่มีโปรแกรมอยู่ และเครื่อง ๆ ที่อยู่ตาม Interface ต่าง ๆ ซึ่งจะทดสอบว่าสามรถจะป้องกัน และ ทำงานได้ตามกฎที่สร้างได้ถูกต้อง

เครื่องที่ใช้ทดสอบมีรายละเอียดดังนี้

- CPU Intel Pentium 4 1.3 GHz
- Ram 256 MB
- OS Linux Redhat 8
- ไฟร์วอล IPTables
- การทำงานของ server จะอยู่บน vmware

4.3.1โดยมีขั้นตอนทดสอบดังนี้

1.ทดสอบการสร้างและลบกฎ และดูผลของกฎที่ iptables

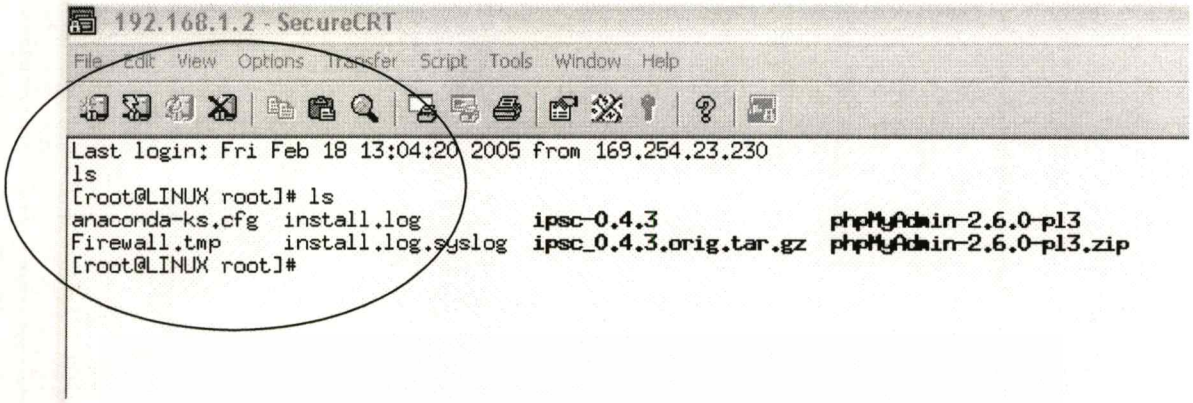


รูปที่ 4.3 ตัวอย่างการเพิ่มกฎเข้าระบบ

```
target     prot opt source               destination
ACCEPT     all  --  anywhere             anywhere            state RELATED,ESTABLISHED

Chain FORWARD (policy DROP)
target     prot opt source               destination
ACCEPT     all  --  anywhere             anywhere            state RELATED,ESTABLISHED
CHECK_FLAGS  tcp  --  anywhere             anywhere
ACCEPT     tcp  --  anywhere             anywhere            tcp spt:ftp-data dpts:1024:65535
WATCH      tcp  --  192.168.0.0/24       192.168.1.0/24      tcp dpt:ftp
WATCH      tcp  --  169.254.23.0/24      192.168.0.2         tcp dpt:ssh
ALLOW_ICMP  icmp --  169.254.23.0/24      192.168.0.0/24

Chain OUTPUT (policy ACCEPT)
target     prot opt source               destination
ACCEPT     all  --  anywhere             anywhere            state RELATED,ESTABLISHED

Chain ALLOW_ICMP (1 references)
```

รูปที่ 4.4 ตัวอย่างกฎที่เข้าไปที่ iptables แล้ว

2. ทดลองใช้งานตามกฎที่ได้เพิ่มเข้าไป โดยการ ทำ ssh ไปที่ server



รูปที่ 4.5 ตัวอย่างใช้งานจริงผ่านกฎที่เพิ่มไว้

4.3.2 สรุปผลการทดสอบ

- ระบบสามรถที่ทำงานได้ตามที่ต้องการ
- สามรถที่จะเพิ่มและลบกฎได้ตามปกติ โดยทำงานได้อย่างถูกต้อง
- กฎทุกกฎสามารถทำงานได้จริงตามที่ต้องการและ ออกแบบเอาไว้
- สามารถที่จะทำงานร่วมกับ iptables ได้ถูกต้อง
- การทำงานส่วนใหญ่ไม่มีปัญหา แต่อาจจะต้องเพิ่มความยืดหยุ่นของระบบให้มากขึ้น

บทที่ 5
บทสรุปและข้อเสนอแนะ

5.1 บทสรุป

ในโครงการนี้ได้ดำเนินการศึกษาข้อมูล และ ทฤษฎีของระบบความปลอดภัย ซึ่งเป็นโปรแกรมที่ใช้งานค่อยข้างยุ่งยาก จึงนำเอาเทคโลยีที่เป็นเว็บมาช่วยทำให้การดูแลและการกำนดนโยบายเป็นไปได้ง่าย โดยจะไม่จำเป็นจะต้องเข้าใจการทำงานมากนัก

ในช่วงของการวิเคราะห์และออกบบระบบนั้น ได้ทำการศึกษาการใช้งาน iptable และ การสร้าง scipt เพื่อที่จะสร้างนโยบายของ iptables จากนั้นจึงทำการออกแบบ และพัฒนาระบบ Linux Scurity Firewall Software ซึ่งในระหว่างการพัฒนาระบบ ได้มีการทดสอบ ปรับปรุงโครงสร้างและฟังก์ชันการทำงานของระบบ จนสามารถใช้งานได้ตามต้องการ

5.2 ข้อดีและข้อเสียของระบบ

5.2.1 ข้อดีของระบบ

- การทำงานของระบบจะทำให้ผู้ดูแลง่ายที่จะดูแล Firewall ได้
- ไม่จำเป็นจะต้องไปซื้อ software ที่มีราคาแพงในการที่จะดูแล
- โดยที่จะเพิ่มหรือแก้ไขนโยบายได้ง่าย
- ผู้ใช้ไม่จำเป็นจะต้องมาศึกษา iptable

5.2.2 ข้อเสียของระบบ

- ระบบไม่อาจจะครอบคลุม function ทั้งหมดของไฟล์วอลได้
- ไม่รองรับการแก้ไข policy จะต้องลบก่อนเท่านั้น
- ไม่รองรับ interface card มากกว่าสามใบ
- ยังไม่มีระบบแจ้งเตือนไปยังผู้ดูแลระบบ

5.3 ข้อเสนอแนะ

ระบบงานในการพัฒนาครั้งนี้ ถึงแม้จะเป็นระบบที่จะช่วยในการดูแล iptable เพื่อที่จะทำให้ผู้ใช้งานทำงานง่ายขึ้น แต่ยังคงมีจุดปรับปรุงอีก ดังนี้

1. ควรเพิ่มความสามารถในการรักษาความปลอดภัยระหว่างผู้ใช้กับระบบ
2.ต่อไปอยากให้สามารถรองรับฟังก์ชั่นทั้งหมดของ iptable ได้
3. ควรจะเพิ่มความสามารถของการทำรายงานมากขึ้น
4. การรองรับ interface cardที่มากกว่าสามใบ

บรรณานุกรม

กิตติ ภักดีวัฒนะกุล. 2546. คัมภีร์ PHP. เคทีพี คอมพ์ แอนด์ คอนซัลท์.

ภูวดล ด่านระหาญ. 2544. **Linux** 2.4 Stateful Firewall : IPTABLES. [Online]. Available: http://www.thaicert.nectec.or.th/paper/firewall/iptables.php.

ภูวดล ด่านระหาญ. 2544. การสร้าง **rule** สำหรับไฟร์วอลล์. [Online]. Available: http://www.thaicert.nectec.or.th/paper/firewall/fwrulebase.php.

ภูวดล ด่านระหาญ. 2544. การตรวจสอบไฟร์วอลล์ **(Firewall Auditing)**. [Online]. Available: http://www.thaicert.nectec.or.th/paper/firewall/fwaudit.php.

สงกรานต์ ทองสว่าง. 2544. My SQL ระบบฐานข้อมูลสำหรับอินเตอร์เน็ต.บริษัท ซีเอ็ดยูเคชั่น จำกัด (มหาชน).

ภาคผนวก

ภาคผนวก ก
การติดตั้งระบบและการใช้งาน

ความต้องการของระบบ

สำหรับเครื่องที่สามารถติดตั้งได้นั้นมีรายละเอียดดังนี้

- ระบบปฏิบัติการลินุกซ์ Redhat หรือ อื่น ๆ
- มีไฟร์วอล IPtables
- มีโปรแกรม Apache ,PHP ,Mysql

การติดตั้ง

สำหรับการติดตั้งนั้น ต้องติดตั้งบนเครื่องเดียวกับ Snort และ firewall ซึ่งไฟล์ที่ใช้ติดตั้งจะอยู่ในรูปแบบบีบอัดไว้แล้ว ไฟล์คือ fw.tar

- ขยายไฟล์ของโปรแกรมไปยังที่จะให้ Apache ทำงาน

tar –xvf fw.tar

- คัดลอกไฟล์ S96FWservice ไปไว้ที่ /etc/rc3.d

cd /etc/rc3.d

chmod u+x S96FWservice

คราวนี้ก็จะได้ตัวที่ start service แล้ว หลังจากนั้นให้ restart ระบบดู

- สร้างฐานข้อมูล

ใช้ sql ไฟล์ในโฟลเดอร์ย่อย sql ชื่อcr_db.sql ใน mysql client ก็จะสร้างฐานข้อมูลของระบบไว้

- เริ่มต้นระบบใหม่ของ Server ก็จะทำงานได้

การใช้งานของระบบ

เข้าไปใช้งานโดยผ่านทางเว็บเพจ ก็จะใช้งานระบบได้ทันที โดยใช้ user =admin ,password = admin ซึ่งเป็นค่าที่กำหนดไว้แล้ว เมื่อเข้าสู่ระบบควรจะเปลี่ยน password ก่อน

ประวัติผู้เขียน

ชื่อผู้เขียน	นายวิชิตฤกษ์ บริบูรณ์
วันเกิด	5 สิงหาคม 2521
สถานที่เกิด	นราธิวาส
วุฒิการศึกษาระดับปริญญาตรี	วท.บ. (วิทยาการคอมพิวเตอร์) คณะวิทยาศาสตร์ มหาวิทยาลัยสงขลานครินทร์ วิทยาเขตหาดใหญ่
การทำงาน	Network System Engineer บริษัท เทเลเมติกส์ จำกัด