การป้องกันเว็บโดยใช้ไอพีเทเบิลส์

Web prevention using iptables

นายนพสรณ์ เกษจันทร์

นายศุภกร ชะอุ่มดี

ปริญญานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิศวกรรมศาสตรบัณฑิต

ภาควิชาวิศวกรรมคอมพิวเตอร์

คณะวิศวกรรมศาสตร์

สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

ปีการศึกษา 2546

การป้องกันเว็บโดยใช้ไอพีเทเบิลส์

Web prevention using iptables

โดย

นายนพสรณ์ เกษจันทร์

นายศุภกร ชะอุ่มดี

อาจารย์ที่ปรึกษา

อาจารย์ธนา หงษ์สุวรรณ

อาจารย์อัครเดช วัชระภูพงษ์

อาจารย์ธนัญชัย ตรีภาค

ปริญญานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิศวกรรมศาสตรบัณฑิต

ภาควิชาวิศวกรรมคอมพิวเตอร์

คณะวิศวกรรมศาสตร์

สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

ปีการศึกษา 2546

ปริญญานิพนธ์ ปีการศึกษา 2546

ภาควิชาวิศวกรรมคอมพิวเตอร์

คณะวิศวกรรมศาสตร์ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

เรื่อง การป้องกันเว็บโดยใช้ไอพีเทเบิลส์

Web Prevention Using iptables

คณะผู้จัดทำ นายนพสรณ์ เกษจันทร์ รหัสประจำตัว 44015332

นายศุภกร ชะอุ่มดี รหัสประจำตัว 44015350

.....อาจารย์ที่ปรึกษา

(อาจารย์ธนา หงษ์สุวรรณ)

.....อาจารย์ที่ปรึกษา

(อาจารย์อัครเดช วัชระภูพงษ์)

.....อาจารย์ที่ปรึกษา

(อาจารย์ธนัญชัย ตรีภาค)

การป้องกันเว็บโดยใช้ไอพีเทเบิลส์

นาย นพสรณ์ เกษจันทร์ 44015332
นาย ศุภกร ชะอุ่มดี 44015350
อ. ธนา หงษ์สุวรรณ อาจารย์ที่ปรึกษา
อ. อัครเดช วัชระภูพงษ์ อาจารย์ที่ปรึกษา
อ.ธนัญชัย ตรีภาค อาจารย์ที่ปรึกษา
ปีการศึกษา 2546

บทคัดย่อ

การรักษาความปลอดภัยในระบบเครือข่ายคอมพิวเตอร์ในปัจจุบันนั้น องค์กรทุกๆ องค์กรนั้นจะใช้ตัวไฟร์วอลในการรักษาระบบความปลอดภัย ซึ่งเรียกได้ว่าระบบเครือข่ายนั้นจะขาดไฟร์วอลเสียไม่ได้

โดยปกตินั้นไฟร์วอลที่เป็นแบบสเตทฟูลจะไม่สามารถที่จะป้องกันการโจมตีในระดับชั้นแอพพลิเคชั่นได้ เช่น การโจมตีด้วยการส่งยูนิโค้ดเข้าไปยังตัว IIS เซิร์ฟเวอร์ และอื่นๆ ที่อยู่ในระดับชั้นแอพพลิเคชั่นได้

ซึ่งกลุ่มผู้วิจัยเล็งเห็นว่ามีความจำเป็นที่จะต้องทำให้ตัวไฟร์วอลนั้นมีความสามารถที่จะป้องกันในส่วนนี้ได้ จึงได้สนใจพัฒนาตัวไฟร์วอล โดยเน้นไปที่ เน็ทฟิลเตอร์/ไอพีเทเบิล ซึ่งการพัฒนานี้เป็นการเพิ่มประสิทธิภาพให้กับตัวไอพีเทเบิลให้สามารถป้องกันการโจมตีในระดับชั้นแอพพลิเคชั่นโดยเอาแนวคิดของการทำงานของ ระบบตรวจจับผู้บุกรุก (IDS) มาเป็นตัวช่วยตรวจสอบ ซึ่งจะเอากฎการตรวจจับผู้บุกรุกมาเป็นกฎที่ใช้ในการตรวจสอบแพ็คเก็ตที่เข้าออกในระบบเครือข่าย

Web Prevention Using iptables

Mr. Noppasorn Kadchan
Mr. Supagorn Chaoumdee
Mr. Thana Hongsuwan Advisor
Mr. Akkradach Watcharapupong Advisor
Mr. Thanunchai Threepak Advisor
Academic Year 2003

ABSTRACT

Computer Network System, in the present. The organizations are used firewall that it can be protected network system. So maybe called the network system it without firewall

Normally, firewall was be stateful inspection that can't protected, attacking in application layer such as UNICODE attacking into IIS server and so on.

So that, my research team focus this inpoint improved firewall that proteced from them. Which we interested in netfilter/iptables so this devalopment was improved on iptables that can proteced from them in application layer. By used conceptual of Intrusion detection System (IDS) that its can improved. So the rules of instrusion detection can got from it that used in detected incoming and outgoing packet which it

กิตติกรรมประกาศ

ปริญญานิพนธ์ฉบับนี้สำเร็จลุล่วงได้ด้วยดี ก็เนื่องมาจากการให้โอกาส การดูแล ให้คำแนะนำต่างๆ การสนับสนุน การให้คำสั่งสอนและให้คำปรึกษาเป็นอย่างดีเสมอมาจาก จากอาจารย์ ธนา หงษ์สุวรรณ อาจารย์ อัครเดช วัชระภูพงษ์ และอาจารย์ ธนัญชัย ตรีภาค ซึ่งต้องขอขอบพระคุณอาจารย์ทั้ง 3 ท่านเป็นอย่างสูง

ขอขอบคุณภาควิชาวิศวกรรมคอมพิวเตอร์ และสถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบังที่ได้จัดเตรียมสิ่งอำนวยความสะดวก เพื่อให้งานวิจัยดำเนินไปได้อย่างสะดวกและรวดเร็วไม่ว่าจะเป็นห้องวิจัยที่สะดวกสบาย อินเตอร์เน็ตความเร็วสูงสำหรับการค้นคว้าหาข้อมูลและการทดลองต่างๆเพื่อเป็นความรู้และประสบการณ์ที่ดี

ขอขอบคุณสมาชิกห้องวิจัยไอแซก (ISAG) ทุกคนไม่ว่าจะเป็นรุ่นพี่ รุ่นน้อง และเพื่อนๆร่วมห้องทุกคนที่คอยสร้างความสนุกสนานครื้นเครงช่วยสร้างบรรยากาศที่ดีและเป็นกันเองระหว่างรุ่นพี่ รุ่นน้องและเพื่อนๆ ช่วยคลายความเคร่งเครียดจากการเรียนและการทำวิจัย ทั้งยังคอยให้กำลังใจกันเสมอมา และต้องขอบคุณห้องวิจัยไอแซก (ISAG) ที่เป็นที่พักผ่อนในยามอ่อนล้า เพื่อให้มีแรงกายในการทำงานต่อไป

และท้ายที่สุดต้องขอขอบพระคุณบุคคลที่สำคัญที่สุดในชีวิตของข้าพเจ้าที่ทำให้ข้าพเจ้ามีทุกวันนี้ คือ บิดา มารดา และบุคคลทุกคนในครอบครัวของข้าพเจ้า อันเป็นที่เคารพรัก คอยอุ้มชูเลี้ยงดู อบรมสั่งสอนข้าพเจ้ามาเป็นอย่างดี ทั้งยังให้ความรักความห่วงใย และกำลังใจที่ดีเสมอมา ข้าพเจ้าต้องขอขอบพระคุณมา ณ ที่นี้ด้วย

นพสรณ์ เกษจันทร์
ศุภกร ชะอุ่มดี

สารบัญ

สารบัญ (ต่อ)

สารบัญ (ต่อ)

สารบัญ (ต่อ)

สารบัญภาพประกอบ

สารบัญตาราง

บทที่ 1
บทนำ

1.1 ความสำคัญและที่มา

เนื่องจากในปัจจุบันการใช้งานอินเตอร์เน็ตมีการใช้งานกันอย่างแพร่หลายและมีการใช้งานอินเตอร์เน็ตก็มีอยู่หลายอย่างไม่ว่าจะเป็นการเปิดเว็บไซด์ การสนทนาผ่านทางอินเตอร์เน็ต การดูหนังฟังเพลง แต่จะเห็นได้ว่าการใช้งานที่มากที่สุดนั่นก็คือการเปิดเว็บไซด์ ซึ่งการเปิดเว็บไซด์นี้จะต้องมีการเชื่อมต่อไปยังผู้ที่ให้บริการเว็บไซด์ต่างๆซึ่งเราเรียกว่าเว็บเซิร์ฟเวอร์ นั่นแสดงให้เห็นว่าเว็บเซิร์ฟเวอร์เป็นส่วนสำคัญในการใช้งานอินเตอร์เน็ตเพื่อใช้ในการเปิดเว็บ

สิ่งที่ต้องคำนึงถึงในโลกของอินเตอร์เน็ตในปัจจุบันอย่างหนึ่งคือเรื่องของความปลอดภัยทางเครือข่ายคอมพิวเตอร์ เนื่องจากเว็บเซิร์ฟเวอร์ทุกๆแห่งจะต้องมีการป้องกันการเข้ามาบุกรุกเว็บเซิร์ฟเวอร์จากผู้ที่ไม่หวังดี เพราะถ้าหากเว็บเซิร์ฟเวอร์โดนบุกรุกจากผู้ไม่หวังดีแล้วก็อาจจะทำให้เกิดความเสียหายให้แก่ข้อมูลต่างๆของเซิร์ฟเวอร์และอาจจะทำให้เว็บเซิร์ฟเวอร์ไม่สามารถให้บริการกับเครื่องคอมพิวเตอร์เครื่องอื่นที่เข้ามาขอใช้บริการในการเปิดดูเว็บไซด์ได้

เมื่อพูดถึงเรื่องความปลอดภัยทางเครือข่ายสิ่งที่เป็นเครื่องมือที่ถูกกล่าวถึงเป็นสิ่งแรกๆเสมอคือไฟร์วอล และปัจจุบันไฟร์วอลก็แทบจะกลายเป็นเครื่องมือที่จำเป็นและขาดไม่ได้สำหรับเครือข่ายคอมพิวเตอร์ที่เชื่อมต่อกับอินเตอร์เน็ตเสียแล้ว แต่ไฟร์วอลโดยส่วนใหญ่ป้องกันการโจมตีหรือการบุกรุกได้เพียงระดับชั้นเน็ตเวิร์คเท่านั้นแต่ไม่มีความสามารถในการป้องกันการโจมตีหรือการบุกรุกในระดับแอพพลิเคชั่นได้ และเนื่องจากรูปแบบการโจมตีทางเว็บเซิร์ฟเวอร์นั้นโดยส่วนใหญ่จะอยู่ในระดับแอพพลิเคชั่น ทำให้เว็บเซิร์ฟเวอร์ยังไม่มีความปลอดภัยเพียงพอต่อการป้องกันการโจมตีทางเว็บเซิร์ฟเวอร์ได้

จากเหตุผลข้างต้นจึงได้พัฒนาไฟร์วอลให้มีความสามารถในการป้องกันการโจมตีในระดับแอพพลิเคชั่น โดยเลือกที่จะพัฒนาโปรแกรมบนระบบปฏิบัติการลีนุกซ์เนื่องจากโดยส่วนใหญ่ในปัจจุบันมีการใช้ระบบปฏิบัติการลีนุกซ์มาใช้เป็นเครื่องไฟร์วอลและเลือกที่จะพัฒนาโปรแกรมไอพีเทเบิลส์ เนื่องจากโปรแกรมไอพีเทเบิลส์ เป็นโปรแกรมไฟร์วอลพื้นฐานที่มีอยู่ในระบบปฏิบัติการลีนุกซ์อยู่แล้วแต่ยังไม่มีความสามารถในการป้องกันการโจมตีในระดับแอพพลิเคชั่น

1.2 วัตถุประสงค์ของโครงงาน

1. เพื่อศึกษาระบบการรักษาความปลอดภัยด้านเครือข่ายคอมพิวเตอร์
2. เพื่อศึกษาหลักการทำงานและโครงสร้างของไฟร์วอล
3. เพื่อศึกษาหลักการทำงานและโครงสร้างของไอพีเทเบิลส์ ซึ่งเป็นโปรแกรมไฟร์วอลบนระบบปฏิบัติการลีนุกซ์

4. เพื่อพัฒนาโมดูลของไอพีเทเบิลส์ เพื่อให้โปรแกรมไอพีเทเบิลส์ มีความสามารถในการป้องกันการโจมตีทางเว็บเซิร์ฟเวอร์เพิ่มมากยิ่งขึ้น
5. เพื่อเพิ่มระบบความปลอดภัยให้กับเว็บเซิร์ฟเวอร์

1.3 ขอบเขตของโครงงาน

โครงงานนี้คือการพัฒนาไอพีเทเบิลส์ โดยพัฒนาในส่วนที่เป็นโมดูลของไอพีเทเบิลส์ที่สามารถนำเอา โมดูลที่พัฒนาขึ้นมานี้ไปใช้งานกับ ไอพีเทเบิลส์ เพื่อเพิ่มประสิทธิภาพของ ไอพีเทเบิลส์ให้สามารถป้องกันการโจมตีทางเว็บเซิร์ฟเวอร์ได้โดยการพัฒนาโมดูลนี้จะแบ่งออกเป็น 2 ส่วนหลักๆคือ

- แมชท์โมดูล คือโมดูลที่ทำหน้าที่ในการตรวจสอบเนื้อข้อมูลของแพ็กเก็ตว่าเป็นข้อมูลที่ปลอดภัยกับเว็บเซิร์ฟเวอร์หรือไม่โดยโมดูลนี้จะทำหน้าที่ในการตรวจสอบเนื้อข้อมูลของแพ็กเก็ตแล้วนำไปทำการเปรียบเทียบว่าเนื้อข้อมูลนั้นตรงกับรูปแบบ (Pattern) ที่เราห้ามไว้หรือไม่ โดยในส่วนของการเปรียบเทียบเนื้อข้อมูลของแพ็กเก็ตกับรูปแบบที่ตั้งไว้นั้นจะใช้หลักการวิเคราะห์ของโบเยอร์-มัวร์ (Boyer-moore) ในการเปรียบเทียบ
- ทาร์เก็ต โมดูล คือโมดูลที่ทำหน้าที่ในการจัดการกับแพ็กเก็ตที่มีเนื้อข้อมูลที่มีลักษณะเป็นการโจมตีเว็บเซิร์ฟเวอร์ โดยโมดูลนี้จะทำการครอปแพ็กเก็ตที่มีเนื้อข้อมูลมีลักษณะเป็นการโจมตีเว็บเซิร์ฟเวอร์ทิ้งไปแล้วทำการส่งทีซีพีรีเซ็ทแพ็กเก็ตกลับไปยังผู้ที่ส่งแพ็กเก็ต นั้นมาเพื่อเป็นการยกเลิกการเชื่อมต่อและบันทึกลงล็อกไฟล์

1.4 ขั้นตอนการดำเนินงาน

1. ศึกษารายละเอียดเกี่ยวกับโปรโตคอลทีซีพี/ไอพี
2. ศึกษาโครงสร้างและการทำงานของไฟร์วอล
3. ศึกษาโครงสร้างและการทำงานของ ไอพีเทเบิลส์
4. ศึกษาหลักการทำงานของระบบการตรวจจับผู้บุกรุกทางเครือข่าย
5. ศึกษาการเขียนเคอร์เนลโมดูล
6. ศึกษาหลักการวิเคราะห์ของโบเยอร์-มัวร์(Boyer-Moore)
7. พัฒนาโมดูลของ ไอพีเทเบิลส์ ที่สามารถตรวจสอบเนื้อข้อมูลของแพ็กเก็ต และโมดูลของไอพีเทเบิลส์ ที่สามารถครอป แพ็กเก็ตและส่งทีซีพีรีเซ็ทแพ็กเก็ตตอบกลับได้
8. ทดสอบการทำงานของโมดูลที่พัฒนาขึ้นและทำการแก้ไขข้อผิดพลาดต่างๆ

บทที่ 2
โพรโตคอลทีซีพี/ไอพี

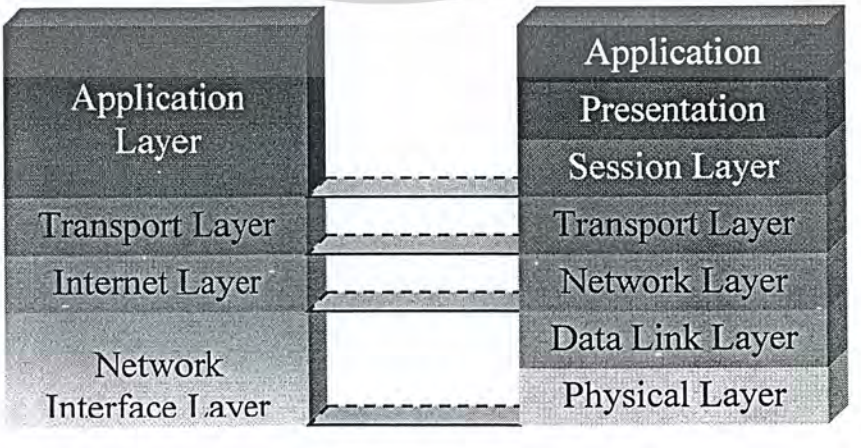
2.1 ความเป็นมาของโพรโตคอลทีซีพี/ไอพี

ทีซีพี/ไอพี เป็นโพรโตคอลมาตรฐานที่ใช้กันอยู่ในระบบปฏิบัติการแบบยูนิกซ์ เริ่มพัฒนาโดยกระทรวงกลาโหมของสหรัฐฯ ในปี ค.ศ. 1969 เพื่อเชื่อมโยงเครื่องคอมพิวเตอร์ทางทหารของแต่ละหน่วยที่อยู่ห่างไกลกัน โดยมีจุดประสงค์ คือสร้างระบบเครือข่ายให้เครื่องคอมพิวเตอร์สามารถรับส่งข้อมูลกันได้ แม้ว่าสายส่งข้อมูลบางส่วนจะถูกทำลายเสียหายไปก็ตาม เพื่อใช้งานในยามเกิดสงคราม โดยเครือข่ายที่จัดตั้งในระยะแรกชื่อว่า Advanced Research Projects Agency Network หรือ อาร์พาเน็ต (ARPANET)

ต่อมาได้พัฒนาเป็นเครือข่ายอินเทอร์เน็ต (INTERNET) โพรโตคอลนี้เหมาะสำหรับเชื่อมต่อคอมพิวเตอร์ทั้งใกล้ และไกลเข้าด้วยกัน และมีมาตรฐานรองรับทำให้ผู้ผลิตฮาร์ดแวร์ และซอฟต์แวร์สามารถสร้างอุปกรณ์ และโปรแกรมที่จะรองรับการทำงานของโพรโตคอลนี้ ทำให้เครื่องคอมพิวเตอร์สามารถรับส่งข้อมูลกันได้ไม่ว่าจะเป็นเครื่องขนาดเล็กหรือขนาดใหญ่ หรือใช้ระบบปฏิบัติการอะไรก็ตาม ทีซีพี/ไอพี (TCP/IP) เป็นชุดโพรโตคอลที่ประกอบด้วยโพรโตคอลต่างๆ หลายโพรโตคอล แต่ละโพรโตคอลมีคุณลักษณะ และมีความสามารถในการทำงานแตกต่างกัน โดยที่ในบทนี้ได้กล่าวถึงรายละเอียดและคุณสมบัติของโพรโตคอลที่สำคัญบางโพรโตคอล

2.2 การเชื่อมต่อของโพรโตคอลทีซีพี/ไอพี (TCP/IP Linking)

ทีซีพี/ไอพี (TCP/IP หรือ Transmission Control Protocol/Internet Protocol) เป็นโพรโตคอลในการสื่อสารในระบบอินเทอร์เน็ต และอินทราเน็ต มีหน้าที่ตรวจสอบการรับส่งข้อมูลระหว่างเครื่องคอมพิวเตอร์ของฝ่ายรับ และฝ่ายส่งให้ได้รับข้อมูลที่ถูกต้องครบถ้วน หากข้อมูลที่ส่งมาเกิดการสูญหายระหว่างทางจะมีการแจ้งให้ต้นทางส่งข้อมูลมาใหม่ การทำงานของทีซีพี/ไอพีสามารถเปรียบเทียบกับโมเดลอ้างอิงโอเอสไอ (Open System Interconnection Reference Model: OSI) ตามมาตรฐานไอเอสโอ (International Organization for Standardization: ISO) ได้ดังรูปที่ 2-1

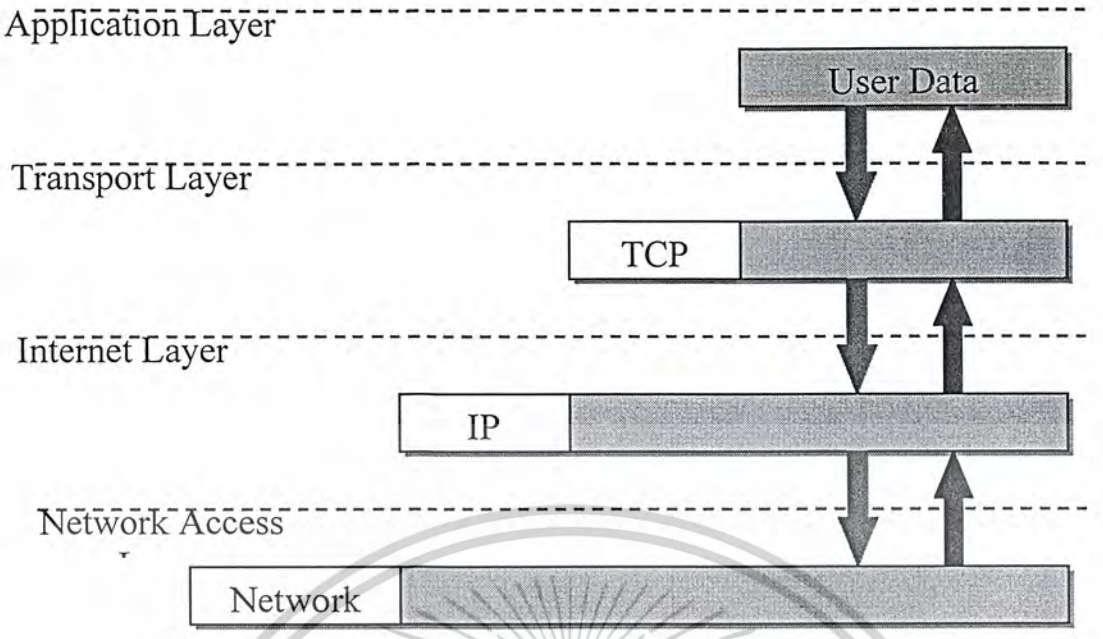


รูปที่ 2-1 แสดงการเปรียบเทียบเลเยอร์ของโอเอสไอกับเลเยอร์ของทีซีพี/ไอพี

ในแต่ละระดับชั้นของทีซีพี/ไอพีมีการทำงานที่แตกต่างกัน ตั้งแต่การติดต่อกับแอพพลิเคชันจนกระทั่งแปลงเป็นสัญญาณส่งไปตามสายสัญญาณ ซึ่งการทำงานในแต่ละระดับชั้นของทีซีพี/ไอพี มีดังตารางที่ 2-1

ชื่อระดับชั้น	หน้าที่
1. ชั้นแอพพลิเคชัน (Application Layer)	รองรับการทำงานของแอพพลิเคชันต่างๆ ที่ทำงานเป็นโพรเซสอยู่ในเครื่องต้นทางและปลายทาง โดยจัดการเชื่อมต่อระหว่างโพรเซส หรือแอพพลิเคชันที่อยู่ต่างเครื่องกัน โดยการทำงานของแอพพลิเคชันต่างๆมีการติดต่อกันตามแต่ละโพรโตคอลเฉพาะแล้วแต่แอพพลิเคชันที่ใช้งาน ซึ่งจะขอบริการจากชั้นทรานสปอร์ตอีกทีหนึ่ง
2. ชั้นทรานสปอร์ต (Transport Layer)	สร้างการเชื่อมต่อกันระหว่างแอพพลิเคชันแบบ end-to-end โดยจุดที่เชื่อมต่อกันเพื่อรับส่งข้อมูลนี้เรียกว่า พอร์ต (port) หรือซ็อกเก็ต (Socket) ในชั้นนี้มีบริการหลักอยู่ 2 แบบ คือ Connection Oriented โดยเรียกผ่านโพรโตคอลทีซีพี (TCP: Transmission Control Protocol) และ Connectionless ซึ่งเรียกผ่านโพรโตคอลยูดีพี (UDP: User Datagram Protocol) ซึ่งกล่าวถึงในหัวข้อถัดไป
3. ชั้นอินเทอร์เน็ต (Internet Layer)	ส่งผ่านข้อมูลระหว่างเครือข่าย โดยมีโพรโตคอลที่ทำงานเป็นกลไกสำคัญในการส่งผ่านข้อมูลไปยังเครือข่ายใดๆ ในอินเทอร์เน็ต คือ ไอพี (Internet Protocol: IP) ซึ่งกล่าวถึงในหัวข้อถัดไป นอกจากนี้ในชั้นนี้ยังมีโพรโตคอลทำงานอยู่ด้วยอีก 2 ชนิด คือ ไอซีเอ็มพี (Internet Control Message Protocol: ICMP) และเออาร์พี (Address Resolution Protocol: ARP)
4. ชั้นเน็ตเวิร์กอินเตอร์เฟส (Network Interface Layer)	แปลงข้อมูลให้อยู่ในรูปที่เหมาะสมกับเครือข่ายแต่ละแบบ ซึ่งแตกต่างกันออกไป และแปลงเป็นสัญญาณไฟฟ้าส่งไปยังเครือข่าย

ตารางที่ 2-1 การทำงานของแต่ละระดับชั้นของทีซีพี/ไอพี

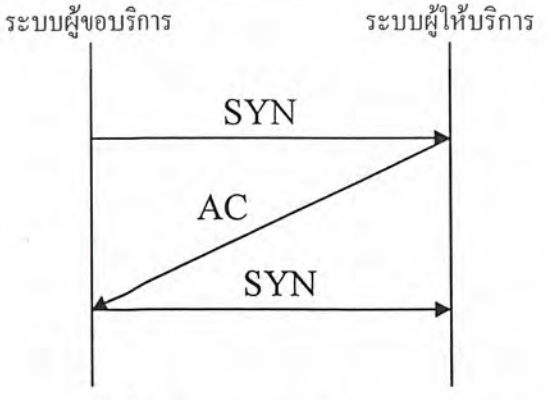


รูปที่ 2-2 แสดงการข้อมูลที่ส่งผ่านในโมเดลของทีซีพี/ไอพี

ในชุดโพรโตคอลทีซีพี/ไอพีนี้ มีโพรโตคอลหลัก ที่ขอกล่าวถึง 3 โพรโตคอล ได้แก่ โพรโตคอลทีซีพี โพรโตคอลยูดีพี ซึ่งทำงานในชั้นทรานสปอร์ต และโพรโตคอลไอพี ซึ่งทำงานในชั้นอินเทอร์เน็ต โดยมีรายละเอียดดังต่อไปนี้

2.3 โพรโตคอลทีซีพี (TCP: Transmission Control Protocol)

การทำงานที่สำคัญอย่างหนึ่งของโพรโตคอลทีซีพี คือ การทำ "3-Way Handshake" ซึ่งเป็นกระบวนการเริ่มต้นในการสร้างการเชื่อมต่อในชั้นทรานสปอร์ต กล่าวคือ ในการติดต่อกันระหว่างระบบในเครือข่ายต้องมีการสร้างการเชื่อมต่อไปยังระบบที่ให้บริการก่อน โดยผู้ขอบริการส่งสัญญาณ SYN เพื่อขอบริการ จากนั้นผู้ให้บริการจะส่งสัญญาณ ACK เพื่อตอบรับการเชื่อมต่อที่ร้องขอมา จึงสามารถรับส่งข้อมูลกันได้ ดังรูปที่ 2-3



รูปที่ 2-3 แสดงการทำ 3-Way Handshake

การเชื่อมต่อแบบ 3-Way Handshake นี้ เป็นการตรวจสอบความพร้อมของทั้งฝ่ายส่ง และฝ่ายรับ และการกำหนดค่าเริ่มต้นของพารามิเตอร์ต่างๆ ของทั้งสองฝ่ายให้ตรงกัน หลังจากกระบวนการทำ 3-Way Handshake สิ้นสุด ทั้งสองฝ่ายจึงสามารถรับ และส่งข้อมูลซึ่งกัน และกันได้

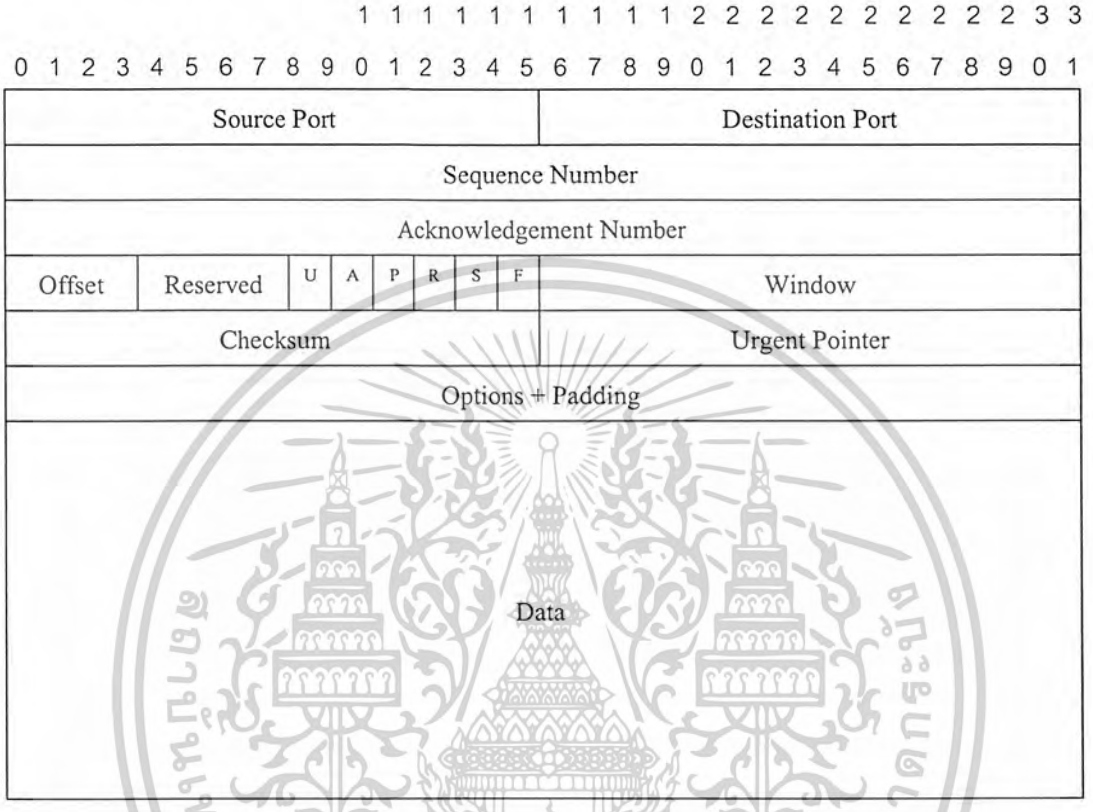
ดังนั้นโพรโตคอลทีซีพีจึงเป็นโพรโตคอลที่มีการรับส่งข้อมูลแบบ "Connection Oriented" ทำให้การทำงานของทีซีพีมีความน่าเชื่อถือมากขึ้น หน้าที่การทำงานของทีซีพีในการรับส่งข้อมูลมีหน้าที่หลัก 6 ข้อ คือ

1. ควบคุมการรับส่งข้อมูล (Basic Data Transfer)
2. ความน่าเชื่อถือในการรับส่งข้อมูล (Reliability)
3. ควบคุมการไหลของข้อมูล (Flow Control)
4. การทำมัลติเพล็ก (Multiplexing)
5. ควบคุมการเชื่อมต่อ (Connection)
6. ความปลอดภัยในการรับส่งข้อมูล (Security)

<u>ส่วนประกอบของทีซีพีเฮดเดอร์</u>

1. *Source Port* : เป็นหมายเลขพอร์ตของบริการที่เครื่องต้นทาง
2. *Destination Port* : เป็นหมายเลขพอร์ตของบริการเครื่องปลายทาง
3. *Sequence Number* : เป็นหมายเลขที่บอกลำดับของการรับส่งข้อมูลของเครื่องที่ต้องการขอส่งข้อมูล
4. *Acknowledgement Number* : เป็นหมายเลขที่บอกลำดับของการรับส่งข้อมูลที่ฝั่งรับข้อมูล ปกติ ค่าของ Acknowledgement Number มีค่าเท่ากับ Sequence Number (ของอีกฝั่งหนึ่ง) + 1 เสมอ
5. *Data Offset* : เป็นตัวบอกค่าออฟเซตของข้อมูล เพราะทีซีพีนั้นไม่มีการกำหนดความยาวที่แน่นอนของข้อมูล จึงต้องมีออฟเซตเป็นตัวบอก
6. *Flag* : เป็นบิตที่บอกชนิดของข้อมูล ได้แก่
   - URG : Urgent Pointer Field Significant - แสดง Urgent Pointer
   - ACK : Acknowledgement Field Significant – แสดงการ Acknowledgement
   - PSH : Push Function
   - RST : Reset The Connection - แสดงเมื่อรีเซ็ทการเชื่อมต่อ
   - SYN : Synchronize Sequence Number - หมายเลขแพ็กเก็ตที่ส่งแบบซิงโครนัส
   - FIN : No more data from sender - แสดงว่าไม่มีข้อมูลที่ส่งจากผู้ส่งแล้ว
7. *Window* : เป็นเลขบอกจำนวนของอ็อกเต็ต (octet) ของข้อมูล จัดการในส่วนของ end-to-end flow control
8. *Checksum* : เป็นส่วนที่ตรวจสอบความถูกต้องของข้อมูล
9. *Urgent Pointer* : เป็นตัวชี้ตำแหน่งของ Urgent Data

10. *Option and Padding* : เป็นตัวบอกออปชันของโพรเซสที่ใช้ทีซีพี
11. *Data* : เนื้อข้อมูลที่ต้องการสื่อสาร มีขนาดได้ไม่ต่ำกว่า 5 32-บิตเวิร์ด (6 บิตแรกสงวนไว้และกำหนดให้เป็นศูนย์)



รูปที่ 2-4 แสดงแพ็กเก็ตทีซีพี

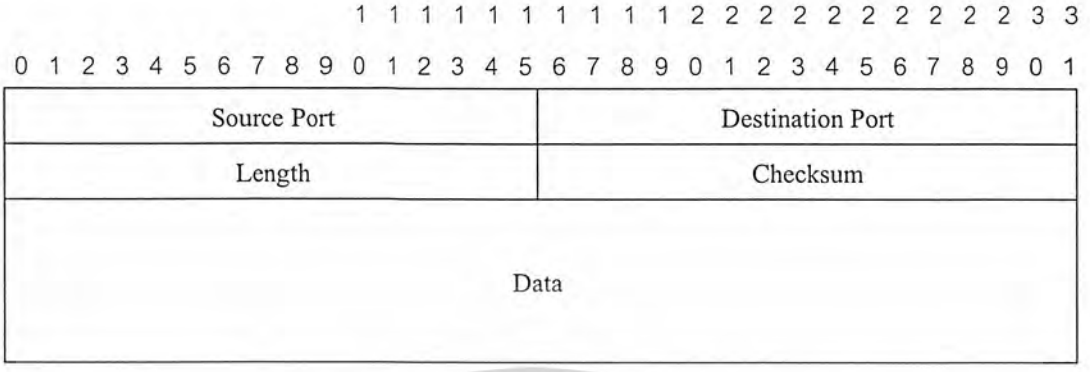
2.4 โพรโตคอลยูดีพี (UDP: User Datagram Protocol)

โพรโตคอลยูดีพีเป็นโพรโตคอลในการติดต่อสื่อสารในชั้นทรานสปอร์ต (Transport Layer) การทำงานคล้ายกับทีซีพีมาก คือจัดการเกี่ยวกับการสื่อสารระหว่างเครื่อง แต่เป็นแบบ Connectionless คือทั้งฝ่ายส่ง และฝ่ายรับไม่จำเป็นต้องอาศัยการสร้างช่องทางเชื่อมต่อกัน โดยไม่ต้องมีการแจ้งให้ฝ่ายรับข้อมูลเตรียมรับข้อมูลเหมือนโพรโตคอลทีซีพี และไม่มีการส่งสัญญาณตรวจสอบว่าข้อมูลถึงเครื่องปลายทางอย่างถูกต้องครบถ้วนในการส่งข้อมูลแต่ละครั้ง จึงไม่มีการส่งข้อมูลใหม่อีกในกรณีที่เกิดความผิดพลาดของการส่งข้อมูล

<u>ส่วนประกอบของ UDP Frame</u>

1. *Source Port* : เป็นค่าตัวเลข 16 บิต บอกพอร์ตของบริการที่เครื่องต้นทาง
2. *Destination Port* : เป็นค่าตัวเลข 16 บิต บอกพอร์ตของบริการที่เครื่องปลายทาง
3. *Length* : เป็นค่าตัวเลข 16 บิต บอกความยาวของข้อมูล

4. *Checksum* : เป็นค่าตัวเลข 16 บิต ตรวจสอบความถูกต้องของข้อมูลที่ส่ง

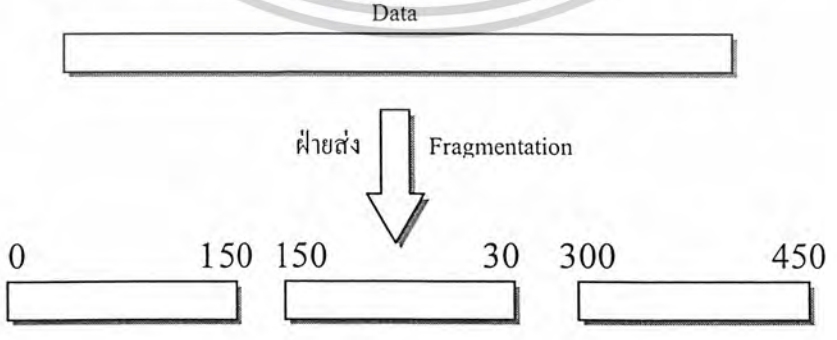


รูปที่ 2-5 แสดงแพ็กเก็ตยูดีพี

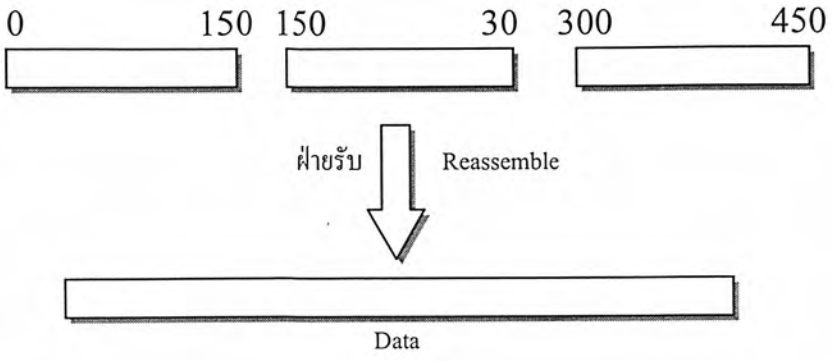
2.5 โพรโตคอลไอพี (IP: Internet Protocol)

โพรโตคอลไอพีเป็นโพรโตคอลที่จัดการเกี่ยวกับแอดเดรสของแต่ละแพ็กเก็ต เพื่อให้ส่งแพ็กเก็ตต่างๆ ไปยังเป้าหมายได้ถูกต้อง การทำงานของไอพีเป็นเพียงการส่งข้อมูลไปยังเครื่องเป้าหมายเท่านั้น ไม่มีการส่งสัญญาณขอบริการ หรือสัญญาณให้บริการระหว่างกันเหมือนทีซีพี เรียกว่าการเชื่อมต่อแบบ Connectionless ซึ่งระบบทั้งสองตั้งสมมติฐานว่าการเชื่อมต่อระหว่างกันไม่มีความผิดพลาดเกิดขึ้นแน่

เนื่องจากมาตรฐานในเครือข่ายมีหลากหลาย ขนาดของแพ็กเก็ตในแต่ละมาตรฐานจึงมีความแตกต่างกันออกไป ทำให้การส่งข้อมูลระหว่างอุปกรณ์ในเครือข่ายนั้นอาจมีการแบ่งข้อมูลออกเป็นแพ็กเก็ตย่อยๆ ในระหว่างการส่ง เรียกว่า การทำแฟร็กเมนเตชัน (Fragmentation) เช่น แพ็กเก็ตของ FDDI มีขนาด 4,500 ไบต์ หากเครื่องปลายอยู่ในเครือข่าย Ethernet ซึ่งมีขนาดของแพ็กเก็ตสูงสุดเพียง 1,500 ไบต์ ดังนั้นการส่งแพ็กเก็ตไปยังเครื่องปลายทางจึงต้องมีการแบ่งเป็นแพ็กเก็ตย่อย และเมื่อแพ็กเก็ตย่อยมาถึงเครื่องเป้าหมายก็จะมารวมกันเป็นแพ็กเก็ตเดิมที่มีขนาด 4,500 ไบต์อีกครั้ง เรียกการรวมกันนี้ว่า การรีแอสเซมเบิ้ล (Reassemble) ซึ่งทำให้ได้ข้อมูลเหมือนที่ส่งมาจากเครื่องต้นทาง



รูปที่ 2-6 แสดงการทำแฟร็กเมนเตชัน



รูปที่ 2-7 แสดงการรีแอสเซมเบิ้ล

ส่วนประกอบของแพ็กเก็ตไอพี

1. *version* : เป็นค่าตัวเลข 4 บิต บอกเวอร์ชันของมาตรฐานไอพีที่ใช้ โดยปกติมีค่าเป็น 4 ซึ่งหมายถึง IPv4
2. *Internet Header Length (IHL)* : เป็นตัวบอกความยาวเฮดเดอร์ของไอพี
3. *Type of Service* : เป็นส่วนที่บอกการทำงานของแพ็กเก็ตที่ส่งว่าทำหน้าที่อะไร มีทั้งหมด 8 บิต โดย

   Bit 0-2 : บอกรายละเอียดการทำงานของแพ็กเก็ตนั้นๆ

   111 - Network Control

   110 - Internetwork Control

   101 - CRITIC / ECP

   100 - Flash Override

   011 - Flash

   010 - Immediate

   001 - Priority

   000 - Routine

   Bit 3 : บอกถึงลักษณะของดีเลย์

   0 = Normal Delay - มีดีเลย์ปกติ

   1 = Low Delay - มีดีเลย์ต่ำ

   Bit 4 : บอกถึงประเภทของทรูพุต

   0 = Normal Throughput – มีทรูพุตปกติ

   1= High Throughput – มีทรูพุตสูง

   Bit 5 : บอกถึงประเภทของความน่าเชื่อถือ

0 = Normal Reliability - มีความน่าเชื่อถือพอประมาณ

1 = High Reliability - มีความน่าเชื่อถือสูง

Bit 6-7 : กันไว้ใช้ในอนาคต

4. *Total Length* : มีขนาด 16 บิต บอกถึงความยาวในดาต้าแกรมของไอพี
5. *Identification field* : เป็นตัวเลข 16 บิต เป็นค่าประจำตัวของไอพีนั้น โดยโฮสต์ที่ส่งเป็นผู้กำหนด และเพิ่มค่าขึ้นหนึ่งเมื่อมีการส่งดาต้าแกรมของไอพีใหม่ ซึ่งใช้ในการประกอบกลับ
6. *Flag* : เป็นตัวเลข 3 bit บอกลักษณะของแพ็กเก็ตว่ามีการแฟร็กเมนต์หรือไม่

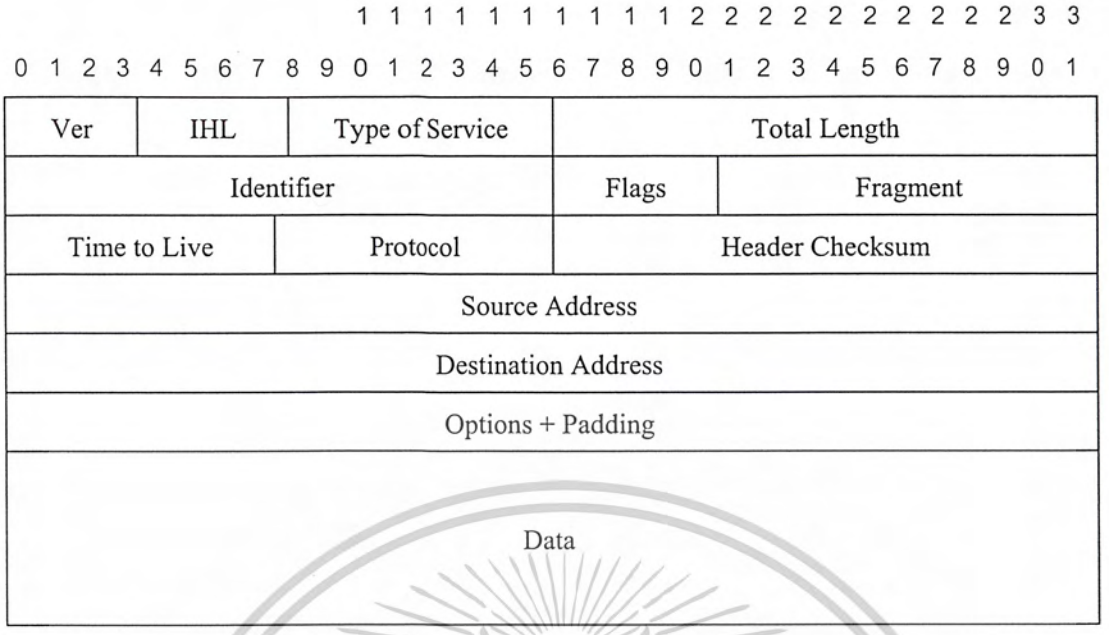
   Bit 0 : สงวนไว้ ปกติเป็น 0

   Bit 1 : 0 = บอกว่าแพ็กเก็ตมีการแตกแพ็กเก็ตย่อย

   1 = บอกว่าแพ็กเก็ตไม่มีการแตกแพ็กเก็ตย่อย

   Bit 2 : 0 = บอกว่าแพ็กเก็ตนั้นเป็นแพ็กเก็ตสุดท้ายที่ได้จากการแตกแพ็กเก็ตย่อย

   1 = บอกว่าแพ็กเก็ตนั้นยังไม่ใช่แพ็กเก็ตสุดท้ายที่ได้จากการแตกแพ็กเก็ตย่อย
7. *Fragment Offset* : เป็นค่าตัวเลข 13 บิต บอกออฟเซตของแฟร็กเมนต์เมื่อเทียบในดาต้าแกรม
8. *Time To Live (TTL)* : เป็นตัวเลข 8 บิต บอกช่วงเวลาของแพ็กเก็ตที่ยังอยู่ในเครือข่ายได้ โดยกำหนดค่าเป็นจำนวนเราท์เตอร์สูงสุดที่ดาต้าแกรมผ่านได้ ซึ่งโดยทั่วไปที่ค่าระหว่าง 32 ถึง 64 และลดค่าลงเรื่อยๆ เมื่อผ่านเราเตอร์ เพื่อเป็นการป้องกันแพ็กเก็ตล้นเครือข่าย
9. *Protocol* : เป็นตัวเลข 8 bit บอกถึงโพรโตคอลที่อยู่เหนือขึ้นไป ว่าเป็นโพรโตคอลระดับสูงกว่าประเภทใด
10. *Header Checksum* : เป็นค่าตัวเลข 32 บิต ใช้ตรวจสอบความถูกต้องของเฮดเดอร์
11. *Source Address*_: เป็นค่าตัวเลข 32 บิต บอกถึงไอพีแอดเดรสของเครื่องต้นทาง
12. *Destination Address* : เป็นค่าตัวเลข 32 บิต บอกถึงไอพีแอดเดรสของเครื่องปลายทาง



รูปที่ 2-8 แสดงแพ็กเก็ตไอพี

บทที่ 3
ไฟร์วอล

3.1 คุณสมบัติทั่วไปของไฟร์วอล

ไฟร์วอลเป็นเครื่องมือรักษาความปลอดภัยที่ทำงานในเชิงป้องกัน (Protect) ซึ่งทำหน้าที่ในการควบคุมการเข้าถึงเน็ตเวิร์ค (Access Control) โดยอาศัยกฎเป็นพื้นฐาน (Rule based) สำหรับคุณสมบัติแต่ละอย่างของไฟร์วอลนั้นมีรายละเอียดดังนี้

1. การป้องกัน (Protect) : ไฟร์วอลเป็นเครื่องมือที่ใช้ทำงานในเชิงป้องกัน โดยแพ็กเก็ตที่สามารถผ่านเข้า-ออกเน็ตเวิร์คได้นั้น จะต้องเป็นแพ็กเก็ตที่ไฟร์วอลเห็นว่ามีความปลอดภัย แพ็กเก็ตที่ไฟร์วอลเห็นว่าไม่ปลอดภัยหรืออาจจะนำมาซึ่งความไม่ปลอดภัยก็จะถูกกำจัด คือ ทิ้งไปเสียเฉยๆไม่ส่งต่อ โดยการที่ไฟร์วอลจะตัดสินใจว่าแพ็กเก็ตใดปลอดภัยและแพ็กเก็ตใดไม่ปลอดภัยนั้นจะอยู่บนพื้นฐานของกฎที่ผู้ดูแลไฟร์วอล (Firewall Administrator) เป็นผู้กำหนดไว้ล่วงหน้า ซึ่งเงื่อนไขของกฎเหล่านี้เองทำให้ไฟร์วอลสามารถป้องกันแพ็กเก็ตที่อาจจะส่งผลร้ายไม่ให้ผ่านเข้าไปถึงเครือข่ายคอมพิวเตอร์ได้
2. ควบคุมการแอคเซส (Access Control): "แอคเซส" หมายถึงการที่โฮสต์ใดโฮสต์หนึ่งสามารถสื่อสารข้อมูลที่ต้องการไปยังโฮสต์ปลายทางได้สำเร็จ การแอคเซสในแต่ละระดับจะมีวิธีการแตกต่างกันออกไป ทำให้การควบคุมการแอคเซสสำหรับแต่ละระดับแตกต่างกันไปด้วย ไฟร์วอลจึงมีการทำงานหลายลักษณะตามวิธีที่ไฟร์วอลใช้ควบคุมการแอคเซส
3. กฎพื้นฐาน (Rule Based) : ไฟร์วอลจะควบคุมการแอคเซสโดยอาศัยการเปรียบเทียบคุณสมบัติของแพ็กเก็ตที่จะผ่านไฟร์วอลกับกฎของการแอคเซสที่ได้กำหนดไว้ หากพบว่าไม่มีกฎที่ห้ามไว้ก็จะอนุญาตให้แพ็กเก็ตนั้นผ่านไปได้ หากมีกฎที่ห้ามไว้แพ็กเก็ตนั้นก็จะถูกสกัดกั้นไว้ด้วยวิธีใดวิธีหนึ่ง

3.2 ประเภทของไฟร์วอล

ไฟร์วอลสามารถจำแนกประเภทจากลักษณะการทำงานได้ดังนี้

3.2.1 แพ็กเก็ตฟิลเตอริงไฟร์วอล / สกรีนิงเราเตอร์ (Packet Filtering Firewall / Screening Router)

เป็นไฟร์วอลพื้นฐานที่มีความสามารถในการควบคุมแทรฟฟิกโดยอาศัยการตรวจสอบข้อมูลที่ปรากฏอยู่ในแพ็กเก็ต ไฟร์วอลประเภทนี้อาจจะเป็นความสามารถที่เพิ่มเติมมาในเราเตอร์ โดยอาศัยโครงสร้างพื้นฐานที่เราเตอร์มีอยู่ให้ทำหน้าที่มากกว่าการเราต์ (Route คือการจัดเส้นทาง) ให้แพ็กเก็ตไปตามทิศทางที่เหมาะสมเพียงอย่างเดียว แต่จะทำการตรวจสอบเปรียบเทียบกับเงื่อนไขที่กำหนดไว้ก่อนจึงจะทำการเราต์แพ็กเก็ตออกไป

ก่อนที่จะรู้จักการทำงานของแพ็กเก็ตฟิลเตอริงไฟร์วอลนั้น จะขอกล่าวถึงองค์ประกอบของแพ็กเก็ตเสียก่อน

แพ็กเก็ตเป็นหน่วยพื้นฐานของการรับส่งข้อมูลของเน็ตเวิร์คเลเยอร์ การรับส่งข้อมูลแต่ละครั้งของเน็ตเวิร์คเลเยอร์จะส่งข้อมูลออกไปชุดหนึ่ง โดยที่ความยาวของข้อมูลนี้จะมีค่าเท่าใดนั้นจะเป็นไปตามคุณสมบัติของเน็ตเวิร์คเลเยอร์นั้นๆ ข้อมูลแต่ชุดนั้นเรียกว่าแพ็กเก็ต สำหรับโปรโตคอล ทีซีพี/ไอพี นั้นจะใช้ ไอพีเป็นโปรโตคอลหลักในการขนส่งข้อมูลระหว่างโฮสต์โดยไอพีซึ่งอยู่ในอินเตอร์เน็ตเลเยอร์จะส่งข้อมูลลงไปยังเน็ตเวิร์คเลเยอร์ตามลำดับ โดยที่หากขนาดของดาต้าแกรม (ไอพีดาต้าแกรม) ที่จะส่งนั้นสามารถส่งไปได้โดยใช้แพ็กเก็ตเดียว ไอพีก็จะส่งดาต้าแกรมนั้นไปในทันที และแพ็กเก็ตนั้นก็คือข้อมูลของไอพี 1 ดาต้าแกรม แต่หากขนาดของไอพีดาต้าแกรมใหญ่กว่าขนาดของเน็ตเวิร์คเลเยอร์แล้วไอพีก็จะต้องทำการแบ่งส่วนหรือ แฟร็กเมนเตชัน (Fragmentation) คือกระจายดาต้าแกรมออกเป็นส่วนย่อยเสียก่อนแล้วจึงค่อยส่งลงไปที่เน็ตเวิร์คเลเยอร์ ซึ่งในกรณีนี้ ข้อมูล 1 แพ็กเก็ตจะเป็นเพียงส่วนย่อยหรือแฟร็กเมนต์หนึ่งของดาต้าแกรมเท่านั้น ดังนั้นข้อมูล 1 แพ็กเก็ตจึงไม่จำเป็นต้องเป็นข้อมูล 1 ดาต้าแกรมเสมอไป แต่อย่างไรก็ตามแพ็กเก็ตทุกแพ็กเก็ตที่ส่งมาจากไอพีจะมีข้อมูลอย่างน้อยที่สุดคือ ไอพีแอดเดรสต้นทางและปลายทางเสมอ ซึ่งจำเป็นสำหรับให้แพ็กเก็ตนั้นวิ่งออกไปจนถึงที่หมายปลายทางได้

หากข้อมูล 1 แพ็กเก็ตนั้นบรรจุครบถ้วนทั้ง ไอพีดาต้าแกรมแล้ว ก็จะทำให้สามารถทราบถึงข้อมูลของโปรโตคอลเลเยอร์ที่สูงขึ้นไปด้วยว่าเป็น ไอซีเอ็มพี, ทีซีพี, ยูดีพี หรือโปรโตคอลอื่นใดที่อาศัยอยู่ใน ไอพีดาต้าแกรม นั้น แต่หากแพ็กเก็ตนั้นไม่สามารถบรรจุข้อมูลได้ครบทั้งดาต้าแกรมแล้ว ก็จะทำให้เพียงแต่ทราบว่าแพ็กเก็ตนั้นเป็นไอพีแพ็กเก็ตเท่านั้น

3.2.1.1 ข้อมูลที่สำคัญของแพ็กเก็ต

ภายในแพ็กเก็ตแต่ละแพ็กเก็ตนั้นจะประกอบด้วยข้อมูลที่สำคัญซึ่งสามารถนำมาใช้เพื่อเป็นเงื่อนไขสำหรับการควบคุมแทรฟฟิกโดยไฟร์วอลดังนี้

1. ไอพีแอดเดรส ต้นทาง : ไอพีแอดเดรสของต้นทาง เพื่อใช้ในการพิจารณาต้นทางของข้อมูลว่าอยู่ในเงื่อนไขที่อนุญาตหรือไม่
2. ไอพีแอดเดรสปลายทาง : ไอพีแอดเดรส ของปลายทางเพื่อใช้ในการพิจารณาปลายทางของข้อมูลว่าอยู่ในเงื่อนไขที่อนุญาตหรือไม่
3. โปรโตคอล : ระบุโปรโตคอลที่อาศัยอยู่ในไอพีดาต้าแกรม ที่กำลังพิจารณานี้
4. พอร์ตต้นทาง : ระบุพอร์ตต้นทางสำหรับโปรโตคอลที่ใช้พอร์ตคือ ทีซีพี และ ยูดีพี ซึ่งข้อมูลพอร์ตต้นทางนี้ส่วนใหญ่มักจะมีความสำคัญในลำดับรองลงไป และไม่ถูกนำมาใช้ควบคุมแทรฟฟิกมากนัก
5. พอร์ตปลายทาง : ระบุพอร์ตปลายทางที่แพ็กเก็ตนี้ต้องการติดต่อด้วยสำหรับโปรโตคอลที่ใช้พอร์ตเช่น ทีซีพี และ ยูดีพี
6. ข้อมูลสำคัญอื่นๆตามลักษณะของโปรโตคอลเช่น ทีซีพี แฟลก, ไอซีเอ็มพีเมสเซสจ์ เป็นต้น

ข้อมูลทั้ง 6 ส่วนนี้จะมีได้อย่างครบถ้วนสมบูรณ์ก็ต่อเมื่อแพ็กเก็ตนั้นมีข้อมูลครบถ้วนทั้งหมดของไอพีดาต้าแกรมหากข้อมูลแพ็กเก็ตนั้นเป็นแฟรกเมนต์ อาจจะทำให้ข้อมูลในส่วนที่ 3 เป็นต้นไปซึ่งอยู่ในโปรโตคอลที่เลเยอร์สูงกว่าไอพีไม่สมบูรณ์ อย่างไรก็ตามไฟร์วอลส่วนใหญ่ทำการติดตั้งใช้งานใน

เครือข่าย ซึ่งมีขนาดของแพ็กเก็ตที่ใหญ่พอสำหรับรองรับไอพีดาต้าแกรมได้ทั้งหมด จึงมักไม่ค่อยมีปัญหาแต่อย่างใดยกเว้นมีการแฟร็กเมนต์โดยความประสงค์ของไอพีเอง

จากข้อมูลที่สำคัญของแพ็กเก็ตข้างต้นนี้ จะสามารถนำมาใช้เป็นเงื่อนไขสำหรับควบคุมการผ่านเข้าออกของข้อมูลได้ โดยการพิจารณาข้อมูลทั้งหมดให้เป็นไปตามกฎที่ระบุไว้ ซึ่งเรียกว่าแอคเซสรูล (Access Rules) หรือกฎของการควบคุมการผ่านเข้าออกของแพ็กเก็ต โดยทั่วไปรูปแบบของการแอคเซสรูลเบื้องต้นจะเป็นดังนี้

Source Adress	Destination Address	Protocol	Services(Dest.Port)	Action

โดยที่ข้อมูลทั้งหมดจะเป็นเสมือนตัวแปรที่จะนำมาเปรียบเทียบกับค่าที่ได้ระบุไว้ในแอคเซสรูลทีละค่า เงื่อนไขในการเปรียบเทียบของแต่ละตัวแปรจะเป็นตรรกะ “และ” ส่วนข้อมูลฟิลด์สุดท้ายหมายถึงสิ่งที่ไฟร์วอลกระทำเมื่อค่าในแพ็กเก็ตนั้นตรงกับเงื่อนไข

ตัวอย่างเช่น

Source Adress	Destination Address	Protocol	Services(Dest.Port)	Action
161.246.5.14	ANY	TCP	80	Accept

นั่นหมายถึงแอคเซสรูลนี้ได้ระบุไว้ว่าจะอนุญาตให้แพ็กเก็ตที่มีต้นทาง ไอพีแอดเดรส 161.246.5.14 และปลายทางใดๆ และใช้โปรโตคอล ทีซีพี และหมายเลขพอร์ตปลายทางเท่ากับ 80 ผ่านไฟร์วอลไปได้ หากไฟร์วอลมีแอคเซสรูลนี้เพียงข้อเดียว ก็เท่ากับอนุญาตให้โฮสต์เพียงโฮสต์เดียวคือโฮสต์ที่มี ไอพีแอดเดรส 161.246.5.14 เท่านั้นที่สามารถใช้บริการเอชทีทีพี (ทีซีพี พอร์ต 80) ไปยังโฮสต์อื่นที่อยู่อีกฟากหนึ่งของไฟร์วอลได้

นี่เป็นเพียงหลักการพื้นฐานในการควบคุมแทรฟฟิกของแพ็กเก็ตฟิลเตอร์ริ่งไฟร์วอลและไฟร์วอลชนิดนี้โดยทั่วไปจะเรียกว่าสกรีนิงเราเตอร์ (Screening Router) เพราะว่าเป็นการนำเอาเราเตอร์ทั่วไปที่มีความสามารถกำหนดแอคเซสรูลได้มาดัดแปลงใช้ในการควบคุมแทรฟฟิก ซึ่งการกำหนดแอคเซสรูลของแทรฟฟิกทำได้โดยพิจารณาจากข้อมูลจากข้อมูลของแต่ละแพ็กเก็ต แต่เนื่องจากเราเตอร์เป็นอุปกรณ์ที่มีพื้นฐานจากการทำงานในอินเตอร์เน็ตเลเยอร์ ทำหน้าที่เราต์แพ็กเก็ตเป็นหลักโดยพิจารณาจากไอพีแอดเดรส และจะทำการเราต์ไปทีละแพ็กเก็ต ดังนั้นจึงสามารถควบคุมแทรฟฟิกได้ดีในระดับไอพี คือดูจากไอพีแอดเดรส ทั้งต้นทางและปลายทางเท่านั้น สำหรับข้อมูลในส่วนของโปรโตคอลในเลเยอร์สูงขึ้นไป เช่น ทีซีพี, ยูดีพี, ไอซีเอ็มพี นั้น เนื่องจากเราเตอร์มีขีดจำกัดในการรับรู้ข้อมูลในเลเยอร์บนถัดขึ้นไปคือ ทรานสปอร์ตเลเยอร์จึงทำให้สามารถควบคุมแทรฟฟิกโดยระบุเงื่อนไขของโปรโตคอลในทรานส

ปอร์ตได้อย่างจำกัด คือจะสามารถควบคุมแทรฟฟิกได้เฉพาะเมื่อข้อมูลในทรานสปอร์ตเลเยอร์นั้นจะสามารถบรรจุได้ในแพ็กเก็ตเดียว หากมีการแฟร็กเมนต์และต้องเชื่อมโยงกันระหว่างหลายแพ็กเก็ตแล้วเราเตอร์จะไม่สามารถรับรู้การเชื่อมโยงนั้นได้

3.2.1.2 ข้อดีของแพ็กเก็ตฟิลเตอร์ริง

1. ราคาถูกเพราะเป็นคุณสมบัติที่มักมีในเราเตอร์อยู่แล้ว อาศัยเพียงการกำหนดแอคเซสรูลที่เหมาะสมเท่านั้น หากยังไม่มีไฟร์วอลอยู่เลย ก็สามารถใช้เพื่อช่วยป้องกันเน็ตเวิร์กภายในได้ดีพอสมควรในระดับหนึ่ง
2. หากเน็ตเวิร์กภายในไม่ใหญ่มาก และมีการใช้งานอินเตอร์เน็ตอย่างจำกัด ก็สามารถใช้ทดแทนไฟร์วอลได้ทันที
3. การใช้สกรีนนิ่งเราเตอร์ควบคู่กับไฟร์วอลจะเป็นการแบ่งเบาภาระของไฟร์วอลได้มาก หากทำการกำหนดแอคเซสรูลได้อย่างสอดคล้องกันแล้ว จะทำให้มีการป้องกันที่เข้มแข็ง
4. การป้องกันบางประเภทไม่สามารถป้องกันได้โดยไฟร์วอล จะต้องทำโดยการกำหนดที่เราเตอร์เท่านั้น

3.2.1.3 ข้อเสียของแพ็กเก็ตฟิลเตอร์ริง

1. การกำหนดแอคเซสรูลทำได้ยาก ไม่มีระบบยูสเซอร์อินเตอร์เฟสเพื่อช่วยในการทำงาน ส่วนใหญ่ก็จะใช้วิธีเทลเน็ตเข้าไปยังเราเตอร์ แล้วป้อนคำสั่งในลักษณะของคอมมานด์ไลน์เข้าไปโดยตรงที่เราเตอร์ ทำให้โอกาสที่จะกำหนดผิดพลาดเนื่องจากการป้อนข้อมูลผิดรูปแบบ (Syntax) เป็นไปได้สูง
2. คำสั่งในการทำงานจะผูกติดกับยี่ห้อของเราเตอร์ ไม่มีมาตรฐานของคำสั่ง หากเปลี่ยนยี่ห้อของเราเตอร์ก็จะต้องศึกษารูปแบบของคำสั่งใหม่
3. ไม่สามารถกำหนดกฎที่ซับซ้อนได้ เนื่องจากขีดจำกัดของเราเตอร์ที่ทำงานโดยพิจารณาครั้งละแพ็กเก็ตเท่านั้น
4. มีความสามารถจำกัด เช่นไม่สามารถบันทึกล็อก (Log) ของแพ็กเก็ตที่ต้องสงสัยไว้ตรวจสอบภายหลังได้
5. เราเตอร์มีกำลังในการประมวลผลจำกัด หากเน็ตเวิร์กมีขนาดใหญ่และมีการสื่อสารข้อมูลหนาแน่น เราเตอร์จะทำงานหนักอยู่แล้ว เมื่อต้องมาทำการประมวลผลแอคเซสรูลด้วยก็อาจจะทำให้ประสิทธิภาพในการเราต์ (Route) แพ็กเก็ตต่ำลงไปมาก และการสื่อสารข้อมูลก็จะติดขัดเป็นคอขวดที่เราเตอร์

3.2.2 เซอร์กิตเลเวลไฟร์วอล / สเตตฟูลอินสเปกชันไฟร์วอล (Circuit-Level Firewall / Stateful Inspection Firewall)

การสื่อสารโดยทั่วไปจะเป็นการสื่อสารแบบต่อเนื่อง โต้ตอบกันไปมาระหว่างผู้รับและผู้ส่งอยู่เสมอ โปรโตคอลที่อยู่ในเลเยอร์ที่สูงกว่าอินเตอร์เน็ตเลเยอร์ ไม่ว่าจะเป็นทรานสปอร์ตอย่างเช่น ทีซีพี ยูดีพี หรือเลยไปถึงแอพพลิเคชันเลเยอร์เช่น เอฟทีพี, เอชทีทีพี, เอสเอ็มทีพี ล้วนแล้วแต่จะต้องมีสถานะของการสื่อสาร (State) เสมอ สถานะนี้จะทำให้ทั้งสองฝั่งสามารถสื่อสารกันได้อย่างต่อเนื่องคือทราบว่าตอนนี้กำลังอยู่ ณ จุดใดและจะต้องส่งหรือรับข้อมูลใดเป็นลำดับต่อไป

3.2.2.1 ความแตกต่างของการพิจารณาข้อมูลแบบแพ็กเก็ตฟิลเตอร์ริงกับสเตตฟูล (Stateful)

อันที่จริงสองเรื่องนี้มิได้ขัดแย้งกันแต่ประการใด แพ็กเก็ตนั้นเป็นการสื่อสารที่เป็นส่วนย่อยของการสื่อสารทั้งหมด ผลของการสื่อสารข้อมูลก็คือผลรวมของการสื่อสารข้อมูลหลายๆแพ็กเก็ตนั่นเอง แต่อย่างไรก็ตามการฟิลเตอร์หรือกรอง โดยพิจารณาทีละแพ็กเก็ตของทุกแพ็กเก็ตที่ผ่านเข้าออกนั้นอาจจะมีผลลัพธ์แตกต่างจากการฟิลเตอร์ของในแบบที่มองสถานะและภาพรวมหรือที่เรียกว่าสเตตฟูล (Stateful) หากเปรียบเทียบการพิจารณาข้อมูลครั้งละแพ็กเก็ตกับการพิจารณาแบบสเตตฟูลแล้ว ตัวอย่างที่น่าจะช่วยให้เข้าใจได้ง่ายขึ้นคือ

เปรียบเทียบการสื่อสารข้อมูลทั้งหมดเสมือนภาพยนตร์ แพ็กเก็ตก็จะหมายถึงภาพนิ่งแต่ละภาพที่นำมาต่อรวมกันแล้วเปิดดูอย่างรวดเร็ว ภาพนิ่งเหล่านั้นก็จะกลายเป็นภาพเคลื่อนไหว ดังนั้นการพิจารณาแพ็กเก็ตก็เป็นเสมือนการดูภาพนิ่งทีละภาพ โดยที่แต่ละภาพไม่มีส่วนเกี่ยวข้องกัน ดังนั้นข้อมูลที่จะรับรู้ได้ก็คือข้อมูลที่ปรากฏอยู่บนแต่ละภาพ แต่จะไม่สามารถเซ็นเซอร์เนื้อเรื่องซึ่งเป็นสิ่งที่เกิดขึ้นจากความสัมพันธ์ของภาพหลายๆภาพได้ มีโอกาสเป็นไปได้ว่ากิจกรรมบางชนิดที่หากดูเป็นภาพนิ่งแล้วจะไม่รู้สึกว่าเป็นสิ่งที่ไม่เหมาะสม แต่หากนำภาพนิ่งมาดูอย่างต่อเนื่องเป็นภาพเคลื่อนไหวแล้วก็อาจจะเป็นสิ่งที่ไม่พึงปรารถนาที่จะให้ปรากฏบนภาพยนตร์ก็เป็นได้

เซอร์กิตเลเวลไฟร์วอลเป็นไฟร์วอลที่ทำงานโดยที่สามารถเข้าใจสถานะการสื่อสารทั้งกระบวนการ เพราะถือว่าการสื่อสารข้อมูลจะสมบูรณ์ได้นั้นจะต้องมีทั้งการส่งและการรับอย่างสอดคล้องสัมพันธ์กันนั่นเอง หมายถึงหากไฟร์วอลจะสามารถควบคุมการสื่อสารได้จริงก็จะต้องสามารถเข้าใจกระบวนการของการสื่อสารตั้งแต่ต้นจนจบ โดยทั่วไปเราจะเรียกไฟร์วอลแบบนี้ว่า “สเตตฟูลอินสเปกชันไฟร์วอล” (หรือเรียกย่อๆว่าสเตตฟูลไฟร์วอล) เป็นไฟร์วอลที่ทำการควบคุมแทรฟฟิกโดยใช้หลักการของแพ็กเก็ตฟิลเตอร์ริงและการกำหนดแอคเซสรูลเช่นเดียวกับสกรีนิงเราเตอร์แต่สเตตฟูลไฟร์วอลจะมีความสามารถในการวิเคราะห์และรับรู้ความต่อเนื่องของแพ็กเก็ตในโปรโตคอลในระดับที่สูงขึ้นไปมากว่า ไม่ว่าจะเป็น ทีซีพี, เอฟทีพี, เอชทีทีพีหรือแม้กระทั่งโปรโตคอลในระดับแอพพลิเคชัน ที่จะมีวิธีการกำหนดสเตตของตนเอง

สเตตฟูลไฟร์วอลเป็นเครื่องมือที่ถูกออกแบบมาเพื่อทำหน้าที่ในการควบคุมแทรฟฟิกโดยเฉพาะไม่ได้เป็นการดัดแปลงการทำงานมาจากเราเตอร์จึงมีความสามารถในการควบคุมแทรฟฟิกการกำหนด

แอคเซสรูล การบริหาร รวมไปถึงความยืดหยุ่นของการควบคุมแทรฟฟิก และประสิทธิภาพในการทำงานที่สูงกว่าสกรีนิงเราเตอร์เป็นอย่างมาก โดยทั่วไปหากพูดถึงไฟร์วอลจะหมายถึงไฟร์วอลประเภทนี้เอง

ตามที่ได้กล่าวไว้ข้างต้นว่า ความแตกต่างที่สำคัญของไฟร์วอลทั้งสองชนิดนี้ในแง่ของการตรวจสอบแทรฟฟิกคือ สเตตฟูลไฟร์วอล มีความสามารถในการวิเคราะห์แทรฟฟิกที่ผ่านไปมาในโปรโตคอลที่เลเยอร์สูงขึ้นไปไม่ว่าจะเป็น ทีซีพี, ยูดีพี, ไอซีเอ็มพี ได้อย่างสมบูรณ์ต่างจาก สกรีนิงเราเตอร์ที่สามารถวิเคราะห์ได้เฉพาะเท่าที่จะมีข้อมูลใน 1 แพ็กเก็ตเท่านั้นเพราะบางครั้งแทรฟฟิกที่ผ่านไปมานั้นมีการเชื่อมโยงกันหลายแพ็กเก็ต โดยเฉพาะ ทีซีพี ซึ่งจะมีลำดับของการติดต่อสื่อสารที่สัมพันธ์กันในแต่ละแพ็กเก็ต การพิจารณาแพ็กเก็ตใดแพ็กเก็ตหนึ่งโดยไม่พิจารณาแพ็กเก็ตอื่นที่เกี่ยวข้องก็อาจจะไม่สามารถควบคุมแทรฟฟิกของ ทีซีพี ได้ นอกจากนี้ยังรวมไปถึงการที่สเตทฟูลไฟร์วอลมีความสามารถในการประกอบรวมแฟร็กเมนต์เข้าด้วยกันให้เป็นดาต้าแกรมที่สมบูรณ์ ก่อนหลังจากนั้นจึงนำดาต้าแกรมนั้นมาทำการตรวจสอบเปรียบเทียบกับแอคเซสรูล

นอกจากการเชื่อมโยงกันของหลายแพ็กเก็ตสำหรับแพ็กเก็ตสำหรับโปรโตคอล ทีซีพี ในทรานสปอร์ตเลเยอร์แล้ว ในแอพพลิเคชันเลเยอร์ก็มีแอพพลิเคชันบางชนิดที่จะต้องอาศัยการพิจารณาแทรฟฟิกอย่างต่อเนื่องเพื่อที่จะนำมากำหนดเป็นแอคเซสรูล ยกตัวอย่างเช่น การทำงานของเอฟทีพี ซึ่งในระหว่างการทำงานของแอพพลิเคชันนั้น โฮสต์ที่เป็นไคลเอนต์จะสามารถกำหนดพอร์ตชั่วคราวขึ้นมาเป็นเซิร์ฟเวอร์พอร์ตใช้สำหรับรับ-ส่งไฟล์ได้ โดยพอร์ตเหล่านี้จะปิดลงเมื่อการรับ-ส่งข้อมูลเสร็จสิ้นสมบูรณ์ ซึ่งในกรณีนี้หากไม่มีการพิจารณาแทรฟฟิกที่มีมาก่อนหน้าแล้ว ไฟร์วอลอาจจะถือได้ว่าการเปิดเซิร์ฟเวอร์พอร์ตชั่วคราวของเอฟทีพีไคลเอนท์ นั้นเป็นการเปิดให้บริการใหม่ขึ้นมาก็ได้ ดังนั้นสเตตฟูลไฟร์วอลจึงมีการทำงานที่ค่อนข้างใกล้ชิดกับแอพพลิเคชันเป็นอย่างมาก จะต้องสามารถเข้าใจคุณสมบัติของการสื่อสารข้อมูลของแต่ละแอพพลิเคชันได้ค่อนข้างดี เพราะแอพพลิเคชันที่ใช้งานอยู่ในเน็ตเวิร์กไม่ได้มีเฉพาะแอพพลิเคชันพื้นฐานเท่านั้น มีแอพพลิเคชันอื่นๆอีกมาก แต่หากแอพพลิเคชันใดมีการใช้งานอย่างแพร่หลายและเป็นที่นิยมของผู้ใช้ โดยส่วนใหญ่ผู้ผลิตจะใส่บิวต์อิน(Built-in) การควบคุมแทรฟฟิกสำเร็จรูปมาให้อยู่ในไฟร์วอลเลย

3.2.2.2 ข้อดีของสเตตฟูลไฟร์วอล

1. ใช้งานง่ายเพราะถูกออกแบบมาทำหน้าที่ของไฟร์วอลโดยเฉพาะ ตรวจสอบแก้ไขแอคเซสรูลได้ง่าย ทำให้ผู้ใช้ไม่ต้องคอยกังวลถึงคำสั่ง และรูปแบบของคำสั่ง ถึงแม้ว่าจะต่างยี่ห้อกันก็สามารถเรียนรู้ใหม่ได้อย่างรวดเร็ว
2. ประสิทธิภาพในการทำงานสูง เนื่องจากออกแบบมาทำหน้าที่ไฟร์วอลโดยเฉพาะ สามารถรองรับแอคเซสรูลที่ซับซ้อนได้ โดยที่ความสามารถในการทำงานไม่ลดลง
3. มีคุณสมบัติเพิ่มเติมให้ใช้ได้มากนอกเหนือจากการควบคุมแทรฟฟิก เช่นสามารถนำไปใช้ร่วมกับระบบการตรวจจับการบุกรุกหรือ IDS (Intrusion Detection System) เพื่อป้องกันการโจมตีได้อัตโนมัติ, สามารถบันทึกข้อมูลเอาไว้กลับมาดูภายหลังได้, สามารถใช้งานร่วมกับระบบป้องกันไวรัสได้เป็นต้น

4. การกำหนดแอคเซสรูลทำได้ง่าย เพราะไฟร์วอลมีความเข้าใจในโปรโตคอลระดับสูง ดังนั้น ผู้ใช้อาจจะไม่จำเป็นต้องมีความเชี่ยวชาญในเรื่องเน็ตเวิร์กมาก ก็พอจะใช้งานไฟร์วอลได้ โดยกำหนดกฎพื้นฐานของแอพพลิเคชันที่ผู้ใช้รู้จัก มากกว่าการกำหนดกฎโดยใช้ข้อมูลบนแพ็กเก็ตโดยตรง เช่นแทนที่จะต้องกำหนดแอคเซสรูลให้อนุญาต ICMP Time exceed in Transit ให้ผ่านได้ เพื่อจะใช้คำสั่ง Traceroute (ซึ่งโดยทั่วไปผู้ใช้ไม่ทราบว่าโปรแกรมใดใช้โปรโตคอลอะไร แต่จะรู้ว่าตนเองต้องการใช้โปรแกรมหรือแอพพลิเคชันอะไรบ้าง) ก็ระบุในไฟร์วอลว่าอนุญาตให้คำสั่ง Traceroute ทำงานได้หลังจากนั้นไฟร์วอลจึงกำหนดเป็นแอคเซสรูลที่ระบุโปรโตคอลเอง
5. สามารถเพิ่มเติมบริการอื่นได้ เช่น เวอร์ชวลไพรเวตเน็ตเวิร์ค (Virtual Private Network), ทันแนลิง (Tunneling)
6. สามารถเพิ่มเติมความปลอดภัยโดยระบบการตรวจสอบผู้ใช้ (Authenticate) ได้
7. การสื่อสารระหว่างไฟร์วอลกับแอดมินคอนโซล (Adminstration Console : โฮสต์ที่ทำหน้าที่ในการบริหารไฟร์วอล) จะมีความปลอดภัยสูง มีการตรวจสอบสิทธิ์ของผู้ที่เป็นแอดมินรวมทั้งการสื่อสารระหว่างไฟร์วอลกับคอนโซลจะมีการรักษาความปลอดภัยที่เข้มงวด มีการเข้ารหัสเพื่อป้องกันการดักอ่านข้อมูล

3.2.2.3 ข้อเสียของสเตตฟูลไฟร์วอล

1. มีราคาแพง ถึงแม้ว่าปัจจุบันจะลดลงไปมากแล้วแต่ก็ยังแพงอยู่
2. ในกรณีที่เป็นไฟร์วอลแบบซอฟต์แวร์ที่ทำงานอยู่บนระบบปฏิบัติการทั่วไปเช่น Solaris, Windows NT, Windows 2000 ต่างก็มีความเสี่ยงที่จะถูกเจาะได้เนื่องจากปัญหาของแต่ละระบบปฏิบัติการเอง ซึ่งจะสามารถเจาะได้ง่ายกว่าการเจาะเราเตอร์ เพราะรูรั่วของระบบปฏิบัติการมีมากกว่าของเราเตอร์
3. ในกรณีไฟร์วอลเป็นประเภทเน็ตเวิร์คแอพพลิแอนซ์ (Network Applicance) คือออกแบบทั้งซอฟต์แวร์และฮาร์ดแวร์เป็นเครื่องเดียวกันเพื่อทำหน้าที่เป็นไฟร์วอลโดยเฉพาะ ผู้ใช้จำเป็นต้องพึ่งพาผู้ผลิตค่อนข้างมาก หากมีปัญหาอาจจะไม่สามารถแก้ไขโดยการใช้อะไหล่ทดแทนจากที่อื่นได้

3.2.3 แอพพลิเคชันเลเวลไฟร์วอล (พร็อกซี่) (Application Level Firewall (Proxy))

พร็อกซี่เป็นเครื่องมือในการควบคุมแทรฟฟิกชนิดหนึ่งซึ่งทำงานที่ระดับของแอพพลิเคชันในลักษณะที่เป็นตัวกลางในการสื่อสารระหว่างไคลเอนต์กับเซิร์ฟเวอร์ โดยทำหน้าที่ป้องกันไม่ให้มีการสื่อสารโดยตรงระหว่างไคลเอนต์กับเซิร์ฟเวอร์ แต่ยังคงให้ไคลเอนต์สามารถใช้งานแอพพลิเคชันบนเซิร์ฟเวอร์ได้ตามปกติ และผู้ใช้ซึ่งใช้งานแอพพลิเคชันนั้นๆจะไม่ได้รับผลกระทบแต่อย่างใด

3.2.3.1 ลักษณะการทำงานของพร็อกซี่

โดยปกติทั่วไปแล้วการสื่อสารระหว่างไคลเอนต์กับเซิร์ฟเวอร์นั้น จะต้องมีการเชื่อมต่อหรือคอนเน็คชัน (Connection) เกิดขึ้นระหว่างไคลเอนต์กับเซิร์ฟเวอร์อยู่ตลอดเวลาที่สื่อสารกันอยู่ จุดสำคัญอยู่ตรงที่การเชื่อมต่อโดยตรงนั้นจะมีความเสี่ยงหลายประการ จึงมีการทำพร็อกซี่เข้ามาใช้งาน

หน้าที่ในการทำงานของพร็อกซี่ คือ เป็นตัวกลางรับข้อมูลจากไคลเอนต์มาแล้วทำการส่งต่อไปยังเซิร์ฟเวอร์ และรับข้อมูลที่ตอบกลับจากเซิร์ฟเวอร์กลับมายังไคลเอนต์ที่ทำการร้องขอ และจะทำหน้าที่นี้อยู่ตลอดเวลาที่ไคลเอนต์และเซิร์ฟเวอร์นั้นติดต่อกัน ซึ่งการที่มีพร็อกซี่มาเป็นตัวกลางระหว่างไคลเอนต์กับเซิร์ฟเวอร์นั้นทำให้โฮสต์ทั้งคู่ไม่จำเป็นต้องติดต่อโดยตรง เพียงแค่ติดต่อกับตัวกลางคือพร็อกซี่เท่านั้น และการทำงานของแอพพลิเคชันทั้งสองฝั่งยังคงทำได้เช่นเดิม

3.2.3.2 ขั้นตอนการนำพร็อกซี่เข้ามาใช้งาน

1. การเริ่มต้นการทำงานของแอพพลิเคชันโดยทั่วไป เริ่มจากการที่แอพพลิเคชันบนไคลเอนต์ขอรับข้อมูลจากเซิร์ฟเวอร์ตามโปรโตคอลในแอพพลิเคชันเลเยอร์ที่กำหนดไว้เช่น เว็บบราวเซอร์กับเว็บเซิร์ฟเวอร์ จะใช้โปรโตคอล เอชทีทีพี ในการสื่อสารระหว่างกัน
2. เมื่อเว็บเซิร์ฟเวอร์ได้รับการขอข้อมูลจากบราวเซอร์แล้วก็จะตอบกลับไปและเริ่มการติดต่อสื่อสารกัน และทั้งฝั่งไคลเอนต์และเซิร์ฟเวอร์ก็จะทำการติดต่อสื่อสารกันตามที่โปรโตคอล เอชทีทีพี กำหนดจนจบการสื่อสารเซสชั่น (Session) นั้น อย่างไรก็ตาม โปรโตคอล เอชทีทีพี นั้นจะต้องอาศัย ทีซีพี ในการรับส่งข้อมูลระหว่างไคลเอนต์กับเซิร์ฟเวอร์ นั่นหมายถึงไคลเอนต์จะต้องสามารถติดต่อกับเซิร์ฟเวอร์ได้ด้วย ทีซีพี เสียก่อน เนื่องจาก ทีซีพี เป็นโปรโตคอลเลเยอร์ที่อยู่ภายใต้ เอชทีทีพี อีกเลเยอร์หนึ่ง ดังนั้นสภาวะการทำงานปกติของ เอชทีทีพี บราวเซอร์จะต้องสามารถติดต่อกับเซิร์ฟเวอร์โดยตรงเสมอ นั่นคือในสภาวะการทำงานปกตินั้นแพ็กเก็ตของ ทีซีพี/ไอพี จะต้องสามารถส่งถึงกันระหว่างโฮสต์ทั้งคู่ได้
3. เมื่อนำพร็อกซี่มาใช้งานจะต้องติดตั้งตรงจุดที่คั่นกลางระหว่างไคลเอนต์กับเซิร์ฟเวอร์เพื่อเป็นตัวกลาง โดยที่พร็อกซี่จะต้องมี 2 อินเตอร์เฟส โดยอินเตอร์เฟสหนึ่งต่ออยู่กับเน็ตเวิร์กของไคลเอนต์และอีกอินเตอร์เฟสหนึ่งต่ออยู่กับเซิร์ฟเวอร์ ซึ่งหากพิจารณาที่พร็อกซี่แล้วจะเห็นว่าสามารถติดต่อได้โดยตรงกับทั้งไคลเอนท์และเซิร์ฟเวอร์ แต่สำหรับไคลเอนต์และเซิร์ฟเวอร์จะติดต่อได้แต่เพียงกับพร็อกซี่เท่านั้น

ในลักษณะที่มีพร็อกซี่มาคั่นกลางระหว่างเน็ตเวิร์กทั้งสองนั้น การสื่อสารระหว่างไคลเอนต์และเซิร์ฟเวอร์ด้วยวิธีการเดิมโดยใช้ เอชทีทีพี เช่นเดิมเหมือนกับมีการสื่อสารกันโดยตรงนั้นย่อมไม่สามารถจะกระทำได้ เพราะการสื่อสารในเลเยอร์ล่างของ ทีซีพี/ไอพี นั้นไม่สามารถทำได้สำเร็จดังนั้นจึงจำเป็นต้องมีการปรับปรุงแก้ไขโปรโตคอลให้สามารถรองรับการสื่อสารที่มีตัวกลางมาถ่ายทอดข้อมูลได้ โดยให้ในระดับ ทีซีพี/ไอพี นั้นกำหนดให้เพียงโฮสต์แต่ละฝั่งสามารถติดต่อกับพร็อกซี่เท่านั้น ส่วนในระดับ เอชทีทีพี นั้นพร็อกซี่จะทำการส่งต่อระหว่างทั้งสองฝั่งให้ดูประหนึ่งว่าสามารถติดต่อกันได้โดยตรง ซึ่งจุดสำคัญของพร็อกซี่ก็จะอยู่ตรงนี้เอง อาจจะกล่าวโดยสรุปคือพร็อกซี่จะทำให้โฮสต์ไม่สามารถติดต่อกันได้โดยโปรโตคอล ทีซีพี/ไอพี แต่จะสามารถติดต่อกันได้ด้วยโปรโตคอลในระดับแอพพลิเคชันเลเยอร์

อย่างที่กล่าวข้างต้นคือแอพพลิเคชันที่ใช้งานพร็อกซี่นั้นจะต้องมีการแก้ไขในระดับแอพพลิเคชันในบางส่วนเพื่อให้สามารถสื่อสารผ่านพร็อกซี่ได้ ดังเช่นเว็บบราวเซอร์ หากจะทำการสื่อสารโดยผ่านพร็อกซี่นั้นจะต้องทำการปรับแต่งเพื่อให้บราวเซอร์ทราบว่าจะให้ติดต่อกับเว็บไซท์โดยผ่านพร็อกซี่หรือจะติดต่อกับเว็บเซิร์ฟเวอร์โดยตรง จะได้ทำการสื่อสารกันได้ถูกต้องและเมื่อเว็บบราวเซอร์ต้องการจะติดต่อไปยังเซิร์ฟเวอร์ใดก็เพียงแต่ส่งคำขอไปยังพร็อกซี่เท่านั้น หลังจากนั้นก็เป็นภาระของ พร็อกซี่ในการติดต่อกับเว็บเซิร์ฟเวอร์ตัวจริง แล้วจึงนำผลที่ได้จากเว็บเซิร์ฟเวอร์ตอบกลับมายังบราวเซอร์

เมื่อปรับแต่งให้บราวเซอร์ทำการสื่อสารผ่านพร็อกซี่ การทำงานจะมีการเปลี่ยนแปลงไปคือจากเดิมเมื่อเว็บบราวเซอร์ต้องการติดต่อกับเว็บเซิร์ฟเวอร์ก็จะส่งคำขอในระดับแอพพลิเคชัน ซึ่งในกรณีนี้คือ เอชทีทีพี ไปยังเซิร์ฟเวอร์ปลายทาง แต่สำหรับในระดับเน็ตเวิร์กนั้นแพ็กเก็ตของคำขอดังกล่าวจะมีไอพีแอดเดรส ของปลายทางคือพร็อกซี่แค่นั้น ไม่ว่าเว็บเซิร์ฟเวอร์จะติดต่อไปยังเว็บเซิร์ฟเวอร์ซึ่งอยู่ที่ใดก็ตาม แพ็กเก็ตจริงๆก็จะเดินทางไปแค่พร็อกซี่เท่านั้น ในขณะเดียวกันพร็อกซี่ก็คอยโต้ตอบในระดับของเอชทีทีพี กลับไปยังไคลเอนต์ประหนึ่งว่าตนเองเป็นเว็บเซิร์ฟเวอร์ปลายทางจริง

โดยทั่วไปพร็อกซี่ที่มีการนำมาใช้งานมากที่สุดคือเว็บพร็อกซี่ แต่จริงๆแล้วยังมีแอพพลิเคชันหลายชนิดที่สามารถใช้พร็อกซี่ได้เช่น เมล์พร็อกซี่, เอฟทีพีพร็อกซี่ เป็นต้น ซึ่งหากแอพพลิเคชันที่ใช้งานอยู่ปกติไม่สามารถปรับแต่งให้ใช้พร็อกซี่ได้ เช่น เอฟทีพีก็จำเป็นจะต้องติดตั้งโปรแกรมพร็อกซี่ไคลเอนท์ (Proxy Client) เพื่อใช้งานกับโปรแกรมเอฟทีพีเพื่อทำหน้าที่ดัดแปลงโปรโตคอลเดิมให้รองรับการสื่อสารพร็อกซี่ได้

3.2.3.3 ข้อดีของการใช้งานพร็อกซี่

1. สามารถควบคุมการติดต่อสื่อสารระหว่างอินเตอร์เน็ตกับเน็ตเวิร์กภายในให้อยู่ในระดับแอพพลิเคชันเท่านั้น ตัดขาดการติดต่อโดยตรงในระดับเน็ตเวิร์กเลเยอร์ระหว่างอินเตอร์เน็ตกับเน็ตเวิร์กภายในออกจากกันอย่างเด็ดขาด ทำให้ลดความเสี่ยงต่อการถูกควบคุมจากการสแกน การถูกเจาะระบบ การก่อกวนโดยใช้เทคนิคในระดับเน็ตเวิร์กเลเยอร์ที่จะเข้ามายังเน็ตเวิร์กภายในได้อย่างเด็ดขาด

2. สามารถเพิ่มเติมหน้าที่การทำงานอย่างอื่นเข้าไปในพร็อกซี่ได้ เช่นสำหรับเว็บพร็อกซี่นอกจากจะเป็นตัวกลางในการติดต่อแล้ว ยังสามารถควบคุมไม่ให้เว็บบราวเซอร์ติดต่อกับเว็บไซท์ที่ไม่ต้องการได้อีกด้วย โดยการกำหนดรายชื่อเว็บไซท์เหล่านั้นไว้ในพร็อกซี่
3. สามารถทำการแคชข้อมูลเก็บไว้ในตัวพร็อกซี่ สำหรับข้อมูลใดที่มีการเรียกใช้บ่อยๆก็ไม่จำเป็นต้องไปอ่านจากเซิร์ฟเวอร์ใหม่ทุกครั้ง แต่ส่วนนี้จะใช้กับข้อมูลที่เป็นสแตติคเท่านั้น ข้อมูลที่มีการเปลี่ยนแปลงตลอดเวลาเป็นไดนามิคอาจจะไม่สามารถแคชไว้ได้
4. ทำให้ผู้ใช้มีการใช้แบนด์วิดธ์ร่วมกันอย่างมีประสิทธิภาพ โดยเฉพาะเมื่อใช้ร่วมกับการแคชที่มีอยู่ในพร็อกซี่ทำให้ช่วยประหยัดการใช้งานแบนด์วิดธ์ไปได้มาก
5. สามารถเพิ่มเติมส่วนการตรวจสอบผู้ใช้ (Authenticate) เข้าไปในหน้าที่หนึ่งของพร็อกซี่ได้ โดยการอนุญาตให้สามารถใช้งานพร็อกซี่นั้นจะขึ้นอยู่กับสิทธิ์การใช้งานที่ผู้ใช้มีอยู่ ทำให้สามารถควบคุมการใช้งานได้ใกล้ชิดมากขึ้นกว่าการควบคุมแทรฟฟิกโดยพิจารณาจาก ไอพี แอดเดรส ของโฮสต์เพียงอย่างเดียว
6. สามารถทำการกลั่นกรองเนื้อหาของข้อมูลได้ (Content Filtering) ทำให้สามารถนำมาเป็นเงื่อนไขในการอนุญาตให้ข้อมูลเหล่านั้นผ่านผ่านเข้าออกได้ เช่นเว็บพร็อกซี่สามารถตรวจสอบเนื้อหาของเว็บไซท์ที่ผู้ใช้เข้าไปดู หากปรากฏว่ามีข้อความที่ไม่เหมาะสมพร็อกซี่ก็จะสามารถดรอปเซสชันที่ผู้ใช้ขอมาได้ หรือในกรณีที่เป็นอีเมล์ พร็อกซี่ก็จะสามารถตรวจสอบเนื้อหาในอีเมล์ได้ว่ามีข้อความที่ไม่เหมาะสมหรือไม่ และอาจจะครอบคลุมถึงการตรวจสอบหาไวรัสที่แนบมากับจดหมายได้อีกด้วย

3.2.3.4 ข้อเสียของการใช้งานพร็อกซี่

1. ขึ้นอยู่กับแอพพลิเคชัน หากแอพพลิเคชันไม่รองรับการสื่อสารโดยผ่านพร็อกซี่ก็ไม่สามารถใช้งานได้
2. ไม่สามารถใช้งานกับแอพพลิเคชันที่ต้องการการสื่อสารโดยตรงแบบ end-to-end ซึ่งแพ็กเก็ตจะต้องมาจากโฮสต์ปลายทางทั้งคู่เท่านั้น ผ่านตัวกลางไม่ได้
3. เสี่ยงต่อการละเมิดความเป็นส่วนบุคคล (Privacy) เนื่องจากข้อมูลทั้งหมดที่สื่อสารจะต้องผ่านพร็อกซี่ก่อนเสมอ และพร็อกซี่ก็มีความสามารถที่จะเก็บข้อมูลเหล่านั้นไว้ตรวจสอบได้ หากมีผู้นำข้อมูลเหล่านั้นไปวิเคราะห์จะสามารถทราบการใช้งานหรืออาจจะทราบข้อมูลทั้งหมดของผู้ใช้ได้
4. เนื่องจากลักษณะของแต่ละแอพพลิเคชันนั้นจะแตกต่างกันออกไป ดังนั้นพร็อกซี่ของแต่ละแอพพลิเคชันจึงทำหน้าที่เฉพาะแอพพลิเคชันนั้นๆ ไม่สามารถใช้ร่วมกันได้ หากโฮสต์ที่อยู่หลังพร็อกซี่มีการใช้งานหลายแอพพลิเคชันก็จะต้องมีพร็อกซี่จำนวนมากเปิดให้บริการตามจำนวนแอพพลิเคชันนั้นๆ
5. ความสามารถในการประมวลผลของโฮสต์ที่ทำหน้าที่พร็อกซี่อาจจะเป็นคอขวดของระบบได้เพราะการสื่อสารทั้งหมดของไคลเอนต์และเซิร์ฟเวอร์จะถูกรวมศูนย์อยู่ที่ พร็อกซี่ก่อน

เสมอ แทนที่จะกระจายไปยังไคลเอนต์และเซิร์ฟเวอร์ ปัญหาลักษณะนี้จะสามารถพบได้ชัดเมื่อมีไคลเอนต์จำนวนมาก

6. เนื่องจากพร็อกซี่เป็นแอพพลิเคชันชนิดหนึ่งเช่นกัน การติดต่อกับในระบบจะอาศัยระบบปฏิบัติการเป็นหลัก จึงมีความสามารถในการป้องกันตัวเองต่ำกว่าไฟร์วอลทั่วไป ตัวพร็อกซี่เองจึงมีความเสี่ยงต่อการถูกโจมตีได้มากและเปราะบางต่อการโจมตีให้ปิดบริการด้วยเทคนิคในระดับเน็ตเวิร์ก ซึ่งอาจส่งผลให้พร็อกซี่อาจจะหยุดทำงานลงได้โดยง่าย โดยเฉพาะเมื่อพร็อกซี่นั้นเป็นโฮสต์ที่ต่อโดยตรงกับอินเตอร์เน็ต จึงเป็นเสมือนด่านหน้าของเน็ตเวิร์กที่จะต้องถูกสแกน ถูกเจาะอย่างแน่นอน แต่ในระดับความต้านทานของพร็อกซี่นั้นต่ำกว่าไฟร์วอลทั่วไป จึงมีแนวโน้มว่าหากใช้พร็อกซี่โดยปราศจากไฟร์วอลร่วมด้วยโอกาสที่พร็อกซี่จะโดนเจาะนั้นมีอยู่สูงมาก

บทที่ 4
รูปแบบการโจมตี

รูปแบบของการโจมตีเครือข่ายคอมพิวเตอร์นั้นมีอยู่หลายประเภท โดยการโจมตีที่ผู้ไม่หวังดีนิยมใช้กันคือการโจมตีเพื่อให้ปิดบริการและการโจมตีอีกรูปแบบหนึ่งที่สร้างความเสียหายให้กับเซิร์ฟเวอร์ได้ค่อนข้างมากคือการโจมตีเว็บแอพพลิเคชั่น โดยการโจมตีแต่ละประเภทมีดังต่อไปนี้

4.1 การโจมตีเพื่อให้ปิดบริการ (Denial of Services : DoS)

4.1.1 ความหมายของการโจมตีเพื่อให้ปิดบริการ

การโจมตีเพื่อให้ปิดบริการ หมายถึง การกระทำใดๆ ที่ทำให้ระบบเป้าหมายไม่สามารถให้บริการบางอย่างได้ หรือไม่ สามารถให้บริการต่อไปได้อีก โดยทั่วไปโจมตีที่ พอร์ตของทีซีพี/ไอพี ซึ่งเชื่อมต่อกับบริการ (Services) ที่รองรับพอร์ตนั้นๆ ดังนั้นการโจมตีพอร์ตจึงเท่ากับการโจมตีบริการของระบบนั่นเอง และอาจมีผลทำให้ระบบนั้นไม่สามารถให้บริการบางอย่างได้ หรือไม่สามารถให้บริการใดๆ ได้เลย

4.1.2 ประเภทของการโจมตีเพื่อให้ปิดบริการ

ในที่นี้ประเภทของการโจมตีสามารถแบ่งได้ดังต่อไปนี้

4.1.2.1 การโจมตีประเภทอยู่ในชั้นทรานสปอร์ต หรือชั้นอินเทอร์เน็ต

การโจมตีในระดับชั้นนี้สามารถแบ่งได้เป็น 3 แบบหลักๆ ได้แก่

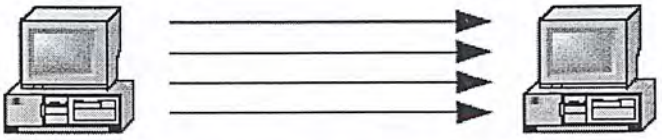
4.1.2.1.1 การส่งแพ็กเก็ตจำนวนมาก (Amount of Packets Sending)

การโจมตีแบบนี้เป็นการส่งแพ็กเก็ตปริมาณมากเข้าไปยังระบบเป้าหมายอาจทำให้ระบบเป้าหมายไม่สามารถให้บริการบางอย่าง หรือไม่สามารถทำงานต่อไปได้ ซึ่งแพ็กเก็ตที่ส่งออกไปนี้สามารถแบ่งออกได้เป็น

(1) แพ็กเก็ตข้อมูล (Data Packets)

การโจมตีวิธีนี้ทำได้โดยการส่งแพ็กเก็ตข้อมูลปริมาณมาก เมื่อข้อมูลเข้ามาสู่เครื่องเป้าหมายก็เก็บไว้ในบัฟเฟอร์ก่อนนำมาประมวลผลอีกครั้ง ดังนั้นหากส่งแพ็กเก็ตเข้ามาเป็นปริมาณมาก อาจทำให้บัฟเฟอร์ของเครื่องเป้าหมายไม่เพียงพอที่จะสามารถรองรับแพ็กเก็ตเหล่านั้นได้ทั้งหมด ซึ่งอาจทำให้เครื่องเป้าหมายให้บริการได้ช้าลงหรือต้องหยุดการให้บริการไปเลย ตัวอย่างการโจมตีประเภทนี้เช่น Ping Flood Attack เป็นต้น

Ping Flood เป็นการโจมตีในยุคแรกๆ ของ DoS หลักการคือส่ง ICMP Echo Request (รูปแบบเดียวกับคำสั่ง Ping) ไปยังเป้าหมายมากๆ ในระยะเวลาติดต่อกัน ทำให้เป้าหมายต้องคอยตอบ ICMP Echo Reply ตลอดเวลาจนไม่สามารถให้บริการอย่างอื่นได้ ความรุนแรงของการโจมตีขึ้นอยู่กับปริมาณแพ็กเก็ตที่โจมตีไปยังเครื่องเป้าหมาย หากเครื่องที่ทำการโจมตีมีประสิทธิภาพสูงและเครือข่ายมีแบนด์วิดธ์มากอาจส่งผลทำให้เครื่องเป้าหมายหยุดการทำงานลงได้



รูปที่ 4-1 การโจมตีด้วย Ping Flood Attack

ข้อสังเกตสำหรับการโจมตีประเภทนี้คือ จะปรากฏแพ็กเก็ต ICMP Echo Request และ ICMP Echo Reply ปริมาณมหาศาล โดยมีการรับส่งกันระหว่างเครื่องเป้าหมายที่ถูกโจมตีกับเครื่องอื่นๆ ที่อาจมีหรือไม่มีตัวตนในอินเตอร์เน็ตก็ได้ เนื่องจากกระบวนการสำคัญอย่างหนึ่งของการโจมตีลักษณะนี้คือ ผู้โจมตีต้องปลอมหมายเลขไอพี (IP Spoofing) เสมอ เพื่อป้องกันไม่ให้แพ็กเก็ต ICMP Echo Reply ถูกส่งกลับมายังเครื่องตัวเอง ซึ่งจะทำให้ผู้โจมตีได้รับผลกระทบจากการโจมตีด้วย และการปลอมไอพียังเป็นหลักประกันได้ว่าจะไม่สามารถติดตามได้ว่าผู้ใดเป็นผู้โจมตี รูปแบบแพ็กเก็ตที่เกิดขึ้นจากการโจมตีมีลักษณะดังนี้

14:49:43.217137 62.51.12.23 > 10.1.1.10 : icmp: echo request
14:49:43.217175 10.1.1.10 > 62.51.12.23 : icmp: echo reply
14:49:43.217195 62.51.12.23 > 10.1.1.10 : icmp: echo request
14:49:43.217219 10.1.1.10 > 62.51.12.23 : icmp: echo reply
14:49:43.217245 96.141.106.124 > 10.1.1.10 : icmp: echo request
14:49:43.217279 10.1.1.10 > 96.141.10.124 : icmp: echo reply
14:49:43.219017 172.19.251.18 > 10.1.1.10 : icmp: net 162.75.127.79 unreachable
14:49:43.237136 75.126.62.65 > 10.1.1.10 : icmp: echo request
14:49:43.237169 10.1.1.10 > 75.126.62.65 : icmp: echo reply
14:49:43.237193 75.126.62.65 > 10.1.1.10 : icmp: echo request
14:49:43.237216 10.1.1.10 > 75.126.62.65 : icmp: echo reply
14:49:43.237240 218.155.179.58 > 10.1.1.10 : icmp: echo request
14:49:43.237272 10.1.1.10 > 218.155.17.58 : icmp: echo reply

รูปที่ 4-2 รูปแบบแพ็กเก็ตที่เกิดขึ้นจากการโจมตีจาก Ping Flood Attack

นอกจาก Ping flooding Attack จะสร้างความเสียหายแก่เครื่องเป้าหมายแล้วยังสร้างความเสียหายแก่ระบบเครือข่ายของเครื่องเป้าหมายด้วย เพราะการโจมตีวิธีนี้จะสร้างแพ็กเก็ตเป็นจำนวนมากขึ้นในเครือข่ายที่เครื่องเป้าหมายตั้งอยู่ ทำให้ระบบเครือข่ายเกิดความคับคั่งของข้อมูล(Congestion) อาจส่งผลให้เครือข่ายเป็นอัมพาตได้การป้องกันการการโจมตีลักษณะนี้ทำได้โดยการกำหนดที่อุปกรณ์เราเตอร์หรือไฟร์วอล โดย

กำหนดไม่ให้แพ็กเก็ต ICMP Echo เข้ามายังเซิร์ฟเวอร์ แต่อย่างไรก็ตามแพ็กเก็ต ICMP Echo ได้ถูกใช้ในโปรแกรม Ping หากมีการปิดกั้นแพ็กเก็ต ICMP Echo จะทำให้ไม่สามารถตรวจสอบสถานะของเซิร์ฟเวอร์ได้ แนวทางที่ควรปฏิบัติคือกำหนดแบนด์วิดธ์ของแพ็กเก็ต ICMP Echo ให้เหมาะสมในอุปกรณ์เราเตอร์ โดยให้เพียงพอต่อการใช้งานสำหรับการตรวจสอบสถานะของระบบ แต่ไม่มากจนทำให้เกิดการโจมตีได้

(2) แพ็กเก็ตสำหรับการควบคุม (Control Packets)

นอกจากแพ็กเก็ตที่ เป็นตัวข้อมูลแล้ว ยังมีแพ็กเก็ตอีกรูปแบบหนึ่งที่สำคัญมากสำหรับการติดต่อสื่อสารบนโพรโตคอลทีซีพี/ไอพี นั่นคือแพ็กเก็ตส่วนการควบคุม ตัวอย่างแพ็กเก็ตประเภทนี้คือ สัญญาณ SYN หรือ ACK สำหรับการสถาปนาการเชื่อมต่อ หรือสัญญาณ FIN สำหรับการยกเลิกการเชื่อมต่อ เป็นต้น ตัวอย่างของการโจมตีแบบนี้ ได้แก่ การทำ SYN flood เนื่องจากปกติการเชื่อมต่อแบบ 3-way handshake เป็นไปตามลักษณะที่ได้อธิบายในหัวข้อ 2.4 แต่ในการโจมตีลักษณะนี้ใช้วิธีทำให้การทำ 3-way handshake ไม่สมบูรณ์กล่าวคือ เครื่องที่ขอบริการส่งสัญญาณ SYN ไปแต่เมื่อได้รับสัญญาณ ACK จากเครื่องที่ให้บริการแล้ว ไม่ส่งสัญญาณ ACK ตอบกลับไป ทำให้เครื่องที่ให้บริการต้องเปิดการเชื่อมต่อรอการตอบกลับ ดังรูปที่ 4-1 ซึ่งการเปิดการเชื่อมต่อรอเอาไว้นี้ต้องใช้ทรัพยากรของระบบส่วนหนึ่ง และหากมีการส่งสัญญาณในลักษณะนี้ มากๆ และทรัพยากรของระบบมีไม่เพียงพอ อาจทำให้ระบบไม่สามารถให้บริการอย่างอื่น หรือให้บริการกับผู้ร้องขอรายอื่นได้

รูปที่ 4-3 แสดงการส่งแพ็กเก็ตแบบ SYN Flood

สำหรับการโจมตีแบบ SYN Flood จะทำให้เกิดแพ็กเก็ตในระบบเครือข่ายในลักษณะดังรูป

10:09:43.143 10.0.0.3 **> isag14.ce.kmitl.ac.th.80: S** 399259715:399259715(0)

10:09:43.149 10.0.0.4 **> isag14.ce.kmitl.ac.th.80: S** 399259715:399259715(0)

10:09:43.158 10.0.0.5 **> isag14.ce.kmitl.ac.th.80: S** 399259715:399259715(0)

10:09:43.165 10.0.0.6 **> isag14.ce.kmitl.ac.th.80: S** 399259715:399259715(0)

10:09:43.177 10.0.0.7 **> isag14.ce.kmitl.ac.th.80: S** 399259715:399259715(0)

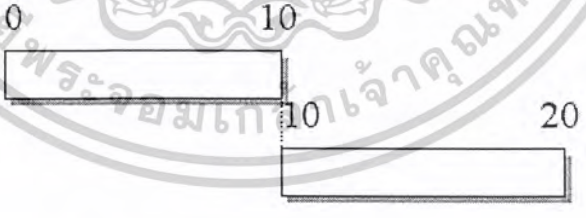
10:09:43.185 10.0.0.8 **> isag14.ce.kmitl.ac.th.80: S** 399259715:399259715(0)

รูปที่ 4-4 แพ็กเก็ตที่เกิดขึ้นจากการโจมตีแบบ SYN Flood

ในปัจจุบันนี้การโจมตีแบบ SYN Flood ถือได้ว่าเป็นการโจมตีที่ได้ผลและหาทางป้องกันได้ยาก เนื่องจากยากที่จะแยกลักษณะของแพ็กเก็ตที่ใช้ในการโจมตีกับแพ็กเก็ตที่ขอเริ่มต้นเชื่อมต่อทั่วไป นอกจากนี้ไฟร์วอลหรือเราเตอร์ทั่วไปยังไม่สามารถป้องกันการโจมตีประเภทนี้ได้อย่างสมบูรณ์ หนทางที่เป็นไปได้คือการใช้ระบบตรวจจับผู้บุกรุกทางระบบเครือข่ายทำการตรวจจับการโจมตี เพื่อนำข้อมูลจากการโจมตีกลับไปตั้งค่าอุปกรณ์เราเตอร์หรือไฟร์วอลเพื่อป้องกันการโจมตีมายังเซิร์ฟเวอร์

4.1.2.1.2 ความผิดปกติของแฟร็กเมนต์ (Abnormal Fragmentation)

การโจมตีวิธีนี้อาศัยหลักการแฟร็กเมนเตชันและรีแอสเซมเบิลที่กล่าวไว้ข้างต้น โดยทำให้แพ็กเก็ตนั้นต้องมีการรีแอสเซมเบิล (กำหนดค่า MF flag = 0) ซึ่งปกติการรีแอสเซมเบิลแพ็กเก็ตทั้งหมดต้องสามารถเชื่อมต่อกันได้สนิท ดังรูปที่ 4-5 แต่แพ็กเก็ตที่ผู้บุกรุกส่งไปมีการแก้ไขข้อมูลในบางฟิลด์ ทำให้เกิดความผิดปกติในกระบวนการรีแอสเซมเบิล ซึ่งการโจมตีในลักษณะนี้ แบ่งได้ดังต่อไปนี้

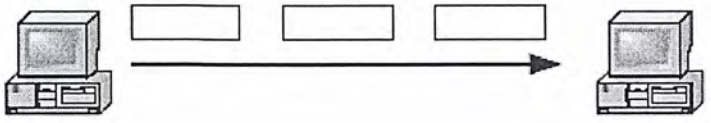


รูปที่ 4-5 แสดงการรีแอสเซมบลีแบบปกติ

(1) การส่งแพ็กเก็ตที่มีลำดับผิดปกติ (Abnormal Sequences of Packets Sending)

ปกติการส่งแพ็ กเก็ตมั กเรียงตามลำดับกันไป หากไม่เรียงลำดับก็ต้องรอจนกว่าแพ็กเก็ตก่อนหน้านี้ มาถึง เพื่อเรียงลำดับแพ็กเก็ตที่เครื่องรับ แต่การโจมตีแบบนี้กลับส่งเฉพาะแพ็กเก็ตสุดท้าย เพื่อให้ระบบเป้าหมายรอแพ็กเก็ตก่อนหน้า และส่งไปเป็นปริมาณมากๆ ซึ่งจะส่งผลให้ระบบต้องจองทรัพยากร

ส่วนหนึ่งเพื่อรองรับแพ็กเก็ตที่ต้องรอเหล่านั้นในปริมาณมาก จนกระทั่งระบบไม่สามารถจัดหาทรัพยากรได้เพียงพอ ส่งผลให้ระบบไม่สามารถให้บริการได้อย่างอื่นได้



รูปที่ 4-6 แสดงการส่งเฉพาะแพ็กเก็ตสุดท้ายไปยังเป้าหมาย

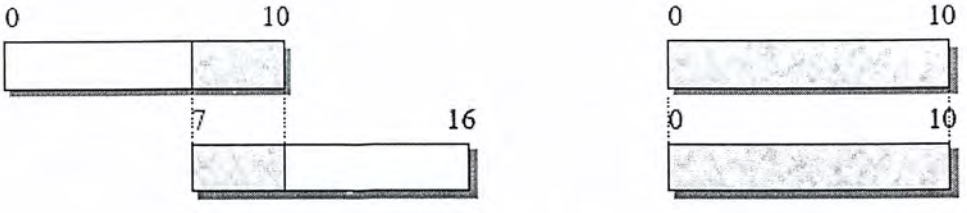
โดยปกติแล้วการโจมตีในรูปแบบนี้ผู้โจมตีจะแก้ไขข้อมูลในฟิลด์แสดงลำดับของแพ็กเก็ต (Fragment Offset) ของแพ็กเก็ตไอพี ซึ่งเป็นส่วนที่แสดงลำดับของข้อมูลหลังจากกระบวนการแฟร็กเมนต์เตชัน โดยแก้ให้ส่งแพ็กเก็ตสุดท้ายหรือแพ็กเก็ตหลังๆ เพียงแพ็กเก็ตเดียวเลย ทำให้ระบบเป้าหมายต้องรอแพ็กเก็ตก่อนหน้านี้

(2) การส่งแพ็กเก็ตที่มีขนาดเหลื่อมกัน (Overlapped Packets' Size Sending)

ปกติแพ็กเก็ตที่ส่งมาต้องนำมาต่อกันที่ระบบเป้าหมายได้ พอดี แต่การโจมตีแบบนี้เป็นการส่งแพ็กเก็ตที่มีขนาดเหลื่อมกัน หรือซ้อนทับกันทำให้ข้อมูลเมื่อมาต่อกันแล้วเกิดความผิดพลาด หรือไม่สามารถเชื่อมต่อกันได้โดยปกติแล้วการโจมตีแบบนี้ ผู้บุกรุกสามารถแก้ไขข้อมูลได้ 2 แห่งใหญ่ๆ ได้แก่

- การแก้ ไขข้อมูลที่ฟิลด์แสดงลำดับของแพ็กเก็ต (Fragment Offset) ของแพ็กเก็ต ไอพี หลังจากกระบวนการรีแอสเซมเบิล ซึ่งทำให้ลำดับในการส่งมีความผิดพลาด และอาจเกิดการเหลื่อมล้ำของแพ็กเก็ต กระบวนการรีแอสเซมเบิลอาจเกิดปัญหาได้

- การแก้ไขฟิลด์แสดงความยาวของ (Total Length) ของแพ็กเก็ตไอพี หลังจากกระบวนการรีแอสเซมเบิล ขนาดของแพ็กเก็ตที่มาต่อไม่พอดีกันทำให้ไม่สามารถรวมแพ็กเก็ตได้ หรือหากรวมได้ ข้อมูลที่ได้ก็ไม่ถูกต้อง



รูปที่ 4-7 แสดงการรีแอสเซมบลีแบบแพ็กเก็ตมีขนาดเหลื่อมกัน

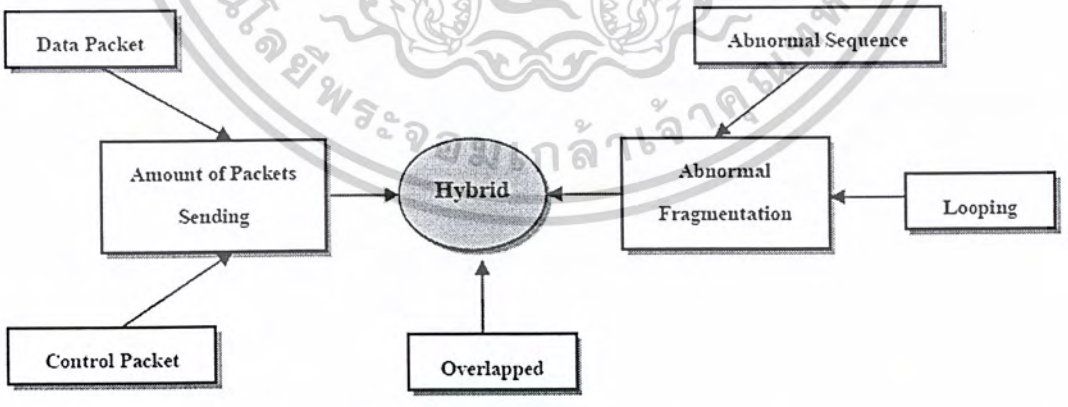
4.2.1.3 การส่งแพ็กเก็ตแบบวนลูป (Looping)

เป็นการโจมตีโดยการส่ งโดยกำหนดค่าแอดเดรสต้นทาง (Source Address) และแอดเดรสปลายทาง (Destination Address) ให้เหมือนกันทำให้เกิดการรับส่งวนไปวนมาอยู่ที่เครื่องเป้าหมายเอง เช่น LAND Attack ซึ่งเป็นโปรแกรมโจมตีที่มีการกำหนดแอดเดรสต้นทาง และแอดเดรสปลายทางเป็นค่าเดียวกัน คือ เป็นแอดเดรสของเครื่องเป้าหมายนั่นเอง ทำให้เกิดการส่งวนไปวนมาอยู่ที่เครื่องเป้าหมาย

รูปที่ 4-9 แสดงการโจมตีโดยส่งแพ็กเก็ตแบบวนลูป

4.2.1.4 การโจมตีแบบผสม (Hybrid)

คือ การโจมตีที่อาศัยวิธีการผสมกันระหว่างสามแบบแรกที่ได้กล่าวมาแล้ว ดังรูปที่ 4-10



รูปที่ 4-10 แสดงแผนภูมิแสดงประเภทของการโจมตีเพื่อให้ปิดบริการสำหรับสแตกทีซีพี/ไอพี

4.2 การโจมตีเว็บแอพพลิเคชั่น (Web Server Attacking)

เว็บแอพพลิเคชั่น หมายถึงซอร์ฟแวร์ที่สามารถเข้าใช้งานผ่านเว็บบราวเซอร์ องค์ประกอบของเว็บแอพพลิเคชั่นนั้นประกอบด้วย

1. เว็บเซิร์ฟเวอร์ (Web Server)
2. แอพพลิเคชันเซิร์ฟเวอร์ (Application Server)
3. ดาต้าเบสเซิร์ฟเวอร์ (Database Server)

การโจมตีระบบเว็บแอพพลิเคชั่นนั้นสามารถโจมตีได้หลายๆอย่างโดยมีสาเหตุมาจาก

1. ความผิดพลาดของผู้ดูแลระบบที่ติดตั้งและตั้งค่าระบบต่างๆ ไม่ดีพอ
2. ความผิดพลาดจากผู้เขียนซอฟต์แวร์ที่เกี่ยวข้องกับการทำงานเช่น โปรแกรมไมโครซอฟต์ไอไอเอส (Microsoft IIS)
3. ความผิดพลาดจากผู้เขียนเว็บแอพพลิเคชั่นเองที่ไม่ได้ตระหนักถึงการทำงานให้เกิดความปลอดภัยในระบบ

ตัวอย่างเทคนิคต่างๆ ที่ใช้ในการโจมตีเว็บแอพพลิเคชั่น

1. Hidden Field Manipulation
2. Cookie Poisoning
3. Backdoor and debug options
4. Application buffer overflows
5. Steath commanding
6. 3rd party misconfigurations
7. Known vulnerabilities
8. Parameter tempering
9. Cross site scripting
10. Forceful browsing

4.2.1 ฮิดเด็นฟิลด์ (Hidden Field)

สาเหตุเกิดจากเว็บแอพพลิเคชั่น ส่งข้อมูลส่วนหนึ่งไปเก็บไว้ที่ไคลเอนท์โดยใช้ ฮิดดั่นฟิลด์แล้วนำค่าดังกล่าวมาใช้งานอีกครั้งหนึ่งตอนโพรเซสเพื่อแสดงหน้าเพจถัดไป

ในการทำงานลักษณะนี้ผู้บุกรุกสามารถเปลี่ยนแปลงค่าฮิดดั่นฟิลด์เพื่อสร้างความเสียหาย เปลี่ยนแปลงการทำงานของเว็บแอพพลิเคชั่น หรือให้ได้ผลลัพธ์ที่ต้องการ

4.2.2 คุกกี้พอยซันนิง (Cookie Poisoning)

ข้อมูลเกี่ยวกับเซสชันของการเชื่อมต่อไปยังเว็บแอพพลิเคชั่น จะเก็บอยู่ในคุกกี้ (cookie) เมื่อมีการเปลี่ยนแปลงค่าในคุกกี้ย่อมสามารถเปลี่ยนแปลงเซสชันไอดี (Session ID) ของการเชื่อมต่อได้

สำหรับข้อมูลที่เก็บอยู่ในคุกกี้นั้นอาจไม่มีความปลอดภัย คือไม่มีการเข้ารหัส หรือเข้ารหัสไว้ไม่ดีพอ

4.2.3 แบ็คดอร์และดีบักออพชัน (Backdoor & Debug Options)

สำหรับแอพพลิเคชั่นที่พัฒนาขึ้นมาหลายๆแอพพลิเคชั่น จะมีการดีบัก (debug) การทำงานของแอพพลิเคชั่นโดยการป้อนพารามิเตอร์บางอย่างเข้าไปในระบบผ่านทางหน้าเว็บเพจ ซึ่งทำให้ผู้พัฒนาและผู้ดูแลระบบทราบค่าตัวแปรและการทำงานของแอพพลิเคชั่นนั้นๆได้ ทำให้ง่ายต่อการแก้ไขข้อผิดพลาด (bug) ต่างๆในระบบ โดยโค้ดส่วนดีบักนี้ผู้พัฒนาเว็บแอพพลิเคชั่นจะเป็นคนใส่ในระบบเอง นอกจากนี้ผู้พัฒนาระบบอาจสร้างแบ็กดอร์ เพื่อใช้เป็นช่องทางในการติดต่อเข้าไปในระบบด้วย

การทำงานในลักษณะนี้จะเป็นอีกช่องทางหนึ่งที่ผู้ไม่หวังดีสามารถเข้าไปใช้งานระบบได้โดยได้สิทธิในทรัพยากรทุกๆอย่างในระบบ ซึ่งปกติจะได้สิทธิสูงสุดในระบบ ในการพัฒนาเว็บแอพพลิเคชั่นจึงควรปิดโหมดการดีบัก (Disable debug mode) และไม่ควรมีแบ็กดอร์ในระบบด้วย

4.2.4 แอพพลิเคชันบัฟเฟอร์โอเวอร์โฟลว์ (Application Buffer Overflow)

การโจมตีเว็บแอพพลิเคชั่นในอีกรูปแบบหนึ่งคือการทำบัฟเฟอร์โอเวอร์โฟลว์ โดยจะทำตรงส่วนของเท็กท์บอกซ์ (Text Box) ที่รับข้อมูลจากผู้ใช้งานเว็บเพจนั้นๆ การโจมตีทำได้โดยการป้อนอินพุทปริมาณมากๆลงในช่อง หรือส่วนในการรับอินพุทจากหน้าเว็บเพจ เมื่อเว็บเพจนั้นส่งข้อมูลไปยังเซิร์ฟเวอร์แล้ว ข้อมูลที่มีขนาดมากกว่าที่กำหนดไว้จะไปทำให้แอพพลิเคชั่นหยุดการทำงานได้ การป้องกันก็คือที่ฝั่งเซิร์ฟเวอร์ควรมีการตรวจสอบขนาดของข้อมูลที่รับเข้ามาด้วย ไม่ให้เกินจากค่าที่กำหนดไว้

4.2.5 สเทลคอมมานดิ้ง (Stealth Commanding)

เป็นการโจมตีสู่เว็บเซิร์ฟเวอร์โดยการส่งคำสั่งการทำงานต่างๆ แนบไปกับข้อมูลต่างๆในระบบ โดยการทำลักษณะนี้ได้นั้นเกิดจากการที่เว็บแอพพลิเคชั่นคิดว่าข้อมูลที่ได้รับมานั้นเป็นเพียงข้อมูลที่ไม่สามารถเอ็กซีคิวได้ ความเสียหายที่อาจจะเกิดขึ้นได้กับระบบก็คือ การถูกเปลี่ยนหน้าเว็บเพจ การปิดบริการ หรือการขโมยข้อมูลจากเซิร์ฟเวอร์ เป็นต้น

4.2.6 เทิดปาร์ตี้มิสคอนฟิกกูเรชั่น (3rd Party Misconfiguration)

ความผิดพลาดอีกข้อหนึ่งที่ทำให้เกิดช่องโหว่ในระบบได้คือการตั้งค่าต่างๆ ในระบบไม่เหมาะสม หรือมีความผิดพลาดขณะติดตั้งโปรแกรม ซึ่งอาจทำให้เกิดปัญหาเช่น ยังมีการใช้รหัสผ่านที่โปรแกรมให้มา (Default Password) อยู่หรือค่าบางอย่างที่ทำให้เกิดความความปลอดภัยไม่ถูกเซ็ตไว้ ทำให้ผู้โจมตีระบบสามารถใช้ช่องโหว่นี้มาโจมตีระบบได้

4.2.7 โนลเวอร์เนอร์บิลิตี้ (Known Vulnerbilities)

ความไม่ปลอดภัยในลักษณะนี้เกิดจากจุดอ่อนในโปรแกรมที่เรานำมาใช้งาน ซึ่งโปรแกรมบางอย่างที่มีการใช้งานกันอย่างแพร่หลายก็อาจมีข้อผิดพลาดในโปรแกรมได้เช่นกัน เช่น ไมโครซอฟท์ไอไอเอส ซึ่งปัญหาที่เกิดขึ้นอาจเกิดในจุดเล็กๆ ในระบบแต่ทำให้ระบบเกิดความไม่ปลอดภัยขึ้นได้

การแก้ปัญหานี้จะต้องแพ็ทช์ (Patch) โปรแกรมที่มีปัญหา โดยการนำโปรแกรมสำหรับแก้ไขจุดอ่อนที่ออกโดยผู้พัฒนาแอพพลิเคชั่นนั้นๆ ซึ่งปกติแล้วจะมีการออกโปรแกรมแก้ไขออกมาอย่างรวดเร็ว แต่ปัญหาอยู่ที่ผู้ที่พัฒนาแอพพลิเคชั่นยกภาระการแก้ไขจุดอ่อนในระบบให้กับผู้ดูแลระบบเอง ซึ่งทำให้ช้ามากและมักจะไม่ทันการ

อันตรายจากปัญหาลักษณะนี้มักลุกลามอย่างรวดเร็ว โดยเฉพาะอย่างยิ่งเมื่อเกิดกับโปรแกรมที่มีการใช้งานกันอย่างแพร่หลายเช่น ไมโครซอฟท์ ไอไอเอส โดยสาเหตุที่ทำให้การลุกลามเป็นไปอย่างรวดเร็วนั้นจะเกิดจากผู้ดูแลระบบไม่ได้ติดตามข่าวจุดอ่อนของระบบที่ตัวเองดูแลอยู่อย่างสม่ำเสมอ แต่ผู้ไม่หวังดีกลับติดตามข่าวเหล่านี้อยู่เสมอๆ เมื่อมีข้อผิดพลาดของโปรแกรมเกิดขึ้น ก็จะมีผู้ประกาศตามหน้าเว็บไซต์ต่างๆทางด้านความปลอดภัยต่างๆ ซึ่งผู้ที่ทราบก่อนมักจะเป็นผู้ไม่หวังดีมากกว่าผู้ดูแลระบบ

4.2.8 พารามิเตอร์เทมเพอร์ริง (Parameter Tempering)

ปัญหานี้เกิดจากการที่เว็บแอพพลิเคชั่นใช้ค่าพารามิเตอร์จากไคลเอนต์ซึ่งการเปลี่ยนค่าพารามิเตอร์นั้นสามารถทำได้ง่ายดาย ซึ่งผู้เขียนเว็บแอพพลิเคชั่นนั้น มักจะคิดว่าค่าต่างๆจะเป็นค่าที่ถูกต้องแล้ว จะมีน้อยคนนักที่จะคำนึงถึงการเปลี่ยนแปลงพารามิเตอร์จะทำให้ระบบมีปัญหาอย่างไร

ปัญหาที่เกิดขึ้นจากการเปลี่ยนแปลงค่าพารามิเตอร์ไปเป็นค่าที่ไม่ถูกต้องนั้น ทำให้เกิดความไม่ปลอดภัยในข้อมูลหลายๆอย่าง เช่น การดึงข้อมูลของลูกค้าคนอื่นๆได้ การเปลี่ยนสิทธิของตนเองไปเป็นของคนอื่นๆเพื่อดึงข้อมูลส่วนตัวของคนอื่นๆเป็นต้น

4.2.9 ครอสไซต์สคริปท์ (Cross Site Script)

ครอสไซต์สคริปท์ เป็นกระบวนการหนึ่งที่อาศัยจุดอ่อนของโฮสที่ไม่มีการตรวจสอบว่าพารามิเตอร์ที่ป้อนเข้ามานั้นคือพารามิเตอร์จากไคลเอนต์จริงหรือไม่ จากจุดอ่อนดังกล่าวทำให้ผู้ไม่หวังดีสามารถโจมตีระบบได้โดยการฝากสคริปท์ไปรันที่เครื่องเป้าหมาย โดยผู้ใช้งานฝั่งไคลเอนต์เป็นผู้นำพาสคริปท์ไปยังเครื่องเป้าหมายได้

โดยการโจมตีจะมีการสร้างลิ้งค์ที่เว็บบอร์ดหรือหน้าเว็บเพจต่างๆ แล้วป้อนพารามิเตอร์เป็นจาวาสคริปท์ (javascript) รอให้คนอื่นๆติดต่อจากเครื่องไคลเอนต์ทำการคลิก แล้วสคริปท์จะทำงานทันที โดยความสามารถของสคริปท์นั้นจะมีตั้งแต่การขโมยเล็กๆน้อยๆ ไปจนถึงการขโมย session หรือข้อมูล Username และ Password ภายในเครื่องเป้าหมาย

4.2.10 ฟอร์ซฟูลบราวซิ่ง (Forceful Browsing)

การโจมตีลักษณะนี้ผู้ไม่หวังดีจะใช้การคาดเดาว่าข้อมูลนั้นๆ อยู่ในไดเรกทอรีไหน แล้วป้อนตำแหน่งของข้อมูลนั้นๆ โดยตรงซึ่งปัญหานี้เกิดจากการใช้ดีฟอลต์ไฟล์ (Default File) ขณะติดตั้งโปรแกรม และไม่มีการลบไฟล์ที่ไม่ใช้ออกไปจากระบบ

ผลที่เกิดจากปัญหานี้คือข้อมูลต่างๆเช่น ล็อกไฟล์ โค้ดต้นแบบของโปรแกรมต่างๆ อาจถูกขโมยไปได้ถ้าเปิดสิทธิให้สามารถอ่านไฟล์หรือไดเร็กทอรี่นั้นๆได้

และนอกจากการโจมตีเว็บแอพพลิเคชั่นทั้ง 10 รูปแบบที่ได้กล่าวมาแล้วยังมีการโจมตีเว็บแอพพลิเคชั่นในลักษณะประยุกต์ (Advanced Attack) ดังนี้

4.2.11 ซอร์สโค้ดดิสโคล้ดเชอร์ (Source Code Disclosures)

การทำซอร์สโค้ดดิสโคล้ดเชอร์ เป็นการใช้จุดอ่อนในการออกแบบแอพพลิเคชั่น ทำให้ผู้บุกรุกสามารถดึงข้อมูลของคอนฟิกกูเรชั่นไฟล์ (Configuration file) หรือข้อมูลอื่นๆได้

ตัวอย่างของการทำซอร์สโค้ดดิสโคล้ดเชอร์

- ข้อผิดพลาดใน WebLogic / WebSpere โดยข้อผิดพลาดนี้ผู้บุกรุกสามารถดึงข้อมูลของไฟล์นามสกุลเจเอสพี (JSP) และ เจเอชทีเอ็มแอล (JHTML) ได้ ซึ่งเกิดจากการตั้งค่าในเว็บเซิร์ฟเวอร์ผิดพลาด โดยการดึงข้อมูลนั้นสามารถทำได้โดยการเปลี่ยนตัวอักษร "jsp" ในยูอาร์แอล (URL) ให้กลายเป็นตัวพิมพ์ใหญ่ จะทำให้เซิร์ฟเวอร์ส่งรายละเอียดในไฟล์ .jsp มาแทนผลลัพธ์ในการทำงานของไฟล์ .jsp นั้น
- การทำซอร์สโค้ดดิสโคล้ดเชอร์จะเป็นบักในไมโครซอฟท์ไอไอเอส ซึ่งมีปัญหากับไฟล์ ".HTR" โดยผู้บุกรุกสามารถดูรายละเอียดในไฟล์นามสกุล .ASA และ .ASP ได้ ยกตัวอย่างยูอาร์แอลที่ทำให้เกิดปัญหาคือ

  http://10.0.0.1/global.asa+.htr

  เมื่อเซิร์ฟเวอร์ได้รับการร้องขอยูอาร์แอลดังกล่าวแล้วจะทำงานโดย .htr จะทำให้ ISM.DLL ทำงานกับยูอาร์แอลดังกล่าวและเครื่องหมาย + จะทำให้ ISM.DLL ไม่ประมวลผล ตัวอักษรหลังเครื่องหมาย + นั้น
- ปัญหาของ Microsoft IIS showcode.asp ซึ่งเป็นโปรแกรมในการดูรายละเอียดในโค้ดของไฟล์ต่างๆได้ โดย showcode.asp จะถูกรวมอยู่กับไอไอเอสของ Windows NT Option Pack 4.0 โดยผู้บุกรุกที่ต้องการดูข้อมูลของไฟล์ต่างๆในระบบสามารถดูได้จากการป้อนยูอาร์แอล เช่น

  http://10.0.0.1/msadc/showcode.asp?Source=/msadc/../../../../../path/to/file.name

4.2.12 การโจมตีทางสถาปัตยกรรมของเว็บเซิร์ฟเวอร์ (Web Server Architecture Attack)

ในบางครั้งก็มีปัญหาในการออกแบบสถาปัตยกรรมของเว็บเซิร์ฟเวอร์ ทำให้เกิดช่องโหว่ขณะใช้งานเว็บเซิร์ฟเวอร์ได้ การโจมตีช่องโหว่ทางสถาปัตยกรรมของเว็บเซิร์ฟเวอร์นี้ จะให้วิธีการบายพาส (bypass) การทำงานบางส่วนของเว็บเซิร์ฟเวอร์ แล้วไปใช้งานบิวท์อินโพรซีเยอร์ (built-in procedure) ของเว็บเซิร์ฟเวอร์โดยตรง การแก้ปัญหานี้สามารถทำได้โดยการตรวจสอบสถาปัตยกรรมของเว็บเซิร์ฟเวอร์ให้ละเอียดเพื่อตรวจสอบดูว่าจะมีการทำงานที่นอกเหนือจากการทำงานปกติเกิดขึ้นในกรณีไหนบ้าง แล้วทำการแก้ไข

ในสถาปัตยกรรมเว็บเซิร์ฟเวอร์นั้นจะมีการตั้งค่าให้แฮนด์เลอร์ (handler) ต่างๆรับผิดชอบการประมวลผลไฟล์ต่างๆในระบบเมื่อถูกร้องขอเช่น เอชทีเอ็มแอลแฮนด์เลอร์ (html handler) จะทำงานเกี่ยวกับการรับส่งข้อมูลภายในไฟล์เอชทีเอ็มแอล ไปยังเครื่องที่ร้องขอ แต่ซีจีไอแฮนด์เลอร์ (cgi handler)จะรับผิดชอบในการเรียกให้ซีจีไอทำงานแต่ในบางกรณีจะมี default handler สำหรับการทำงานกับข้อมูลที่อยู่นอกเหนือหน้าที่ของแฮนด์เลอร์อื่นๆ ซึ่งอาจทำให้ผู้บุกรุกสามารถเรียกใช้ default handler นี้เพื่อทำการอ่านไฟล์ซีจีไอขึ้นมาแสดงผลได้ หรืออาจส่งค่าไฟล์เอชทีเอ็มแอลไปยังเจเอสพีแฮนด์เลอร์ (jsp handler) ทำให้ระบบคอมไพล์ไฟล์เอชทีเอ็มแอล โดยจาวาคอมไพเลอร์และเอ็กซีคิวต์โดยจาวารันทาม (java run-time) ในกรณีนี้ก็ทำให้ผู้บุกรุกสามารถมีการทำงานบางอย่างในระบบได้ ดังตัวอย่างการทำแฮนด์เลอร์ฟอร์สซิ่ง

(handler forcing) ในเครื่องเว็บเซิร์ฟเวอร์ของซัน (Sun Java Web Server) โดยผู้บุกรุกสามารถป้อนยูอาร์แอลดังตัวอย่าง

http://10.0.0.2/servlet/com.sun.server.http.pagecompile.jsp.runtime.JspServlet/path/to/file.html

โดยจะมีการเรียกให้เซิร์ฟเลท (servlet) ทำงานจากการป้อนพาร์ท (path) /servlet/ แล้วเรียก PageCompile handler (Servlet) มาแฮนด์เดิล (handle) ไฟล์ข้อมูลข้างหลัง แล้วป้อนพาร์ทไปยังไฟล์ของข้อมูลที่ต้องการให้แฮนด์เดิลซึ่งในทางปฏิบัติผู้บุกรุกอาจจะป้อนข้อมูลที่เป็นโปรแกรมสำหรับการเชื่อมต่อทางไกลส่งไปให้จาวารันทามเป็นตัวเอ็กซีคิวต์ แล้วเปิดพอร์ตขึ้นรอรับการเชื่อมต่อจากผู้บุกรุกสำหรับการทำในลักษณะนี้จะทำให้การเชื่อมต่อนั้นมีสิทธิเทียบเท่าผู้ดูแลระบบทันที

4.2.13 เอสคิวแอลพอยซันนิ่งแอนด์อินเจ็คชั่น (SQL Poisoning & Injections)

เป็นการโจมตีโดยใช้จุดอ่อนของการเขียนแอพพลิเคชั่น ที่มีการใช้งานเอสคิวแอลสเตตเมนต์ (sql statement) โดยรับข้อมูลจากไคลเอนท์แต่ไม่ได้ตรวจสอบก่อนว่าข้อมูลที่รับเข้ามานั้นถูกต้องหรือไม่ซึ่งเอสคิวแอลสเตตเมนต์ จะเชื่อมต่อไปยังดีบีเอ็มเอสโดยตรง (ผ่าน SQL Query) ทำให้ผู้บุกรุกสามารถเพิ่มเติมและเปลี่ยนแปลงเอสคิวแอลสเตตเมนต์เพื่อให้ทำงานอื่นได้ ยกตัวอย่างโค้ดในการดึงข้อมูลฐานข้อมูล คือ

```
Dim sql_con, result, sql_qry
Const CONNECT_STRING =
"Provider=SQLOLDEB;SERVER=WEB_DB;UID=sa;PWD=xyzzy"
sql_qry= "SELECT * FROM PRODUCT WHERE ID ="
        & Request.QueryString("DB")
Set objCon = Server.CreateObject("ADODB.Connection")
ObjCon.Open CONNECT_STRING
Set objRS – objCon.Execute(strSQL);
```

จากตัวอย่างโค้ดที่อยู่ในเว็บแอพพลิเคชันนั้น จะเห็นได้ว่าไม่มีการตรวจสอบค่าของอินพุทที่รับเข้ามาเลย ดังนั้นถ้ามีการร้องขอในลักษณะ

http://10.0.0.3/showtable.asp?ID=3+OR+1=1

ผลลัพธ์เมื่อโปรแกรมทำงานในคิวรี สเตตเมนต์ (Query statement) คือ

SELECT * FROM PRODUCT WHERE ID = 3 OR 1=1

ซึ่งจะผลให้มีการทำงานคำสั่งต่อไปนี้ที่แอพพลิเคชันเซิร์ฟเวอร์

SELECT * FROM PRODUCT WHERE ID = 3

DROP TABLE PRODUCT

นอกจากจะสามารถส่งคำสั่งเพื่อทำงานกับเอสคิวแอลสเตตเมนต์ได้แล้วยังสามารถส่งคำสั่งเพื่อการทำงานอื่นๆ ได้ด้วยเช่นกันยกตัวอย่าง

http://10.0.0.3/showtable.asp?ID=3%01EXEC+master..xp_cmdshell+'copy+\winnt\system32\cmd.exe+\inetpub\scripts'

ซึ่งจะส่งคำสั่งไปทำงานที่ฝั่งเซิร์ฟเวอร์คือ

Copy \winnt\system32\winnt\cmd.exe \inetpub\scripts

4.2.14 ไมโครซอฟท์ไอไอเอสยูนิโค้ดบัก (Microsoft IIS Unicode bug)

สำหรับบักที่สร้างความเสียหายต่อองค์กรธุรกิจอย่างมาก เห็นจะไม่พ้นยูนิโค้ดบัก (Unicode bug) ในไมโครซอฟท์ไอไอเอส ซึ่งช่องโหว่นี้ทำให้ผู้บุกรุกสามารถส่งคำสั่งต่างๆ ไปทำงานยังเซิร์ฟเวอร์ได้อย่างง่ายดาย เพียงแค่ส่ง URL ที่มียูนิโค้ดที่มีปัญหาเข้าสู่ระบบ แล้วให้ระบบรับคำสั่งจาก URL ไปทำงานเช่น

http://10.0.0.3/scripts/..%c0%af../winnt/system32/cmd.exe?c+dir

ซึ่งจะส่งคำสั่ง dir ไปทำงานที่เว็บเซิร์ฟเวอร์ และส่งผลลัพธ์การทำงานมาที่หน้าจอบราวเซอร์

บทที่ 5
โครงสร้างและการทำงานของไอพีเทเบิลส์

ลีนุกซ์สามารถใช้งานเป็นไฟร์วอลล์ได้ตั้งแต่เคอร์เนล 1.1 ซึ่งเป็นเวอร์ชันแรก โดยอลัน ค็อกซ์ (Alan Cox) ใช้ชื่อว่า ipfw (จาก BSD) ต่อมา ลีนุกซ์ 2.0 ได้ถูกพัฒนาและปรับปรุงได้เครื่องมือที่มีชื่อว่า ipfwadm โดยเครื่องมือชิ้นนี้อนุญาตให้ผู้ใช้สามารถควบคุมกฎการกรองแพ็กเก็ตได้ และต่อมา ลีนุกซ์ 2.2 ก็ได้สร้างเครื่องมือตัวใหม่ชื่อไอพีเชนส์ (ipchains) ซึ่งเผยแพร่ในปี 1998 โดยรัสตี้ รุสเซล (Rusty Russel) และทีมงาน ทั้งนี้ไอพีเชนส์นี้ถือได้ว่าเป็นพัฒนาการขั้นที่สามของลีนุกซ์ไฟร์วอล จวบจนกระทั่งในปัจจุบัน ก็มีเน็ทฟิลเตอร์และ ไอพีเทเบิลส์ซึ่งถือว่าเป็นพัฒนาการขั้นที่สี่ของ ลีนุกซ์ ไฟร์วอล

เน็ทฟิลเตอร์นั้นเป็นชื่อใหม่ของโค้ดที่ทำหน้าที่เป็น แพ็กเก็ต handler (stateful inspection) ใน ลีนุกซ์ เคอร์เนล 2.4 (ที่จริงคือเวอร์ชัน 2.3.15 และเวอร์ชันต่อๆ มา) ซึ่งได้ถูกออกแบบและปรับปรุงใหม่จากเวอร์ชันก่อนหน้านี้ เป็นเรื่องที่น่ายินดีคือเน็ทฟิลเตอร์นั้นสามารถทำงานย้อนหลังร่วมกับไอพีเชนและ ipfwadm ได้ และคำสั่งในการเรียกใช้งานคือไอพีเทเบิลส์

ความแตกต่างระหว่างไอพีเทเบิลส์ และ ไอพีเชนส์

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

ดาวน์โหลด

ก่อนที่ใช้งานไฟร์วอลล์ตัวใหม่นี้ได้ ให้ล็อกอินเป็น root และทดลองพิมพ์คำว่า iptables ดูว่าคำสั่งนี้มีอยู่หรือไม่ ถ้าไม่มี ต้องดาวน์โหลดจาก http://www.netfilter.org โดยให้ดาวน์โหลดเวอร์ชันล่าสุดมา ในที่นี้คือ iptables-1.2.9.tar.bz2 จากนั้นให้ขยายไฟล์และติดตั้งดังคำสั่งด้านล่าง

tar -jxvf iptables-1.2.9.tar.bz2

cd iptables-1.2.9

make

make install

ปรับแต่งเคอร์เนล

หลังจากที่มีคำสั่ง iptables อยู่ในระบบแล้ว บางระบบที่มีเคอร์เนลต่ำกว่า 2.4 จำเป็นต้องคอมไพล์เคอร์เนลใหม่ เพื่อให้ใช้งานiptablesได้ ทั้งนี้โครงงานนี้ได้พัฒนาโดยใช้เคอร์เนลเวอร์ชั่น 2.4.23 ปัจจุบันซึ่งสามารถดาวน์โหลดได้จาก http://www.kernel.org

หลังจากที่ดาวน์โหลดมาแล้วให้ขยายไฟล์ไปไว้ที่ /usr/src/linux จากนั้นใช้คำสั่ง make menuconfig หรืออาจใช้ make xconfig ในกรณีที่ได้ติดตั้ง XWindows ไว้แล้ว โดยต้องมั่นใจว่าได้ enable option ต่างๆ ด้านล่างนี้ ภายใต้ Networking options

<*> Packet socket

[*] Network packet filtering (replaces ipchains)

<*> Unix domain sockets

[*] TCP/IP networking

[*] IP: advanced router

[*] IP: policy routing

[*] IP: use netfilter MARK value as routing key

[*] IP: fast network address translation

[*] IP: use TOS value as routing key

และภายใต้เมนู "IP: Netfilter Configuration -- >" ให้ enable ทุกออปชันเพื่อให้ใช้งาน netfilter ได้เต็มประสิทธิภาพ

แต่มีอีกจุดหนึ่งที่ห้าม enable ภายใต้ "Networking options" คือ

[ ] IP: TCP Explicit Congestion Notification support

เพราะออปชันนี้ จะทำให้บาง แพ็กเก็ต ที่ออกจาก ลีนุกซ์ ถูกเซ็ตบิต ECN ไปด้วย ซึ่งจะทำให้ไม่สามารถสื่อสารกับ internet router บางตัวได้

จากนั้นก็ให้คอมไพล์และติดตั้งเคอร์เนลโดยใช้คำสั่งดังนี้

คำสั่งที่ใช้ในการคอมไพล์เคอร์เนล

make dep

make clean

make bzImage

make modules

#รันคำสั่งนี้ในกรณีที่คอมไพล์เคอร์เนลแบบเลือก module ด้วย

make modules_install

#รันคำสั่งนี้ในกรณีที่คอมไพล์เคอร์เนลแบบเลือก module ด้วย

cp /usr/src/linux/arch/i386/boot/bzImage /boot/bzimage-2.4.23

#การ copy bzImage นั้น ขึ้นอยู่กับสถาปัตยกรรมของเครื่องคอมพิวเตอร์นั้นๆ ซึ่งในที่นี้ใช้ i386

cp /usr/src/linux/System.map /boot/System.map-2.4.23

ln -s /boot/System.map-2.4.23 .boot/System.map

#ถ้าได้รับ error ให้ลบไฟล์ /boot/System.map ทิ้งก่อน

จากนั้นให้แก้ไข /etc/lilo.conf เพื่อเพิ่ม configuration ใหม่ ตัวอย่างด้านล่างแสดงเรดแฮท (Red Hat) ลีนุกซ์ ที่มีเคอร์เนลเดิมเป็น 2.4.18 และเคอร์เนลใหม่เป็น 2.4.23

boot=/dev/hda

map=/boot/map

install=/boot/boot.b

prompt

timeout=50

message=/boot/message

default=linux-2.4.23

image=/boot/vmlinuz-2.4.23

label=linux-2.4.23

read-only

root=/dev/hda1

image=/boot/vmlinuz-2.4.18

label=linux-2.4.18

read-only

root=/dev/hda1

หลังจากนั้นรันคำสั่ง lilo -v และรีบูตใหม่ ถ้าไม่มีข้อผิดพลาดใดๆ ก็จะได้เคอร์เนลของ ลีนุกซ์ เวอร์ชันล่าสุดที่สนับสนุนการทำงานของ iptables อย่างเต็มที่

5.1 รูปแบบการใช้งาน iptables เบื้องต้น

iptables จะมีรูปแบบการใช้งานดังนี้คือ

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

โดย กฎ ที่เขียนขึ้นจะเป็นเป็นตัวบอกเคอร์เนลว่าให้กระทำ 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 กฎ 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 ถ้าแพ็กเก็ต 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 เป็นตารางที่ใช้เปลี่ยนแปลงหรือแก้ไข แพ็กเก็ต เช่น เปลี่ยนค่า TTL, MARK ซึ่งปกติจะใช้ในการทำ routing ที่มีความซับซ้อนสูง มี 2 built-in chain คือ PREROUTING chain (ใช้แก้ไข แพ็กเก็ต ก่อนที่จะเข้าสู่ไฟร์วอลล์และก่อนเข้าสู่ routing decision) และ OUTPUT chain (ใช้แก้ไข แพ็กเก็ต ที่ถูกสร้างโดยไฟร์วอลล์ก่อนที่มันจะถูกส่งไปยัง routing decision) ทั้งนี้ไม่สามารถทำ network address translation หรือ masquerading ที่ table นี้ได้ และโครงงานนี้ไม่ได้ศึกษาในส่วนนี้

Command

- -A เพิ่ม กฎ ใหม่ต่อท้าย chain (Append rule) เช่น
  # iptables -A INPUT -p ALL -i eth0 -j ACCEPT

- -D ลบ กฎ (Delete rule) เช่น
  # iptables -D INPUT --dport 80 -j DROP

- -I เพิ่ม กฎ ใหม่ ใน chain (Insert rule) เช่น
  # iptables -I OUTPUT -p ALL -s 127.0.0.1/32 -j ACCEPT

- -R แทนที่ กฎ เดิม ด้วย กฎ ใหม่ (Replace rule)

- -L แสดง กฎ ทั้งหมดใน chain (ถ้าไม่ระบุ chain จะแสดง กฎ ทั้งหมดใน filter table ทั้งสาม built-in chain) เช่น
  # iptables -L
  # iptables -L -t nat
  # iptables -L INPUT

- -F ลบ กฎ ทั้งหมดใน chain ทิ้ง เช่น
  # iptables -F INPUT
  # iptables -F mychain

- -Z ใช้ reset byte counter สำหรับทุก กฎ ใน chain ที่กำหนด เช่น
  # iptables -Z INPUT

- -N ใช้สร้าง chain ใหม่ เช่น
  # iptables -N mychain

- -X ลบ chain ที่ไม่มี กฎ ซึ่งสามารถลบ user-defined chain ที่ไม่มี กฎ 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 กฎ ใดๆ เลย แพ็กเก็ต นั้นก็ต้องถูกตัดสินใจโดย policy ของ chain นั้นๆ เช่น
  # iptables -P FORWARD DROP

ซึ่งหาก แพ็กเก็ต ถูกส่งเข้ามายัง FORWARD chain และไม่ match กับ กฎ 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 กฎ 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 แสดงข้อมูลไอพีแอดเดรสและ พอร์ต เป็นตัวเลขเท่านั้น เช่น
  # iptables -L OUTPUT -n
- --line-numbers ใช้ร่วมกับ -L เพื่อแสดงเลขบรรทัดของ กฎ ซึ่งตัวเลขที่แสดงนี้จะสามารถใช้ได้กับคำสั่ง แทรก กฎ ที่ระบุเป็นลำดับที่ของ กฎ เช่น
  # iptables -L --line-numbers
- --modprobe=*command* เพื่อโหลด module ที่เกี่ยวข้อง

5.2 ฟังก์ชั่นแมทช์ (Match)

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

- การระบุ source, destination IP address
  สามารถระบุ 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.ce.kmitl.ac.th
  2. ระบุไอพีแอดเดรสโดยตรง เช่น 127.0.0.1 หรือ 161.246.5.14

3. ระบุเป็น group ของไอพีแอดเดรส เช่น 161.246.5/24 ซึ่งหมายถึงไอพีแอดเดรสตั้งแต่ 161.246.5.0 – 161.246.5.255
4. หรืออาจจะใช้ 161.246.5.0/255.255.255.0 แทน 161.246.5.0/24 ได้

- การทำ Inversion
  ในบางกรณีนั้นหากต้องการระบุเป็น inverse เช่น อนุญาตให้ทุกไอพียกเว้นไอพีที่ระบุไว้ ซึ่งการใช้คำสั่งดังกล่าวสามารถทำได้โดยใช้เครื่องหมาย ! นำหน้า argument ที่ต้องการ (เครื่องหมาย ! หมายถึง NOT) เช่น -p ! TCP 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, ไอซีเอ็มพี หรือสามารถใช้ตัวเลขแทนได้ (สำหรับ *NIX อ้างอิงได้จาก /etc/protocols) และยังสามารถใช้ได้ทั้งตัวอักษรเล็กหรือใหญ่ (ใช้ได้ทั้ง tcp และ TCP) เช่น -p TCP หรือ -p ! tcp

- การระบุ interface
  -i หรือ --in-interface ตามด้วยชื่อ interface ใช้เพื่อระบุ incoming interface ซึ่งหมายถึงว่า แพ็กเก็ต ที่จะ match กับ กฎ นี้ต้องเข้ามาจาก interface ที่กำหนด เช่น -i eth0 หมายความว่า ทุก แพ็กเก็ต ที่เข้ามาทาง eth0 จะ match กับ กฎ นี้ ทั้งนี้ชื่อ interface ที่สามารถใช้ได้นั้น สามารถตรวจสอบได้โดยใช้คำสั่ง ifconfig

  และ -o หรือ --out-interface ตามด้วยชื่อของ interface ใช้เพื่อระบุ outgoing interface ซึ่งหมายถึงว่า แพ็กเก็ต ที่จะ match กับ กฎ นี้ กำลังจะเดินทางผ่าน interface ที่ระบุไว้ เช่น -o eth1 หรือ -o ! eth1

  ข้อสังเกต
  - สำหรับ INPUT chain นั้นไม่มี output interface ดังนั้นหากใช้ -o ร่วมกับ INPUT chain ก็จะไม่มี แพ็กเก็ต ใดที่ match กับ กฎ นี้เลย
  - ทำนองเดียวกันกับ OUTPUT chain ที่ไม่มี input interface ดังนั้นหากใช้ -i ร่วมกับ OUTPUT chain ก็ไม่มีประโยชน์อันใด
  - FORWARD chain มีได้ทั้ง input และ output interface
  - หากระบุ interface ที่ไม่มีอยู่จริง ก็จะไม่มี แพ็กเก็ต ใดที่ match กับ กฎ นั้นเลย

- หากใช้เครื่องหมาย + ร่วมกับ interface เช่น ppp+ นั้นจะหมายถึงทุกๆ ppp interface เช่น ppp0, ppp1

- fragment packet

  ในการส่งข้อมูลใน ip network นั้นเป็นเรื่องปกติที่จะเกิดการ fragment ของ แพ็กเก็ต เนื่องจากขนาดของ แพ็กเก็ต มีขนาดใหญ่เกินไปที่จะส่งไปในครั้งเดียว จำเป็นต้องมีการแบ่ง แพ็กเก็ตออกเป็นหลายๆ ชิ้นทยอยส่งไป ซึ่งเรียกกันว่าการทำ fragment โดยเครื่องปลายทางจะทำหน้าที่ประกอบ fragment packet รวมกันเป็น แพ็กเก็ต ที่สมบูรณ์ดังเดิม

ข้อมูลที่เป็น fragment packet นั้นจะมี header ที่สมบูรณ์แค่ แพ็กเก็ต แรกเท่านั้น ตัว แพ็กเก็ต ที่ตามมาจะมีแค่ header บางส่วนคือ ไอพีแอดเดรสเท่านั้น ไม่มีข้อมูลของโพรโตคอลแนบมาด้วย ดังนั้นการตรวจสอบข้อมูล header ของ ทีซีพี, ยูดีพี, ไอซีเอ็มพี จึงไม่สามารถทำได้ใน แพ็กเก็ต ที่สองเป็นต้นมา

หากใช้ NAT บรรดา fragment packet จะถูกประกอบเข้าด้วยกันจนสมบูรณ์ก่อนที่ แพ็กเก็ต จะเข้าไปถึง packet filtering ดังนั้นจึงไม่มีความจำเป็นที่จะต้องกังวลเกี่ยวกับ fragment packet

ดังนั้นถ้าไม่ได้ใช้ NAT ก็ควรทำความเข้าใจไว้ว่า iptables มีกระบวนการในการทำงานกับ fragment packet อย่างไร หลังจากที่ fragment packet แรกผ่านเข้ามาแล้ว iptables สามารถตรวจสอบได้ว่าจะอนุญาตให้ผ่านหรือไม่ ในขณะที่ fragment packet ที่สองและหลังจากนั้นที่ตามมานั้น จะไม่สามารถ match กับ กฎ ใดๆ เลย เช่น -p TCP --sport www หรือแม้แต่ -p TCP --sport ! www

อย่างไรก็ตาม สามารถเขียน กฎ ให้ตรวจสอบทั้ง fragment packet ตัวที่สองและหลังจากนั้นที่ตามมาได้ด้วยการใช้ -f หรือ --fragment ทั้งนี้อาจจะเขียนในทางตรงข้ามคือไม่ต้องตรวจสอบ fragment packet ที่สองและหลังจากนั้น โดยใช้ ! -f ก็ได้

ทั้งนี้โดยปกติแล้วมักจะปล่อยให้ fragment packet ผ่านไป เนื่องจากถ้าสามารถ DROP ตัว fragment packet ตัวแรกได้แล้ว มันก็ไม่สามารถถูกประกอบที่เครื่องปลายทางได้ แต่ทั้งนี้ fragment packet ที่ถูกปล่อยไปดังกล่าวอาจจะทำให้เครื่องที่ได้รับ hang หรือ crash ได้ หรืออาจจะเกิดการโจมตีแบบ Denial of Service โดยใช้ fragment packet ได้

- TCP extension

  ถ้ามีการเรียกใช้ -p tcp ตัว TCP extension ก็จะถูกโหลดมาใช้งานโดยอัตโนมัติ โดยมีออปชั่นให้เลือกใช้งานดังนี้

  - --tcp-flags *mask flags* : mask นั้นหมายถึง flag ที่ต้องการตรวจสอบ และ flag เป็นตัวที่บ่งชี้ว่า flag ใดต้องถูก set บ้าง

    เช่น # iptables -A INPUT -p tcp --tcp-flags ALL SYN,ACK -j DROP

    โดย ALL นั้นหมายถึงทุกๆ flag (SYN,ACK,FIN,RST,URG,PSH) และถ้า flag

SYN,ACK ถูก set พร้อมกันก็ให้ drop packet นั้นทิ้งไป นอกจากนี้ยังสามารถใช้ NONE ซึ่งหมายถึงไม่มี flag ใดถูก set ได้

- --syn เป็นสัญลักษณ์ย่อของ --tcp-flags SYN,RST,ACK SYN
- --source-port หรือ --sport สามารถใช้ได้ทั้งตัวเลขและตัวอักษร (อ้างอิงจากไฟล์ /etc/services) และระบุเป็น พอร์ต เดียว หรือช่วงของ พอร์ต ได้
  เช่น --sport 21:25 หมายถึง พอร์ต 21 - 25 , --sport 25: หมายถึงพอร์ตที่มากกว่าหรือเท่ากับ 25 , --sport :25 หมายถึงพอร์ตที่น้อยกว่าหรือเท่ากับ 25
- --destination-port หรือ --dport มีรูปแบบการใช้งานเช่นเดียวกันกับ --sport
- --tcp-option ใช้ตรวจสอบ TCP option ว่าตรงกับเลขที่ระบุไว้หรือไม่

- อธิบาย flag ของ ทีซีพี เพิ่มเติม
  การเชื่อมต่อโดยใช้ ทีซีพี นั้น ผู้ที่เริ่มสร้าง connection จะเป็นผู้ส่ง SYN แพ็กเก็ต มายังเครื่องปลายทาง ดังนั้นหากไม่ต้องการให้ให้เครื่องใดเป็นผู้เริ่มสร้างการติดต่อก็สามารถ block ไอพี ดังกล่าวได้ โดยใช้ --syn เช่น -p TCP -s x.x.x.x --syn หากยังไม่เข้าใจรูปแบบการเชื่อมต่อแบบ ทีซีพี นี้แล้ว ก็เป็นการยากที่จะสร้าง กฎ สำหรับ iptables ดังนั้นจึงขอแนะนำให้ไปศึกษาหลักการทำงานเบื้องต้นของทั้ง ทีซีพี, ยูดีพี, ไอซีเอ็มพี มาก่อน

- UDP extension
  คล้ายกันกับ ทีซีพี ตัว ยูดีพี extension มีออปชันให้เลือกใช้เพียงแค่ 2 อย่างเท่านั้นคือ --source-port (--sport) และ --destination-port (--dport) โดยต้องระบุ -p udp ด้วย

- ICMP extension
  โดยการระบุ -p icmp ก็สามารถใช้งาน ICMP extension ได้ โดยมีออปชันให้เลือกคือ --icmp-type เช่น --icmp-type host-unreachable (หรือใช้เลข 3 แทนได้) นอกจากนี้ยัง สามารถระบุ type/code ได้ เช่น 3/3 ซึ่งหมายถึง port unreachable

TYPE	CODE	Description	Query	Error
0	0	Echo Reply	x	
3	0	Network Unreachable		x
3	1	Host Unreachable		
3	2	Protocol Unreachable		
3	3	Port Unreachable		
3	4	Fragmentation needed but no frag. bit set		

TYPE	CODE	Description	Query	Error
3	5	Source routing failed		
3	6	Destination network unknown		
3	7	Destination host unknown		
3	8	Source host isolated (obsolete)		
3	9	Destination network administratively prohibited		
3	10	Destination host administratively prohibited		
3	11	Network unreachable for TOS		
3	12	Host unreachable for TOS		
3	13	Communication administratively prohibited by filtering		
3	14	Host precedence violation		
3	15	Precedence cutoff in effect		
4	0	Source quelch		
5	0	Redirect for network		
5	1	Redirect for host		
5	2	Redirect for TOS and network		
5	3	Redirect for TOS and host		
8	0	Echo request	x	
9	0	Router advertisement		
10	0	Route sollicitation		
11	0	TTL equals 0 during transit		x
11	1	TTL equals 0 during reassembly		
12	0	IP header bad (catchall error)		
12	1	Required options missing		
13	0	Timestamp request (obsolete)		

TYPE	CODE	Description	Query	Error
14	0	Timestamp reply (obsolete)		
15	0	Information request (obsolete)		
16	0	Information reply (obsolete)		
17	0	Address mask request		
18	0	Address mask reply		

ตารางที่ 5-1 แสดงรายละเอียดของ ICMP message

- Match Extension

  เป็น netfilter package ที่อยู่ในช่วงทดลองใช้ รูปแบบการใช้งานให้ใช้ -m แล้วตามด้วย match ที่ต้องการ เช่น -m mac ทั้งนี้มีออปชันให้เลือกใช้งานดังต่อไปนี้

  - mac

    รูปแบบการใช้งาน: -m mac หรือ --match mac

    ใช้ตรวจสอบ source MAC address ว่าตรงกับค่าที่ระบุไว้หรือไม่ มีประโยชน์สำหรับ PREROUTING, INPUT chain โดยมีออปชันให้ใช้งานคือ

    - --mac-source เช่น --mac-source 00:55:81:CC:42:FF

  - limit

    รูปแบบการใช้งาน: -m limit หรือ --match limit

    ใช้เพื่อจำกัดจำนวนของการ match ที่อาจจะมากเกินไป เป็นประโยชน์สำหรับ กฎ ที่วางไว้ตอนท้ายสุดของ chain (ใช้ร่วมกันกับ DROP policy) ซึ่งส่วนใหญ่เป็น กฎ ที่ใช้เก็บข้อมูลลงล็อกไฟล์ ซึ่งถ้าผู้บุกรุกส่ง แพ็กเก็ต ที่ไม่เข้าข่าย กฎ ใดๆ ใน chain จนกระทั่งมาถึง กฎ ที่ทำหน้าที่เก็บล็อกนี้ ถ้า แพ็กเก็ต ที่เข้ามามีจำนวนมากก็อาจจะทำให้ฮาร์ดดิสก์เต็มได้ ดังนั้นจึงต้องใช้จำกัดจำนวนในการเก็บข้อมูลลงล็อก ซึ่งมีออปชันให้ใช้งานดังนี้คือ

    - --limit ตามด้วยตัวเลข ซึ่งบอกถึงจำนวนครั้งสูงสุดของการ match กับ กฎ ที่ยินยอมต่อ 1 วินาที เช่น --limit 5/s ทั้งนี้ยังสามารถใช้หน่วยเวลาอื่นได้ เช่น /second /minute /hour /day เช่น -m limit --limit 3/minute
    - --limit-burst ตามด้วยตัวเลข แสดงถึงจำนวนมากที่สุดของ แพ็กเก็ต ที่ match กับ กฎ นี้ ค่า default ของมันคือ 5

ตัวอย่างการใช้ --limit และ --limit-burst ร่วมกัน เช่น

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

โดยส่วนใหญ่นิยมวาง กฎ นี้ไว้เป็น กฎ สุดท้ายใน chain โดยเฉพาะ chain ที่มี default policy เป็น DROP เพื่อเป็นตัวเก็บหลักฐานว่ามี แพ็กเก็ต ใดที่ถูกส่งมาและไม่ผ่านการตรวจสอบจาก กฎ และกำลังจะถูก DROP โดย default policy โดย กฎ ด้านบนนี้กำหนดจำนวน match สูงสุดไว้ 3 ครั้งต่อนาที ซึ่งแสดงว่าใน 1 นาทีจะมีการบันทึกล็อกได้สูงสุด 3 ครั้งเท่านั้น และมีค่า burst เท่ากับ 3 ซึ่งอธิบายได้ว่า ถ้าสมมุติมี แพ็กเก็ต ที่ match กับ กฎ 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 "Packet died: "

นอกจากนี้ยังใช้ป้องกันการโจมตีแบบ Denial of Service เช่น SYN flood ได้ด้วย เช่น

iptables -A FORWARD -p tcp --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 ไอซีเอ็มพี แพ็กเก็ต ทิ้งทั้งหมด เพราะถือว่า ไอซีเอ็มพี แพ็กเก็ต เป็นข้อมูลที่มีอันตรายยิ่งและสามารถปลอมแปลงได้ง่าย

หรือใช้ป้องกันการถูก scan

iptables -A FORWARD -p tcp --tcp-flags SYN,ACK,FIN,RST RST -m limit --limit 1/s -j ACCEPT

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

- owner

  รูปแบบการใช้งาน: -m owner หรือ --match owner

  ใช้ตรวจสอบลักษณะของ แพ็กเก็ต ว่าใครเป็นผู้สร้าง ซึ่งสามารถใช้ได้กับ OUTPUT chain เท่านั้น และใช้ได้กับบาง แพ็กเก็ต ที่มีเจ้าของ เช่น ไอซีเอ็มพี แพ็กเก็ต นั้นใช้ไม่ได้เพราะไม่มีเจ้าของ มีออปชันให้ใช้งานดังนี้คือ

- --uid-owner *userid*
  ใช้ตรวจสอบว่า แพ็กเก็ต ถูกสร้างโดย user id ที่ระบุไว้หรือไม่ (ใช้ตัวเลขแทน userid เท่านั้น)
- --gid-owner *groupid*
  ใช้ตรวจสอบว่า แพ็กเก็ต ถูกสร้างโดย user ที่อยู่ใน group id ที่ระบุไว้หรือไม่ (ใช้ตัวเลขแทน groupid เท่านั้น)
- --pid-owner *processid*
  ใช้ตรวจสอบว่า แพ็กเก็ต ถูกสร้างขึ้นจาก process ที่มี process id ตรงกับที่ระบุไว้หรือไม่
- --sid-owner *sessionid*
  ใช้ตรวจสอบว่า แพ็กเก็ต ถูกสร้างโดย process ที่อยู่ใน session group ที่กำหนดไว้หรือไม่

o unclean
รูปแบบการใช้งาน: -m unclean หรือ --match unclean
เป็นโมดูลที่อยู่ในระหว่างการทดลองใช้งาน นอกจากนี้ยังไม่มีออปชันสำหรับใช้งาน และโปรดระมัดระวังหากจะนำไปใช้กับเครื่องที่ต้องการความปลอดภัย เนื่องจากอาจจะยังมีข้อบกพร่องของโปรแกรมอยู่

โดย แพ็กเก็ต ที่เข้าข่าย unclean คือ

- แพ็กเก็ต ที่มี header ของ ไอซีเอ็มพี/ทีซีพี/ยูดีพี สั้นหรือไม่สมบูรณ์
- ทีซีพี, ยูดีพี แพ็กเก็ต ที่มี source หรือ destination ip address เป็นศูนย์
- ทีซีพี แพ็กเก็ต ที่ใช้ flag ผสมกันแบบผิดปกติ
- แพ็กเก็ต ที่ใช้ ทีซีพี option, IP option เกินความยาวที่กำหนดไว้ หรือมีความยาวของออปชันเป็นศูนย์
- fragment แพ็กเก็ต ที่ไม่สมบูรณ์ ทั้งด้านความยาวและค่า offset ที่เหลื่อมซ้อนกัน เช่น Ping of Death

o multiport ใช้ร่วมกับ --sport หรือ --dport ในกรณีที่ต้องการระบุ พอร์ต จำนวนมากกว่าหนึ่ง เช่น

-m multiport -p tcp --sport 25,80,53

- The State Match
  รูปแบบการใช้งาน: -m state หรือ --match state
  เป็นโมดูลที่ใช้ประโยชน์ได้เป็นอย่างดี มีออปชันให้ใช้งานดังนี้

- NEW

  รูปแบบการใช้งาน: -m state --state new หรือ --match state --state new

  หมายถึง แพ็กเก็ต ที่เป็นตัวสร้าง connection ใหม่

- ESTABLISHED

  รูปแบบการใช้งาน: -m state --state established หรือ --match state --state established

  หมายถึง แพ็กเก็ต ที่เกี่ยวข้องกันกับ connection ที่สร้างไว้แล้ว เช่น echo-reply แพ็กเก็ต หรือ แพ็กเก็ต ที่ส่งข้อมูลออกไปจาก web server เมื่อมี request web service เข้ามา

- RELATED

  รูปแบบการใช้งาน: -m state --state related หรือ --match state --state related

  เป็น แพ็กเก็ต ที่เกี่ยวข้องกับ connection ที่สร้างไว้แล้ว แต่ไม่ใช่ส่วนหนึ่งส่วนใดของ connection นั้น เช่น FTP data แพ็กเก็ต (พอร์ต 20) ที่เกิดขึ้นจากการใช้คำสั่งใน FTP command (พอร์ต 21)

- INVALID

  รูปแบบการใช้งาน: -m state --state invalid หรือ --match state --state invalid

  เป็น แพ็กเก็ต ที่ไม่เกี่ยวข้องกับส่วนอื่นเลย เช่น ไอซีเอ็มพี echo-reply ที่เกิดขึ้น โดยที่ไม่มีเครื่องได้ในระบบส่ง echo-request ออกไปเลย (กรณีเช่นนี้เกิดขึ้นได้เนื่องจากอาจจะโดนโจมตีแบบ Smurf attack)

5.3 การระบุทาร์เก็ต (target)

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

- user-defined chain

  เนื่องจาก iptables อนุญาตให้ผู้ใช้สามารถสร้าง chain ขึ้นมาได้ใหม่นอกเหนือจาก built-in chain ทั้งสามตัว (INPUT, OUTPUT, FORWARD) ทั้งนี้จะต้องใช้ตัวอักษรตัวเล็กทั้งหมดสำหรับ chain ที่ผู้ใช้สร้างขึ้นเอง

  เมื่อ แพ็กเก็ต match กับ กฎ ที่เป็น user-defined chain ตัว แพ็กเก็ต จะถูกนำไปตรวจสอบใหม่โดย user-defined chain นั้นๆ และถ้าใน chain นั้นๆ ไม่มีการตัดสินใจใดๆ ตัว แพ็กเก็ต ก็สามารถย้อนกลับมายัง กฎ ถัดไปใน chain ที่เริ่มต้นเดินทางได้ (ศึกษารายละเอียดได้จากตัวอย่างด้านล่าง)

INPUT
Rule1: -p ICMP -j DROP
Rule2: -p TCP -j test
Rule3: -p UDP -j DROP

test
Rule1: -s 192.168.1.1
Rule2: -d 192.168.1.1

เช่น ถ้า ทีซีพี แพ็กเก็ต เดินทางจาก 192.168.1.1 ไปยัง 1.2.3.4 ดังนั้น แพ็กเก็ต จะเข้าสู่ INPUT chain และไม่ match กับ กฎ1 แต่ match กับ กฎ2 ซึ่งมี target เป็น test ดังนั้น แพ็กเก็ต จะเข้าสู่ test chain และ match กับ กฎ1 แต่เนื่องจาก กฎ1 ของ test ไม่ได้ระบุ target ดังนั้น แพ็กเก็ต จึงผ่านไปยัง กฎ2 ซึ่งไม่ match จากนั้น แพ็กเก็ต จึงจะเดินทางกลับไปยัง กฎ3 ของ INPUT chain อีกครั้ง ซึ่งก็ไม่ match เช่นกัน ในกรณีที่ผ่าน กฎ 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 กฎ ที่ระบุ target เป็น LOG มีออปชันให้เลือกใช้งานดังนี้คือ

    - --log-level

      เป็นการระบุ priority level ของ log ซึ่งกำหนดได้ตั้งแต่ debug, info ,notice, warning, crit, alert, emerg รายละเอียดเกี่ยวกับ syslog สามารถอ่านได้ที่ http://thaicert.nectec.or.th/paper/unix_linux/linux_syslog.php

    - --log-prefix

      ตามด้วยชุดของตัวอักษรยาวไม่เกิน 29 ตัว ซึ่งชุดของตัวอักษรดังกล่าวจะปรากฏอยู่บนล็อกไฟล์

  - REJECT

    คล้ายกับ DROP เพียงแต่จะส่ง ไอซีเอ็มพี port unreachable กลับไปยังผู้ที่ส่ง แพ็กเก็ตมา (ข้อยกเว้นคือ ไอซีเอ็มพี error message ไม่ response กับ ไอซีเอ็มพี error message ด้วยกันเอง เพราะอาจจะทำให้เกิดลูปที่ไม่รู้จบ) ทั้งนี้สามารถใช้ร่วมกับ --reject-with ตามด้วย argument ที่ต้องการได้ รายละเอียดโปรดศึกษาจากคู่มือการใช้งาน iptables ที่มาพร้อมตัวโปรแกรม (#man iptables)

- special built-in target
    - RETURN

      กรณีที่ แพ็กเก็ต match กับ กฎ ที่มี target เป็น RETURN นั้นเสมือนกับเป็นคำสั่งให้ออกไปจาก chain ปัจจุบัน เช่น หาก match กับ กฎ ที่อยู่ใน built-in chain (INPUT, FORWARD, OUTPUT) แพ็กเก็ต ดังกล่าวจะถูกโยนไปยัง default policy ของ chain นั้นๆ และหาก แพ็กเก็ต match กับ กฎ ที่เป็น user-defined chain ตัว แพ็กเก็ต จะถูกโยนออกมา chain ก่อนหน้านั้น

    - QUEUE

      เป็น chain พิเศษ ใช้สำหรับส่งต่อ แพ็กเก็ต ไปยัง application ที่เขียนขึ้นมารองรับโดยเฉพาะ โดยจะต้องมี queue handler และ application เป็นส่วนประกอบที่จะทำงานร่วมกัน

5.4 การเดินทางของ แพ็กเก็ต ในระบบ

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

Step	Table	Chain	Comment
1			ข้อมูลอยู่ในระหว่างการเดินทาง เช่น กำลังมาจากอินเทอร์เน็ต
2			ข้อมูลเข้ามายังเครื่องไฟร์วอลล์ผ่านทาง incoming interface (เช่น eth0)
3	mangle	PREROUTING	ใช้สำหรับการทำ mangling แพ็กเก็ต เท่านั้น เช่น เปลี่ยนค่า TOS ของแพ็กเก็ต ซึ่งในกรณีปกติแล้วแทบไม่ได้ใช้งาน
4	nat	PREROUTING	chain นี้ใช้สำหรับทำ Destination Network Address Translation ไม่ควรสร้าง กฎ เพื่อกรอง แพ็กเก็ต ที่ chain นี้ เพราะอาจจะมีบาง แพ็กเก็ต ที่ไม่เข้าสู่ chain นี้ (มีแค่ แพ็กเก็ต แรกเท่านั้นที่ผ่านเข้าสู่ chain ส่วน แพ็กเก็ต ถัดไปใน connection เดียวกันนั้น จะถูกกระทำเหมือนกับที่ แพ็กเก็ต แรกได้รับ)

Step	Table	Chain	Comment
5			เข้าสู่ Routing decision เพื่อตัดสินใจว่า แพ็กเก็ต จะถูกส่งไปที่ใด
6	filter	FORWARD	เนื่องจากในตัวอย่างนี้ แพ็กเก็ต จะถูกส่งไปยังเครื่องอื่นในเครือข่าย ดังนั้น แพ็กเก็ต จึงต้องเข้า FORWARD chain ของ filter table ซึ่งสามารถเขียน กฎ สำหรับควบคุมการผ่านเข้าออกของ แพ็กเก็ต สำหรับ forwarded แพ็กเก็ต ได้ที่นี่
7	nat	POSTROUTING	และก่อนที่ แพ็กเก็ต จะออกไปจากเครื่องไฟร์วอลล์ โดยส่วนใหญ่(ไม่ใช่ทั้งหมด) จะผ่าน chain นี้ ซึ่งใช้ทำ Source Network Address Translation ไม่ควรสร้าง กฎ 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)

อย่างที่เห็นจากตาราง forwarded แพ็กเก็ต คือ แพ็กเก็ต ที่มีปลายทางที่ไม่ใช่เครื่อง ไฟร์วอลล์ อาจจะเป็น แพ็กเก็ต ที่มาจากอินเทอร์เน็ต หรืออาจจะเป็น แพ็กเก็ต ที่มาจากเครื่องลูกในเครือข่ายที่ต้องการส่งออกไปอินเทอร์เน็ต ซึ่งไม่ว่าจะส่งในทิศทางใด แพ็กเก็ต ก็จะต้องผ่าน chain ในลักษณะด้านบนนี้เสมอ

ตารางที่ 5-2 การเดินทางของฟอร์วาร์ดแพ็กเก็ต (Forwarded แพ็กเก็ต)

Step	Table	Chain	Comment
1			ข้อมูลอยู่ในระหว่างการเดินทาง เช่น กำลังมาจากอินเทอร์เน็ต
2			ข้อมูลเข้ามายังเครื่องไฟร์วอลล์ผ่านทาง incoming interface (เช่น eth0)
3	mangle	PREROUTING	ใช้สำหรับการทำ mangling แพ็กเก็ต เท่านั้น เช่น เปลี่ยนค่า TOS ของ แพ็กเก็ต ซึ่งในกรณีปกติแล้วแทบไม่ได้ใช้งาน
4	nat	PREROUTING	chain นี้ใช้สำหรับทำ Destination Network Address Translation ไม่ควรสร้าง กฎ เพื่อกรอง แพ็กเก็ต ที่ chain นี้ เพราะอาจจะมีบาง แพ็กเก็ต ที่ไม่เข้าสู่ chain นี้ (มีแค่ แพ็กเก็ต แรกเท่านั้นที่ผ่านเข้าสู่ chain ส่วน แพ็กเก็ต ถัดไปใน connection เดียวกันนั้น จะถูกกระทำเหมือนกับที่ แพ็กเก็ต แรกได้รับ)

Step	Table	Chain	Comment
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 5-3 การเดินทางของแพ็กเก็ตที่โฮสต์ปลายทาง (Destination localhost)

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

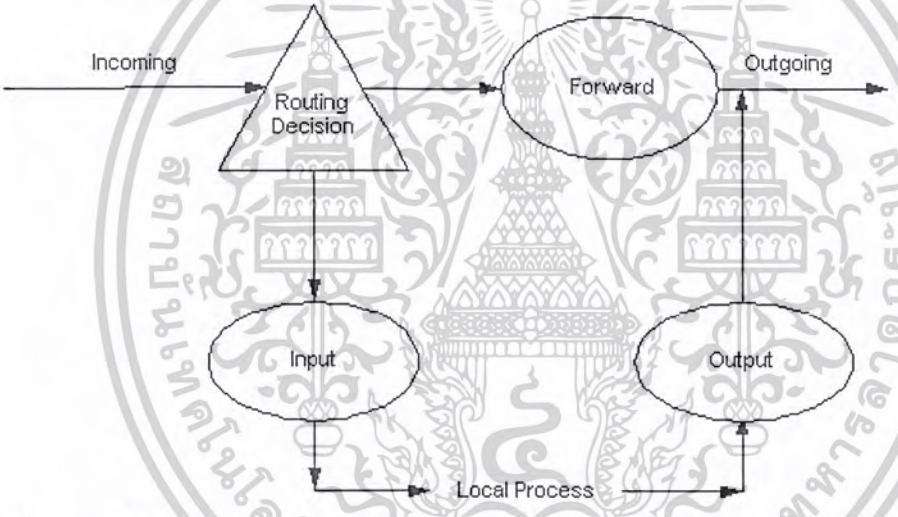
ตารางที่ 5-4 การเดินทางของแพ็กเก็ตที่โฮสต์ต้นทาง (Source localhost)

จากสามตัวอย่างด้านบน คงพอจะมองเห็นภาพออกแล้วว่า แพ็กเก็ต เดินทางเข้าออกในระบบ อย่างไร อย่างไรก็ตาม จะสรุปความสำคัญของแต่ละตาราง (table) อีกครั้ง ดังนี้

5.5 ฟิลเตอร์เทเบิล (Filter Table)

เป็นตารางที่ใช้งานมากที่สุด เป็นจุดที่ใช้ในการตรวจสอบและควบคุมการผ่านเข้าออกของ แพ็กเก็ต ถ้าหากจะพิจารณาการไหลเวียนของ แพ็กเก็ต เฉพาะในส่วนของ filter table โดยไม่สนใจ table อื่นๆ นั้น ก็พอจะแสดงให้เห็นได้ดังภาพที่ 1 โดยเมื่อ แพ็กเก็ต เข้ามาในระบบ มันจะเข้าไปยัง routing decision เพื่อตัดสินใจว่า แพ็กเก็ต จะถูกส่งไปที่ใด

- ในกรณีที่ แพ็กเก็ต ถูกส่งผ่านไปยังเครื่องอื่น แพ็กเก็ต นั้นจะต้องถูกตรวจสอบโดย กฎ ใน FORWARD chain
- ถ้า แพ็กเก็ต นั้น มีเป้าหมายเป็นเครื่องปัจจุบัน (เครื่องที่รัน iptables อยู่นี้ เรียกอีกอย่างว่า ลีนุกซ์ box) ตัว แพ็กเก็ต จะถูกตรวจสอบโดย กฎ ใน INPUT chain
- และในกรณีที่ แพ็กเก็ต ถูกสร้างจากเครื่องปัจจุบัน (ลีนุกซ์ box) ตัว แพ็กเก็ต จะถูกตรวจสอบจาก กฎ ใน OUTPUT chain ก่อนที่จะถูกส่งออกไป



รูปที่ 5-1 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 สามารถกำหนดให้ enable IP forwarding (forward=1) ได้โดยใช้คำสั่ง echo "1" > /proc/sys/net/ip_forward เพื่อกำหนดให้ IP forwarding เป็น enable เพื่อให้ ลีนุกซ์ box สามารถ forward ip แพ็กเก็ต ได้ ในบางครั้งนั้นการใช้คำสั่งดังกล่าวทุกครั้งอาจจะไม่สะดวก สามารถแก้ไขไฟล์ configuration ที่ /etc/sysctl.conf แล้ว set ให้ net.ipv4.ip_forward=1 เพื่อเป็นการแก้ไขแบบถาวร ในกรณีที่ต้องการให้

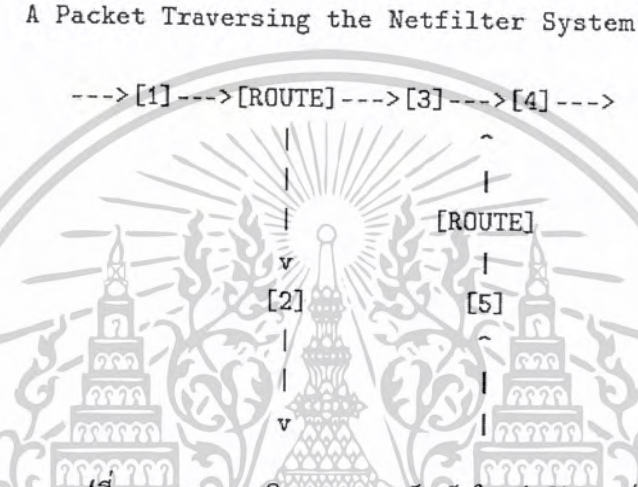
สนับสนุนการทำงานกับ dynamic IP ด้วย เช่น PPP, SLIP, DHCP ก็สามารถทำได้โดยใช้คำสั่ง echo "1" > /proc/sys/net/ipv4/ip_dynaddr ได้เช่นเดียวกัน

*** เนื่องจากโครงงานนี้ศึกษาเฉพาะ Filter Table จึงไม่ขอกล่าวถึง NAT Table และ Mangle Table ในที่นี้

บทที่ 6
เน็ตฟิลเตอร์

6.1 สถาปัตยกรรมของเน็ตฟิลเตอร์ (Netfilter Architecture)

เน็ตฟิลเตอร์นี้เป็นเพียงแค่ส่วนหนึ่งของฮุคส์ที่มีอยู่หลายส่วนในโพรโตคอลสแตค โดยรูปแบบทางเดินของแพ็กเก็ตในแนวความคิด(Idealized) ของเน็ทฟิลเตอร์เป็นดังรูปที่ 6-1



รูปที่ 6-1 แสดงการเดินทางของแพ็กเก็ตในเน็ทฟิลเตอร์

จากรูปที่ 6-1 เริ่มจากทางด้านซ้ายซึ่งเป็นทางที่แพ็กเก็ตเข้ามาและถูกตรวจสอบขั้นต้นด้วยโครงสร้างเน็ทฟิลเตอร์ชื่อ NF_IP_PRE_ROUTING [1] hook

จากนั้นแพ็กเก็ตก็จะเข้าสู่ routing code ซึ่งจะทำหน้าที่ตัดสินใจว่าจะให้แพ็กเก็ตเดินทางต่อไปอย่างไร โดย Routing code นี้สามารถส่งต่อแพ็กเก็ตไปยังเครื่องอื่นหรือส่งแพ็กเก็ตให้กับเครื่องตัวเองและสามารถ ดรอป แพ็กเก็ตนั้นทิ้งไปก็ได้

ถ้าหาก แอดเดรสปลายทาง ของแพ็กเก็ตนั้นเป็น แอดเดรส ของ Interface card ที่แพ็กเก็ต นั้นเข้ามาก็จะเรียก NF_IP_LOCAL_IN [2] hook ก่อนที่จะส่งผ่านไปยัง process ต่างๆ

แต่ถ้า แอดเดรสปลายทาง ของแพ็กเก็ตนั้นเป็น แอดเดรส ของอีก Interface card หนึ่งที่ไม่ใช่ Interface card ที่เข้ามา ก็จะเรียก NF_IP_FORWARD [3] hook จากนั้นแพ็กเก็ตก็จะถูกส่งไปยัง hook ตัวสุดท้ายของ Netfilter คือ NF_IP_POST_ROUTING [4] hook เพื่อส่งต่อไปยังเครื่องอื่นต่อไป

ส่วน NF_IP_LOCAL_OUT [5] hook จะถูกเรียกสำหรับแพ็กเก็ตที่ถูกสร้างขึ้นภายในเครื่องตัวเองเพื่อส่งแพ็กเก็ตออกไปยังเครื่องอื่น

6.2 พื้นฐานของเน็ทฟิลเตอร์ (Netfilter Base)

การทำงานพื้นฐานของ Netfilter นั้นมีอยู่หลักๆ 5 อย่างดังนี้

1. NF_ACCEPT : คือตอบรับแพ็กเก็ตนั้นเช่นนำไปประมวลผลหรือส่งต่อไปยังเครื่องอื่น
2. NF_DROP : คือกำจัดแพ็กเก็ตนั้นทิ้งไป
3. NF_STOLEN : คือเข้าไปจัดการกับแพ็กเก็ตบางอย่าง โดยไม่ให้แพ็กเก็ตเดินทางต่อ
4. NF_QUEUE : คือให้แพ็กเก็ตไปเข้าคิวเพื่อส่งต่อไปยังแอพพลิเคชั่นที่เขียนขึ้นมารองรับโดยเฉพาะ
5. NF_REPEAT : คือสั่งให้เรียก hook นี้อีกครั้ง

6.3 การพิจารณาแพ็กเก็ต (Packet Selection : IP Tables)

การพิจารณาแพ็กเก็ตที่เข้ามาในเครื่องคอมพิวเตอร์ ระบบจะทำการเรียก IP Tables ซึ่งฝังตัวอยู่ในโครงสร้างของเน็ทฟิลเตอร์ซึ่ง IP Tables นี้พัฒนาโดยตรงจากไอพีเซนส์ซึ่งการพิจารณาแพ็กเก็ตนี้จะใช้สำหรับการกรองแพ็กเก็ต (Packet Filtering), เน็ตเวิร์ค แอดเดรส ทรานส์เลชั่น และการกระจายแพ็กเก็ตโดยทั่วไป (General Pre-Route Packet Mangling) มีรายละเอียดดังนี้

- การกรองแพ็กเก็ต (Packet Filtering)

การกรองแพ็กเก็ตนี้จะใช้ตารางกรอง (Filter Table) ซึ่งตารางนี้จะไม่เปลี่ยนแปลงข้อมูลต่างๆ ของแพ็กเก็ตแต่จะมีหน้าที่เพียงกรองแพ็กเก็ตเท่านั้น ประโยชน์อย่างหนึ่งของ iptables ที่มีเหนือกว่า ipchains คือมีขนาดที่เล็กกว่าและทำงานได้รวดเร็วกว่า และมันเข้าไปทำงานตรงจุดที่เรียกว่า NF_IP_LOCAL_IN, NF_IP_FORWARD และ NF_IP_LOCAL_OUT ของ Netfilter นั่นคือเมื่อแพ็กเก็ตใดๆก็ตามที่เข้า-ออกจาก Netfilter มันจะถูกกรองด้วยจุดๆเดียวจาก 3 จุดที่ของ Netfilter ได้กล่าวไว้ข้างต้น ซึ่งจะทำให้เป็นเรื่องที่ง่ายยิ่งขึ้นสำหรับผู้ใช้ในการพิจารณาแพ็กเก็ตว่าจะถูกกรองด้วยจุดใดคือ แพ็กเก็ตที่มาจากเครื่องอื่นแล้วเข้ามายัง Netfilter ก็จะถูกกรองด้วย NF_IP_LOCAL_IN และหากเป็นแพ็กเก็ตที่เข้ามาแล้วต้องส่งให้เครื่องอื่นต่อก็จะถูกกรองด้วย NF_IP_FORWARD หรือหากเป็นแพ็กเก็ตที่สร้างจากเครื่องตัวเองแล้วต้องการส่งออกไปยังเครื่องอื่นก็จะถูกกรองด้วย NF_IP_LOCAL_OUTPUT

เนื่องจากโครงงานนี้เกี่ยวข้องเฉพาะการกรองแพ็กเก็ต (Packet Filtering) ดังนั้นจึงไม่ขอกล่าวถึง เน็ตเวิร์ค แอดเดรส ทรานส์เลชั่น และการกระจายแพ็กเก็ตโดยทั่วไป (General Pre-Route Packet Mangling) ในที่นี้

6.4 ส่วนขยายของไอพีเทเบิล (Extending iptables)

ประกอบด้วยส่วนของเคอร์เนล (Kernel Space) และส่วนของผู้ใช้ (userspace Tool)

6.4.1 ส่วนของเคอร์เนล (Kernel Space)

แบ่งเป็นแมทช์ฟังก์ชั่น (New Match Function) และ ทาร์เก็ตฟังก์ชั่น (New Target Function) ดังนี้

6.4.1.1 แมทช์ฟังก์ชั่น (New Match Function) โดยแกนหลักของแมทช์ฟังก์ชั่น นี้ คือ struct ipt_match ซึ่งจะถูก register ลงไปในแมทช์ฟังก์ชั่นของไอพีเทเบิลส์โดยใช้ ipt_register_match() โครงสร้างของ struct ipt_match มีส่วนประกอบเป็นฟิลด์ต่างๆดังนี้

list

ฟิลด์นี้จะถูกเซ็ทไว้เป็น { NULL, NULL } (บางทีถูกเรียกว่าฟิลส์ขยะ หรือ junk field)

name

ฟิลด์นี้คือชื่อของแมทช์ฟังก์ชั่นที่จะใช้ในการป้อนคำสั่ง iptables เช่น แมทช์ฟังก์ชั่น ที่พัฒนานี้ชื่อ webprevent ก็จะเซ็ทให้ name เป็น webprevent และ name นี้ควรสอดคล้องกับชื่อของไฟล์แมทช์โมดูล ตัวอย่างเช่น webprevent ก็จะสอดคล้องกับ ipt_webprevent.o ซึ่งเป็นชื่อของไฟล์แมทช์โมดูล

match

ฟิลด์นี้คือฟังก์ชั่นหลักของแมทช์ฟังก์ชั่น การทำงานหลักๆจะใช้ฟิลด์นี้เนื่องจากภายในฟิลด์ match นี้ จะมี skb ซึ่งเป็นพอยท์เตอร์ที่ชี้ไปยังซ็อกเก็ตบัฟเฟอร์ซึ่งเป็นหัวใจหลักในการทำงานกับแพ็กเก็ตที่รับเข้ามา

checkentry

ฟิลด์นี้จะเป็นฟังก์ชั่นซึ่งมีหน้าที่ในการเช็คการป้อนกฎที่มีลักษณะเฉพาะเจาะจงหรือกฎที่มีลักษณะพิเศษจากผู้ใช้

destroy

ฟิลด์นี้เป็นฟังก์ชั่นที่จะถูกเรียกให้ขึ้นมาทำงานเมื่อกฎของไอพีเทเบิลส์ที่ใช้แมทช์ฟังก์ชั่นนี้อยู่ถูกลบออกไป

me

ฟิลด์นี้จะถูกเซ็ตให้เป็น 'THIS_MODULE' ซึ่งเป็นพอยท์เตอร์ชี้มายังโมดูลนี้

6.4.1.2 ทาร์เก็ตฟังก์ชั่น (New Target Function) โดยแกนหลักของทาร์เก็ตฟังก์ชั่นคือ struct ipt_target ซึ่งจะถูก register ลงไปในทาร์เก็ตฟังก์ชั่นของ iptables โดยใช้ ipt_register_target() โครงสร้างของ struct ipt_target ประกอบด้วยฟิลด์ต่างๆดังนี้

list

ฟิลด์นี้จะถูกเซ็ทให้เป็น { NULL, NULL } (บางทีถูกเรียกว่าฟิลส์ขยะ หรือ junk field)

name

ฟิลด์นี้คือชื่อของทาร์เก็ตฟังก์ชั่นที่จะใช้ในการป้อนคำสั่งไอพีเทเบิลส์เช่น ทาร์เก็ตฟังก์ชั่นที่พัฒนานี้ชื่อ WEBDROP ก็จะเซ็ทให้ name เป็น WEBDROP และ name นี้ควรสอดคล้องกับชื่อของไฟล์ทาร์เก็ต โมดูล ตัวอย่างเช่น WEBDROP ก็จะสอดคล้องกับ ipt_WEBDROP.o ซึ่งเป็นชื่อของไฟล์ทาร์เก็ตโมดูล

target

ฟิลด์นี้คือฟังก์ชั่นหลักของแม็ทฟังก์ชั่น การทำงานหลักๆจะใช้ฟิลด์นี้เนื่องจากภายในฟิลด์ทาร์เก็ตนี้จะมี skbuff ซึ่งเป็นพอยท์เตอร์ที่ชี้ไปยังซ็อกเก็ตบัฟเฟอร์ซึ่งเป็นหัวใจหลักในการทำงานกับแพ็กเก็ตที่รับเข้ามา

checkentry

ฟิลด์นี้คือฟังก์ชั่นที่จะใช้เช็คการป้อนกฎของที่มีลักษณะเฉพาะเจาะจงหรือกฎที่มีลักษณะพิเศษจากผู้ใช้

destroy

ฟิลด์นี้เป็นฟังก์ชั่นที่จะถูกเรียกให้ขึ้นมาทำงานเมื่อกฎของไอพีเทเบิลส์ที่ใช้ทาร์เก็ตฟังก์ชั่นนี้อยู่ถูกลบออกไป

me

ฟิลด์ นี้จะถูกเซ็ตให้เป็น 'THIS_MODULE' ซึ่งเป็นพอยท์เตอร์ชี้มายังโมดูลนี้

6.4.2 ส่วนของผู้ใช้ (Userspace Tool)

การพัฒนาโมดูลของไอพีเทเบิลส์นั้นจะต้องมีการพัฒนาส่วนของผู้ใช้ควบคู่กับส่วนของเคอร์เนลด้วยซึ่งส่วนของผู้ใช้นี้พัฒนาขึ้นมาในรูปแบบของแชร์ไลบรารี่ (Shared library) ซึ่งประกอบด้วยฟังก์ชั่นหลักคือฟังก์ชั่น ' _init()' โดยมันจะถูกเรียกขึ้นมาโดยอัตโนมัติ ซึ่งหากเปรียบเทียบกับเคอร์เนลโมดูลแล้วก็เหมือนกับฟังก์ชั่น 'init_module()' โดย ฟังก์ชั่น _init() นี้จะไปเรียกฟังก์ชั่น 'register_match()' หรือฟังก์ชั่น 'register_target()' ก็ขึ้นอยู่กับว่าเราพัฒนาแชร์ไลบรารี่ นี้ขึ้นมาสำหรับแมทช์ฟังก์ชั่นหรือสำหรับทาร์เก็ตฟังก์ชั่น โดยประกอบด้วย Function ที่จำเป็นซึ่งประกาศไว้ในไฟล์ 'iptables.h' ดังนี้

check_inverse()

เป็นฟังก์ชั่นที่ใช้ในการตรวจสอบการป้อนคำสั่งจากผู้ใช้ว่ามีเครื่องหมาย '!' หรือไม่ถ้ามีก็จะทำการเซ็ทแฟลก 'invert'

string_to_number()

คือฟังก์ชั่นที่ใช้ในการเปลี่ยนตัวอักษรไปเป็นตัวเลข

exit_error()

เป็นฟังก์ชั่นที่ถูกเรียกใช้ถ้ามีความผิดพลาดเกิดขึ้น โดยทั่วไปก็จะมีอาร์กิวเมนต์ (argument) 'PARAMETER_PROBLEM' ซึ่งหมายถึง ผู้ใช้ป้อนคำสั่งผิดพลาด

โดยสรุปส่วนของผู้ใช้ คือ ฟังก์ชั่นที่ใช้ตรวจสอบการป้อนคำสั่งไอพีเทเบิลส์จากผู้ใช้ซึ่ง ส่วนของผู้ใช้ที่พัฒนานี้มี 2 ประเภท คือส่วนของผู้ใช้ของแมทช์ฟังก์ชั่นและส่วนของผู้ใช้ของทาร์เก็ตฟังก์ชั่น ซึ่งทั้ง 2 ฟังก์ชั่นมีโครงสร้างที่คล้ายกันดังนี้

6.4.2.1 ส่วนของผู้ใช้ของแมทช์ฟังก์ชั่น (Userspace Tool of New Match Function)

ส่วนของผู้ใช้ของแมทช์ฟังก์ชั่นที่เขียนขึ้นมานี้จะถูกเรียกอีกอย่างว่าแชร์ไลบรารี่ ซึ่งมี static struct iptables_match เป็นฟังก์ชั่นหลักในการ register แชร์ไลบรารี่นี้เข้าไปในแม็ทฟังก์ชั่น ของ iptables โดย static struct iptables_match มีโครงสร้างประกอบด้วยฟิลด์ต่างๆดังนี้

next

ฟิลด์นี้จะถูกเซ็ทเริ่มต้นให้เป็น NULL ในตอนเริ่มต้น

name

ฟิลด์นี้คือชื่อของแม็ทฟังก์ชั่นซึ่งควรตรงกันกับชื่อของแชร์ไลบรารี่ เช่นชื่อของแม็ทฟังก์ชั่น คือ webprevent ดังนั้นชื่อของแชร์ไลบรารี่ต้องเป็น libipt_webprevent.so

version

ฟิลด์นี้จะถูกเซ็ทไว้เป็น 'IPTABLES_VERSION' เพื่อให้ iptables ใช้แชร์ไลบรารี่ที่ตรงกับ version ของตัวมันเอง

size

คือการบอกขนาดที่ใช้สำหรับข้อมูลในแมชต์ฟังก์ชั่นนี้ซึ่งควรใช้ IPT_ALIGN() macro เพื่อความถูกต้อง

userspacrsize

ฟิลด์นี้จะเซ็ทค่าไว้เหมือนกับฟิลด์ size

help

ฟิลด์นี้คือฟังก์ชั่นที่ใช้ในการแสดงข้อความเมื่อผู้ใช้ใส่ option ' -h ' เพื่อแสดงข้อความอธิบายให้ผู้ใช้ เข้าใจ

init

ฟิลด์นี้ใช้เพื่อเป็นการเริ่มต้นการอินนิเชียลไลซ์เอ็กซ์ตราสเปส (initialize extra space)ใน โครงสร้างของ ipt_entry_match โดยปกติจะเซ็ทไว้เป็น *nfcache |= NFC_UNKNOWN

parse

ฟิลด์นี้จะถูกเรียกขึ้นมาใช้งานเมื่อมีการใส่อ็อพชั่นของแม็ทฟังก์ชั่นในตอนที่ป้อนคำสั่ง iptables

final_check

ฟิลด์นี้จะถูกเรียกขึ้นมาใช้งานหลังจากป้อนคำสั่ง iptables ผ่านไปแล้วมันจะทำการเช็คว่าการป้อนคำสั่ง iptables เพื่อใช้กับแม็ทฟังก์ชั่นนี้ถูกต้องหรือไม่ เช่นถ้าเช็คแล้วปรากฏว่าไม่ถูกต้องก็อาจจะสั่งให้เรียกฟังก์ชั่น ' exit_eror() '

print

ใช้ในการแสดงรายละเอียดของ rule ที่เกิดจากการป้อนคำสั่ง iptables ที่ใช้แม็ทฟังก์ชั่นนี้

save

ฟิลด์นี้จะถูกเรียกใช้เมื่อมีการใช้คำสั่ง iptables-save

extra_opts

ฟิลด์นี้จะถูกเรียกใช้ถ้าต้องการให้มีการทำงานออพชั่นต่างๆ

6.4.2.2 ส่วนของผู้ใช้ของทาร์เก็ตฟังก์ชั่น (Userspace Tool of New Target Function)

ส่วนของผู้ใช้ของทาร์เก็ตฟังก์ชั่นที่เขียนขึ้นมานี้จะถูกเรียกอีกอย่างว่าแชร์ไลบรารี่ ซึ่งมี static struct iptables_target เป็นฟังก์ชั่นหลักในการ register แชร์ไลบรารี่ นี้เข้าไปในทาร์เก็ตฟังก์ชั่นของ iptables โดย static struct iptables_target มีโครงสร้างประกอบด้วยฟิลด์ต่างๆที่เหมือนกับฟิลด์ของ static struct iptables_match ที่ได้อธิบายไว้แล้วในส่วนของผู้ใช้ของแมทช์ฟังก์ชั่นเพียงแต่เปลี่ยนจากแมทช์เป็นทาร์เก็ตเท่านั้น

next

ฟิลด์นี้จะถูกเซ็ทเริ่มต้นให้เป็น NULL ในตอนเริ่มต้น

name

ฟิลด์นี้คือชื่อของทาร์เก็ตฟังก์ชั่นซึ่งควรตรงกันกับชื่อของแชร์ไลบรารี่ เช่นชื่อของทาร์เก็ตฟังก์ชั่น คือ webprevent ดังนั้นชื่อของแชร์ไลบรารี่ต้องเป็น libipt_webprevent.so

version

ฟิลด์นี้จะถูกเซ็ทไว้เป็น 'IPTABLES_VERSION' เพื่อให้ไอพีเทเบิลส์ใช้แชร์ไลบรารี่ที่ตรงกับ version ของตัวมันเอง

size

คือการบอกขนาดที่ใช้สำหรับข้อมูลในทาร์เก็ตฟังก์ชั่นนี้ซึ่งควรใช้ IPT_ALIGN() macro เพื่อความถูกต้อง

userspacrsize

ฟิลด์นี้จะเซ็ทค่าไว้เหมือนกับฟิลด์ size

help

ฟิลด์นี้คือฟังก์ชั่นที่ใช้ในการแสดงข้อความเมื่อผู้ใช้ใส่ option ' -h ' เพื่อแสดงข้อความอธิบายให้ผู้ใช้ เข้าใจ

init

ฟิลด์นี้ใช้เพื่อเป็นการเริ่มต้นการอินนิเชียลไลซ์เอ็กซ์ตราสเปส (initialize extra space)ในโครงสร้างของ ipt_entry_target โดยปกติจะเซ็ทไว้เป็น *nfcache |= NFC_UNKNOWN

parse

ฟิลด์นี้จะถูกเรียกขึ้นมาใช้งานเมื่อมีการใส่อ็อพชั่นของทาร์เก็ตฟังก์ชั่นในตอนที่ป้อนคำสั่ง iptables

final_check

ฟิลด์นี้จะถูกเรียกขึ้นมาใช้งานหลังจากป้อนคำสั่ง iptables ผ่านไปแล้วมันจะทำการเช็คว่าการป้อนคำสั่ง iptables เพื่อใช้กับทาร์เก็ตฟังก์ชั่นนี้ถูกต้องหรือไม่ เช่นถ้าเช็คแล้วปรากฏว่าไม่ถูกต้องก็อาจจะสั่งให้เรียกฟังก์ชั่น ' exit_error() '

print

ใช้ในการแสดงรายละเอียดของ rule ที่เกิดจากการป้อนคำสั่ง iptables ที่ใช้ทาร์เก็ตฟังก์ชั่นนี้

save

ฟิลด์นี้จะถูกเรียกใช้เมื่อมีการใช้คำสั่ง iptables-save

extra_opts

ฟิลด์นี้จะถูกเรียกใช้ถ้าต้องการให้มีการทำงานออพชั่นต่างๆ

บทที่ 7
หลักการออกแบบและการทำงาน

7.1 หลักการออกแบบ

จากโครงสร้างของไอพีเทเบิลส์ที่ได้กล่าวไว้ในบทที่ 6 ซึ่งโครงงานนี้จะทำการพัฒนาโครงสร้างของ iptables โดยส่วนที่พัฒนาขึ้นมานี้คือ โมดูลที่ทำงานในส่วนของ แม็ทช์ และ ทาร์เก็ต ซึ่งส่วนที่พัฒนาขึ้นมานี้เรียกว่า ส่วนขยายของไอพีเทเบิล (Extending iptables) ซึ่งประกอบด้วยเคอร์เนลโมดูล และโมดูลส่วนของผู้ใช้โดยทั้ง 2 โมดูล จะถูกเรียกใช้งานโดย iptables ซึ่งส่วนขยายของไอพีเทเบิลนี้ทางผู้พัฒนาได้ทำการพัฒนาขึ้นมาใน 2 ส่วนหลักๆ เรียกว่า

- แมทช์ฟังก์ชั่น (New Match Functions)
- ทาร์เก็ตฟังก์ชั่น (New Target Functions)

โดยในการสร้างแมทช์ฟังก์ชั่น และทาร์เก็ตฟังก์ชั่น ทั้ง 2 ส่วนนี้จะต้องมี เคอร์เนล โมดูลและส่วนของผู้ใช้เพื่อรองรับการใช้งานของไอพีเทเบิลส์ซึ่ง ส่วนของผู้ใช้จะทำหน้าที่ในส่วนของการตรวจสอบการป้อนคำสั่งของผู้ใช้ ส่วนเคอร์เนลโมดูล จะทำหน้าที่ในส่วนของเคอร์เนลที่จะตรวจสอบแพ็กเก็ตที่มีลักษณะตรงกับกฎที่ผู้ใช้ได้ป้อนเข้ามาซึ่งทั้งแมทช์ฟังก์ชั่นและทาร์เก็ตฟังก์ชั่นที่พัฒนาขึ้นมามีรายละเอียดดังนี้

7.1.1 แมทช์ฟังก์ชั่น

แมทช์ฟังก์ชั่นที่พัฒนาขึ้นมานี้มีชื่อว่า webprevent โดยแมทช์ฟังก์ชั่นนี้จะทำการตรวจสอบข้อมูลของแพ็กเก็ตว่าภายในแพ็กเก็ตมีเนื้อหาข้อมูลที่มีลักษณะบ่งบอกว่าเป็นการโจมตีหรือไม่ โดยเทียบจากรูปแบบการโจมตีว่าเนื้อหาข้อมูลในแพ็กเก็ตนั้นตรงกับรูปแบบที่ตั้งไว้หรือไม่ โดยรูปแบบการโจมตีที่ใช้เป็นกฎของการตรวจสอบนี้ได้มาจากการอ่านกฎจากไฟล์ ซึ่งทำให้การปรับปรุงหรือเพิ่มเติมกฎทำได้อย่างสะดวก และเนื่องจากโมดูลที่พัฒนาขึ้นมานี้เป็นเคอร์เนลโมดูลจึงทำให้การอ่านไฟล์ไม่สามารถอ่านแบบทั่วไปได้ คือต้องทำการอ่านไฟล์จากโพรเซสไฟล์ (Process File) โดยต้องทำการคัดลอกรูปแบบการโจมตีจากไฟล์ต้นฉบับไปยังโพรเซสไฟล์ชื่อ rule ซึ่งอยู่ในไดเร็คทอรี /proc/net/ipt_rulelist/

ส่วนการเปรียบเทียบข้อมูลในแพ็กเก็ตกับรูปแบบการโจมตีที่เป็นกฎนั้นทำโดยใช้ฟังก์ชั่นในการค้นหาและเปรียบเทียบตัวอักขระ ซึ่งฟังก์ชั่นนี้เป็นการนำเอาเทคนิคโบเยอร์-มัวร์ (Boyer-moore) ซึ่งเป็นเทคนิคในการค้นหาและเปรียบเทียบตัวอักขระ ที่มีประสิทธิภาพ เนื่องจากการค้นหาและเปรียบเทียบทำได้อย่างรวดเร็ว เพื่อประสิทธิภาพในการตอบสนองเมื่อมีการร้องขอการเชื่อมต่อจากไคลเอนต์จำนวนมาก

7.1.2 ทาร์เก็ตฟังก์ชั่น

ทาร์เก็ตฟังก์ชั่นที่พัฒนาขึ้นมานี้มีชื่อว่า WEBDROP โดยทาร์เก็ตฟังก์ชั่นนี้จะทำการกำจัดแพ็กเก็ตนั้นทิ้งไปและจะทำการส่งทีซีพีรีเซ็ตแพ็กเก็ต (TCP Reset Packet) กลับไปยังต้นทางที่ส่งแพ็กเก็ตนี้มาด้วย เพื่อยกเลิกการเชื่อมต่อ โดยทาร์เก็ตฟังก์ชั่นนี้จะทำการล็อกกิ้ง (Logging) คือการบันทึก

ข้อมูลของแพ็กเก็ตที่มีลักษณะโจมตีนี้โดยจะทำการบันทึกลงล็อกไฟล์ (Log file) และแจ้งเตือน (alert) ออกมาทางหน้าจอเพื่อเตือนให้ผู้ดูแลระบบรับรู้ถึงการโจมตี ซึ่งรายละเอียดที่เก็บบันทึกและแจ้งเตือนจะประกอบด้วยไอพีแอดเดรสและแมคแอดเดรส (MAC Address) ของผู้โจมตี

โดยทั้งแมทช์ฟังก์ชั่นและทาร์เก็ตฟังก์ชั่น นั้นใช้เทคนิคการเขียนเคอร์เนลโมดูลขึ้นมาแล้วทำการแทรก (insert) โมดูลใหม่เข้าไปในเคอร์เนลโดยไม่ต้องทำการคอมไพล์เคอร์เนลใหม่ซึ่งเป็นเทคนิคที่นิยมใช้กันในการเขียนเคอร์เนลโมดูล

7.2 การทำงานของโปรแกรม

การทำงานของโปรแกรมนั้นแบ่งออกเป็นคุณลักษณะของอินพุทและเอาท์พุท (Input / Output Specification) ดังนี้

7.2.1 คุณลักษณะของอินพุท (Input Specification)

โมดูลของไอพีเทเบิลส์ที่พัฒนาขึ้นมานี้มีคุณลักษณะของอินพุทของโปรแกรม 2 ประเภทคือ

7.2.1.1 คุณลักษณะของอินพุทจากการป้อนคำสั่ง (command) ของไอพีเทเบิลซึ่งคุณลักษณะของอินพุทประเภทแรกนี้จะมีส่วนของผู้ใช้ (Userspace Tool) เป็นโปรแกรมตรวจสอบการป้อนคำสั่ง ตัวอย่างการป้อนคำสั่งไอพีเทเบิลส์มีรูปแบบคร่าวๆดังนี้ เช่น

iptables –A INPUT –p tcp --dport 80 –m webprevent - -webprevent 'rule' –j WEBDROP

จากการป้อนคำสั่งด้านบน จะหมายความว่าเป็นการเพิ่มกฎให้กับเชนอินพุท (INPUT chain) ซึ่งเชน (Chain) นี้อาจจะเป็นเชนอินพุทหรือเชนฟอร์เวิร์ด (FORWARD chain) ก็ได้ขึ้นอยู่กับว่าเราวางเครื่องไฟร์วอลกับเครื่องเว็บเซิร์ฟเวอร์ตำแหน่งไหนของเครือข่ายซึ่งในกรณีตัวอย่างนี้สมมุติให้เครื่องไฟร์วอลกับเครื่องเซิร์ฟเวอร์เป็นเครื่องเดียวกันจึงตั้งกฎไว้ที่เชนอินพุท แต่ถ้าหากเครื่องไฟร์วอลอยู่หน้าเครื่องเว็บเซิร์ฟเวอร์ก็ให้ตั้งกฎไว้ที่เชนฟอร์วาร์ดโดยให้ตรวจสอบแพ็กเก็ตที่เป็นโปรโตคอลทีซีพี ที่มีหมายเลขพอร์ตปลายทางเท่ากับ 80 ซึ่งเป็นหมายเลขพอร์ตของเว็บเซิร์ฟเวอร์แล้วนำแพ็กเก็ตนั้นไปแม็ทช์ (match) กับ webprevent ซึ่งเป็นแมทช์ฟังก์ชั่นที่ทำหน้าที่ค้นหาและเปรียบเทียบเนื้อหาข้อมูลในแพ็กเก็ตว่าตรงกับรูปแบบการโจมตีที่ตั้งไว้หรือไม่โดย ส่วน - -webprevent 'rule' เป็นออพชั่นของแมทช์ฟังก์ชั่นเพื่อให้ไปอ่านรูปแบบการโจมตีที่เป็นกฎที่ไฟล์ชื่อ rule ซึ่งเป็น process file อยู่ในไดเร็คทอรี่ /proc/net/ipt_rulelist ถ้าทำการค้นหาและเปรียบเทียบเนื้อหาข้อมูลในแพ็กเก็ตแล้วพบว่าตรงกับรูปแบบการโจมตีที่ได้ตั้งกฎไว้ก็จะทำการเรียก WEBDROP ซึ่งเป็นทาร์เก็ตฟังก์ชั่นให้ทำการกำจัดแพ็กเก็ตนั้นทิ้งไป

7.2.1.2 คุณลักษณะของอินพุทจากแพ็กเก็ต ที่เข้ามาสู่กฎของไอพีเทเบิลที่ตั้งไว้ จากตัวอย่างการป้อนคำสั่งในข้อ 7.2.1.2 ก็คือแพ็กเก็ตที่เข้ามาสู่เชนอินพุท โดยแพ็กเก็ตที่เข้ามานี้ก็จะถูกนำเข้าไปประมวลผลภายในแมทช์ฟังก์ชั่นที่สร้างขึ้นนั่นเอง

7.2.2 คุณลักษณะของเอาท์พุท (Output Specification)

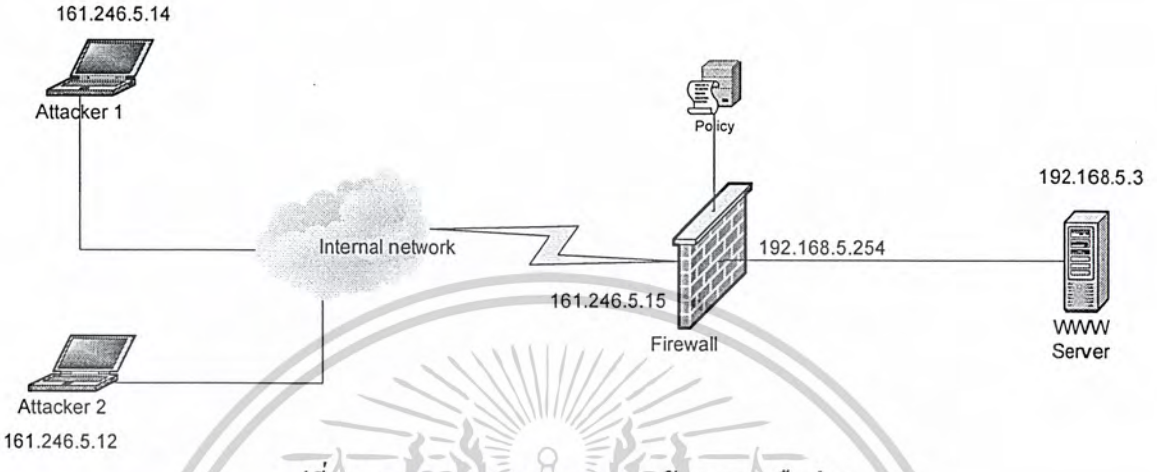
โมดูลของไอพีเทเบิลที่พัฒนาขึ้นมีคุณลักษณะของเอาท์พุทของโปรแกรม 2 ประเภท คือ

7.2.2.1 คุณลักษณะของเอาท์พุทที่เกิดจากการกำจัดแพ็กเก็ต ที่ตรงกับรูปแบบการโจมตีที่ตั้งไว้ นั่นคือคุณลักษณะของเอาท์พุทประเภทนี้เกิดขึ้นจากการที่แพ็กเก็ตมีลักษณะตรงกับ webprevent ซึ่งเป็นแมทช์

ฟังก์ชั่นที่ตั้งไว้ แล้วถูก WEBDROP ซึ่งเป็นทาร์เก็ตฟังก์ชั่นให้ทำการกำจัดแพ็กเก็ตนั้นทิ้งไป แล้วทาร์เก็ตฟังก์ชั่น WEBDROP นี้ก็จะทำการส่งทีซีพีรีเซ็ตแพ็กเก็ตกลับไปยังเครื่องที่ส่งแพ็กเก็ตนี้มา

7.2.2.2 คุณลักษณะของเอาท์พุทที่เกิดจากการแสดงและบันทึก (Logging) การโจมตี คือการแสดงข้อมูลของแพ็กเก็ตที่มีลักษณะโจมตีออกมาทางหน้าจอเพื่อแจ้งเตือนให้ผู้ดูแลระบบทราบว่ามีการโจมตีเว็บเซิร์ฟเวอร์ และทำการบันทึกลงล็อกไฟล์ (Log File) ด้วยเพื่อให้ผู้ดูแลระบบได้ทำการตรวจสอบความผิดปกติได้ในภายหลัง

บทที่ 8
การทดสอบการทำงาน



รูปที่ 8-1 แสดงภาพจำลองสภาพแวดล้อมของเครือข่าย

สภาพแวดล้อมในการทดลองโปรแกรมนี้นั้นเราได้จำลองสถานะการณ์ที่จะใช้ในการทดลองการทำงานของตัวโมดูลที่เราได้เขียนขึ้นมาแล้วใส่เข้าไปยังเคอร์เนลของตัวไอพีเทเบิล โดยการจำลองสถานะการณ์จะเป็นไปตามลักษณะดังรูปที่ 8-1

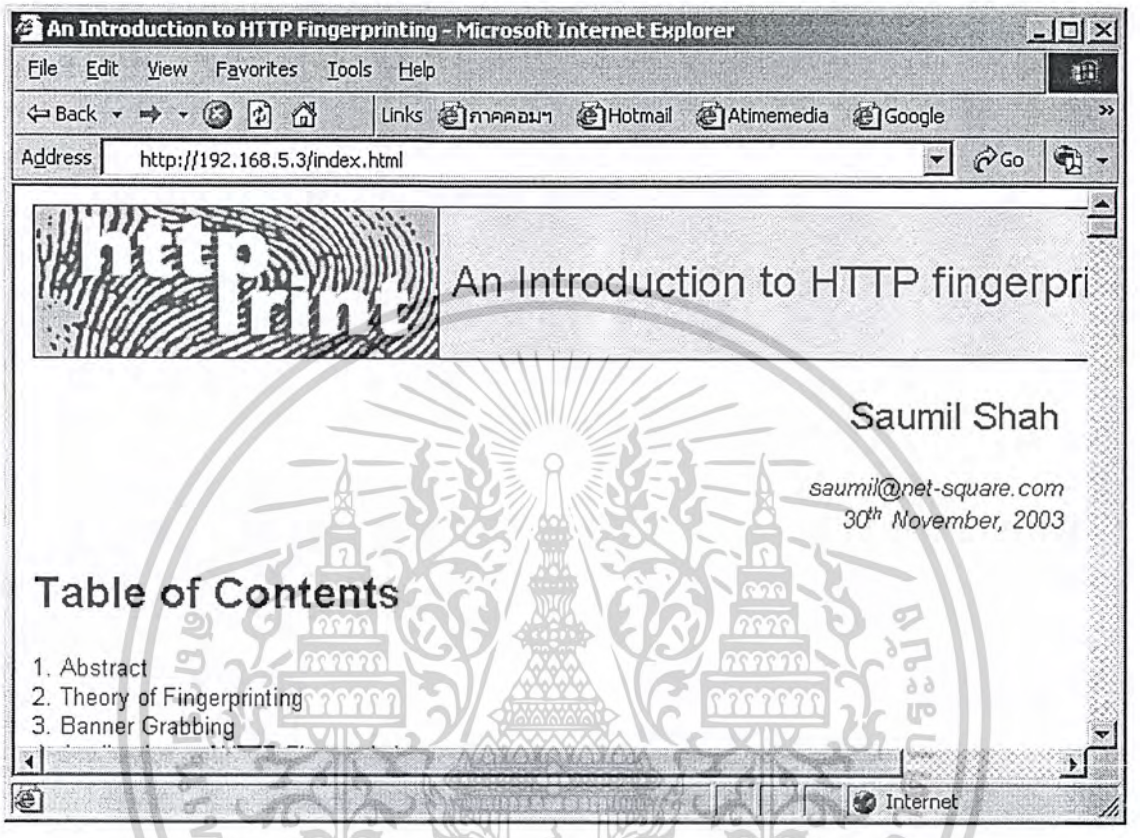
เครื่องผู้โจมตีที่ 1 จะเป็นเครื่องที่มีระบบปฏิบัติการเป็น วินโดว์ 2000 และเครื่องที่เป็นผู้โจมตีที่ 2 จะเป็นระบบปฏิบัติการลีนุกซ์

เครื่องไฟร์วอลจะเป็นไฟร์วอลที่ใช้ ระบบปฏิบัติการลีนุกซ์เคเบียน เคอร์เนล เวอร์ชั่น 2.4.23 และเน็ทฟิลเตอร์/ไอพีเทเบิลส์ เวอร์ชั่น 1.2.9 ในการใช้งาน และเครื่องเวบเซิร์ฟเวอร์ระบบปฏิบัติการจะเป็น วินโดว์ 2000 แอ็ดวานค์เซิร์ฟเวอร์

การโจมตีจะยกตัวอย่างมาเพียงสองวิธีที่นิยมใช้กันคือ การโจมตีแบบการทำไดเร็คทรอรี่ทราเวอร์ซอล และการโจมตีด้วยการใช้ จิล (jill)

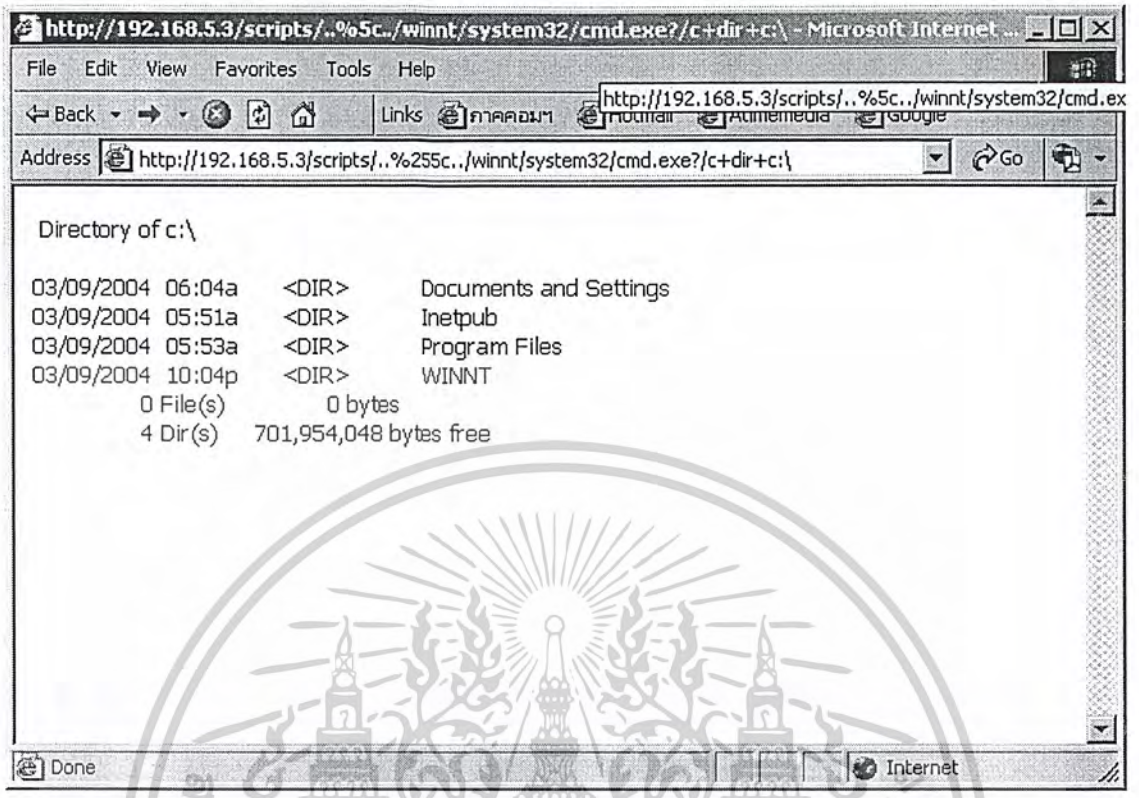
ซึ่งผลการทดลองที่ได้จะมีดังต่อไปนี้

การทดสอบโปรแกรมโดยการเปิดเว็บไซต์โดยใส่แอดเดรส ในช่อง URL ของเว็บบราวเซอร์ ด้วยชื่อของเว็บเซิร์ฟเวอร์ทำให้สามารถเปิดเว็บไซท์นั้นได้ตามปกติ ดังรูปที่ 8-2



รูปที่ 8-2 แสดงการเปิดเว็บไซท์ตามปกติ

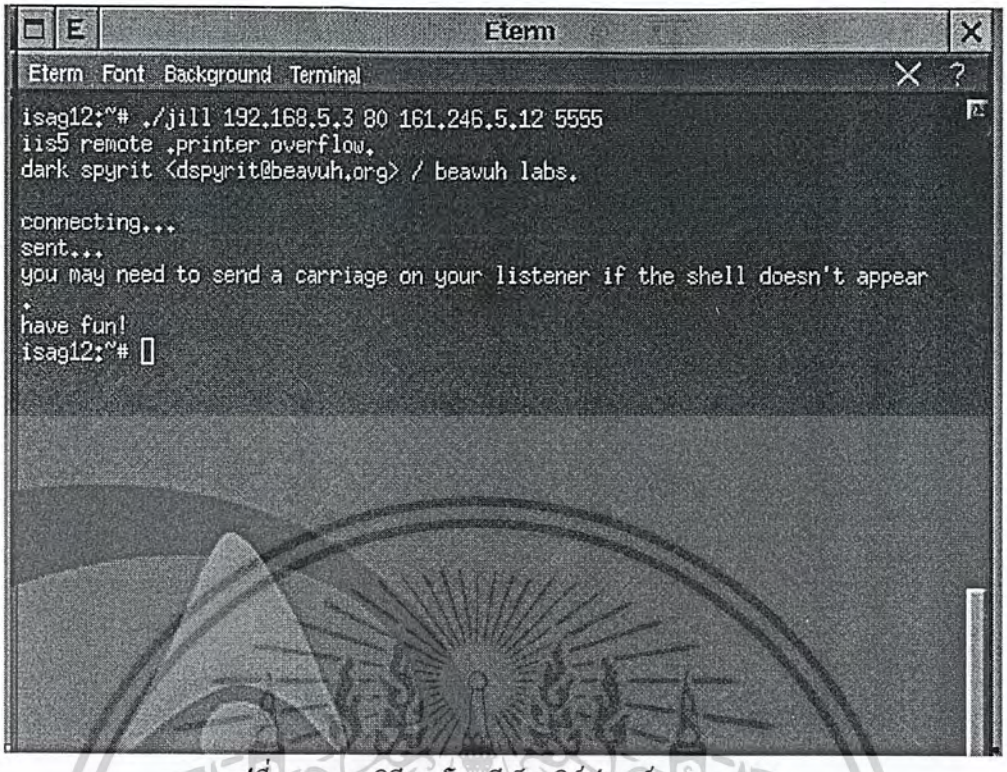
หลังจากนั้นทำการทดลองป้อนเนื้อหาของข้อมูลที่มีลักษณะเป็นการโจมตีเช่น http://192.168.5.3/scripts/..%255c../winnt/system32/cmd.exe?/c+dir+c:\ ซึ่งการป้อนในลักษณะดังกล่าวเป็นการทำไดเร็คทอรี่ทราเวอร์ซอล (Directory Traversal) เพื่อทำการดูข้อมูลในเครื่องเว็บเซิร์ฟเวอร์ว่ามีข้อมูลเป็นแฟ้มข้อมูลหรือเป็นไฟล์อะไรบ้าง ซึ่งผลจากการโจมตีแบบนี้จะได้ผลดังรูปที่ 8-3



รูปที่ 8-3 แสดงการโจมตีโดยใช้วิธีการไดเร็คทอรี่ทราเวอร์ซอล

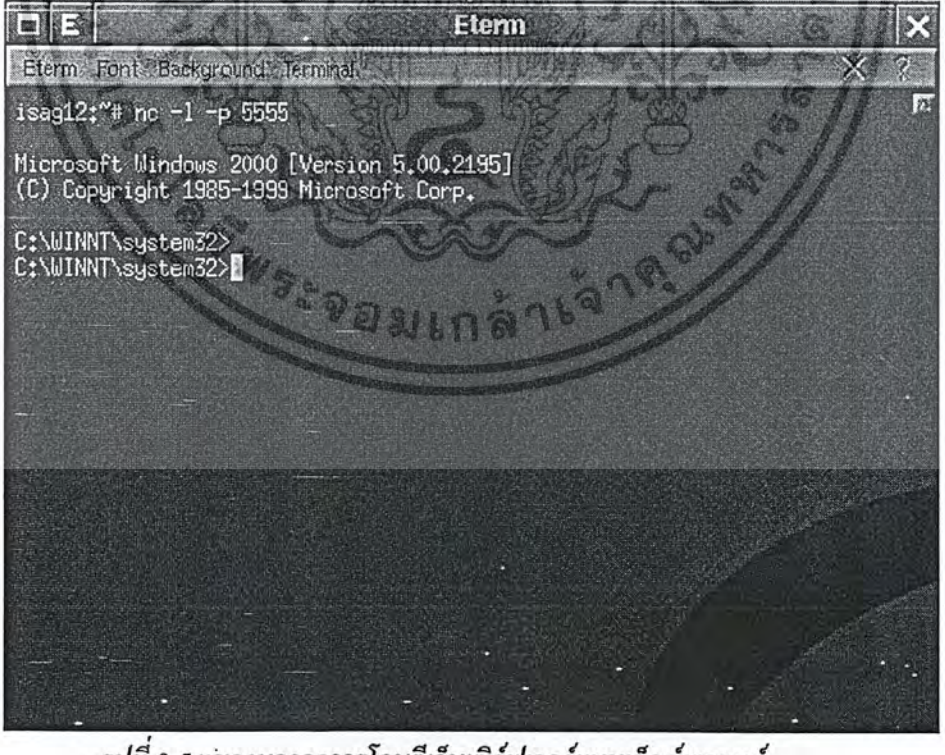
จากรูปจะเห็นว่าผลจากการโจมตีดังกล่าวจะทำให้ผู้ไม่หวังดีสามารถมองเห็นข้อมูลของเว็บเซิร์ฟเวอร์ว่ามีแฟ้มข้อมูลต่างๆ ซึ่งข้อมูลเหล่านี้สามารถนำไปเป็นประโยชน์ในการโจมตีได้ในอนาคต

นอกจากการโจมตีแบบดังกล่าวยังมีการโจมตีอีกรูปแบบหนึ่งที่สามารถทำให้ผู้โจมตีสามารถเข้าไปควบคุมเครื่องเว็บเซิร์ฟเวอร์จนสามารถทำให้ตัวเองเป็นเจ้าของเครื่องเว็บเซิร์ฟเวอร์เลยซึ่งการโจมตีรูปแบบดังกล่าวเรียกว่าการโจมตีแบบเอ็กซ์พลอยท์ซึ่งวิธีการโจมตีคือการใช้โปรแกรม จิลล์ และ เน็ทแคท (netcat) เพื่อทำการเปิดพอร์ตของเว็บเซิร์ฟเวอร์ที่ทำให้ผู้ไม่หวังดีสามารถใช้พอร์ตนั้นทำการเข้าไปควบคุมเครื่องเว็บเซิร์ฟเวอร์โดยการโจมตีแบบนี้จะเป็นดังรูปที่ 8-4



รูปที่ 8-4 แสดงวิธีการโจมตีเว็บเซิร์ฟเวอร์แบบ exploit

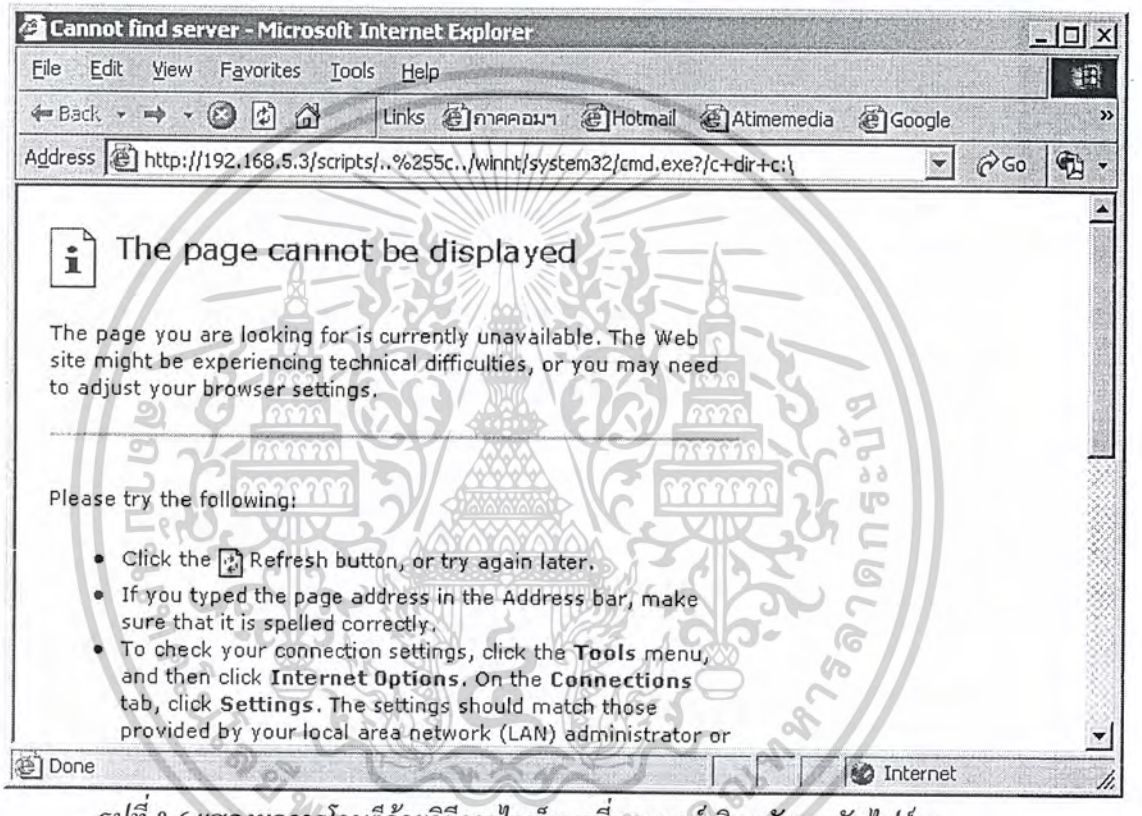
หลังจากการโจมตีตามรูปที่ 8-4 แล้วผู้ไม่หวังดีก็สามารถเข้าไปใช้หรือทำการควบคุมเว็บเซิร์ฟเวอร์ได้ซึ่งผลจาการโจมตีแบบเอ็กซ์พลอยท์ จะได้ผลดังรูปที่ 8-5



รูปที่ 8-5 แสดงผลจากการโจมตีเว็บเซิร์ฟเวอร์แบบเอ็กซ์พลอยท์

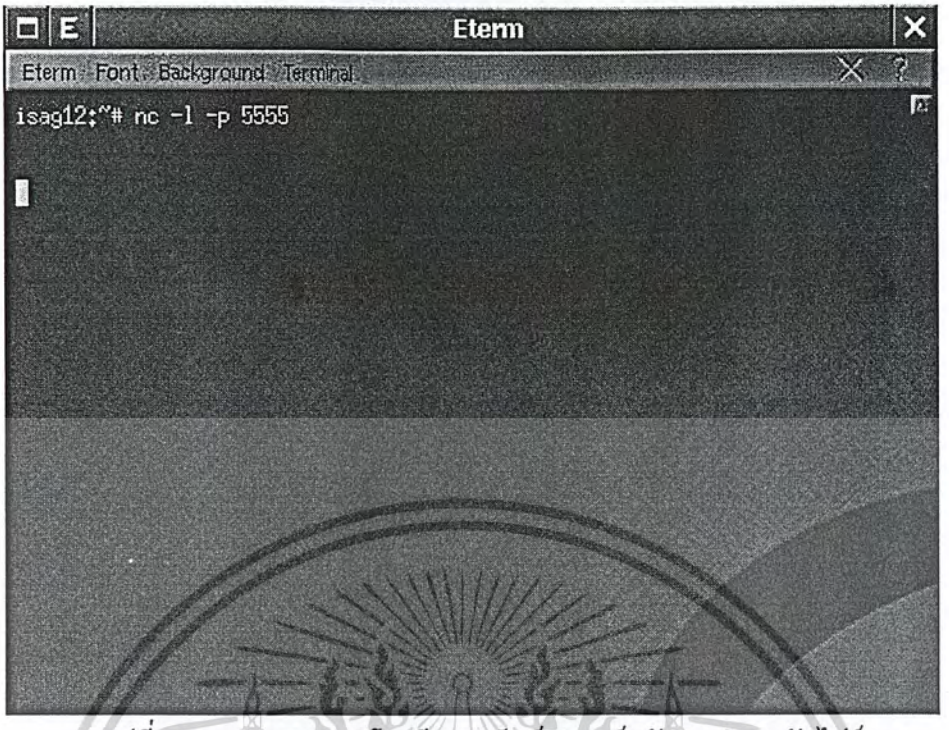
จากรูปที่ 8-5 ซึ่งแสดงผลจากการโจมตีด้วยวิธีการเอ็กซ์พลอยท์ทำให้เครื่องที่ของผู้โจมตีที่ใช้ระบบปฏิบัติการลีนุกซ์สามารถเข้าไปใช้คอมมานด์ไลน์ ของเครื่องเว็บเซิร์ฟเวอร์ที่เป็นระบบปฏิบัติการวินโดวส์ได้

หลังจากนั้นทำการรันโปรแกรมไฟร์วอลที่พัฒนาขึ้นมาแล้วทำการโจมตีด้วยวิธีการเดิมคือการทำไดเร็คทอรี่ทราเวอร์เซิลและการโจมตีแบบเอ็กซ์พลอยท์จะทำให้ผู้ไม่หวังดีไม่สามารถทำการโจมตีได้ นั่นคือเครื่องไฟร์วอลสามารถป้องกันการโจมตีได้สำเร็จ ซึ่งการผลการโจมตีหลังจากรันไฟร์วอลแล้วแสดงดังรูปที่ 8-6 และรูปที่ 8-7



รูปที่ 8-6 แสดงผลการโจมตีด้วยวิธีการไดเร็คทอรี่ทราเวอร์เซิลหลังจากรันไฟร์วอล

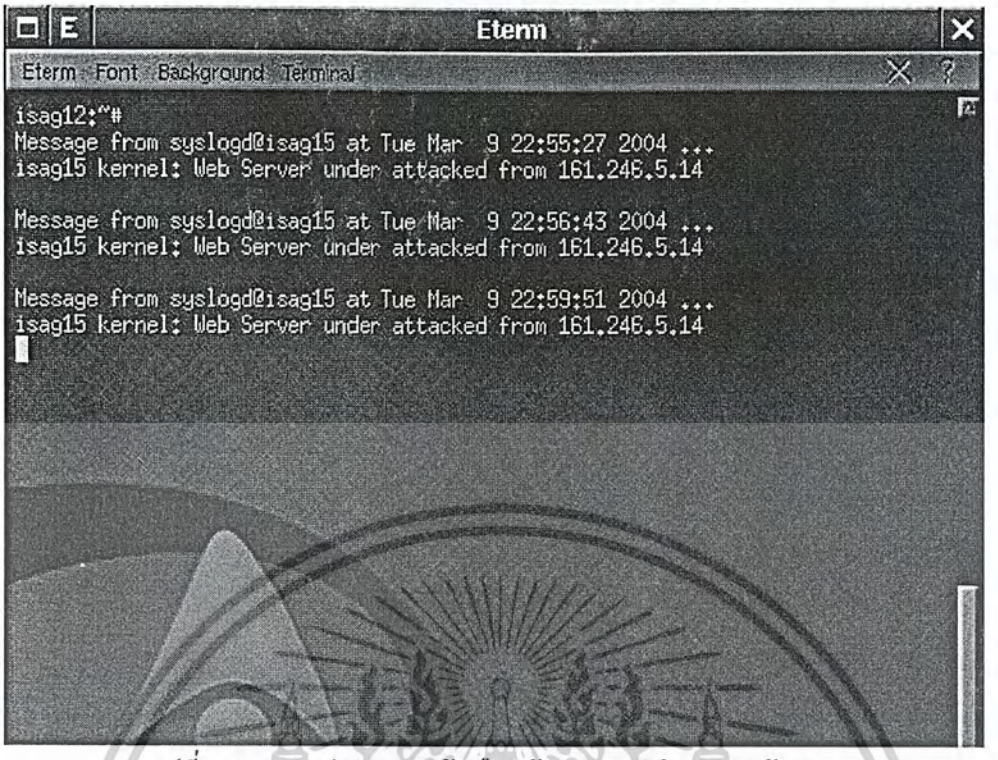
จากรูปที่ 8-6 จะเห็นได้ว่าผู้ไม่หวังดีไม่สามารถทำการโจมตีแบบไดเร็คทอรี่ทราเวอร์ซอลได้เนื่องจากการโจมตีดังกล่าวไม่สามารถผ่านเครื่องไฟร์วอลไปได้



รูปที่ 8-7 แสดงผลจากการโจมตีแบบ เอ็กซ์พลอยท์หลังจากทำการรันไฟร์วอล

จากรูปที่ 8-7 จะเห็นว่าผู้ไม่หวังดีไม่สามารถทำการโจมตีแบบเอ็กซ์พลอยท์ได้เนื่องจากการแบบดังกล่าวไม่สามารถผ่านเครื่องไฟร์วอลไปได้

และหลังจากผู้ไม่หวังดีทำการโจมตีด้วยวิธีการทั้ง 2 แบบที่ได้กล่าวมาแล้วข้างต้น จะทำให้ไฟร์วอลทำการกำจัดแพ็กเก็ตที่มีการโจมตีนั้นทิ้งไปและทำการแจ้งเตือนผู้ดูแลระบบผ่านทางหน้าจอ ซึ่งรูปแบบการแจ้งเตือนผ่านทางหน้าจอแสดงในรูปที่ 8-8



รูปที่ 8-8 แสดงรูปแบบการแจ้งเตือนผู้ดูแลระบบผ่านทางหน้าจอ

จากรูปที่ 8-8 แสดงการแจ้งเตือนผู้ดูแลระบบว่ามีการโจมตีเว็บเซิร์ฟเวอร์ผ่านทางหน้าจอ โดยมีข้อความระบุถึงหมายเลขไอพี่ของผู้โจมตีด้วย และการแจ้งเตือนลักษณะแบบนี้ก็จะทำการบันทึกลงไฟล์ที่ชื่อว่า syslog ด้วยซึ่งไฟล์ syslog เป็นไฟล์ที่ทำการบันทึกข้อมูลสำคัญของระบบ

บทที่ 9
สรุปผลและวิจารณ์

9.1 ปัญหาและอุปสรรคในการพัฒนาโปรแกรม

ในช่วงเวลาที่ผ่านมาที่ได้ทำการพัฒนาโปรแกรมการป้องกันเว็บโดยใช้ไอพีเทเบิลส์นั้นได้พบปัญหาและอุปสรรคต่างๆหลายประการได้แก่

- การเขียนโปรแกรมเคอร์เนลโมดูล และ ยูสเซอร์สเปซ ทูล ของไอพีเทเบิลส์ นั้นมีโครงสร้างที่ตายตัวจึงทำให้ไม่สามารถเขียนโปรแกรมได้อย่างอิสระเหมือนเคอร์เนลโมดูลทั่วไปดังนั้นการเขียนโปรแกรมจึงค่อนข้างมีอุปสรรคเพราะไม่ค่อยมีความอิสระในการสร้าง อัลกอริทึมของตัวเอง
- การเขียนโปรแกรมเคอร์เนลโมดูล นั้นการทำงานของโปรแกรมจะทำงานในส่วนของเคอร์เนล ซึ่งเป็นแกนหลักของระบบปฏิบัติการ ดังนั้นจึงต้องเขียนด้วยความระมัดระวังเนื่องจากหากโปรแกรมทำงานผิดพลาดจะทำให้เครื่องคอมพิวเตอร์ไม่สามารถทำงานต่อได้เป็นอาการที่เรียกว่า "Kernel Panic" ซึ่งจะต้องทำการรีเซ็ตเครื่องคอมพิวเตอร์เพียงเท่านั้นซึ่งการรีเซ็ตเครื่องคอมพิวเตอร์บ่อยๆจะเป็นผลเสียต่อเครื่องคอมพิวเตอร์ด้วย
- การป้องกันของโมดูลที่สร้างขึ้นมานี้ไม่สามารถป้องกันการโจมตีผ่านทางซีเคียวซอกเก็ตเลเยอร์ได้
- การที่จะดึงเอาเนื้อข้อมูลออกมาจากไอพีแพ็คเก็ตจำเป็นจะต้องอาศัยไลบรารี่ของ skbuffer จึงจะสามารถดึงเอาเนื้อข้อมูลออกมาได้
- ฟังก์ชันของไอพีเทเบิลส์มีให้ใช้จำกัดไม่หลากหลายและยืดหยุ่นเหมือนกับการเขียนโปรแกรมไฟร์วอลตัวอื่นๆ

9.2 แนวทางการพัฒนาต่อในอนาคต

เนื่องจากโมดูลของไอพีเทเบิลส์ที่พัฒนาขึ้นมานี้สามารถตรวจจับเนื้อหา (Content) ของข้อมูลในแพ็กเก็ตได้แล้วซึ่งการตรวจจับรูปแบบของข้อมูลนี้เป็นเพียงแค่ส่วนหนึ่งของการตรวจจับการบุกรุกในรูปแบบของการโจมตีเว็บเซิร์ฟเวอร์ แต่ในความเป็นจริงแล้วยังมีการโจมตีรูปแบบอื่นที่มีลักษณะเป็นรูปแบบของข้อมูลได้อีกเช่น การโจมตีโดยการใช้ไวรัสคอมพิวเตอร์ (Virus) ดังนั้นในอนาคตสามารถพัฒนาโมดูลของ iptables ให้สามารถตรวจจับและป้องกันการโจมตีโดยใช้ไวรัสคอมพิวเตอร์ได้ด้วย ซึ่งการโจมตีด้วยไวรัสโดยทั่วไปแล้วจะโจมตีผ่านทางจดหมายอิเล็กทรอนิกส์ ดังนั้นจึงต้องศึกษาโปรโตคอลที่เกี่ยวข้องกับการส่งจดหมายอิเล็กทรอนิกส์เช่น โปรโตคอล SMTP (Simple Mail Transfer Protocol)

และนอกเหนือจากนั้นการพัฒนาโมดูลของไอพีเทเบิลส์นั้นเป็นการพัฒนาในรูปแบบของเคอร์เนลโมดูล ดังนั้นจึงต้องศึกษาการเขียนโปรแกรมแบบเคอร์เนลโมดูลด้วย

บรรณานุกรม

เรืองไกร รังสิผล : “ **เปิดโลก Firewall และ Internet Security** ” : กรุงเทพฯ , โปรวิชั่น , 2544

Alessandro Rubini & Jonathan Corbet : “ **Linux Device Drivers**” : USA , Orielly

www.netfilter.org

www.thaicert.nectec.or.th

www.kernelnewbies.org

www.linuxjournal.com/article.php?sid=7984

ภาคผนวก ก.
วิธีการติดตั้ง

การติดตั้งโปรแกรมนี้แบ่งออกเป็น 2 ประเภทคือ การติดตั้ง เคอร์เนลโมดูล และ แชร์ไลบรารี่ (Shared Library)

1. **การติดตั้ง เคอร์เนลโมดูล** เนื่องจากการติดตั้ง เคอร์เนลโมดูล จะต้องทำการ คอมไพล์ โมดูลใหม่เพื่อให้ตรงกับ เคอร์เนล ที่ใช้อยู่ดังนั้นจึงต้องมี ซอร์ส เคอร์เนล อยู่ด้วย โดยให้เก็บซอร์ส เคอร์เนล ไว้ใน /usr/src/linux โดย ลีนุกซ์ ในที่นี้คือ ซอร์ส เคอร์เนล การติดตั้ง เคอร์เนลโมดูล แบ่งเป็น 2 อย่างคือ เคอร์เนลโมดูล match function และ เคอร์เนลโมดูล targert function แต่ละอย่างมีขั้นตอนในการติดตั้งดังนี้

1.1 การติดตั้ง เคอร์เนลโมดูล match function

1.1.1 เข้าไปใน ไดเร็คทอรี match

1.1.2 Copy ไฟล์ ipt_webprevent.h ไปไว้ในซอร์สเคอร์เนล โดยพิมพ์คำสั่ง

```
cp ipt_webprevent.h  /usr/src/linux/include/linux/netfilter_ipv4/
```

1.1.3 ทำการ คอมไพล์ โดยพิมพ์คำสั่ง

```
make
```

1.1.4 จะได้ไฟล์เคอร์เนลโมดูล ชื่อ ipt_webprevent.o มา แล้วให้ทำการแทรกโมดูล โดยพิมพ์คำสั่ง

```
insmod  ipt_webprevent.o
```

1.1.5 เสร็จการติดตั้งเคอร์เนลโมดูล match function

1.2 การติดตั้งเคอร์เนลโมดูล targert function

1.2.1 เข้าไปใน ไดเร็คทอรี TARGET

1.2.2 Copy ไฟล์ ipt_WEBDROP.h ไปไว้ในซอร์สเคอร์เนล โดยพิมพ์คำสั่ง

```
cp ipt_WEBDROP.h  /usr/src/linux/include/linux/netfilter_ipv4/
```

1.2.3 ทำการ คอมไพล์ โดยพิมพ์คำสั่ง

```
make
```

1.2.4 จะได้ไฟล์ เคอร์เนลโมดูล ชื่อ ipt_WEBDROP.o มาแล้วทำการแทรกโมดูลเข้าไปในเคอร์เนลโดยพิมพ์คำสั่ง

```
insmod ipt_WEBDROP.o
```

1.2.4 เสร็จสิ้นการติดตั้ง เคอร์เนลโมดูล target function

2. **การติดตั้ง แชร์ไลบรารี่ (Shared Library)** เนื่องจากการติดตั้ง แชร์ไลบรารี่ ต้องทำการคอมไพล์ แชร์ไลบรารี่ ใหม่เพื่อให้ตรงกับเวอร์ชั่นของ iptables ที่ใช้งานอยู่บนเครื่องนั้นๆ ดังนั้นจึงต้องมี iptables ซอร์ส อยู่ด้วยโดยสามารถดาวน์โหลดได้จาก www.netfilter.org และ เวอร์ชั่น ของ iptables ที่ใช้ในการพัฒนาคือ เวอร์ชั่น 1.2.9 และ เวอร์ชั่น 1.2.9 จึงต้องเลือก ดาวน์โหลด เฉพาะ 2 เวอร์ชั่น นี้เท่านั้นเพื่อความถูกต้อง และการติดตั้ง แชร์ไลบรารี่ มีอยู่ 2 อย่างคือ แชร์ไลบรารี่ ของ match function และ แชร์ไลบรารี่ ของ target function แต่ละอย่างมีขั้นตอนในการติดตั้งดังนี้

2.1 การติดตั้ง แชร์ไลบรารี่ ของ match function

1.1.1 หลังจาก ดาวน์โหลด iptables มาแล้วทำการแตกไฟล์ไปไว้ที่ /usr/local/bin/ จะได้เป็น ไดเร็คทอรี iptables-1.2.9 (ในที่นี้ใช้ iptables เวอร์ชั่น 1.2.9)

1.1.2 ทำการติดตั้งโปรแกรม iptables โดย เข้าไปใน ไดเร็คทอรี iptables-1.2.9 แล้วพิมพ์คำสั่ง

```
./configure
make
make install
```

1.1.3 เข้าไปใน ไดเร็คทอรี match (ไดเร็คทอรี match คือ ไดเร็คทอรี ของโปรแกรมไฟล์วอลที่พัฒนา)

1.1.4 ทำการ copy ไฟล์ libipt_webprevent.c ไปไว้ใน ไดเร็คทอรี extensions ซึ่งเป็น ไดเร็คทอรี่ย่อยของ ไดเร็คทอรี iptables-1.2.9 โดยการพิมพ์คำสั่ง

```
cp libipt_webprevent.c /usr/local/bin/iptables-1.2.9/extensions/
```

1.1.5 เข้าไปใน ไดเร็คทอรี extension โดยพิมพ์คำสั่ง

```
cd /usr/local/bin/iptables-1.2.9/extensions/
```

1.1.6 ทำการแก้ไขข้อความในไฟล์ Makefile โดยพิมพ์คำสั่ง

```
vi Makefile
```

ทำการเพิ่มข้อความใน Makefile ตรงบรรทัด PF_EXT_SLIB:= โดยเพิ่มคำว่า webprevent ต่อจากข้อความ PF_EXT_SLIB:= แล้วทำการ save Makefile แล้วออกจากโปรแกรม vi

1.1.7 ทำการ คอมไพล์ แชร์ไลบรารี่ ของ match function โดยพิมพ์คำสั่ง

```
make
```

1.1.8 หลังจาก คอมไพล์ จะได้ แชร์ไลบรารี่ ชื่อ libipt_webprevent.so แล้วให้ทำ

1.1.9 การ copy ไฟล์ libipt_webprevent.so ไปไว้ที่ /usr/local/lib/iptables/

1.1.10 เสร็จสิ้นการติดตั้ง แชร์ไลบรารี่ ของ match function

1.2 การติดตั้ง แชร์ไลบรารี่ ของ target function

1.2.1 เข้าไปใน ไดเร็คทอรี target (ไดเร็คทอรี target คือ ไดเร็คทอรี ของโปรแกรมไฟล์วอลที่พัฒนา)

1.2.2 ทำการ copy ไฟล์ libipt_WEBDROP.c ไปไว้ใน ไดเร็คทอรี extensions ซึ่งเป็นไดเร็คทอรี่ย่อยของ ไดเร็คทอรี iptables-1.2.9 โดยการพิมพ์คำสั่ง

cp libipt_WEBDROP.c /usr/local/bin/iptables-1.2.9/extensions/

2.2.3 เข้าไปใน ไดเร็คทอรี extension โดยพิมพ์คำสั่ง

cd /usr/local/bin/iptables-1.2.9/extensions/

2.2.4 ทำการแก้ไขข้อความในไฟล์ Makefile โดยพิมพ์คำสั่ง

vi Makefile

ทำการเพิ่มข้อความใน Makefile ตรงบรรทัด PF_EXT_SLIB:= โดยเพิ่มคำว่า WEBDROP ต่อจากข้อความ PF_EXT_SLIB:= แล้วทำการ save Makefile แล้วออกจากโปรแกรม vi

1.2.3 ทำการ คอมไพล์ แชร์ไลบรารี่ ของ target function โดยพิมพ์คำสั่ง

make

1.2.4 หลังจาก คอมไพล์ จะได้ แชร์ไลบรารี่ ชื่อ libipt_WEBDROP.so แล้วให้ทำการ copy ไฟล์ libipt_WEBDROP.so ไปไว้ที่ /usr/local/lib/iptables/

1.2.5 เสร็จสิ้นการติดตั้ง แชร์ไลบรารี่ ของ match function

*หมายเหตุ ซอร์สเคอร์เนลในการคอมไพล์โครงงานใช้ตัวเคอร์เนลเวอร์ชัน 2.4.23 GNU GCC เวอร์ชัน 2.95

ภาคผนวก ข.

วิธีการใช้งาน

การใช้งานโปรแกรมไฟร์วอลที่พัฒนาขึ้นนี้ต้องใช้งานโดยเรียกผ่านโปรแกรม iptables โดยการใส่คำสั่ง iptables ซึ่งมีตัวอย่างของการป้อนคำสั่งดังนี้

iptables -A INPUT -p tcp --dport 80 -m webprevent --webprevent 'rule' -j WEBDROP

จากการป้อนคำสั่ง iptables ด้านบนใช้ในกรณีที่เครื่องเว็บเซิร์ฟเวอร์และเครื่องไฟร์วอลเป็นเครื่องเดียวกัน ถ้าหากเครื่องเว็บเซิร์ฟเวอร์ตั้งอยู่หลังเครื่องไฟร์วอลก็เพียงแต่เปลี่ยนจากคำว่า INPUT เป็นคำว่า FORWARD ซึ่งจากคำสั่งดังกล่าวอธิบายได้ดังนี้

-A INPUT คือการเพิ่ม rule ให้กับ chain INPUT

-p tcp --dport 80 คือการบ่งบอกว่าเป็น protocol TCP และ destination port เท่ากับ 80

-m webprevent คือการบอกให้ packet ที่เข้ามาโดนตรวจสอบด้วย webprevent ซึ่งเป็น match function ที่พัฒนาขึ้น

--webprevent 'rule' คือออพชั่นของฟังก์ชั่นแมทช์ที่จะไปทำการสร้างไฟล์ชื่อ rule ที่ใช้เป็นกฎไว้ในโพรเซสไฟล์ ซึ่งอยู่ใน /proc/net/ipt_rulelist/

-j WEBDROP คือการสั่งให้ packet ตรงกับ match function webprevent ไปกระทำตาม WEBDROP ซึ่งเป็น target function ที่พัฒนาขึ้นมา

จากนั้นให้ทำการคัดลอกไฟล์ที่เป็นกฎต้นฉบับไปยังโพรเซสไฟล์โดยใช้คำสั่งดังนี้

cat rule.txt > /proc/net/ipt_rulelist/rule

โดยไฟล์ rule.txt เป็นไฟล์ต้นฉบับ

หลังจากนั้นให้ทำการเปิดเว็บด้วยโปรแกรมเว็บบราวเซอร์จากเครื่องไคล์เอนท์(client) โดยใส่ address มายังเครื่องที่เป็นเว็บเซิร์ฟเวอร์ถ้าหากการทำงานถูกต้องจะสามารถเปิดเว็บไซต์ได้เช่นพิมพ์

http://161.246.5.15/

จากการพิมพ์แอดเดรสตามด้านบนจะสามารถเปิดเว็บไซต์ได้ หลังจากนั้นให้ทำการเพิ่มข้อความต่อจากaddress ดังกล่าวโดยข้อความต้องตรงกับแพทเทิร์น (pattern) ที่ได้ตั้งกฎเอาไว้ว่าเป็นแพทเทิร์น ที่มีลักษณะเป็นการโจมตีเช่น /%00 , /%0a.pl , /..%255c../, /cgi-bin/// , /cgi-bin/Count.cgi , /NULL.printer ตัวอย่างเช่น

http://161.246.5.15/scripts/..%255c../winnt/system32/cmd.exe?/c+dir เป็นต้น