ระบบตรวจจับและคัดแยกทราฟฟิกที่ผิดปกติ
โดยวิเคราะห์พฤติกรรมการใช้งานเครือข่าย

BEHAVIOR-BASED ANOMALY NETWORK TRAFFIC
DETECTION AND SEPARATION SYSTEM

ชนาธิป ศรียาภัย

ฐานุตร์ ปิ่มหทัยวุฒิ

ณัฐถกร วังมณี

ปริญญานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิศวกรรมศาสตรบัณฑิต

สาขาวิชาวิศวกรรมคอมพิวเตอร์ คณะวิศวกรรมศาสตร์

สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

ปีการศึกษา 2553

ปริญญานิพนธ์ปีการศึกษา 2553

สาขาวิชาวิศวกรรมคอมพิวเตอร์

คณะวิศวกรรมศาสตร์ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

เรื่อง ระบบตรวจจับและคัดแยกทราฟฟิกที่ผิดปกติโดยวิเคราะห์พฤติกรรมการใช้งานเครือข่าย

BEHAVIOR-BASED ANOMALY NETWORK TRAFFIC DETECTION AND SEPARATION SYSTEM

ผู้จัดทำ

1. นายชนาธิป ศรียาภัย รหัสนักศึกษา 50010298
2. นายฐานุตร์ ปิ่มหทัยวุฒิ รหัสนักศึกษา 50010399
3. นายณัฐถกร วังมณี รหัสนักศึกษา 50010453

_____อาจารย์ที่ปรึกษา

(อาจารย์เกียรติณรงค์ ทองประเสริฐ)

_____อาจารย์ที่ปรึกษา

(อาจารย์ธนัญชัย ตรีภาค)

ระบบตรวจจับและคัดแยกทราฟฟิกที่ผิดปกติ โดยวิเคราะห์พฤติกรรมการใช้งานเครือข่าย

นายชนาธิป	ศรียาภัย	50010298
นายฐานุตร์	ปิ่มหทัยวุฒิ	50010399
นายณัฐถกร	วังมณี	50010453
อาจารย์เกียรติณรงค์	ทองประเสริฐ	อาจารย์ที่ปรึกษา
อาจารย์ธนัญชัย	ตรีภาค	อาจารย์ที่ปรึกษาร่วม

ปีการศึกษา 2553

บทคัดย่อ

โครงงานระบบตรวจจับและคัดแยกทราฟฟิกที่ผิดปกติ โดยวิเคราะห์จากพฤติกรรมการใช้งานเครือข่ายเกิดขึ้นจากแนวความคิดที่ว่าด้วยปัญหาในเรื่องของปริมาณทราฟฟิกที่มากเกินไปในระบบเครือข่าย ซึ่งเป็นผลมาจากการใช้งานจำพวกบิททอเรนท์ (Bit Torrent) หรือเอฟทีพี (FTP) ที่ทำให้เกิดผลกระทบกับระบบเครือข่ายส่วนรวม

ระบบตรวจจับและคัดแยกทราฟฟิกที่ผิดปกตินี้ จะเข้าช่วยแบ่งเบาภาระของเครือข่าย โดยระบบจะตรวจจับแพคเกตข้อมูลและทำการคัดแยกทราฟฟิกออกเป็นสองประเภท คือ ทราฟฟิกที่ปกติ เช่น ข้อมูลจำพวกเอชทีทีพี (HTTP) หรือการเข้าเว็บทั่วไปและทราฟฟิกที่ผิดปกติคือ Bit Torrent และ FTP ที่มีโอนถ่ายปริมาณข้อมูลมากๆ ซึ่งระบบนี้จะทำให้ระบบเครือข่ายสามารถทำงานต่อไปได้เป็นปกติ

ระบบตรวจจับและคัดแยกทราฟฟิกที่ผิดปกติ โดยวิเคราะห์จากพฤติกรรมการใช้งานเครือข่าย พัฒนาขึ้นบนระบบปฏิบัติการลินุกซ์ (Linux) ใช้ภาษาไพธอน (Python) ในการพัฒนาส่วนติดต่อผู้ใช้โดยการนำโปรแกรมสนอร์ท (Snort) และไอพีเทเบิล (IP Tables) มาประยุกต์ใช้งานร่วมกัน โดยใช้ Snort ในการดักแพกเกต (Packet) และใช้ไอพีเทเบิล (IP Tables) สำหรับการ Mark Packet ที่ตรงกับกฎ แล้วส่งต่อให้ IP Route เพื่อทำการ Forward Packet ออกไปตาม Interface ที่ต้องการ และเก็บกฎต่างๆที่ใช้ในการคัดแยกทราฟฟิกไว้ในฐานข้อมูล นอกจากนี้ ระบบสามารถทำการมอนิเตอร์ระบบเครือข่ายจากการแสดงผลปริมาณทราฟฟิกที่วิ่งผ่านอินเตอร์เฟสและเก็บบันทึกการบังคับใช้กฎเป็นลอคไฟล์ (Log File) ได้อีกด้วย

โดยสรุปแล้วระบบที่พัฒนาขึ้นสามารถนำไปปรับใช้หรือพัฒนาต่อโดยผู้ดูแลระบบ เพื่อคัดกรองทราฟฟิกในเน็ตเวิร์คให้มีประสิทธิภาพมากยิ่งๆขึ้นไป

Behavior-Based Anomaly Network Traffic Detection and Separation System

Mr. Chanathip	Sriyapai	50010298
Mr. Thanut	Pimhataivoot	50010399
Mr. Nattakorn	Wangmanee	50010362
Mr. Kiatnarong	Tongprasert	Advisor
Mr. Thanunchai	Threepak	Co-Advisor

Academic Year 2010

ABSTRACT

The Behavior-Based Anomaly Network Traffic Detection and Separation System Project has started from the problem on the massive amount of traffic flowing in the network. Nowadays a lot of people use the Internet in our daily life whether at home, school or work. The problem is when you are at the school or the office, there are many people sharing your Internet connection. With this limited bandwidth situation, some people still use the peer-to-peer communication (also known as Bit Torrent) or FTP to share a big chunk of data which leads to the gigantic consuming of the network bandwidth. The main question is how can we manage these traffics to let them continue to be available to the user but on the counter part not interrupting any other who is using the internet for other purposes.

The System is developed based on Linux operation system using Python 2.7 programming language. Both of them are famous for their great network features. Under the hood is the Snort, IP Route and IP Tables that help doing most of the decent work. Snort is used to analyze payloads of the packet and IP Tables is used to mark each packet then IP Route is used to route packet out from interface configured by each rules which are kept in the database. In addition, the system also monitors the amount of traffic that went through each interface. All the enforcement of rules are kept in the log file

To conclude, the system that we developed can be adjust or further develop by Network Administrator to suit the requirement of each network system to make it work more efficiently.

กิตติกรรมประกาศ

ผู้จัดทำขอขอบพระคุณอาจารย์ที่ปรึกษาคือ อาจารย์เกียรติณรงค์ ทองประเสริฐ อาจารย์ธนัญชัย ตรีภาค อาจารย์อัครเดช วัชระภูพงษ์ สำหรับคำแนะนำที่เป็นประโยชน์ในการทำโครงงาน คุณพ่อคุณแม่ที่เป็นกำลังใจให้ในการทำโครงงานนี้ ห้องวิจัยระบบเครือข่ายที่เป็นที่พึ่งให้ผู้จัดทำโครงการสามารถทำโครงงานได้ด้วยดี แล้วถึงเป็นที่วางเครื่องเซิฟเวอร์เพื่อทดลองของโครงงานนี้ รวมถึงอาจารย์ที่ปรึกษาห้องวิจัยอาจารย์จิระศักดิ์ สิทธิกรที่คอยให้กำลังใจให้เสมอมา

สุดท้ายนี้สาขาวิชาวิศวกรรมคอมพิวเตอร์ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบังที่ประสิทธิ์ประศาสตร์วิชา ขอบคุณเพื่อนๆ พี่ๆ น้องๆที่เป็นกำลังใจฝ่าฟันค่ำคืนอันแสนยาวนานจนโครงงานนี้สำเร็จ ทางคณะผู้จัดทำขอขอบคุณมา ณ ที่นี้ด้วย

ชนาธิป ศรียาภัย
ฐานุตร์ ปิ่มหทัยวุฒิ
ณัฐฌกร วังมณี

สารบัญ

หน้า

บทคัดย่อภาษาไทย I

บทคัดย่อภาษาอังกฤษ II

กิตติกรรมประกาศ III

สารบัญ IV

สารบัญตาราง VI

สารบัญรูป VII

บทที่ 1 บทนำ 1

1.1 ที่มาและความสำคัญของโครงงาน 1

1.2 จุดมุ่งหมายและวัตถุประสงค์ของโครงงาน 1

1.3 ขอบเขตของโครงงาน 1

1.4 ขั้นตอนการดำเนินงาน 2

1.5 ประโยชน์ที่คาดว่าจะได้รับ 2

1.6 ส่วนประกอบของปริญญานิพนธ์ 2

บทที่ 2 ทฤษฎีที่เกี่ยวข้องและเทคโนโลยีที่ใช้ 3

2.1 ทีซีพี/ไอพี 3

2.2 ไฟร์วอลล์ (Firewall) 8

2.3 ระบบตรวจจับผู้บุกรุก (Intrusion Detection System) 10

2.4 ไอพีเทเบิล (IP Tables) 11

2.5 สนอร์ท (Snort) 22

2.6 ลิบพีแคบ (Libpcap) 22

2.7 TCP Dump 24

บทที่ 3 รายละเอียดการพัฒนาโครงการ 26

3.1 รายละเอียดการพัฒนาระบบ (System Specification) 26

3.2 ออกแบบส่วนติดต่อผู้ใช้งาน (User Interfaces) 34

สารบัญ (ต่อ)

หน้า

บทที่ 4 การทดสอบการทำงาน 38

4.1 การทดลองตรวจจับแพคเกตโดยใช้สนอร์ท 38

4.2 การทดลองการอินเจ็กท์โค้ดลงบนเชลล์สคริปต์โดยจียูไอภาษาไพธอน 38

4.3 การทดลองคัดแยกทราฟฟิกโดยใช้ไอพีเทเบิล 40

4.4 การทดสอบระบบจริง 44

บทที่ 5 สรุปผลการทดลอง ปัญหาและแนวทางการพัฒนา 50

5.1 สรุปผลการดำเนินโครงการ 50

5.2 ปัญหาและอุปสรรค 50

5.3 แนวทางการพัฒนาโครงการและประยุกต์ใช้กับงานอื่นๆ 50

บรรณานุกรม 51

ภาคผนวก ก 53

ภาคผนวก ข 56

สารบัญตาราง

ตาราง **หน้า**

2.1 เปรียบเทียบระหว่างไฟร์วอลล์ซอฟต์แวร์กับไฟร์วอลล์ฮาร์ดแวร์ 10
2.2 ใช้ในการเลือกแพคเกตโดยไฟร์วอลล์ 15
2.3 คำสั่งทั่วไปในไอพีเทเบิลที่ใช้ในการเลือก 19
2.4 คำสั่งสำหรับโปรโตคอลไอซีเอ็มพี 19
2.5 คำสั่งที่ใช้ในส่วนของเป้าหมาย/กระโดด 20
2.5 คำสั่งที่ใช้ในส่วนของเป้าหมาย/กระโดด (ต่อ) 21

สารบัญรูป

รูป		หน้า
2.1	ขั้นตอนการ Encapsulation และ Demultiplexing	4
2.2	โครงสร้างทีซีพี/ไอพี	4
2.3	ไอพีเฮดเดอร์	5
2.4	ไอซีเอ็มพีเฮดเดอร์	6
2.5	ยูดีพีเฮดเดอร์	7
2.6	ทีซีพีเฮดเดอร์	7
2.7	เส้นทางการเดินทางของแพคเกตเมื่อเข้ามาในระบบ (Filter table)	12
2.8	แผนภาพการไหลของแพคเกตในไอพีเทเบิล	14
2.9	ข้อมูลดิบของ Libpcap ที่ดักมาได้	23
2.10	โครงสร้างของเฟรม	23
2.11	เฮดเดอร์ของไอพี	24
3.1	โครงสร้างซอร์ฟแวร์	28
3.2	ภาพรวมการทำงานของระบบ	29
3.3	ลำดับขั้นตอนการทำงานของระบบ	30
3.4	Use case Diagram	30
3.5	Flowchart แยกตามโปรโตคอลที่กำหนด	31
3.6	Flowchart แยกตามไอพีแอดเดรส และพอร์ท	32
3.7	Flowchart แสดงจำนวนของ Top Loader	33
3.8	หน้าต่างที่ใช้ในการตั้งค่าเริ่มต้นการใช้งาน	34
3.9	หน้าต่างที่ใช้ในการสร้างกฎ	35
3.10	หน้าต่างลอค	36
3.11	หน้าต่างเลือกใช้กฎในแต่ละอินเตอร์เฟสตลอดเวลา	37
4.1	ลอคไฟล์แพคเกตของสนอร์ท	38
4.2	อินเตอร์เฟสโปรแกรมที่ได้ทำการทำลอง	39
4.3	การอินเจ็กท์โค๊ด	39
4.4	ทดลองเขียนกฎของไอพีเทเบิล	40
4.5	Interface Diagram ของระบบ	40
4.5	จากเครื่องเซิร์ฟเวอร์ ใช้ HFS ในการทดสอบว่า IP Tables ทำงานได้ตามต้องการ	42

สารบัญรูป (ต่อ)

รูป	หน้า
4.6 กำหนดจากเครื่อง Gateway ให้ออก Interface eth0 (IP Address 161.246.5.96)	42
4.7 มีการ Request จากเครื่องที่เป็น Gateway ด้วย IPaddress 161.246.5.96	43
4.8 กำหนดจากเครื่อง Gateway ให้ออก Interface eth2 (IPaddress 161.246.5.99)	43
4.9 มีการ Request จากเครื่องที่เป็น Gateway	44
4.10 กำหนด rule Protocol P2P ให้กับ Alternate WAN	44
4.11 Log File ที่ Alert	45
4.12 เราท์ติ้งเทเบิล	45
4.13 ทราฟฟิก P2P ที่วิ่งผ่าน Alternate WAN	46
4.14 กำหนด rule IP address ให้กับ Alternate WAN	47
4.15 ทราฟฟิกของ IP 192.168.1.2 ที่วิ่งผ่าน Alternate WAN	47
4.16 กำหนด rule Port ให้กับ Alternate WAN	48
4.17 ทราฟฟิกที่ใช้งาน port 80 วิ่งผ่าน Alternate WAN	48
4.18 กำหนด rule Most Bandwidth Usage ให้กับ Alternate WAN	49
4.19 log IP ที่ใช้งานสูงสุด	49
ข.1 หน้าต่างเริ่มต้นโปรแกรม	56
ข.2 หน้าต่าง ตั้งค่าอินเตอร์เฟส	57
ข.3 ทำการใส่ค่าเริ่มต้นของอินเตอร์เฟส	57
ข.4 ข้อความแจ้งเตือนเมื่อคลิกปุ่ม CANCLE	58
ข.5 ข้อความแจ้งเตือนเมื่อคลิกปุ่ม CLEAR	58
ข.6 หน้าต่างปรับแต่งกฎ	59
ข.7 การเพิ่มกฎ	59
ข.8 คลิกปุ่ม SAVE เพื่อนบันทึกกฎ	60
ข.9 รายชื่อกฎ ใน List Box	60
ข.10 ข้อความแจ้งเตือนเมื่อคลิกปุ่ม DELETE	61
ข.11 การลบกฎ	61
ข.12 หน้าต่าง Real-Time	62
ข.13 การสวิตช์กฎให้กับอินเตอร์เฟส	62
ข.14 หน้าต่างลอค	63

บทที่ 1
บทนำ

1.1 ที่มาและความสำคัญของโครงงาน

ในปัจจุบันอินเตอร์เน็ทเป็นที่แพร่หลายอย่างกว้างขวางทั่วโลก ซึ่งเป็นเหตุให้เกิดการวิวัฒนาการของระบบอินเตอร์เน็ทอย่างรวดเร็ว ทั้งความเร็วในการเชื่อมต่อปริมาณของข้อมูลในระบบและการแลกเปลี่ยนข้อมูลในระบบ ซึ่งมีความสะดวกสบายขึ้นอย่างมาก ด้วยเหตุนี้เองจึงทำให้เกิดความต้องการในการใช้งานอินเตอร์เน็ทที่มากขึ้น ทั้งในครัวเรือน องค์กร และสถานศึกษา ทำให้เกิดปัญหาปริมาณแบนด์วิดธ์ (Bandwidth) ไม่พอเพียงต่อการใช้งาน เนื่องจากผู้ใช้แต่ละคนมีการใช้งานในปริมาณที่มากแต่ทรัพยากรนั้นมีอยู่อย่างจำกัด ซึ่งการใช้งานนั้นมีทั้งแบบที่เกี่ยวข้องกับงานภายในองค์กร สถาบันหรือสถานศึกษา และที่ไม่เกี่ยวข้อง เช่น การใช้งานโปรแกรมจำพวกบิททอเรนท์ (Bit Torrent) หรือการถ่ายโอนข้อมูลปริมาณที่มากเกินพอดีด้วยเอฟทีพีซึ่งการใช้งานบางประเภท อาจทำให้เกิดการใช้แบนด์วิดธ์อย่างสิ้นเปลือง และส่งผลกระทบต่อระบบส่วนรวม

ระบบตรวจจับและคัดแยกทราฟฟิกที่ผิดปกตินี้ จะเข้าช่วยแบ่งเบาภาระของเครือข่าย โดยจะทำการแยกทราฟฟิกที่ผิดปกติคือ มีการใช้งานที่เยอะเกินความพอดีออกไปอยู่บนสายเชื่อมสัญญาณอินเตอร์เน็ทสายหนึ่งและทราฟฟิกที่ผิดปกติอีกสายหนึ่ง ซึ่งจะส่งผลให้การใช้งานเกินพอดีเหล่านี้ไม่ส่งผลกระทบต่อส่วนกลางและผู้ใช้งานปกติ

1.2 จุดมุ่งหมายและวัตถุประสงค์ของโครงงาน

1) เพื่อพัฒนาระบบที่สามารถตรวจสอบความผิดปกติของการใช้งานระบบเครือข่าย
2) เพื่อพัฒนาระบบตรวจสอบความผิดปกติของเครือข่ายที่สามารถแบ่งระดับความผิดปกติของเครือข่าย
3) เพื่อพัฒนาระบบตรวจสอบความผิดปกติของเครือข่ายให้สามารถคัดแยกทราฟฟิกที่ผิดปกติออกจากกัน
4) เพื่อพัฒนาระบบตรวจสอบความผิดปกติของเครือข่ายที่สามารถส่งแพคเกตที่ปกติและผิดปกติทั้งสองกลุ่มนี้ไปคนละเส้นทางได้

1.3 ขอบเขตของโครงงาน

1) ระบบสามารถตรวจสอบความผิดปกติของการใช้งานระบบเครือข่ายหรือพฤติกรรมที่สื่อได้ว่าเป็นการใช้งานที่ผิดปกติ

2) ระบบดังกล่าวสามารถระบุความผิดปกติในหลากหลายระดับ เช่น แบ่งแยกตามหมายเลขไอพีแอดเดรส แบ่งแยกตามหมายเลขพอร์ต
3) ระบบสามารถคัดแยกทราฟฟิกของแอปพลิเคชั่นเป้าหมาย เช่น Bit Torrent, HTTP ได้
4) ระบบทำงานบนระบบปฏิบัติการ Linux

1.4 ขั้นตอนการดำเนินงาน

1) ศึกษาหาความรู้ในเรื่องต่างๆ และโปรแกรมที่เกี่ยวข้อง พร้อมทั้งศึกษาวิธีการและลักษณะการใช้งานเพื่อนำมาประยุกต์ใช้กับระบบ
2) ทำการทดลองเขียนโปรแกรม
3) สรุปผลการศึกษาและการทดลองต่างๆพร้อมจัดทำเอกสารโครงงาน

1.5 ประโยชน์ที่คาดว่าจะได้รับ

1) ลดความหนาแน่นของทราฟฟิกในช่องการใช้งานปกติ
2) ลดค่าใช้จ่ายขององค์กรหรือสถาบันในการขยายปริมาณแบนด์วิดธ์ของเครือข่าย
3) สามารถบ่งบอกถึงต้นเหตุของการใช้งานเครือข่ายที่ส่งผลกระทบต่อระบบเครือข่ายส่วนรวม
4) ทำให้เครือข่ายสามารถใช้งานได้ตามปกติ แม้มีผู้ใช้บางกลุ่มที่ใช้งานเครือข่ายปริมาณมากอยู่ในขณะนั้น

1.6 ส่วนประกอบของปริญญานิพนธ์

ปริญญานิพนธ์ฉบับนี้แบ่งออกเป็น 5 บทดังต่อไปนี้

บทที่ 1 กล่าวถึง ความเป็นมา, จุดมุ่งหมายและวัตถุประสงค์, ขอบเขตของโครงงาน, ขั้นตอนการดำเนินงาน รวมไปถึงประโยชน์ที่คาดว่าจะได้รับ

บทที่ 2 กล่าวถึง ทฤษฎีที่เกี่ยวข้องและเทคโนโลยีที่ใช้

บทที่ 3 กล่าวถึง หลักการออกแบบในการพัฒนาระบบซึ่งในบทนี้จะอธิบายรายละเอียดต่างๆในการพัฒนาส่วนต่างๆของโครงงาน

บทที่ 4 กล่าวถึง การทดสอบการทำงาน

บทที่ 5 กล่าวถึง บทสรุปของโครงงานและแนวทางในการพัฒนาต่อ

บทที่ 2
ทฤษฎีที่เกี่ยวข้องและเทคโนโลยีที่ใช้

2.1 ทีซีพี/ไอพี

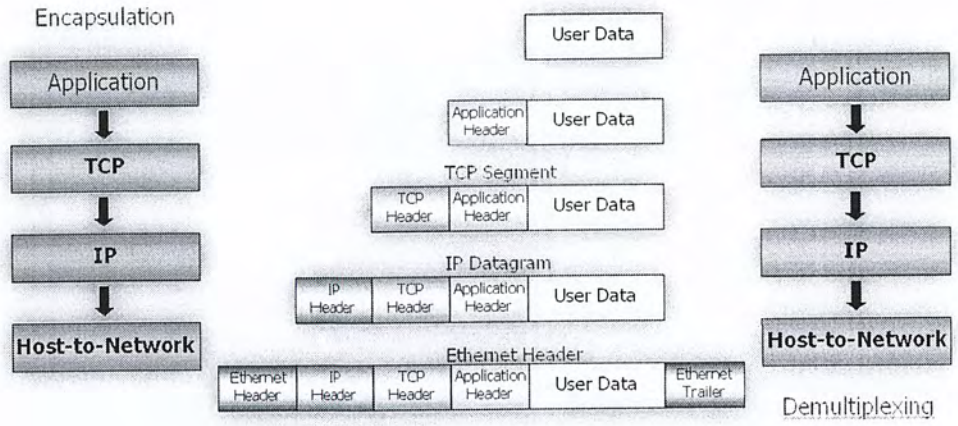
ทีซีพี/ไอพี (TCP/IP-Transmission Control Protocol/Internet Protocol) เป็นชุดของโปรโตคอลที่ถูกใช้ในการสื่อสารผ่านเครือข่ายอินเทอร์เน็ต โดยมีวัตถุประสงค์เพื่อให้สามารถใช้สื่อสารจากต้นทางข้ามเครือข่ายไปยังปลายทางได้ และสามารถหาเส้นทางที่จะส่งข้อมูลไปได้เองโดยอัตโนมัติ ถึงแม้ว่าในระหว่างทางอาจจะผ่านเครือข่ายที่มีปัญหา โปรโตคอลก็ยังคงหาเส้นทางอื่นในการส่งผ่านข้อมูลไปให้ถึงปลายทางได้ ชุดโปรโตคอลนี้ได้รับการพัฒนามาตั้งแต่ปี 1960 ซึ่งถูกใช้เป็นครั้งแรกในเครือข่ายอาร์พาเน็ต (ARPANET) ซึ่งต่อมาได้ขยายการเชื่อมต่อไปทั่วโลกเป็นเครือข่ายอินเตอร์เน็ต ทำให้ ทีซีพี/ไอพี เป็นที่ยอมรับอย่างกว้างขวางจนถึงปัจจุบัน

ทีซีพี/ไอพี มีจุดประสงค์ของการสื่อสารตามมาตรฐานสามประการคือ

1) เพื่อใช้ติดต่อสื่อสารระหว่างระบบที่มีความแตกต่างกัน
2) ความสามารถในการแก้ไขปัญหาที่เกิดขึ้นในระบบเครือข่าย เช่นในกรณีที่ผู้ส่งและผู้รับยังคงมีการติดต่อกันอยู่ แต่โหนดกลางที่ใช้เป็นผู้ช่วยรับ-ส่งเกิดเสียหายใช้การไม่ได้ หรือสายสื่อสารบางช่วงถูกตัดขาด กฎการสื่อสารนี้จะต้องสามารถจัดหาทางเลือกอื่นเพื่อทำให้การสื่อสารดำเนินต่อไปได้โดยอัตโนมัติ
3) มีความคล่องตัวต่อการสื่อสารข้อมูลได้หลายชนิดทั้งแบบที่ไม่มีความเร่งด่วน เช่น การจัดส่งแฟ้มข้อมูล และแบบที่ต้องการรับประกันความเร่งด่วนของข้อมูล เช่น การสื่อสาร

2.1.1 เอนแคปซูเลชั่น/ดีมัลติเพล็กซิ่ง (Encapsulation/Demultiplexing)

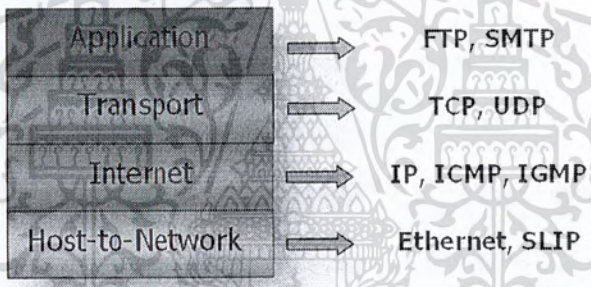
การส่งข้อมูลผ่านในแต่ละเลเยอร์ แต่ละเลเยอร์จะทำการประกอบข้อมูลที่ได้รับมากับข้อมูลส่วนควบคุมซึ่งถูกนำมาไว้ในส่วนหัวของข้อมูลเรียกว่า เฮดเดอร์ (Header) ภายในเฮดเดอร์จะบรรจุข้อมูลที่สำคัญของโปรโตคอลที่ทำการเอ็นแคปซูเลต (Encapsulate) เมื่อผู้รับได้รับข้อมูลก็จะเกิดกระบวนการทำงานย้อนกลับคือ โปรโตคอลเดียวกันทางฝั่งผู้รับก็จะได้รับข้อมูลส่วนที่เป็นเฮดเดอร์ก่อนและนำไปประมวลและทราบว่าข้อมูลที่ตามมามีลักษณะอย่างไร ซึ่งกระบวนการย้อนกลับนี้เรียกว่า ดีมัลติเพล็กซิ่ง (Demultiplexing) ดังรูป 2.1



รูป 2.1 ขั้นตอนการ Encapsulation และ Demultiplexing

2.1.2 โครงสร้างของทีซีพี/ไอพี

ในแต่ละเลเยอร์ (Layer) ของโครงสร้าง TCP/IP สามารถอธิบายได้ดังรูป 2.2



รูป 2.2 โครงสร้างทีซีพี/ไอพี

2.1.2.1 ชั้นโฮสต์-เครือข่าย (Host-to-Network Layer)

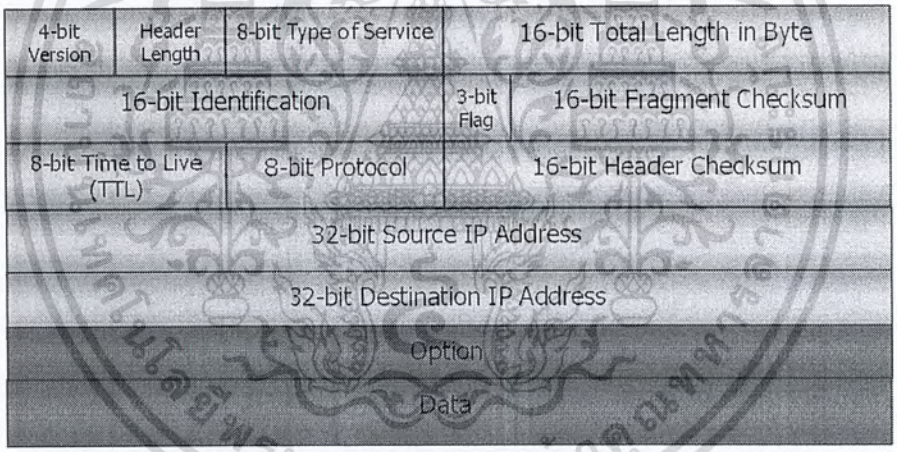
โพรโตคอลสำหรับการควบคุมการสื่อสารในชั้นนี้เป็นสิ่งที่ไม่มีการกำหนดรายละเอียดอย่างเป็นทางการ หน้าที่หลักคือการรับข้อมูลจากชั้นสื่อสารไอพีมาแล้วส่งไปยังโหนด (Node) ที่ระบุไว้ในเส้นทางเดินข้อมูลทางด้านผู้รับก็จะทำงานในทางกลับกัน คือรับข้อมูลจากสายสื่อสารแล้วนำส่งให้กับโปรแกรมในชั้นสื่อสารอินเทอร์เน็ต

2.1.2.2 ชั้นสื่อสารอินเทอร์เน็ต (The Internet Layer)

ใช้ประเภทของระบบการสื่อสารที่เรียกว่า ระบบเครือข่ายแบบสลับช่องสื่อสารระดับแพคเกต (Packet Switching Network) ซึ่งเป็นการติดต่อแบบไม่ต่อเนื่อง (Connectionless) หลักการทำงานคือการปล่อยให้ข้อมูลขนาดเล็กที่เรียกว่า แพคเกต (Packet) สามารถไหลจากโหนดผู้ส่งไปตามโหนดต่างๆ ในระบบจนถึงจุดหมายปลายทางได้โดยอิสระ หากว่ามีการส่งแพคเกตออกมาเป็นชุดโดยมีจุดหมายปลายทางเดียวกันในระหว่าง การเดินทางในเครือข่าย แพคเกตแต่ละ

ตัวในชุดนี้ก็จะเป็นอิสระแก่กันและกัน ดังนั้นแพคเกตที่ส่งไปถึงปลายทางอาจจะไม่เป็นไปตามลำดับก็ได้ ในชั้นนี้มีโปรโตคอลที่สำคัญดังต่อไปนี้

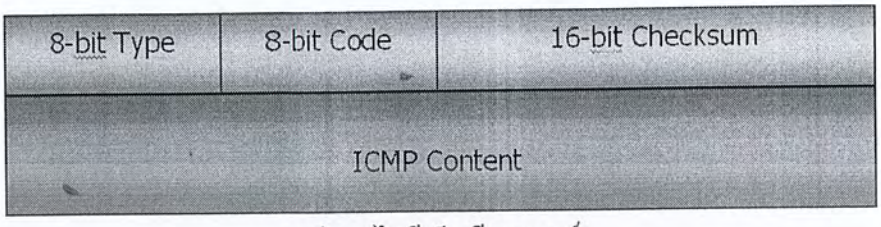
1) ไอพี (IP-Internet Protocol) ไอพีเป็นโปรโตคอลในระดับเน็ตเวิร์คเลเยอร์ ทำหน้าที่จัดการเกี่ยวกับแอดเดรสและข้อมูลและควบคุมการส่งข้อมูลบางอย่างที่ใช้ในการหาเส้นทางของแพคเกตซึ่งกลไกในการหาเส้นทางของไอพีจะมีความสามารถในการหาเส้นทางที่ดีที่สุด และสามารถเปลี่ยนแปลงเส้นทางได้ในระหว่างการส่งข้อมูล และมีระบบการแยกและประกอบดาต้าแกรม (Datagram) เพื่อรองรับการส่งข้อมูลระดับดาต้าลิงค์ที่มีขนาดเอ็มทียู (MTU-Maximum Transmission Unit) ที่แตกต่างกันทำให้สามารถนำไอพีไปใช้บนโปรโตคอลอื่นได้หลากหลาย เช่น อีเทอร์เนต, โทเคนริงหรือ แอปเปิ้ลทอล์ก การเชื่อมต่อของไอพีเพื่อทำการส่งข้อมูลจะเป็นแบบคอนเนกชันเลส (Connectionless) กรณีที่มีการแบ่งข้อมูลออกเป็นส่วนย่อยๆ (Fragmentation) และถูกนำไปรวมเป็นดาต้าแกรมเดิมเมื่อถึงปลายทาง ดังรูป 2.3



รูป 2.3 ไอพีเฮดเดอร์

2) ไอซีเอ็มพี (ICMP-Internet Control Message Protocol) ไอซีเอ็มพีเป็นโปรโตคอลที่ใช้ในการตรวจสอบและรายงานสถานภาพของดาต้าแกรม (Datagram) ในกรณีที่เกิดปัญหากับดาต้าแกรม เช่น เราเตอร์ไม่สามารถส่งดาต้าแกรมไปถึงปลายทางได้ ไอซีเอ็มพีจะถูกส่งออกไปยังโฮสต์ต้นทางเพื่อรายงานข้อผิดพลาดที่เกิดขึ้นอย่างไรก็ดีไม่มีอะไรรับประกันได้ว่าไอซีเอ็มพีเมสเซจที่ส่งไปจะถึงผู้รับจริงหรือไม่ หากมีการส่งดาต้าแกรมออกไปแล้วไม่มีไอซีเอ็มพีเมสเซจฟ้องเออเรอร์ (Error) กลับมา ก็แปลความหมายได้สองกรณีคือ ข้อมูลถูกส่งไปถึงปลายทางอย่างเรียบร้อยหรืออาจจะมีปัญหาในการสื่อสารทั้งการส่ง

ดาต้าแกรมและไอซีเอ็มพีเมสเซจที่ส่งกลับมาก็มีปัญหาระหว่างทางก็ได้ ไอซีเอ็มพีจึงเป็นโปรโตคอลที่ไม่มีความน่าเชื่อถือ (Unreliable) ซึ่งจะเป็นหน้าที่ของโปรโตคอลในเนตเวิร์กเลเยอร์ในการจัดการให้การสื่อสารนั้นๆ มีความน่าเชื่อถือ ดังรูป 2.4



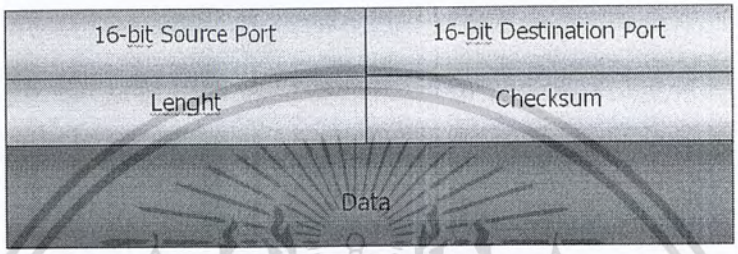
รูป 2.4 ไอซีเอ็มพีเฮดเดอร์

2.1.2.3 ชั้นสื่อสารนำส่งข้อมูล (Transport Layer)

แบ่งเป็นโพรโตคอล 2 ชนิดตามลักษณะ ลักษณะแรกเรียกว่า โปรโตคอลทีซีพี (TCP-Transmission Control Protocol) เป็นแบบที่มีการกำหนดช่วงการสื่อสารตลอดระยะเวลาการสื่อสาร (Connection Oriented) ซึ่งจะยอมให้มีการส่งข้อมูลเป็นแบบไบท์สตรีม (Byte stream) ที่ไว้ใจได้โดยไม่มีข้อผิดพลาดข้อมูลที่มีปริมาณมากจะถูกแบ่งออกเป็นส่วนเล็กๆ เรียกว่า เมสเซจ (Message) ซึ่งจะถูกส่งไปยังผู้รับผ่านทางชั้นสื่อสารของอินเทอร์เน็ท ทางฝ่ายผู้รับจะนำ Message มาเรียงต่อกันตามลำดับเป็นข้อมูลตัวเดิม ทีซีพียังมีความสามารถในการควบคุมการไหลของข้อมูลเพื่อป้องกันไม่ให้ผู้ส่ง ส่งข้อมูลเร็วเกินกว่าที่ผู้รับจะทำงานได้ทันอีกด้วย โปรโตคอลการนำส่งข้อมูลแบบที่สองเรียกว่า โปรโตคอลยูดีพี (UDP-User Datagram Protocol) เป็นการติดต่อแบบไม่ต่อเนื่อง (Connectionless) มีการตรวจสอบความถูกต้องของข้อมูลแต่จะไม่มีการแจ้งกลับไปยังผู้ส่ง จึงถือได้ว่าไม่มีการตรวจสอบความถูกต้องของข้อมูล อย่างไรก็ตามวิธีการนี้มีข้อดีในด้านความรวดเร็วในการส่งข้อมูล จึงนิยมใช้ในระบบผู้ให้และผู้ใช้บริการ (Client/Server system) ซึ่งมีการสื่อสารแบบถาม/ตอบ (Request/Reply) นอกจากนั้นยังใช้ในการส่งข้อมูลประเภทภาพเคลื่อนไหวหรือการส่งเสียง (Voice) ทางอินเทอร์เน็ท

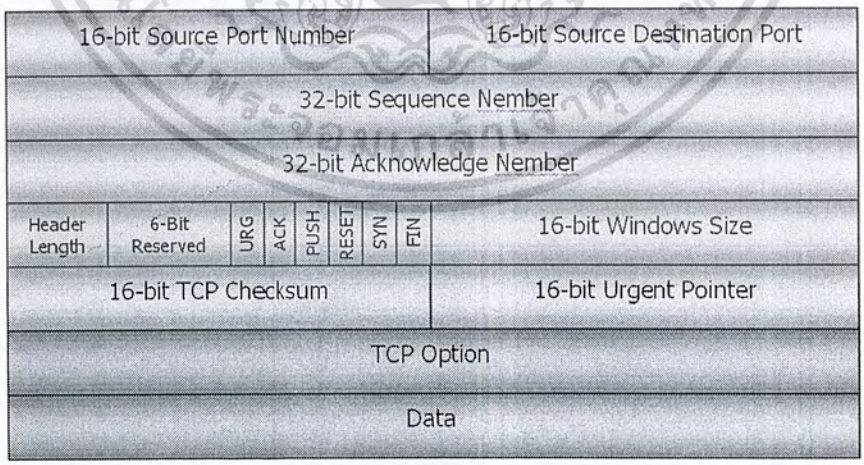
1) ยูดีพี (UDP-User Datagram Protocol) เป็นโปรโตคอลที่อยู่ในชั้น Transport Layer เมื่อเทียบกับ OSI Model โดยการส่งข้อมูลของยูดีพีนั้นจะเป็นการส่งครั้งละ 1 ชุดข้อมูลเรียกว่ายูดีพีดาต้าแกรม (UDP datagram) ซึ่งจะไม่มีความสัมพันธ์กันระหว่างดาต้าแกรมและจะไม่มีกลไกการตรวจสอบความสำเร็จในการรับส่งข้อมูล กลไกการตรวจสอบโดยเช็คซัม (Checksum) ของยูดีพีนั้น เพื่อเป็นการป้องกันข้อมูลที่อาจจะถูกแก้ไขหรือมีความผิดพลาดระหว่างการส่งและหากเกิดเหตุการณ์ดังกล่าวปลายทางจะได้รู้ว่ามีข้อผิดพลาดเกิดขึ้น

แต่มันจะเป็นการตรวจสอบเพียงฝ่ายเดียวเท่านั้น โดยในข้อกำหนดของยูดีพี หากพบว่าเกิดความผิดพลาดก็ให้ผู้รับปลายทางทำการทิ้งข้อมูลนั้น แต่จะไม่มีการแจ้งกลับไปยังผู้ส่งแต่อย่างใด การรับส่งข้อมูลแต่ละครั้งหากเกิดข้อผิดพลาดในระดับไอพี เช่น ส่งไม่ถึง, หมดเวลา ผู้ส่งจะได้รับข้อความจากระดับไอพีเป็นไอซีเอ็มพีเออเรอร์เมสเซจ (ICMP Error Message) แต่เมื่อข้อมูลส่งถึงปลายทางถูกต้องแต่เกิดข้อผิดพลาดในส่วนของยูดีพีเอง จะไม่มีการยืนยันหรือแจ้งให้ผู้ส่งทราบแต่อย่างใด ดังรูป 2.5



รูป 2.5 ยูดีพี เฮดเดอร์

2) TCP (Transmission Control Protocol) อยู่ในชั้นทรานสปอร์ตเช่นเดียวกับยูดีพี ทำหน้าที่จัดการและควบคุมการรับส่งข้อมูล ซึ่งมีความสามารถและรายละเอียดมากกว่ายูดีพีโดยดาต้าแกรมของทีซีพีจะมีความสัมพันธ์ต่อเนื่องกันและมีกลไกควบคุมการรับส่งข้อมูลให้มีความถูกต้อง (Reliable) และมีการสื่อสารอย่างเป็นกระบวนการ (Connection-oriented) ดังรูป 2.6



รูป 2.6 ทีซีพี เฮดเดอร์

2.1.2.4 ชั้นสื่อสารการประยุกต์ (Application Layer)

มีโปรโตคอลสำหรับสร้างจอเทอร์มินัลเสมือนเรียกว่า เทลเนต(TELNET) โปรโตคอลสำหรับการจัดการแฟ้มข้อมูลเรียกว่าเอฟทีพี (FTP) และโพรโตคอลสำหรับการให้บริการจดหมายอิเล็กทรอนิกส์เรียกว่าเอสเอ็มทีพี (SMTP) โดยโปรโตคอลสำหรับสร้างจอเทอร์มินัลเสมือนช่วยให้ผู้ใช้สามารถติดต่อกับเครื่องโฮสต์ที่อยู่ไกลออกไปโดยผ่านอินเทอร์เน็ท และสามารถทำงานได้เสมือนกับว่ากำลังนั่งทำงานอยู่ที่เครื่องโฮสต์นั้น โปรโตคอลสำหรับการจัดการแฟ้มข้อมูลช่วยในการคัดลอกแฟ้มข้อมูลมาจากเครื่องอื่นที่อยู่ในระบบเครือข่ายหรือส่งสำเนาแฟ้มข้อมูลไปยังเครื่องใดๆ ก็ได้ โปรโตคอลสำหรับให้บริการจดหมายอิเล็กทรอนิกส์ช่วยในการจัดส่งข้อความไปยังผู้ใช้ในระบบหรือรับข้อความที่มีผู้ส่งเข้ามา

2.2 ไฟร์วอลล์ (Firewall)

ไฟร์วอลล์เป็นระบบรักษาความปลอดภัยของคอมพิวเตอร์แบบหนึ่ง ซึ่งตอนนี้ถือว่าเป็นระบบรักษาความปลอดภัยที่นิยมใช้กันมากที่สุดและยังถือว่าเป็นการรักษาความปลอดภัยเบื้องต้นอีกด้วย โดยไฟร์วอลล์สามารถรักษาความปลอดภัยได้ทั้งเฉพาะตัวเครื่องหรือทั้งระบบเครือข่ายคอมพิวเตอร์ ซึ่งหากเปรียบไฟร์วอลล์ก็คือกำแพงที่ล้อมบ้านในการทำงานเฉพาะเครื่องหรือหากในเครือข่ายก็จะเป็นกำแพงที่ล้อมเมืองเอาไว้ โดยพอร์ตที่จะใช้ในการทำงานนั้นถือว่าเป็นประตูหากทำการเปิดพอร์ตทั้งหมดไว้จะเปรียบเสมือนเปิดประตูต้อนรับผู้มาเยือนทั้งหมด

ไฟร์วอลล์อาจจะเป็นฮาร์ดแวร์หรือซอฟต์แวร์ หลักการก็คือต้องมีเครือข่ายอย่างน้อยสองเครือข่ายเชื่อมต่อกัน โดยเครือข่ายหนึ่งต้องการที่จะป้องกันไม่ให้ถูกเปิดเผยจากอีกเครือข่ายหนึ่ง ไฟร์วอลล์จะอยู่ระหว่างทางเชื่อมของจุดต่างๆหรืออยู่ที่เกตเวย์ระหว่างสองเครือข่าย ซึ่งโดยทั่วไปจะเป็นเครือข่ายส่วนตัวกับเครือข่ายสาธารณะ เพื่อทำหน้าที่ตรวจสอบข้อมูลที่เข้ามาและจะบลอคหรืออนุญาตให้ข้อมูลนั้นผ่านไปได้นั้นขึ้นอยู่กับกฎหรือการตั้งค่าไฟร์วอลล์การตั้งชื่อของไฟร์วอลล์มาจากความจริงที่ว่า การที่เครือข่ายหนึ่งได้รับความเสียหายและแผ่ขยายไปยังซับเนตอื่นๆ ซึ่งก็เหมือนประตูที่กันไฟหรือกำแพงกันไฟไม่ให้ลุกลามไปทั่วอย่างรวดเร็ว

2.2.1 ประเภทของไฟร์วอลล์

ไฟร์วอลล์สามารถแบ่งได้ 2 ประเภทตามลักษณะการใช้งานคือไฟร์วอลล์ส่วนบุคคลและเนตเวิร์กไฟร์วอลล์ซึ่งมีรายละเอียดดังนี้

2.2.1.1 ไฟร์วอลล์ส่วนบุคคล

เป็นโปรแกรมไฟร์วอลล์ที่ใช้ติดตั้งลงในเครื่องคอมพิวเตอร์ส่วนบุคคลของผู้ใช้ซึ่งมักจะเป็นเครื่องเดสก์ท็อปหรือโน้ตบุ๊ค เพื่อใช้ปกป้องเครื่องไม่ให้ถูกโจมตีหรือบุกรุกผ่านเครือข่าย ตัวอย่างไฟร์วอลล์ส่วนบุคคล ที่เห็นได้ชัดเจนที่สุดคือโปรแกรมวินโดวส์ไฟร์วอลล์ที่ติดมากับระบบปฏิบัติการวินโดวส์

2.2.1.2 Network ไฟร์วอลล์

เรียกอีกอย่างว่าไฟร์วอลล์ระดับองค์กร คือไฟร์วอลล์ที่ใช้ติดตั้งไว้เป็นส่วนหนึ่งของระบบเครือข่ายด่านหน้า (Perimeter Firewall) เพื่อใช้ป้องกันการบุกรุกผ่านทางเครือข่าย Network ไฟร์วอลล์อาจจะมาในรูปของซอฟต์แวร์ที่ติดตั้งลงในเครื่องระดับเซิร์ฟเวอร์หรืออาจจะมาในรูปของอุปกรณ์หรือฮาร์ดแวร์สำเร็จรูปที่เพียงแต่ทำการตั้งค่าหรือคอนฟิกให้ถูกต้องเท่านั้นก็สามารถใช้งานได้เลย โดยทั่วไป Network ไฟร์วอลล์จะมีราคาแพงกว่าไฟร์วอลล์ส่วนบุคคลมาก และมักจะนำไปติดตั้งไว้เป็นส่วนหนึ่งของระบบเครือข่ายด่านหน้า (Perimeter Firewall) ที่เชื่อมต่อกับเครือข่ายภายนอกหรืออินเตอร์เน็ท Network ไฟร์วอลล์มีทั้งแบบไฟร์วอลล์ฮาร์ดแวร์และไฟร์วอลล์ซอฟต์แวร์

2.2.2 คุณสมบัติของไฟร์วอลล์

คุณสมบัติทั่วไปของไฟร์วอลล์ นั้นจะมีอยู่ 3 อย่างด้วยกันคือ

1) ป้องกัน (Protect) ไฟร์วอลล์เป็นเครื่องมือที่ทำงานเชิงป้องกันโดยแพคเกต ที่จะสามารถผ่านเข้าออกได้นั้นจะต้องเป็นแพคเกตที่มันเห็นว่าปลอดภัยหากแพคเกตใดที่มันเห็นว่าไม่ปลอดภัย มันก็จะไม่อนุญาตให้ผ่าน โดยการตัดสินใจว่าแพคเกตปลอดภัยหรือไม่ขึ้นอยู่กับกฎพื้นฐานที่ได้กำหนดไว้
2) ควบคุมการเข้าถึง (Access Control) ไฟร์วอลล์จะควบคุมการเข้าถึงของโฮสต์ต่างๆ ให้เป็นไปตามกฎพื้นฐานที่ได้กำหนดไว้
3) กฎพื้นฐาน (Rule Base) ไฟร์วอลล์จะทำการควบคุมการเข้าถึงโดยอาศัยการเปรียบเทียบคุณสมบัติของแพคเกตที่จะผ่านเข้า-ออกกับกฎพื้นฐานที่ได้กำหนดไว้ ถ้ามีกฎข้อใดข้อหนึ่งห้าม มันก็จะไม่ยอมให้ปล่อยผ่าน

ตาราง 2.1 เปรียบเทียบระหว่างไฟร์วอลล์ซอฟต์แวร์กับไฟร์วอลล์ฮาร์ดแวร์

Software Firewall	Hardware Firewall
การติดตั้งซอฟต์แวร์สามารถติดตั้งลงบันเครื่องเซิร์ฟเวอร์ทั่วๆไปได้ ซึ่งอาจจะเป็นเครื่องที่ใช้ระบบปฏิบัติการ Windows 2000/2003 Server, Linux หรือ Solaris	ระบบปฏิบัติการที่ใช้ในฮาร์ดแวร์จะต้องเป็นของผู้ผลิตโดยเฉพาะเท่านั้น หรืออาจจะเป็นระบบปฏิบัติการ Linux ที่ผ่านการปรับแต่งด้านระบบความปลอดภัยให้เข้มงวดยิ่งขึ้น หรือที่เรียกว่า Security Hardening
ประสิทธิภาพการทำงานของไฟร์วอลล์ขึ้นอยู่กับประสิทธิภาพของเครื่องเซิร์ฟเวอร์นั้นๆ	ทำงานได้รวดเร็วและมีประสิทธิภาพมากกว่า เนื่องจากมีการออกแบบฮาร์ดแวร์ให้เหมาะสมกับการทำงานโดยเฉพาะ
มีความปลอดภัยน้อยกว่า เนื่องจากมักจะช่องโหวในระบบปฏิบัติการที่ถูกค้นพบบ่อยๆ ทำให้กลายเป็นช่องทางให้ระบบถูกเจาะได้ง่าย	มีความปลอดภัยสูงกว่า เนื่องจากระบบปฏิบัติการที่อยู่ภายในอุปกรณ์นั้นได้รับการปรับแต่งให้ทำงานเฉพาะด้านไว้แล้ว จึงมีโอกาสเกิดช่องโหวในตัวระบบปฏิบัติการน้อยมาก
การทำ Redundancy ทำได้ยากกและเสียค่าใช้จ่ายสูง เช่น จะต้องซื้อเครื่องคอมพิวเตอร์ที่เป็นเครื่องเซิร์ฟเวอร์ 2 เครื่อง และยังต้องซื้อค่า License ของตัวระบบปฏิบัติการและซอฟต์แวร์ของตัวไฟร์วอลล์รวมทั้งซอฟต์แวร์ในการทำ Redundancy หรือ Load Balancing เป็นจำนวน 2 ชุด ทำให้สิ้นเปลืองค่าใช้จ่ายมาก	อุปกรณ์บางรุ่นมีความสามารถในด้านของ Redundancy หรือ Fault Tolerance โดยภายในอุปกรณ์จะมีฮาร์ดแวร์สำรองที่พร้อมจะทำงานได้ทันที หากฮาร์ดแวร์ชุดแรกขัดข้อง หรือสามารถเลือกที่จะให้ฮาร์ดแวร์ทั้งสองชุดทำงานพร้อมกันช่วยแบ่งเบาการทำงานของอุปกรณ์ได้ (Load Balacing)
มีความยืดหยุ่นสูงเนื่องจากสามารถติดตั้งโปรแกรมอื่นๆเพิ่มเติมได้ สามารถอัพเกรดฮาร์ดแวร์เพิ่มได้ เช่น การเพิ่มฮาร์ดดิสก์ หน่วยความ การ์ดแลน เป็นต้น	มีความยืดหยุ่นน้อย การอัพเกรดฮาร์ดแวร์อาจจำเป็นต้องอาศัยฮาร์ดแวร์จากผู้ผลิตอุปกรณ์เท่านั้น นอกจากนี้ยังไม่สามารถติดตั้งซอฟต์แวร์อื่นๆเพิ่มเติมได้ ยกเว้นในกรณีที่ผู้ผลิตมีให้เท่านั้น

2.3 ระบบตรวจจับผู้บุกรุก (Intrusion Detection System)

คือ ซอฟท์แวร์หรือฮาร์ดแวร์ที่ได้รับการออกแบบมาเพื่อให้ตรวจสอบการเชื่อมต่อที่ไม่พึงประสงค์หรือความพยายามที่จะเข้ามาทำอันตรายต่อเครือข่าย โดยผ่านระบบต่างๆ เช่น อินเตอร์เน็ต ,แลน เป็นต้น โดยการโจมตีนั้นอาจจะเกิดจาก แครกเกอร์, เวิร์มหรือ มัลแวร์ ต่างๆ และข้อจำกัดของระบบตรวจจับผู้บุกรุก นั้นก็คือไม่สามารถที่จะตรวจสอบแพคเกตที่เข้ารหัสได้องค์ประกอบของไอดีเอสนั้นมีหลายหลายส่วนแต่ส่วนประกอบไอดีเอสที่สำคัญนั้นมีอยู่สามส่วนได้แก่

1) เซนเซอร์ (Sensor) คือส่วนที่จะสร้างเหตุการณ์ที่เกี่ยวกับความปลอดภัย
2) คอนโซล(Console) คือส่วนที่จะตรวจจับเหตุการณ์, แจ้งเตือน รวมไปถึงการควบคุมเซนเซอร์
3) เอนจิน (Engine) เป็นส่วนที่จะบันทึกเหตุการณ์จากเซนเซอร์ลงในฐานข้อมูลและจะแจ้งเตือนตามกฎที่ได้ตั้งเอาไว้

2.3.1 ประเภทของระบบตรวจจับผู้บุกรุก (IDS)

ระบบตรวจจับผู้บุกรุกแบ่งออกเป็น 2 ประเภทคือ

1) เน็ตเวิร์คเบสไอดีเอส (Network-Based IDS) คือ ระบบที่ตรวจดูแพคเกตที่วิ่งอยู่ในเครือข่ายและแจ้งเตือนถ้าพบหลักฐานที่คาดว่าจะเป็นการบุกรุกเครือข่ายไอดีเอส จะมีฐานข้อมูลที่ใช้สำหรับเปรียบเทียบเพื่อบอกว่าเป็นการพยายามที่จะบุกรุกเครือข่ายหรือไม่ ซึ่งข้อมูลนี้จะเรียกว่าซิกเนเจอร์ (Signature) โดยไอดีเอสจะใช้ซิกเนเจอร์ที่เก็บไว้เพื่อเปรียบเทียบกับข้อมูลที่ได้จากการวิเคราะห์แพคเกตที่วิ่งในเครือข่าย โดยส่วนใหญ่แล้วไอดีเอสประเภทนี้จะมีเน็ตเวิร์คการ์ด 2 การ์ดโดยเน็ตเวิร์คการ์ดตัวแรกจะเชื่อมต่อเข้ากับเครือข่ายที่ต้องการเฝ้าระวัง โดยเน็ตเวิร์คการ์ดตัวนี้จะไม่มีหมายเลขไอพี ส่วนเน็ตเวิร์คการ์ดอีกอันหนึ่งจะเชื่อมต่อเข้าอีกเครือข่ายหนึ่งเพื่อใช้สำหรับส่งการแจ้งเตือนภัยไปยังเซิร์ฟเวอร์ โดยเครือข่ายนี้จะต้องไม่ถูกเชื่อมต่อกับเครือข่ายหลักที่ไอดีเอสจะตรวจจับการบุกรุกเพื่อป้องกันไม่ให้ไอดีเอสถูกโจมตีเสียเอง

2) โฮสต์เบสไอดีเอส (Host-Based IDS) คือระบบที่ติดตั้งที่โฮสต์คอยเฝ้าระวังและตรวจจับความพยายามที่จะบุกรุกโฮสต์นั้น Host-Based ไอดีเอสเป็นซอฟแวร์ที่รันบนโฮสต์ โดยปกติไอดีเอสประเภทนี้จะวิเคราะห์ล็อค (Log) เพื่อค้นหาข้อมูลเกี่ยวกับการบุกรุกในระบบยูนิกส์นั้นลอคที่ไอดีเอสจะตรวจสอบ เช่น (ซิสลอค, แมสเซจ, ลาสต์ลอค ส่วนวินโดวส์ไอดีเอสก็จะตรวจสอบอีเวนท์ลอคต่างๆ เช่น ซิสเทม, แอพพลิเคชันและซีเคียวริตี้ เป็นต้น ไอดีเอสจะอ่านเหตุการณ์ใหม่ที่เกิดขึ้นในลอคและเปรียบเทียบกับกฎที่ตั้งไว้ก่อน ถ้าตรงก็จะแจ้งเตือนทันที ดังนั้นการที่ลอคจะตรวจจับการบุกรุกได้ระบบจะต้องบันทึกเหตุการณ์ต่างๆ ที่สำคัญที่เกิดขึ้นกับระบบในการลอคไฟล์ มิเช่นนั้นไอดีเอสก็จะไม่มีข้อมูลที่จะใช้วิเคราะห์ว่ามีการบุกรุกหรือไม่

2.4 ไอพีเทเบิล (IP Tables)

ไฟร์วอลล์บนระบบปฏิบัติการลินุกซ์ที่นิยมใช้คือไอพีเทเบิลเนื่องจากฟรีและค่อนข้างมีประสิทธิภาพสูง โดยสามารถใช้ลินุกซ์ทำไอพีเทเบิลได้โดยการติดตั้งไอพีเทเบิลซึ่งเป็นคำสั่งคอมมานไลน์บนลีนุกซ์ ที่ใช้ปรับแต่งให้กับระบบเพื่อให้สามารถกรองแพคเกต (Filter) และสามารถทำการแปลงแอดเดรส (NAT) ได้อีกด้วยโดยคำสั่งนี้มีมาพร้อมกับลีนุกซ์ เคอร์เนลเวอร์ชัน 2.4 ขึ้นไป สามารถใช้งานได้โดยไม่ต้องคอมไพล์เคอร์เนลใหม่

2.4.1 คำสั่งในไอพีเทเบิล

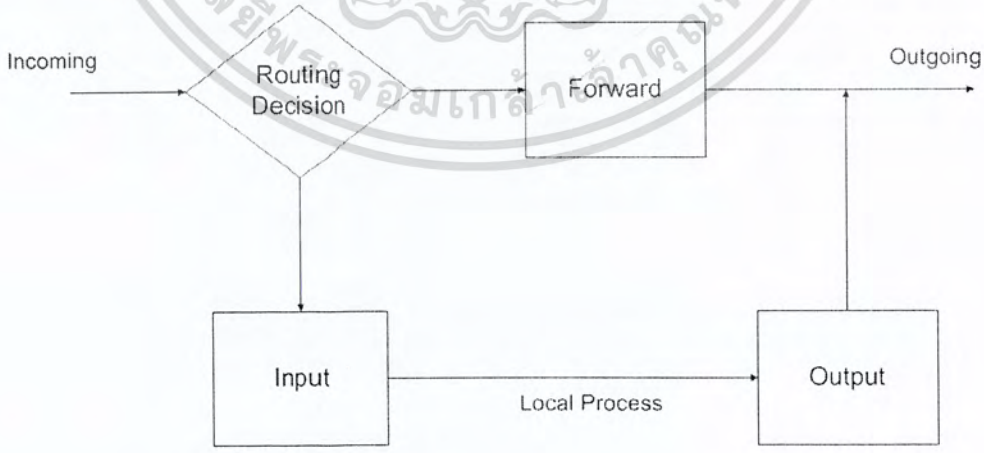
รูปแบบการใช้งานไอพีเทเบิลเบื้องต้น จะมีรูปแบบการใช้งานดังนี้คือ

#iptables [Table]<Command> <Match> <Target/jump>

โดยกฎ(Rule) ที่เขียนขึ้นจะเป็นตัวบอกเคอร์เนลว่าให้ทำอย่างไรในกรณีที่พบแพคเกตตรงตามที่ระบุไว้

[Tables] หมายถึง ตารางที่ต้องการระบุ เช่น iptables -t nat หมายถึงให้ทำงานกับ ตารางแนท ในกรณีที่ไม่ได้ระบุตารางไอพีเทเบิลจะถือว่าคำสั่งดังกล่าวระบุถึงฟิลเตอร์เทเบิลโดยอัตโนมัติ มี 3 ตาราง ดังรูป 2.8 คือ

1) ฟิลเตอร์เทเบิล (Filter Table) ใช้สำหรับกรองแพคเกตมี 3 บิวท์อินเชน คือ อินพุต (Input), เอาพุต (Output), ฟอเวิร์ด (Forward) เป็นตารางที่ใช้งานมากที่สุด เป็นจุดที่ใช้ในการตรวจสอบและควบคุมการผ่านเข้าออกของแพคเกตถ้าหากจะพิจารณาการไหลเวียนของแพคเกตเฉพาะในส่วนของฟิลเตอร์เทเบิล โดยไม่สนใจเทเบิลอื่นๆ นั้น ก็พอจะแสดงให้เห็นได้โดยเมื่อแพคเกตเข้ามาในระบบมันจะเข้าไปยังตัวประมวลผลเส้นทางเพื่อตัดสินใจว่าแพคเกตจะถูกส่งไปที่ใดในกรณีที่แพคเกตถูกส่งผ่านไปยังเครื่องอื่นแพคเกตนั้นจะต้องถูกตรวจสอบโดยกฎใน Forward Chain ถ้าแพคเกตนั้นมีเป้าหมายเป็นเครื่องปัจจุบัน (เครื่องที่รันไอพีเทเบิลอยู่นี้ เรียกอีกอย่างว่าลินุกซ์บอกซ์ (Linux box) ตัวแพคเกตจะถูกตรวจสอบโดยกฎในอินพุตเชนและในกรณีที่แพคเกตถูกสร้างจากเครื่องปัจจุบัน (ลินุกซ์บอกซ์) ตัวแพคเกตจะถูกตรวจสอบจากกฎ ใน Output Chain ก่อนที่จะถูกส่งออกไป ดังรูป 2.7

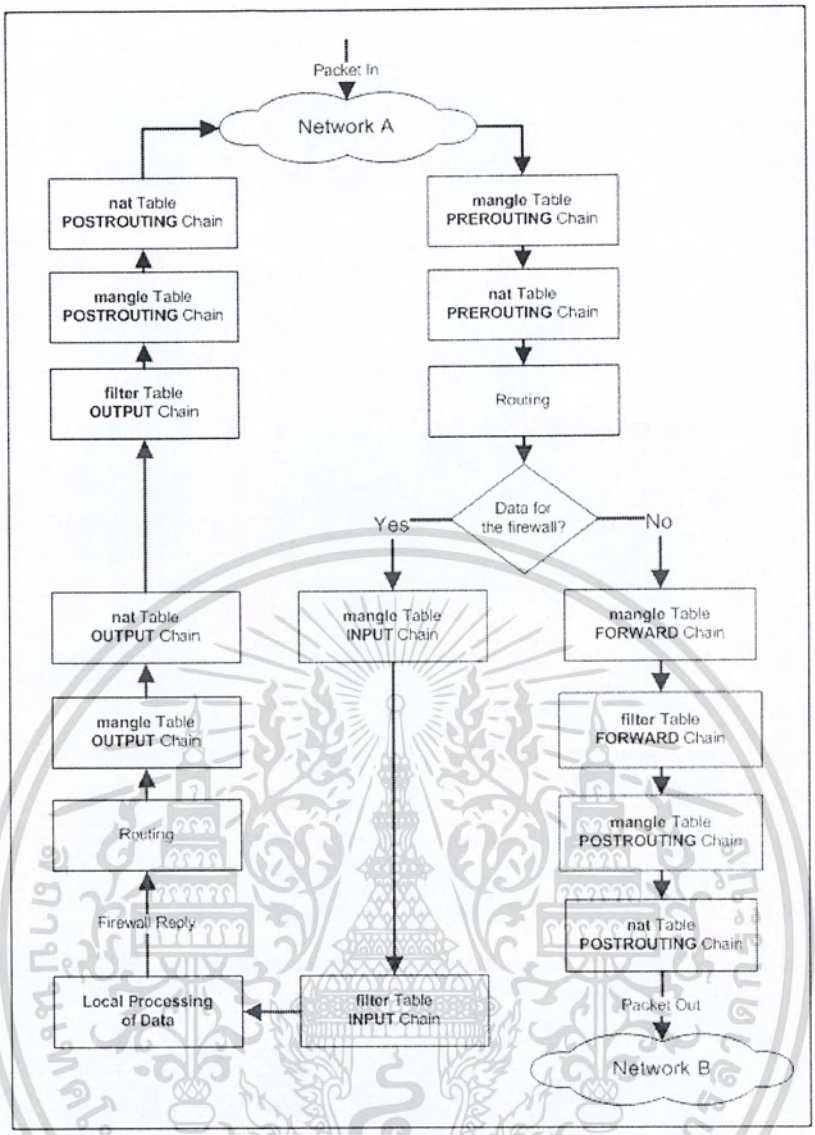


รูป 2.7 เส้นทางการเดินทางของแพคเกตเมื่อเข้ามาในระบบ (Filter table)

จากรูป 2.8 ไอพีเทเบิลประกอบไปด้วยบิวท์อินเชนจำนวน 3 เชน ซึ่งไม่สามารถลบได้คือ Input, Output, Forward เมื่อเครื่องคอมพิวเตอร์เริ่มทำงานในครั้งแรก ทั้งสาม Chain จะมีวิธีการปฏิบัติขั้นพื้นฐานเป็นแอคเซป (ACCEPT) ซึ่งหมายความว่าอนุญาตให้ทุกอย่างผ่านเข้าออกได้หมดและสำหรับฟอเวิร์ดเชนนั้นถึงแม้จะได้กำหนดให้วิธีการปฏิบัติเป็น Accept แล้วแพคเกตก็จะยังไม่สามารถถูก Forward ไปยังจุดหมายที่ต้องการได้ ตราบใดที่ยังไม่ได้ตั้งให้อีนาเบิลไอพีฟอเวิดดิ่ง (Enable IP Forwarding) ทั้งนี้โดยปกติแล้ว Forward=0 สามารถกำหนดให้ Enable IP Forwarding (Forward=1) ได้ โดยใช้คำสั่ง “echo "1" > /proc/sys/net/ip_forward” เพื่อกำหนดให้ IP Forwarding เป็น Enable เพื่อให้ ลินุกซ์บอกซ์ สามารถ Forward IP แพคเกต ได้ ในบางครั้งนั้นการใช้คำสั่งดังกล่าวทุกครั้งอาจจะไม่สะดวก สามารถแก้ไขไฟล์ Configuration ที่ /etc/sysctl.conf แล้วเซตให้ net.ipv4.ip_forward=1 เพื่อเป็นการแก้ไขแบบถาวรในกรณีที่ต้องการให้สนับสนุนการทำงานกับไดนามิกไอพีด้วยเช่น พีพีพี (PPP), เอสแอลไอพี (SSL IP), ดีเอชซีพี(DHCP) ก็สามารถทำได้โดยใช้คำสั่งecho "1" > /proc/sys/net/ipv4/ip_dynaddr ได้เช่นเดียวกัน

2) แมนเกิลเทเบิล (Mangle Table) เป็นตารางที่ใช้สำหรับแก้ไขข้อมูลทีโอเอส (TOS), ทีทีแอล (TTL), มาร์ค (Mark) ของแพคเกตจะใช้ในการทำ Routing ที่มีความซับซ้อนสูง มี 2 Building Chain คือพรีเราติ้งเชน (PREROUTING chain) สำหรับใช้แก้ไขแพคเกตก่อนที่จะเข้าสู่ไฟร์วอลล์และก่อนเข้าสู่ส่วนประมวลผลเส้นทางและ Output Chain สำหรับใช้แก้ไขแพคเกตที่ถูกสร้างโดยไฟร์วอลล์ก่อนที่มันจะถูกส่งไปยังส่วนประมวลผลเส้นทาง ซึ่งโดยปกติแล้วแทบจะไม่ได้ใช้งานและไม่ควรทำการกรองแพคเกตที่ตารางนี้รวมทั้งไม่ควรทำดีแนท (DNAT), เอสแนท (SNAT) หรือแมสคิวเรดดิ่ง (Masquerading) ที่ตารางนี้อย่างเด็ดขาดด้วย

3) แนทเทเบิล (Nat Table) เป็นตารางที่ใช้สำหรับทำการแปลงแอดเดรส (Network Address Translation) มี 3 บิวท์อินเชนคือ พรีเราติ้ง (Pre Routing), โพสเราติ้ง (Post Routing), เอาพุต (Output) เช่น เปลี่ยนค่าไอพีแอดเดรสต้นทางและปลายทาง (Source IP Address, Destination IP Address) จุดสำคัญอีกอย่างหนึ่งที่ต้องรู้ก็คือ มีเพียงแพกเกตแรกเท่านั้นที่เข้ามาที่ Chain นี้ ส่วนแพกเกตถัดไปนั้นจะถูกกระทำเหมือนที่แพกเกตแรกได้รับ ดังนั้นจึงไม่ควรทำการกรอง Packet ที่ Chain เหล่านี้ โดยใช้คำสั่ง “iptables -t nat” หมายถึงให้ทำงานกับ NAT Table ในกรณีที่ไม่ได้ระบุตาราง IP Tables จะถือว่าคำสั่งดังกล่าวระบุถึงฟิลเตอร์เทเบิลโดยอัตโนมัติ



รูป 2.8 แผนภาพการไหลของแพคเกตในไอพีเทเบิล

ตาราง 2.2 ใช้ในการเลือกแพคเกตโดยไฟร์วอลล์

Queue Type	Queue Function	Packet Transformation Chain in Queue	Chain Function
Filter	Packet filtering	FORWARD	คัดกรองแพคเกตที่ไปยังเซิฟเวอร์ผ่านอีกพอร์ตหนึ่งของไฟร์วอลล์
		INPUT	คัดกรองแพคเกตที่เข้ามายังไฟร์วอลล์
		OUTPUT	คัดกรองแพคเกตที่สร้างขึ้นจากไฟร์วอลล์
Nat	Network Address Translation	PREROUTING	การแปลงค่าแอดเดรสเกิดขึ้นก่อนจะทำการเราท์ติ้งจะทำการแปลงค่าแอดเดรสปลายทางให้เข้ากันกับเราท์ติ้งเทเบิลในไฟร์วอลล์ หรือเรียกว่า Destination NAT (DNAT)
		POSTROUTING	การแปลงแอดเดรสเกิดขึ้นหลังจากการเราท์ติ้งหากไม่จำเป็นต้องแก้ไขแอดเดรสปลายทาง ดังเช่น PREROUTING การแปลงนี้จะแปลงแอดเดรสต้นทางโดย NAT ในแบบ one-to-one หรือ one-to-many หรือเรียกว่า Source NAT (SNAT)
		OUTPUT	การแปลงค่าเน็ตเวิร์คแอดเดรสของแพคเกตที่ถูกสร้างโดยไฟร์วอลล์
Mangle	TCP header modification	PREROUTING POSTROUTING OUTPUT INPUT FORWARD	การแก้ไขบิท QoS ของแพคเกต TCP ก่อนเกิดการเราท์ติง

<Command> จะเป็นตัวสั่งให้ไอพีเทเบิลทำในสิ่งที่ต้องการ เช่น

#iptables -A INPUT

หมายถึงให้สร้างกฎต่อท้ายอินพุตเชนโดยเป็นแพคเกตที่วิ่งเข้ามายังตัวไฟร์วอลล์ (INPUT chain) ในฟิลเตอร์เทเบิล

คำสั่ง

-A เพิ่มกฎใหม่ต่อท้ายเชน (Append rule) เช่น

#iptables -A INPUT -p ALL -i eth0 -j ACCEPT

-D ลบกฎ (Delete rule) เช่น

#iptables -D INPUT --dport 80 -j DROP

-I เพิ่มกฎใหม่ ในเชน (Insert rule) เช่น

#iptables -I OUTPUT -p ALL -s 127.0.0.1/32 -j ACCEPT

-R แทนที่กฎเดิมด้วยกฎใหม่ (Replace rule)

-L แสดงกฎทั้งหมดในเชน (ถ้าไม่ระบุเชนจะแสดงกฎทั้งหมดในฟิลเตอร์เทเบิลทั้งสามบิวท์อินเชน) เช่น

#iptables -L

#iptables -L -t nat

#iptables -L INPUT

-F ลบกฎทั้งหมดในเชนทิ้ง เช่น

#iptables -F INPUT

#iptables -F mychain

-Z ใช้รีเซตไบท์เคาเตอร์ (reset byte counter) สำหรับทุก rule ใน chain ที่กำหนด เช่น

#iptables -Z INPUT

-N ใช้สร้างกฎใหม่ เช่น

#iptables -N mychain

-X ลบกฎที่ไม่มีกฎซึ่งสามารถลบยูเซอร์ดีไฟน์เชนที่ไม่มีกฎได้ แต่ไม่สามารถลบบิวท์อินเชนได้ เช่น

#iptables -X emptychain

-P เปลี่ยนปฏิบัติการขั้นพื้นฐานของเชนค่าที่ใช้ได้คือ แอคเซป, ดรอป ทั้งนี้ค่านี้มีความสำคัญอย่างมากเพราะหากแพคเกตถูกส่งเข้ามาในเชนแล้ว และไม่ตรงกับกฎใด ๆ เลยแพคเกตนั้นก็ต้องถูกตัดสินใจโดยขั้นพื้นฐานของเชนนั้นๆ เช่น

#iptables -P FORWARD DROP

ซึ่งหากแพคเกตถูกส่งเข้ามายังฟอเวิดเชนและไม่ตรงกับกฎใดๆ ในฟอเวิดเชนนี้เลย มันก็จะถูกดรอปทันที

-E ใช้เปลี่ยนชื่อเชนใหม่ เช่น

#iptables -E myoldchain mynewchain

การใช้คำสั่งด้านบนนั้นสามารถใช้ร่วมกับออปชั่นบางอย่างได้คือ

-V, --verbose ใช้ร่วมกับ -L, -A, -I, -D, -R เพื่อให้แสดงจำนวนไบท์ที่ตรงกับกฎออกมาด้วย (หน่วยเป็นได้ทั้ง K(x1,000) ,M(x1,000,000), G(x1,000,000,000)) เช่น

#iptables -L –v

-x, --exact ใช้ร่วมกับ -L และ -v เพื่อให้แสดงจำนวนแพคเกตและจำนวนของไบท์ข้อมูลที่ตรงโดยไม่ให้แสดงผลในหน่วยของ K,M,G เช่น

#iptables -L OUTPUT -v –x

-n, --numeric ใช้ร่วมกับ -L เพื่อสั่งให้ไอพีเทเบิลแสดงข้อมูลไอพีแอดเดรสและพอร์ตเป็นตัวเลขเท่านั้น เช่น

#iptables -L OUTPUT -n

--line-numbers ใช้ร่วมกับ -L เพื่อแสดงเลขบรรทัดของกฎซึ่งตัวเลขที่แสดงนี้จะสามารถใช้ได้กับคำสั่งอินเซิสรูลที่ระบุเป็นลำดับที่ของกฎ เช่น

#iptables -L --line-numbers

--modprobe=command เพื่อโหลดโมดูลที่เกี่ยวข้อง

<Match> เป็นส่วนที่ใช้ตรวจสอบว่าแพคเกตมีข้อมูลตรง (Match) กับที่ระบุไว้หรือไม่ เงื่อนไขของการแมทนั้นจะต้องอาศัยความเข้าใจในเรื่อง ไอพี, ทีซีพี, ยูดีพี, และไอซีเอ็มพีมาบ้าง จึงจะสามารถตั้งเงื่อนไขที่เหมาะสมและตรงตามความต้องการได้ ซึ่งมีรายละเอียดดังนี้

1) การระบุ ไอพีต้นทางและปลายทางสามารถระบุไอพีต้นทางของแพคเกตโดยใช้ –s หรือ –source หรือ –src และสำหรับไอพีปลายทางก็ใช้ –d หรือ –destination หรือ –dst การระบุไอพีแอดเดรสนั้นสามารถทำได้ 4 แบบด้วยกันคือ ใช้ชื่อเต็มแทน เช่น localhost หรือ www.ce.kmitl.ac.th ระบุไอพีแอดเดรสโดยตรง เช่น 127.0.0.1 หรือ 202.44.204.33 ระบุเป็นกลุ่มของไอพีแอดเดรส เช่น 202.44.204.0/24 ซึ่งหมายถึงไอพีแอดเดรสตั้งแต่ 202.44.204.0 - 202.44.204.255 หรืออาจจะใช้ 202.44.204.0/255.255.255.0 แทน 202.44.204.0/24 ได้
2) การทำอินเวิสชันในบางกรณีนั้นหากต้องการระบุเป็นอินเวิสเช่น อนุญาตให้ทุกไอพียกเว้นไอพีที่ระบุไว้ ซึ่งการใช้คำสั่งดังกล่าวสามารถทำได้โดยใช้เครื่องหมาย “!” นำหน้าอากิวเมนท์ที่ต้องการ (เครื่องหมาย “!” หมายถึง NOT) เช่น -p ! TCP ซึ่งจะตรงกับโพรโตคอลทุกๆ ตัวที่ไม่ใช่ทีซีพี

3) การระบุโพรโตคอลสามารถระบุโพรโตคอลที่ต้องการได้ดังนี้คือ ทีซีพี, ยูดีพี, ไอซีเอ็มพี หรือสามารถใช้ตัวเลขแทนได้และยังสามารถใช้ได้ทั้งตัวอักษรเล็กหรือใหญ่ (ใช้ได้ทั้ง tcp และ TCP) เช่น -p TCP หรือ -p ! tcp

4) การระบุ interface –I หรือ --in-interface ตามด้วยชื่ออินเตอร์เฟสใช้เพื่อระบุ In Coming อินเตอร์เฟส ซึ่งหมายถึงว่าแพคเกตที่จะตรงกับกฎนี้ต้องเข้ามาจากอินเตอร์เฟสที่กำหนดเช่น -i eth0 หมายความว่าทุกแพคเกตที่เข้ามาทาง eth0 จะตรงกับกฎนี้ ทั้งนี้ชื่ออินเตอร์เฟสที่สามารถใช้ได้นั้น สามารถตรวจสอบได้โดยใช้คำสั่งไอพีคอนฟิคและ –o หรือ --out-interface ตามด้วยชื่อของอินเตอร์เฟสใช้เพื่อระบุเอาโกอิงอินเตอร์เฟส (Outgoing Interface) ซึ่งหมายถึงว่าแพคเกตที่จะตรงกับกฎนี้กำลังจะเดินทางผ่านอินเตอร์เฟสที่ระบุไว้ เช่น -o eth1 หรือ -o ! eth1

ข้อสังเกต

1) สำหรับอินพุตเชนนั้นไม่มีเอาพุตอินเตอร์เฟสดังนั้นหากใช้ -o ร่วมกับ อินพุตเชนก็จะไม่มีแพคเกตใดที่ตรงกับกฎนี้เลย

2) ทำนองเดียวกันกับเอาพุตเชนที่ไม่มีอินพุตอินเตอร์เฟสดังนั้นหากใช้ -i ร่วมกับเอาพุตเชนก็ไม่มีประโยชน์อันใด

3) ฟอเวิดเชน มีได้ทั้งอินพุตและเอาพุตอินเตอร์เฟส

4) หากระบุอินเตอร์เฟสที่ไม่มีอยู่จริงก็จะไม่มีแพคเกตใดที่ตรงกับกฎนั้นเลย

5) หากใช้เครื่องหมาย + ร่วมกับ interface เช่น ppp+ นั้นจะหมายถึงทุกๆ ppp interface เช่น ppp0, ppp1

ตาราง 2.3 คำสั่งทั่วไปในไอพีเทเบิลที่ใช้ในการเลือก

iptables command Switch	คำอธิบาย
-t <-table->	หากไม่มีการบ่งบอก ตาราง ที่ใช้ จะใช้ตารางฟิลเตอร์สามารถตั้งค่าให้เป็นตารางแนทหรือแมนเกิลได้
-j <target>	กระโดดไปยังเชนเป้าหมายที่ต้องการเมื่อแพคเกตตรงกับกฎที่ตั้งไว้
-A	ผนวกกฎเข้ากับด้านท้ายของ
-F	ลบกฎทั้งหมดในตารางที่เลือก
-p <protocol-type>	ตรวจจับโปรโตคอลเช่น ไอซีเอมพี ทีซีพี ยูดีพี
-s <ip-address>	ตรวจจับไอพีแอดเดรสต้นทาง
-d <ip-address>	ตรวจจับไอพีแอดเดรสปลายทาง
-i <interface-name>	ตรวจจับอินเตอร์เฟสที่แพคเกตเข้ามา
-o <interface-name>	ตรวจจับอินเตอร์เฟสที่แพคเกตจะออกไป

ตัวอย่างการใช้คำสั่ง เช่น

#iptables -A INPUT -s 0/0 -i eth0 -d 192.168.1.1 -p TCP -j ACCEPT

มีหมายความว่าแพคเกตที่เข้ามายังไฟร์วอลล์ไอพีเทเบิล (INPUT) ทุกแพคเกตที่ใช้โปรโตคอลทีซีพีและมีไอพีปลายทางเป็น 192.168.1.1 จะยอมให้ผ่านได้หมด (-s 0/0 หมายถึง ไอพีแอดเดรสต้นทางสามารถเป็นอะไรก็ได้)

ตาราง 2.4 คำสั่งสำหรับโปรโตคอลไอซีเอ็มพี

Matches used with icmp	คำอธิบาย
--icmp-type <type>	ชนิดที่มักใช้คือ echo-reply และ echo-request

ตัวอย่างการใช้คำสั่ง เช่น

#iptables -A OUTPUT -p icmp --icmp-type echo-request -j ACCEPT

#iptables -A INPUT -p icmp --icmp-type echo-reply -j ACCEPT

มีความหมายว่า IP Tables อนุญาตให้ไฟร์วอลล์ส่ง ICMP Echo Request หรือคำสั่ง Pingได้และในทางกลับกัน ก็อนุญาตให้รับ ICMP Echo Reply กลับมาได้

<Target/jump> เป็นตัวระบุว่าเมื่อเจอแพคเกตที่ตรง (Match) ก็จะกระทำ (Action) ตามที่ระบุไว้ เช่น ถ้าแพคเกตใดมีไอพีแอดเดรสต้นทาง (Source IP Address) เป็น 1.2.3.4 ให้ทิ้งแพคเกตนั้น (DROP packet)

ตาราง 2.5 คำสั่งที่ใช้ในส่วนของเป้าหมาย/กระโดด

Target	คำอธิบาย	Most Common Options
ACCEPT	iptables หยุดทำการประมวลผลและส่งแพคเกตไปยังแอพพลิเคชั่นปลายทางหรือระบบปฏิบัติการเพื่อประมวลผลต่อไป	N/A
DROP	iptables หยุดการทำงานและบล็อกแพคเกต	N/A
LOG	ข้อมูลของแพคเกตถูกส่งไปยัง syslog daemon เพื่อเก็บ log iptables ทำการประมวลผลต่อไปยังกฎต่อไปในตาราง เนื่องจากไม่สามารถ log และดรอปแพคเกตในเวลาเดียวกัน จึงไม่น่าแปลกที่จะมีกฎที่เหมือนกันซ้ำกันในตารางอันแรกจะทำการลอคแพคเกต และอีกอันทำการดรอปแพคเกต	--log-prefix "string" Tells iptables to prefix all log messages with a user defined string. Frequently used to tell why the logged packet was dropped

ตาราง 2.5 คำสั่งที่ใช้ในส่วนของเป้าหมาย/กระโดด (ต่อ)

Target	คำอธิบาย	Most Common Options
REJECT	ทำงานคล้ายกับดรอป แต่จะมีการส่งกลับค่า Error ไปยังโฮสที่ส่งแพคเกต นี้มาว่าแพคเกตถูกบล็อก	--reject-with qualifier ค่าควอลิไฟเออร์ บอกว่าข้อความตอบกลับชนิดไหนจะถูกส่งกลับไปบ้างซึ่งมีดังนี้ icmp-port-unreachable (default) icmp-net-unreachable icmp-host-unreachable icmp-proto-unreachable icmp-net-prohibited icmp-host-prohibited tcp-reset echo-reply
DNAT	ใช้สำหรับแปลงค่าแอดเดรสปลายทาง หรือ เปลี่ยนค่าไอพีปลายทางนั้นเอง	--to-destination ipaddress บอก iptables ว่าไอพีปลายทางจะเปลี่ยนเป็นอะไร
SNAT	ใช้สำหรับการแปลงแอดเดรสต้นทาง หรือแก้ไข ไอพีต้นทางของแพคเกต ซึ่งสามารถกำหนดค่าได้	--to-source <address>[-<address>][:<port>-<port>] บอกไอพีต้นทาง และพอร์ตต้นทางใช้โดย SNAT.
MASQUERADE	ใช้แก้แปลงแอดเดรสต้นทาง โดยมีค่าเริ่มต้นคือ ไอพีของอินเตอร์เฟสของไฟร์วอลล์	[--to-ports <port>[-<port>]] บ่งบอก Range ของพอร์ตที่จะทำการ Mapping ได้

2.5 สนอร์ท (Snort)

Snort คือระบบป้องกันผู้บุกรุกแบบโอเพนซอร์สซึ่งมีความสามารถในการวิเคราะห์ทราฟฟิกแบบเรียลไทม์และเก็บลอคไฟล์บนระบบเน็ตเวิร์คและยังสามารถทำการวิเคราะห์ลึกลงในระดับโปรโตคอลค้นหาข้อมูลในแพคเกต ซึ่งสามารถนำไปใช้ในการตรวจจับการบุกรุกต่างๆ และการตรวจจับช่องโหว่ชนิดต่างๆ เช่น การทำบัฟเฟอร์โอเวอร์โฟลว์ การทำพอร์ทสแกน โจมตีซีจีไอ (CGI) การตรวจการเปิดแชร์ไฟล์ การตรวจหารุ่นของระบบปฏิบัติการและอื่นๆ อีกมากมาย

สนอร์ทมีลักษณะการใช้งานอยู่ 3 ประเภท คือ

1) การใช้งานแบบแพคเกตสนิฟเฟอร์คือดักดูแพคเกตคล้ายกับโปรแกรม TCP Dump
2) การเก็บลอคของแพคเกต (เป็นประโยชน์ต่อการแก้ปัญหาเน็ตเวิร์ค)
3) ใช้เป็นระบบดักจับผู้บุกรุกแบบเต็มรูปแบบ

2.5.1 กฎและการทำงานของสนอร์ท

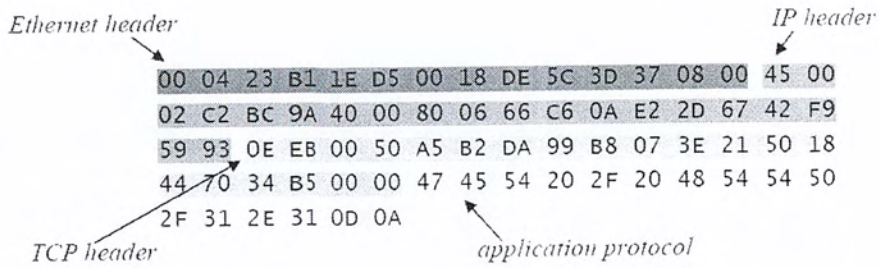
สนอร์ทเป็นโปรแกรมที่สามารถใช้งานได้ 4 โหมด คือ

1) โหมดสนิฟเฟอร์ (Sniffer) โดยจะทำงานโดยอ่านแพคเกตที่วิ่งอยู่ในเครือข่ายและแสดงผลต่อเนื่องไปเรื่อยๆ ในหน้าคอนโซล
2) โหมดแพคเกตลอคเกอร์ (Packet Logger) โดยทำการเก็บ Log แพคเกตลงบนฮาร์ดไดร์ฟ
3) โหมดระบบตรวจจับผู้บุกรุกบนเน็ตเวิร์ค (IDS) เป็นโหมดที่ซับซ้อนและยากต่อการตั้งค่ามากที่สุด ซึ่งจะทำให้ Snort สามารถวิเคราะห์ทราฟฟิกบนระบบเน็ตเวิร์ค เพื่อตรวจหาแพคเกตที่เข้าข่ายกับกฎที่ผู้ใช้ตั้งไว้และสั่งการต่างๆตามที่ตั้งค่าไว้
4) โหมดอินไลน์ (Inline) จะทำการรับแพคเกตมาจากโปรแกรมไอพีเทเบิลแทนที่ลิปพีแคปและสั่งการให้ไอพีเทเบิลดรอปหรือส่งต่อแพคเกตไปตามกฎของ Snort ที่ตั้งไว้

2.6 ลิบพีแคบ (Libpcap)

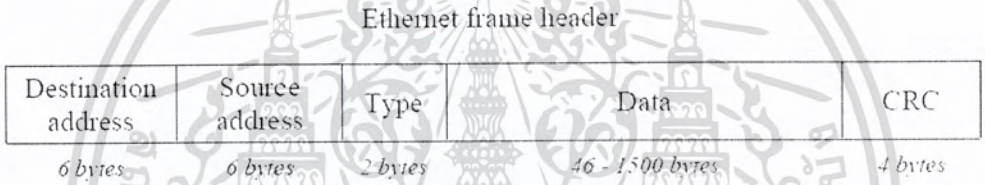
เป็น library สำหรับการดักจับแพคเกตที่ได้รับความนิยมอย่างสูงถูกนำไปใช้งานอย่างแพร่หลาย ในปัจจุบันมีซอฟท์แวร์สำเร็จรูปมากมายที่พัฒนาด้วย Libpcap ที่รู้จักกันดีได้แก่ Ethereal, TCP Dump, Snort, NMap, Ntop จุดเด่นของ Libpcap คือ มีรูปแบบ API ที่ใช้งานง่ายแต่มีประสิทธิภาพสูงสามารถจับแพคเกตได้ลึกถึงระดับดาต้าลิงค์ทั้งในโหมดปกติและโหมด Promiscuous อีกทั้งยังมีความสามารถในการเลือกจับเฉพาะบางชนิดของแพคเกตผู้ใช้ที่สนใจและที่สำคัญ Libpcap เป็น Open-Source Library ที่สามารถให้นำมาใช้งานได้ฟรี

การตีความจากข้อมูลดิบของแพคเกตที่ดักจับมาได้โดยทั่วไปโครงสร้างของแพคเกต ที่รับเข้ามาจะมีลักษณะของเฮดเดอร์หุ้มเป็นชั้นๆ จากระดับชั้นดาต้าลิงค์ไปจนถึงชั้น Application ดังรูป 2.9



รูป 2.9 ข้อมูลดิบของ Libpcap ที่ดักมาได้

สามารถตรวจสอบโปรโตคอลของข้อมูลภายใต้เฟรมได้จากส่วนของ Header ของ Frame ซึ่งในกรณีของโปรโตคอล Ethernet ที่นิยมใช้กันใน Local Area Network นั้น โครงสร้างของ Frame จะเป็นดังรูป 2.10



รูป 2.10 โครงสร้างของเฟรม

Header ของ Ethernet frame เริ่มต้นด้วย Destination Address และ Source Address ขนาด 6 ไบต์ บอกให้ทราบว่า Frame นี้ส่งออกมาจากที่ใดและจะส่งไปยังที่ใดแต่สิ่งที่สนใจจริงๆ จะเป็นส่วนที่อยู่ถัดมาซึ่งจะบ่งบอกชนิดโปรโตคอลของ Payload ของ Frame นั้น ตัวอย่างเช่นหากค่านี้เป็น 0x0800 ก็จะหมายถึง IPv4, 0x0806 จะหมายถึง ARP ฯลฯ ดังนั้นจึงสามารถตรวจสอบโปรโตคอล layer 3 โดยอ่านค่าจากไบต์ที่ 12 และ 13 นับจากส่วนหน้า

สำหรับ Source และ Destination IP address ที่ต้องการนำมาตรวจสอบนั้น ต้องดึงออกมาจาก header ของ IP ซึ่งจะมีโครงสร้างตามรูป 2.11

Version 4 bits	HLen 4 bits	Services 8 bits	Total length 16 bits
Identification 16 bits		Flag 3 bits	Fragmentation offset 13 bits
Time to live 8 bits	Protocol 8 bits	Header checksum 16 bits	
Source IP address 32 bits			
Destination IP address 32 bits			
Option ≤ 40 bytes			

รูป 2.11 เฮดเดอร์ของไอพี

จากรูป 2.11 จะเห็นว่าส่วนที่เป็น Source IP address นั้นมีความยาว 4 ไบต์ เริ่มต้นที่ไบต์ที่ 12 จนถึงไบต์ที่ 15 และ ส่วนที่เป็น Destination IP address จะเริ่มจากไบต์ที่ 16 จนถึงไบต์ที่ 19 ดังนั้นหากนับตั้งแต่ส่วนหัวของ Frame ตำแหน่งเริ่มต้นของ Source และ Destination IP จะอยู่ที่ไบต์ที่ 26 และ 30 ตามลำดับ

กล่าวโดยสรุปก็คือ ในการตรวจสอบว่า pcap_packet ที่จับมาได้นั้นเป็นข้อมูลที่เป็นการ download ของ interface นี้หรือไม่ก็เพียงแต่ตรวจสอบว่า

1) pcap_packet[12] == 0x08 และ pcap_packet[13] == 0x00 ใช่หรือไม่
2) เลข 32 บิตที่ตำแหน่งที่ 30 ของ pcap_packet มีค่าเท่ากับเลข 32 บิตของ IP address ของ interface หรือไม่

2.7 TCP Dump

TCP Dump เป็นโปรแกรม Protocol Analyzers ได้พัฒนามาตั้งแต่ ค.ศ. 1990 ที่ Lawrence Berkeley National Laboratory โดยถูกใช้สำหรับงานทางด้าน Network Monitor ซึ่งเป็นเครื่องมือที่วิเคราะห์คุณภาพของเครือข่ายและวิเคราะห์ข้อมูลที่อยู่บนเครือข่ายการทำงานของ TCP Dump ตัว tcpdump จะไปเปลี่ยนการทำงานของ Interface (LAN Card) ให้ทำงานในลักษณะ promiscuous mode ซึ่งจะคอยรับข้อมูลจากเครือข่ายและสามารถนำข้อมูลเหล่านั้นมาแสดงผลได้หลาย วิธี เช่น แสดงผลบนหน้าจอ (Console) หรือจะแสดงผลในลักษณะของไฟล์เพื่อนำกลับมาวิเคราะห์ในภายหลังก็ได้

ตัวอย่าง option ของ TCP Dump

- tcpdump -i interface

เลือก Interface ที่ TCP Dump จะคอยรับข้อมูลใช้สำหรับเครื่องที่มีหลาย Interface (มี LAN Card หลายใบ)

- tcpdump –n tcpdump

จะไม่พิมพ์ host name ลดการค้นหาชื่อจาก DNS

- tcpdump –l

จะเก็บข้อมูลที่จะแสดงผลไว้ใน buffers ซึ่งสามารถนำข้อมูลเก็บลงไฟล์ได้ ดังนี้ #tcpdump -l | tee “filename” หรือ #tcpdump -l > “filename” & tail –f “filename”

- tcpdump –w file เก็บข้อมูลทั้งหมดลงไฟล์แทนที่จะแสดงผลทางจอภาพ ซึ่งสามารถนำข้อมูลกลับมาวิเคราะห์ได้โดยใช้ option –r (ด่านล่าง)

-tcpdump –r file

อ่านข้อมูลจากไฟล์ (ที่บันทึกจาก option –w)

-tcpdump –t

ไม่แสดง timestamp tcpdump –x แสดงข้อมูลในรูปเลขฐาน 16 (hex)

บทที่ 3
รายละเอียดการพัฒนาโครงการ

การออกแบบและพัฒนาโปรแกรมจะต้องพิจารณาถึงขั้นตอนการทำงานต่างๆ ของโปรแกรม ประสิทธิภาพและการใช้ทรัพยากรของระบบที่มีอยู่จำกัดรวมไปถึงผลลัพธ์การวิเคราะห์ของโปรแกรมจะต้องมีความใกล้เคียงกับปัญหาที่เกิดขึ้นจริงของระบบในขณะนั้น

3.1 รายละเอียดการพัฒนาระบบ (System Specification)

3.1.1 เครื่องมือที่ใช้ในการพัฒนา

1) ระบบปฏิบัติการลินุกซ์ โครงงานนี้ได้เลือกใช้ระบบปฏิบัติการเซนต์ (Cent Os) เนื่องจากเป็นระบบปฏิบัติการที่เป็นฟรีแวร์และเหมาะสำหรับการทำเป็นระบบปฏิบัติการสำหรับเครื่องคอมพิวเตอร์แม่ข่าย
2) ไพธอน เวอร์ชั่น 2.7 เฟรมเวิร์ก PyQt4 ใช้ในการสร้างโปรแกรมควบคุมลักษณะกราฟฟิกในส่วนของการติดต่อผู้ใช้งาน
3) สนอร์ท (nort) 2.8.6 เป็นโปรแกรมแบบโอเพนซอร์สที่ทำหน้าที่ตรวจจับข้อมูลในระบบเครือข่ายเพื่อใช้ในการวิเคราะห์พฤติกรรมการใช้งานเครือข่ายของผู้ใช้ในระบบและเป็นตัวแจ้งความผิดปกติให้
4) ไอพีเทเบิล (IP Tables) 1.3.5 โปรแกรมที่ใช้สำหรับการตั้งกฎและใช้ในการเลือกกำหนดเส้นทางให้กับแพคเกต
5) มายเอสคิวแอล (MySQL) 5.0.77 โปรแกรมที่ใช้สำหรับเป็นฐานข้อมูลใช้ในการเก็บกฎต่างๆ
6) ทีซีพีดัมพ์ (TCP Dump) ใช้ในการหาผู้ใช้ที่ทำการแลกเปลี่ยนข้อมูลมากที่สุดในขณะนั้น

3.1.2 รายละเอียดส่วนนำเข้า (Input Specification)

1) รายละเอียดกฎเกณฑ์ในการคัดแยกทราฟฟิกที่ผิดปกติ ผู้ใช้จะต้องสร้างเกณฑ์ในการคัดแยกแพคเกตว่าแพคเกตนั้นปกติหรือไม่แล้วหลังจากนั้นจึงจะเข้าสู่ขั้นตอนการทำการส่งต่อแพคเกต
2) แพคเกตข้อมูลที่เข้ามาเป็นแพคเกตที่ส่งมาจากภายในเครือข่ายภายในและต้องการออกสู่อินเตอร์เน็ทภายนอก

3.1.3 รายละเอียดส่วนนำออก (Output Specification)

1) แพคเกตข้อมูลที่ต้องการออกสู่อินเตอร์เน็ท โดยระบบจะทำการส่งต่อไปยังเกตเวย์ โดยแยกออกเป็นสองเส้นทางคือ
   1.1) เส้นทางสำหรับแพคเกตที่ปกติ(Normal)
   1.2) เส้นทางสำหรับแพคเกตที่ผิดปกติ(Anomaly)
2) ข้อมูลทราฟฟิกปัจจุบันเมื่อผู้ใช้ทำการมอนิเตอร์

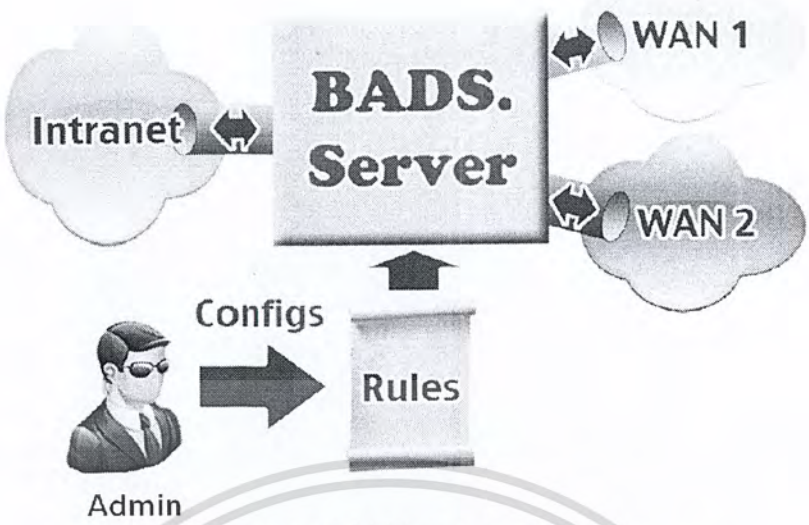
3.1.4 รายละเอียดฟังก์ชั่น (Functional Specification)

1) ผู้ใช้สามารถปรับตั้งกฎเองได้เพื่อตั้งเกณฑ์ในการคัดแยกทราฟฟิก
2) มีระบบตรวจจับแพคเกต
3) ผู้ใช้สามารถคัดกรองข้อมูลทราฟฟิกได้ในแต่ละอินเตอร์เฟส
4) มีระบบการทำการส่งต่อแพคเกตออกสองอินเตอร์เฟส
5) ผู้ใช้สามารถเลือกเส้นทางที่ต้องการให้แพคเกตออกไปได้
6) ระบบสามารถคัดแยกทราฟฟิกตามไอพีแอดเดรส หมายเลขพอร์ต และโปรโตคอลที่กำหนดได้
7) ระบบสามารถทำการเก็บและบันทึกลอคไฟล์ (Log File)
8) ระบบสามารถแสดงจำนวนผู้ใช้แบนด์วิดธ์สูงสุดได้
9) มีการแสดงผลทราฟฟิกแบบเรียลไทม์
10) เก็บกฎที่ใช้ในการคัดแยกทราฟฟิกและลอคของสนอร์ทไว้ในฐานข้อมูล

3.1.5 โครงสร้างของซอฟต์แวร์ (Software Design)

โครงสร้างของระบบตรวจจับและคัดแยกทราฟฟิกที่ผิดปกติโดยวิเคราะห์จากพฤติกรรมการใช้งานเครือข่ายได้แบ่งส่วนการทำงานออกเป็น 2 ส่วนหลักๆ คือ

1) ส่วนของการตรวจจับทราฟฟิก (Detection)
2) ส่วนของการคัดแยกทราฟฟิกออกเป็นสองเส้นทาง (Separation)



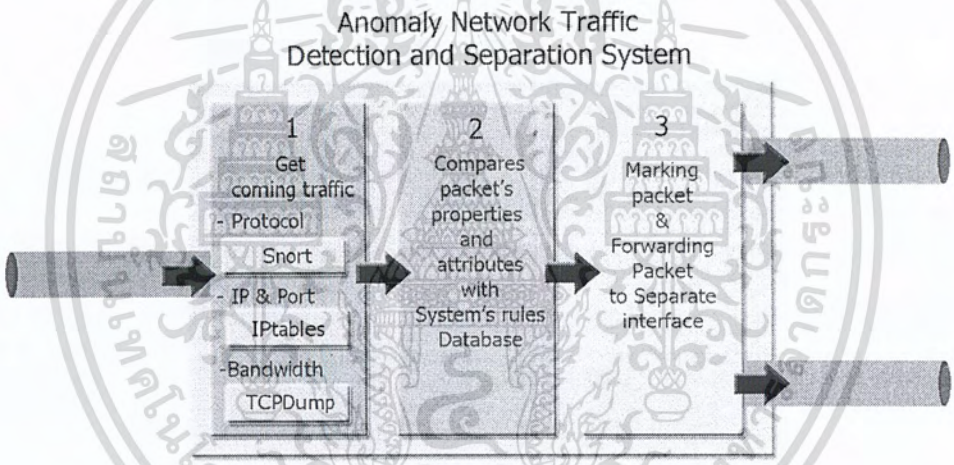
รูป 3.1 โครงสร้างซอร์ฟแวร์

จากรูป 3.1 โครงสร้างซอร์ฟแวร์ของระบบ เมื่อทำการตั้งค่าอินเตอร์เฟสซึ่งมีการโหลดค่าเริ่มต้นจากไฟล์ข้อความเรียบร้อยแล้ว ระบบจะทำการเรียกโปรแกรมสนอร์ทขึ้นมาแล้วเก็บลอคไฟล์ลงในฐานข้อมูลและทีซีพีดัมพ์เพื่อหาไอพีแอดเดรสด้วย เมื่อทำการตั้งค่ากฎต่างๆ เรียบร้อยแล้ว และมีทราฟฟิกที่วิ่งจากเครือข่ายเข้าสู่ระบบ ระบบจะทำการตรวจจับและตรวจสอบว่าทราฟฟิกเหล่านั้นว่าเป็นทราฟฟิกที่ปกติหรือไม่ก่อนที่จะออกสู่อินเตอร์เน็ต โดยผู้ดูแลระบบเป็นผู้เลือกกฎเกณฑ์ที่ใช้ในการคัดแยก โดยมีการแบ่งแยกตามกลุ่มไอพีแอดเดรส แบ่งตามพอร์ต แบ่งแยกตามปริมาณการใช้งาน แบ่งแยกตามโปรโตคอลที่ใช้ซึ่งวิเคราะห์จากพฤติกรรมของผู้ใช้งานเครือข่าย ระบบจะคัดแยกทราฟฟิกที่ปกติไปยังอินเตอร์เน็ทโดยใช้เส้นทางหนึ่งและแยกทราฟฟิกที่ผิดปกติไปอีกเส้นทางหนึ่ง โดยดูจากกฎที่ผู้ดูแลระบบได้ทำการตั้งไว้แล้วส่งไปยังไอพีเทเบิลเพื่อทำการมาร์คแพคเกตแล้วส่งไปไปให้ไอพีเราท์ทำการส่งต่อแพคเกตออกไปยังอินเตอร์เฟสที่ต้องการ

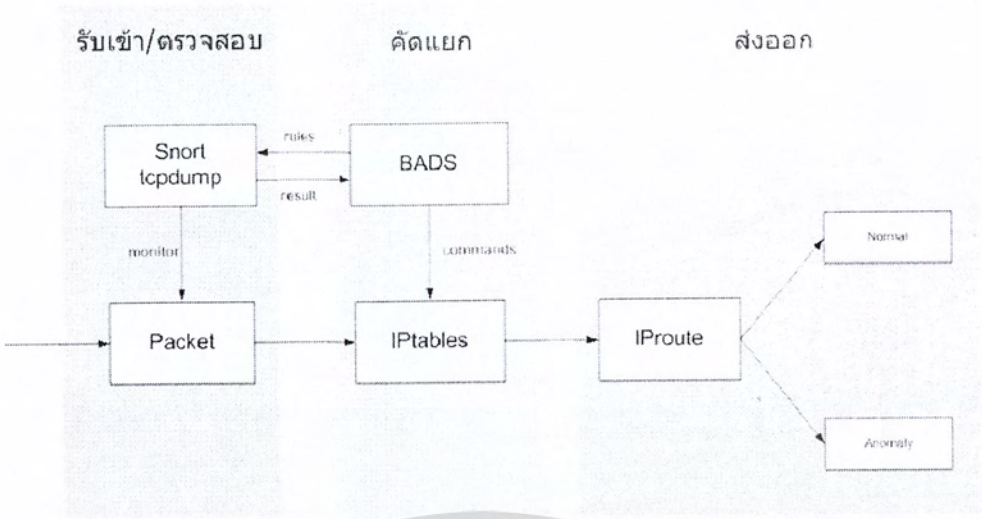
1) การตรวจจับทราฟฟิก จะใช้พัฒนาโปรแกรมโดยใช้สนอร์ทในการช่วยดักจับแพคเกตที่เข้ามาแล้วนำมาให้โปรแกรมวิเคราะห์ข้อมูลของแพคเกตดังเช่น หมายเลขพอร์ท, ไอพีต้นทาง, ไอพีปลายทางและดูปริมาณของแพคเกตที่เข้ามาโดยจะแสดงผลออกทางโปรแกรมด้วย
2) การคัดแยกทราฟฟิก ในส่วนนี้โปรแกรมที่จะพัฒนาจะสามารถคัดแยกทราฟฟิกที่ผิดปกติได้ โดยกฎการคัดแยกนั้นจะพิจารณาจากโปรโตคอลที่ใช้งาน, หมายเลขพอร์ต, ปริมาณการใช้งาน ซึ่งผู้ใช้สามารถทำการปรับแต่งกฎการคัดแยกได้เช่นจะคัดแยกโปรโตคอลอะไรออกบ้าง แยกพอร์ตหมายเลขไหนออกบ้าง โดยทราฟฟิกที่ปกติ จะทำการส่งต่อออกไปยังอินเตอร์เฟสปกติ ส่วนทราฟฟิกที่ตรงกับกฎที่กำหนดไว้จะถูกฟอร์เวิร์ดออกไปยังอีกอินเตอร์เฟสที่เลือกไว้

3.1.6 การทำงานของระบบ

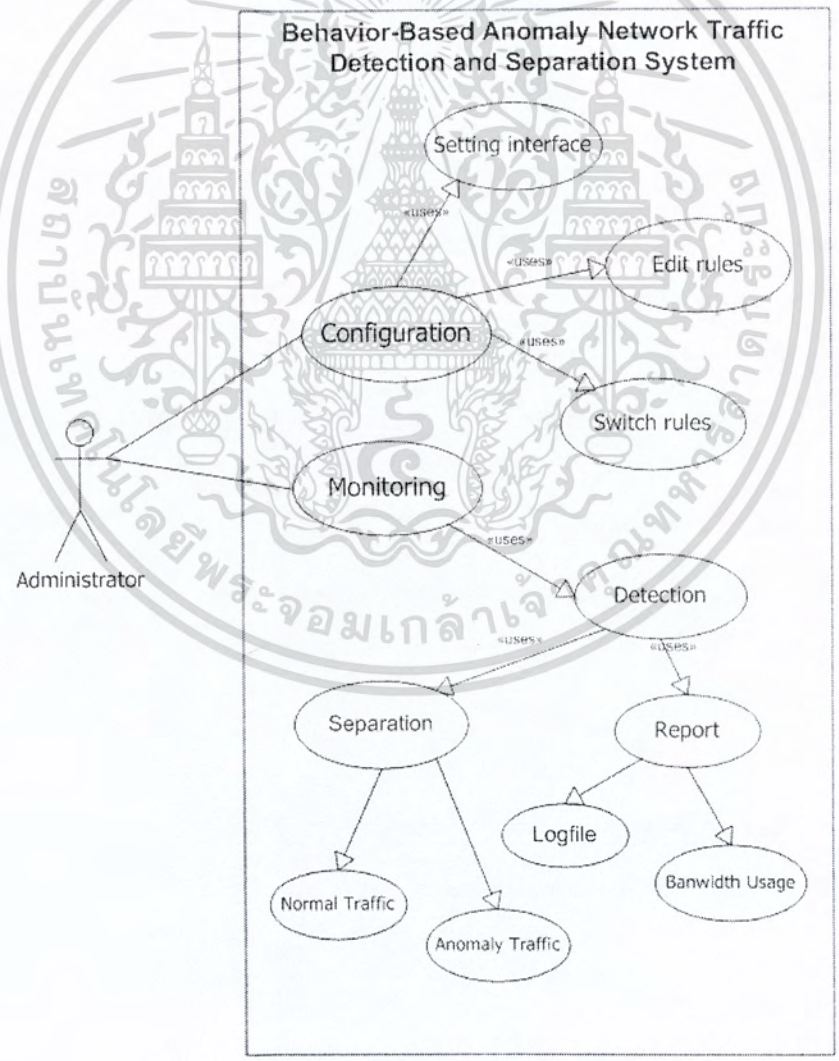
1) ส่วนของการตรวจจับเมื่อรับทราฟฟิกเข้ามาสามารถแยกได้ 3 กรณีดังนี้
   1.1) แยกตามโปรโตคอล จะใช้สนอร์ทดักแพคเกตตามกฎของสนอร์ทเก็บลอคลงฐานข้อมูลเพื่อพร้อมให้โปรแกรมเรียกใช้
   1.2) แยกตามไอพีและพอร์ทโดยใช้ไอพีเทเบิลคัดแยกแพคเกต
   1.3) แยกตามปริมาณการใช้งานจะใช้ TCP Dump ตรวจหาไอพีที่ใช้งานเยอะที่สุด
2) ค่าที่ได้มาจากส่วนที่หนึ่งเปรียบเทียบกับฐานข้อมูลกับกฎที่ตั้งไว้และแสดงผลผลลัพธ์ว่าแพคเกตแต่ละแพคเกตตรงกับกฎข้อใดและส่งต่อผลลัพธ์ไปยังส่วนที่สาม
3) ทำการมาร์คแพคเกตที่ตรงกับกฎแล้วส่งต่อให้ไอพีเราท์เพื่อทำการส่งต่อแพคเกตออกไปตามอินเตอร์เฟสที่ต้องการคือ อินเตอร์เฟสของทราฟฟิกชนิดปกติหรืออินเตอร์เฟสของทราฟฟิกชนิดผิดปกติ ดังรูป 3.2 และ 3.3



รูป 3.2 ภาพรวมการทำงานของระบบ

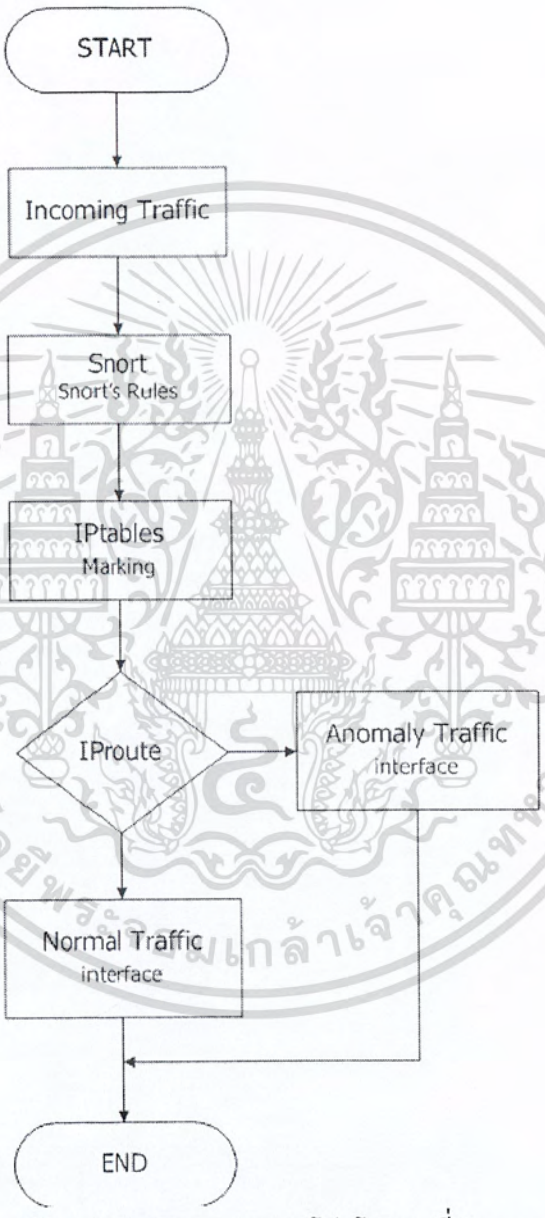


รูป 3.3 ลำดับขั้นตอนการทำงานของระบบ



รูป 3.4 Use case Diagram

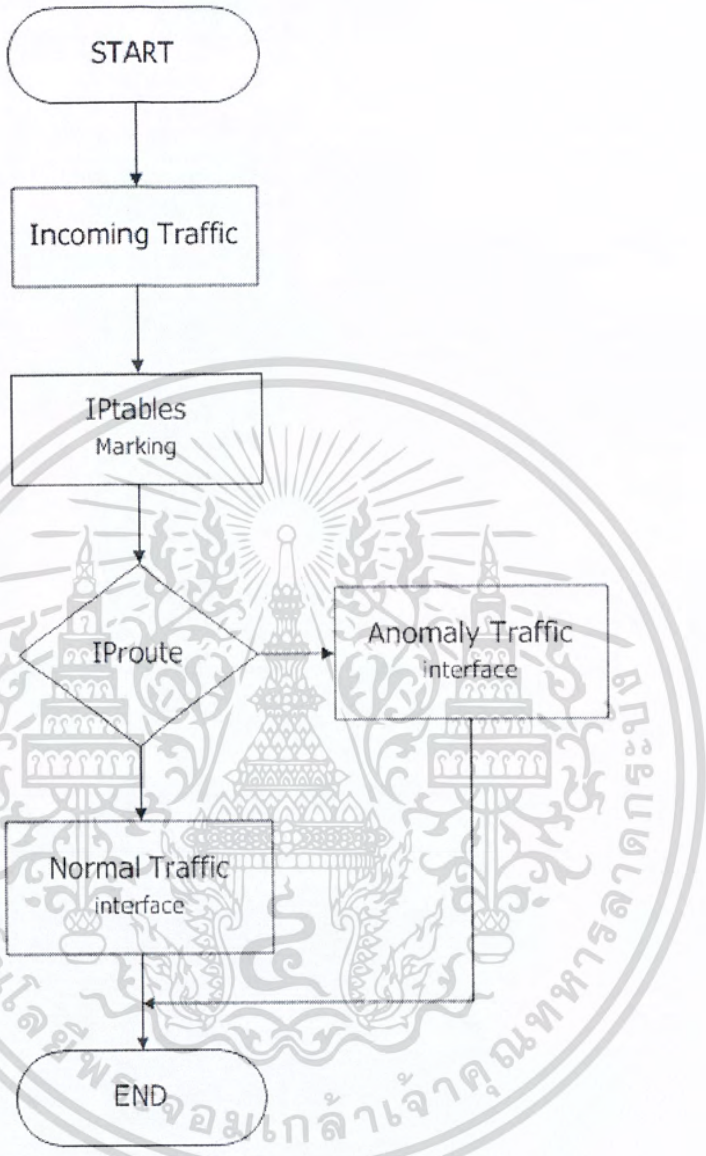
จากรูป 3.4 ผู้ใช้สามารถทำการปรับแต่งระบบโดยทำการตั้งค่าอินเตอร์เฟสที่ใช้และทำการสร้าง ลบ และ แก้ไข กฎที่ใช้ในการคัดแยกทราฟฟิกได้ สามารถเลือกได้ว่าจะใช้กฎใดกับอินเตอร์เฟสไหนโดยทำการสวิตช์กฎให้ตรงกับอินเตอร์เฟสที่ต้องการนอกเหนือจากนั้นแล้วผู้ใช้สามารถดูไฟล์ลอคของกฎที่ใช้และปริมาณการใช้งานแบนด์วิดธ์ของแต่ละอินเตอร์เฟสจากการตรวจจับของระบบ



รูป 3.5 Flowchart แยกตามโปรโตคอลที่กำหนด

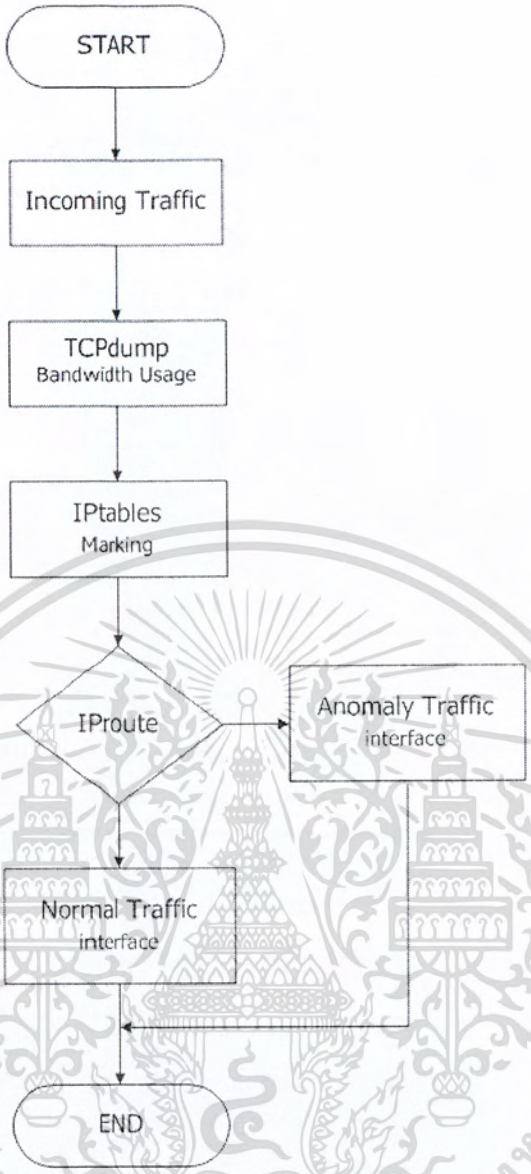
จากรูป 3.5 เมื่อรับแพคเกตเข้ามาในระบบเมื่อเลือกแยกตามโปรโตคอลจะใช้สนอร์ทดักแพคเกตตามกฎของสนอร์ทเก็บลอคลงฐานข้อมูลทำการมาร์คแพคเกตที่ตรงกับกฎแล้วส่งต่อให้ไอ

พีเราท์เพื่อทำการส่งต่อแพคเกตออกไปตามอินเตอร์เฟสที่ต้องการคืออินเตอร์เฟสของทราฟฟิกชนิดปกติหรืออินเตอร์เฟสของทราฟฟิกชนิดผิดปกติ



รูป 3.6 Flowchart แยกตามไอพีแอดเดรส และพอร์ท

จากรูป 3.6 เมื่อรับแพคเกตเข้ามาในระบบเมื่อเลือกแยกตามไอพีแอดเดรสและพอร์ท เมื่อตรงกับกฎที่ตั้งไว้ไอพีเทเบิลจะทำการมาร์คแพคเกตที่ตรงกับกฎแล้วส่งต่อให้ไอพีเร้าท์เพื่อทำการส่งต่อแพคเกตออกไปตามอินเตอร์เฟสที่ต้องการคืออินเตอร์เฟสของทราฟฟิกชนิดปกติหรืออินเตอร์เฟสของทราฟฟิกชนิดผิดปกติ



รูป 3.7 Flowchart แสดงจำนวนของ Top Loader

จากรูป 3.7 เมื่อรับแพคเกตเข้ามาในระบบ ถ้าเลือกการคัดแยกตามปริมาณการใช้งาน จะทำการใช้ทีซีพีดัมพ์คัดแยกไอพีออกมาแล้วไอพีเทเบิลจะทำการมาร์คแพคเกตที่ตรงกับกฎ แล้วส่งต่อให้ไอพีเราท์เพื่อทำการส่งต่อแพคเกตออกไปตามอินเตอร์เฟสที่ต้องการคืออินเตอร์เฟสของทราฟฟิกชนิดปกติหรือ อินเตอร์เฟสของทราฟฟิกชนิดผิดปกติ

3.1.7 ขอบเขตและข้อจำกัดของระบบ

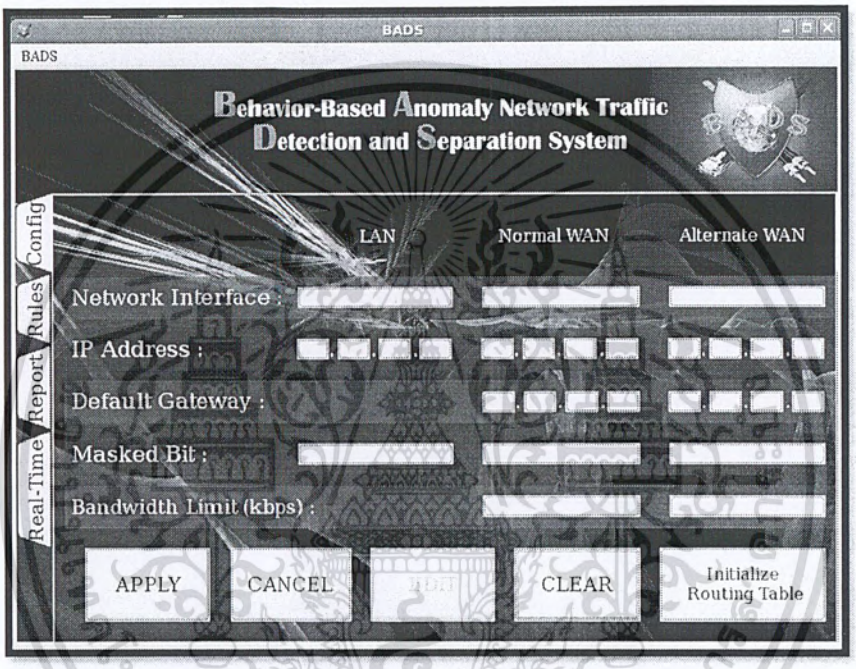
1) โปรแกรมทำงานบนระบบปฏิบัติการลินุกซ์
2) ใช้ในการเชื่อมต่อที่เป็นแบบอีเทอร์เนตเท่านั้น
3) ใช้ได้ในโครงสร้างทีซีพี/ไอพี
4) กฎการทำงานบางตัวอาจไม่ครอบคลุมทุกการใช้งาน

3.1.8 กลุ่มผู้ใช้โปรแกรม

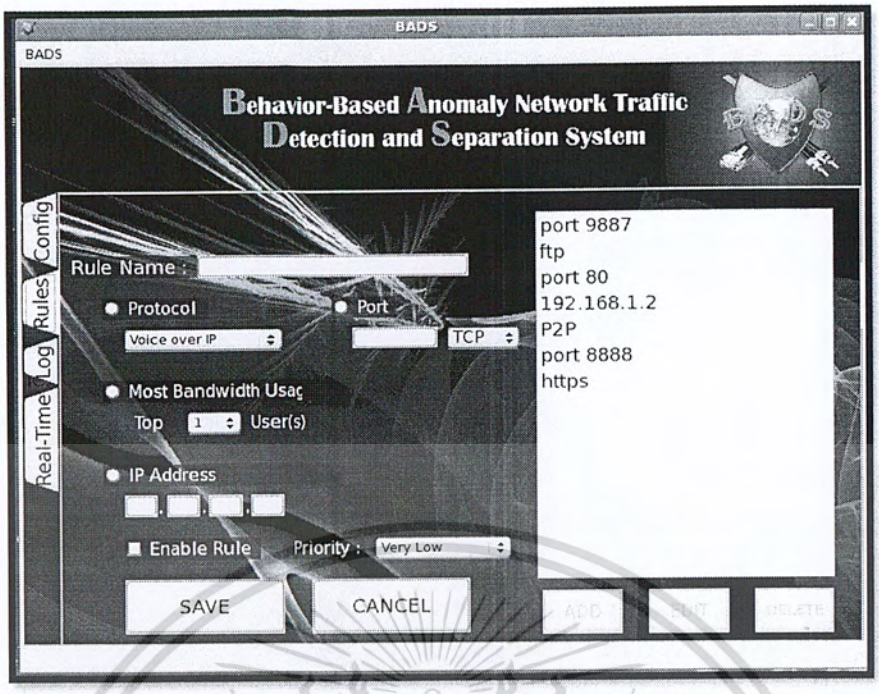
1) กลุ่มผู้ใช้ที่หน้าที่ดูแลระบบเครือข่าย (Administrator)

3.2 ออกแบบส่วนติดต่อผู้ใช้งาน (User Interfaces)

ส่วนของการติดต่อกับผู้ใช้งาน ในหัวข้อนี้จะอธิบายรายละเอียดคร่าวๆ ซึ่งจะอธิบายโดยละเอียดที่คู่มือการใช้งานโปรแกรม ภาคผนวก ข หน้าแรกเป็นหน้าที่ใช้งานการตั้งค่าเริ่มต้นการใช้งานของระบบซึ่งให้ใส่ค่าของ Network Interface Card ,ไอพีแอดเดรส เป็นต้น ดังรูป 3.8

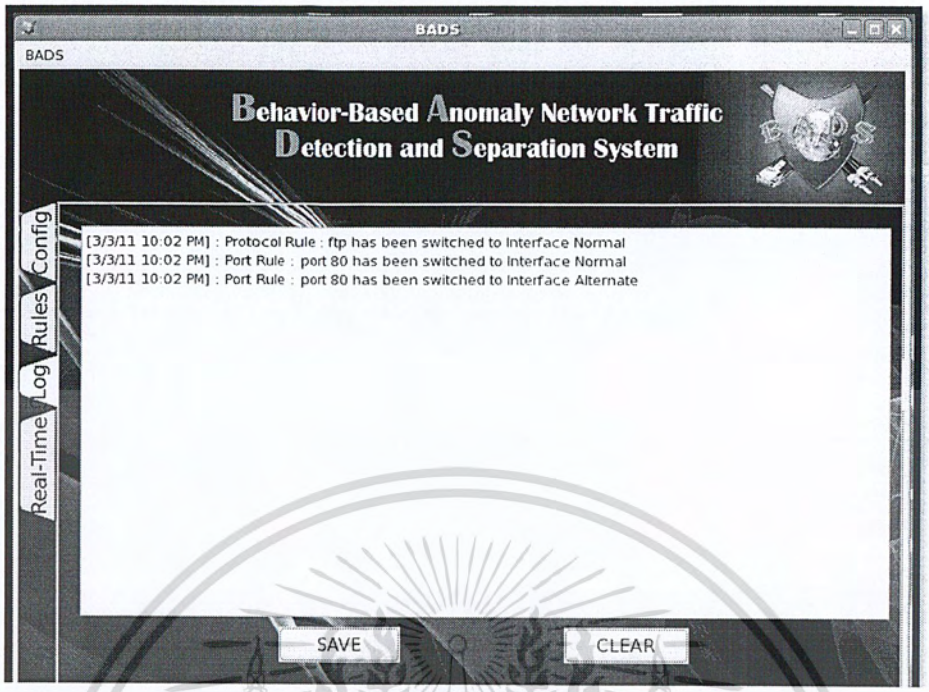


รูป 3.8 หน้าต่างที่ใช้ในการตั้งค่าเริ่มต้นการใช้งาน



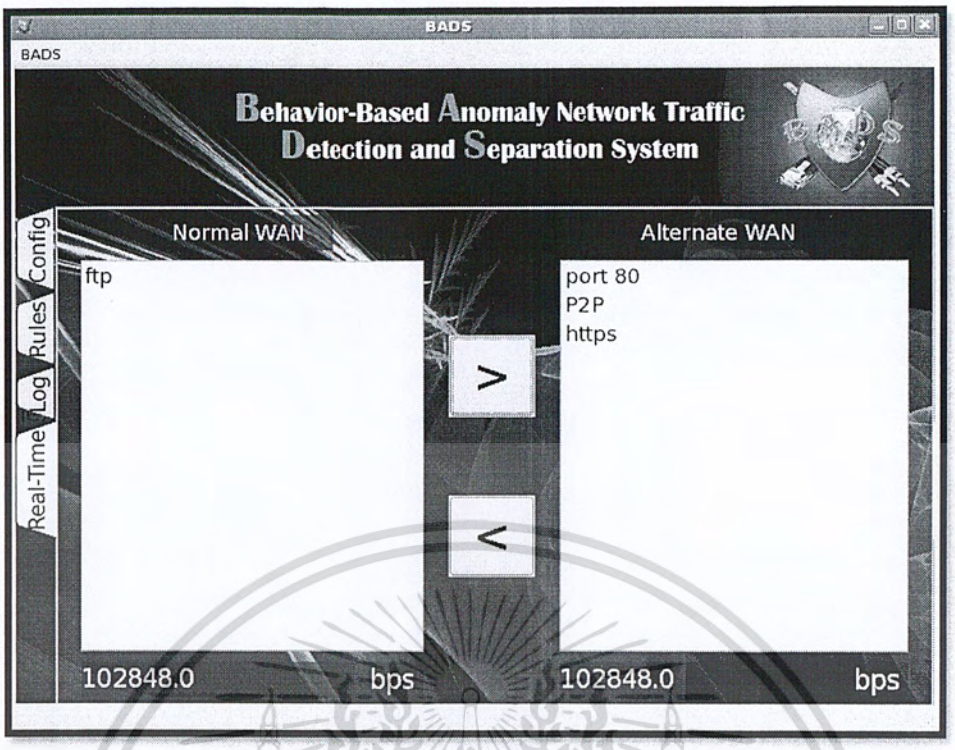
รูป 3.9 หน้าต่างที่ใช้ในการสร้างกฎ

จากรูป 3.9 หน้าต่างนี้ใช้สำหรับการสร้างกฎเพื่อใช้เป็นเกณฑ์ในการคัดแยกทราฟฟิกโดยสามารถ เพิ่ม, ลบ,และแก้ไขกฎเหล่านั้นได้ ในการสร้างผู้ใช้จำเป็นต้องกรอกรายละเอียดต่างๆที่จำเป็นตามที่กำหนดมาให้ทั้งชนิดของโปรโตคอล หมายเลขไอพี พอร์ทที่ต้องการจัดการ เมื่อคลิกที่ปุ่มเพื่อบันทึก (Save) กฎที่ผู้ใช้สร้างขึ้นใหม่จะปรากฏอยู่ในกล่องรายชื่อทางด้านขวามือ หากผู้ใช้งานต้องการแก้ไข หรือลบกฎออก ก็สามารถทำได้โดยคลิกที่ปุ่มทางด้านล่างของกล่องรายชื่อได้ตามความต้องการ



รูป 3.10 หน้าต่างลอค

จากรูป 3.10 หน้าต่างนี้ใช้สำหรับการมอนิเตอร์ระบบเพื่อดูการแสดงผลในรูปแบบลอคไฟล์ โดยดึงข้อมูลมาจากฐานข้อมูล



รูป 3.11 หน้าต่างเลือกใช้กฎในแต่ละอินเตอร์เฟสตลอดเวลา

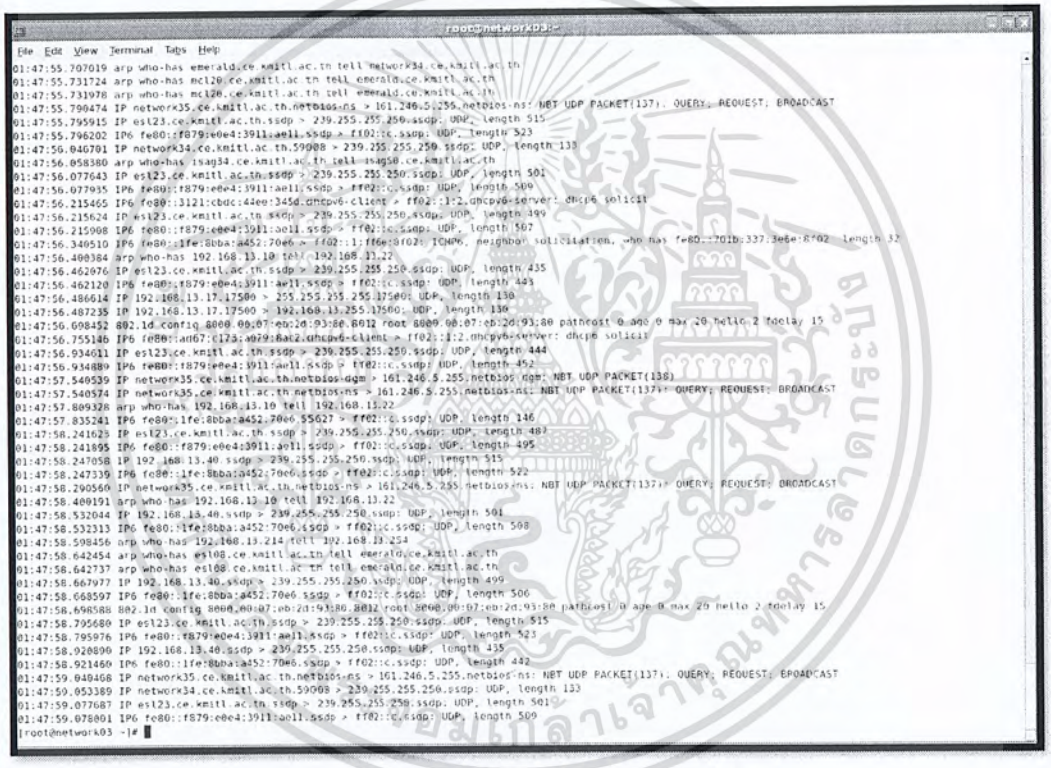
จากรูป 3.11 หน้าต่างนี้แสดงการใช้งานกฎที่ผู้ใช้สร้างไว้เรียบร้อยแล้วเข้ากับอินเตอร์เฟสโดยค่าเริ่มต้นของกฎจะถูกปรับให้ใช้ในอินเตอร์เฟสที่ 1 โดยสามารถสลับกฎไปมาในแต่ละอินเตอร์เฟสได้และสามารถแสดงปริมาณแบนวิดธ์ที่ใช้งานในแต่ละอินเตอร์เฟสได้

บทที่ 4
การทดสอบการทำงาน

4.1 การทดลองตรวจจับแพคเกตโดยใช้สนอร์ท

ทำการทดลองโดยใช้สนอร์ทดักจับแพคเกตที่วิ่งผ่านระหว่างอินเตอร์เฟสโดยสนอร์ทจะทำการเก็บข้อมูลแพคเกตเหล่านั้นลงในลอคไฟล์

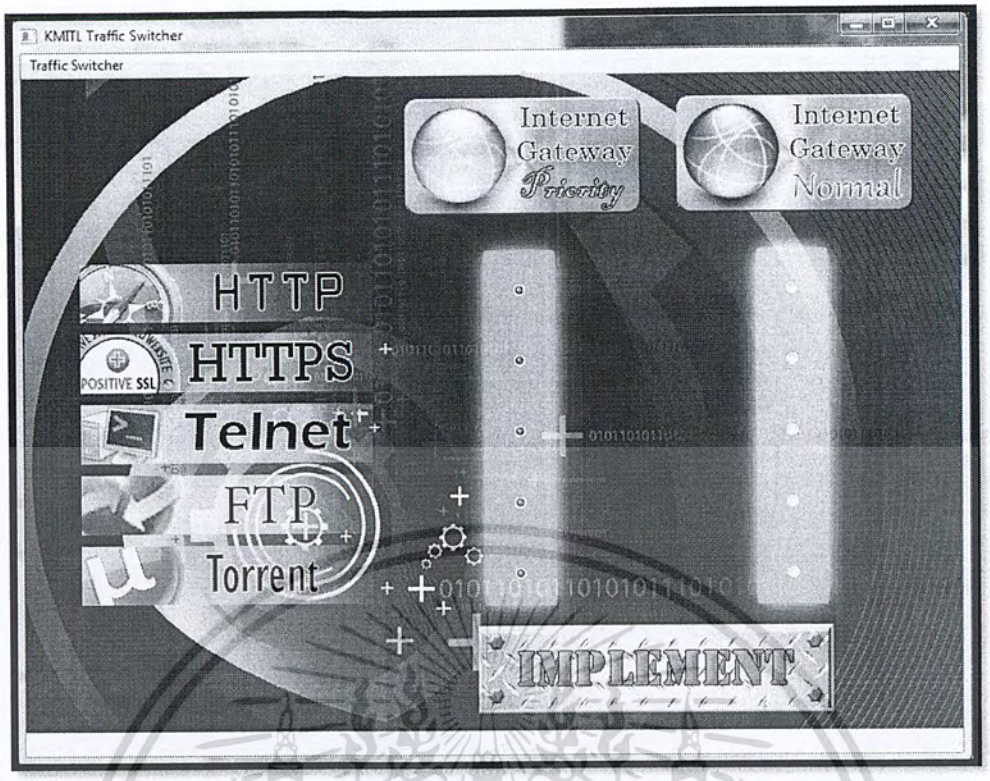
ผลลัพธ์ที่ได้ สามารถอ่านข้อมูลแพคเกตที่วิ่งเข้าออกของทุกๆ อินเตอร์เฟสโดยข้อมูลจะแสดงถึงโปรโตคอลที่ใช้ไอพีต้นทางและไอพีปลายทาง ดังรูป 4.1



รูป 4.1 ลอคไฟล์แพคเกตของสนอร์ท

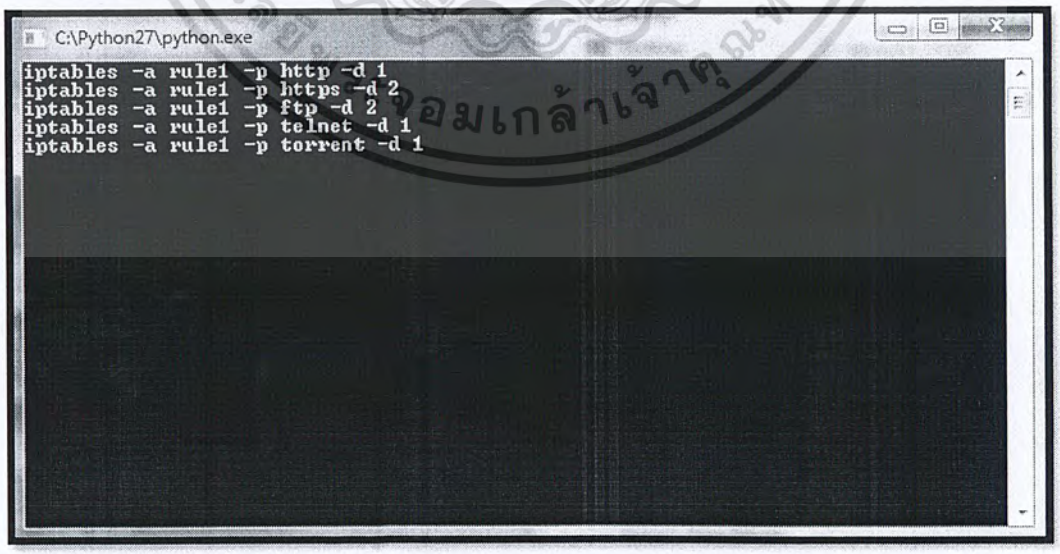
4.2 การทดลองการอินเจ็กท์โค๊ดลงบนเชลล์สคริปต์โดยจียูไอภาษาไพธอน

การทดลองนี้จะทำการทดสอบการอินเจ็กท์โค๊ดลงเชลล์สคริปต์เพื่อทดลองสั่งคำสั่งที่เป็น ซีแอลไอ (CLI) โดยใช้ จียูไอ (GUI) โดยจะสร้างความสะดวกให้แก่ผู้ดูแลระบบเนื่องจากไม่จำเป็นต้องจดจำคำสั่งต่างๆ มากมายแค่สั่งการผ่าน จียูไอ (GUI) เท่านั้นโดยผู้จัดทำได้เขียนโปรแกรมโดยใช้ภาษาไพธอน และไลบรารี่ PyQt4 ดังรูป 4.2



รูป 4.2 อินเตอร์เฟสโปรแกรมที่ได้ทำการทำลอง

โดยการทดลองนี้จะทดสอบการอินเจ็กท์โค๊ดเข้าเชลล์สคริปต์ โดยสมมุติให้มีการลองสั่งไอพีเทเบิลให้ฟอร์เวิร์ดไปยังพอร์ททั้ง 2 โดยมีโปรโตคอลที่จะให้ทำอยู่ 5 โปรโตคอลคือ เอชทีทีพี, เอชทีพีพีเอส, เทลเนต, เอฟทีพี, ทอเรนท์ (พอร์ตดีฟอลต์) โดยในจียูไอจะมีให้เลือกแล้วกดปุ่มอิมพลีเมนท์ (Implement) จากนั้นก็จะเกิดการอินเจ็กท์โค๊ดขึ้นดังรูป 4.3



รูป 4.3 การอินเจ็กท์โค๊ด

ผลลัพธ์ที่ได้คือสามารถอินเจ็กท์โค๊ดได้โดยการเพิ่มไลบรารี่ของระบบปฏิบัติการเข้าไปทำให้สามารถส่งสตริงค์ไปหาตัวเคอร์เนลได้ทำให้สามารถพอร์ทจียูไอลงซีแอลไอได้ซึ่งสามารถนำไปประยุกต์ใช้ต่อไปเมื่อต้องการอินเจ็กท์โค๊ดอื่นๆจากจียูไอ ของโปรแกรม

4.3 การทดลองคัดแยกทราฟฟิกโดยใช้ไอพีเทเบิล

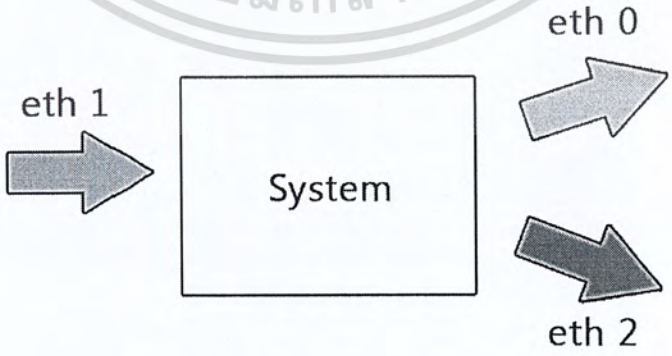
ได้ทำการทดลองเขียนกฎของไอพีเทเบิลเพื่อคัดแยกทราฟฟิกที่ต้องการให้เข้าหรือออกในระบบและไม่ต้องการให้เข้าหรือออกมาในระบบ

ผลลัพธ์ที่ได้ไอพีเทเบิลสามารถคัดแยกทราฟฟิกที่ไม่ต้องการให้เข้าหรือออกในระบบได้ ดังรูป 4.4

```
[root@network03 ~]# iptables -L -v
Chain INPUT (policy ACCEPT 420K packets, 206M bytes)
 pkts bytes target     prot opt in     out     source               destination
 420K  206M RH-Firewall-1-INPUT  all  --  any    any     anywhere             anywhere
    0     0 ACCEPT     tcp  --  any    any     161.246.5.0/24       network03.ce.kmitl.ac.th tcp dpt:ssh
    3   168 DROP       tcp  --  any    any     anywhere             network03.ce.kmitl.ac.th tcp dpt:ssh

Chain FORWARD (policy ACCEPT 30091 packets, 2270K bytes)
 pkts bytes target     prot opt in     out     source               destination
30091 2270K RH-Firewall-1-INPUT  all  --  any    any     anywhere             anywhere

Chain OUTPUT (policy ACCEPT 118K packets, 16M bytes)
 pkts bytes target     prot opt in     out     source               destination

Chain RH-Firewall-1-INPUT (2 references)
 pkts bytes target     prot opt in     out     source               destination
[root@network03 ~]#
```

รูป 4.4 ทดลองเขียนกฎของไอพีเทเบิล

จากนั้นทำการทดลองให้ไอพีเทเบิลสามารถเราท์แพกเกตข้อมูลให้ออกไปยังอินเตอร์เฟสที่ต้องการ โดยที่การเข้า-ออกของข้อมูลแต่ละอินเตอร์เฟสของระบบที่ทดลอง ดังรูป 4.5



รูป 4.5 Interface Diagram ของระบบ

ทำการเซตไอพีเทเบิลให้รองรับการฟอร์เวิร์ดข้อมูลออกสองทางโดยแรกสุดต้องสร้างเทเบิลในไอพีเทเบิลเพื่อใช้ในการเราท์แพคเกตข้อมูลโดยพิมพ์คำสั่งไอพีเทเบิลดังนี้

> echo **10 NRML** >> /etc/iproute2/rt_tables

> echo **20 ANRM** >> /etc/iproute2/rt_tables

จากนั้นทำการเพิ่มข้อมูลลงในเราท์เทเบิลที่สร้างขึ้นโดยใช้คำสั่งต่อไปนี้

> iproute add table NRML **X.X.X.X/X (Network Address of eth0/Subnetmask)**

> **eth0** proto kernel scop link src **X.X.X.X (IPaddress of eth0)**

> iproute add table NRML **X.X.X.X/X (Network Address of eth1/Subnetmask)**

> **eth1** proto kernel scop link src **X.X.X.X (IPaddress of eth1)**

>> iproute add table ANRM **X.X.X.X/X (Network Address of eth2/Subnetmask)**

> **eth2** proto kernel scop link src **X.X.X.X (IPaddress of eth2)**

> iproute add table ANRM **X.X.X.X/X (Network Address of eth1/Subnetmask)**

> **eth1** proto kernel scop link src **X.X.X.X (IPaddress of eth1)**

และเพิ่ม Default Gateway ให้กับแต่ละเทเบิลด้วยคำสั่งต่อไปนี้

> iproute add default via **X.X.X.X (Default Gateway of eth0)** table NRML dev **eth0**

> iproute add default via **X.X.X.X (Default Gateway of eth2)** table ANRM dev **eth2**

ใช้คำสั่งต่อไปนี้เพื่อทำ NAT

> iptables –t nat –A POSTROUTING –o **eth0** –j MASQUERADE

> iptables –t nat –A POSTROUTING –o **eth2** –j MASQUERADE

> echo 1 > /proc/sys/net/ipu4/ip_forward

ใช้ไอพีเทเบิลให้มาร์คแพคเกตเพื่อใช้ในการตัดสินใจส่งแพคเกตไปตามที่ต้องการด้วยคำสั่งต่อไปนี้

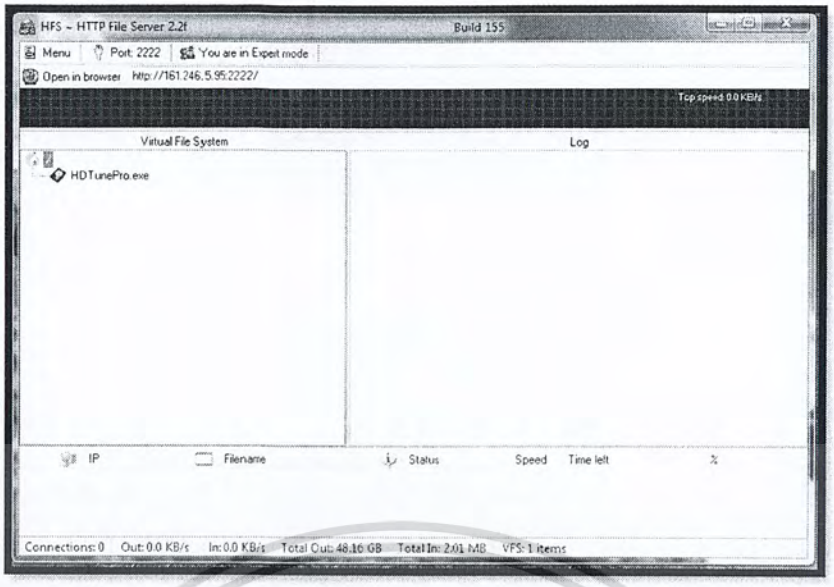
> iptables –A PREROUTING –t mangle –i **eth0** –p tcp –dport **XX (port)** –j MARK –set-mark **10**

> iptables –A PREROUTING –t mangle –i **eth2** –p tcp –dport **XX (port)** –j MARK –set-mark **20**

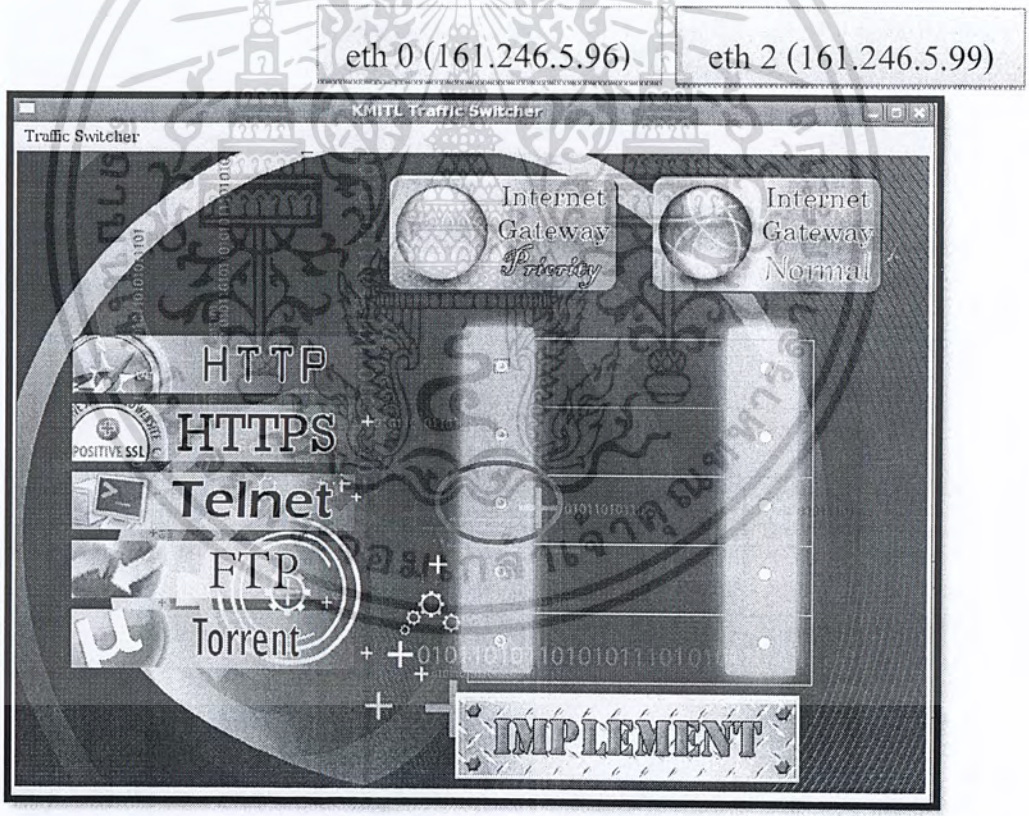
ใช้คำสั่งต่อไปนี้เพื่อให้แพคเกตที่ถูกมาร์คไว้ไปใช้ Route Table ที่ถูกต้องตามต้องการ

> ip rule add from **0.0.0.0/0** fwmark **10** table **NRML**

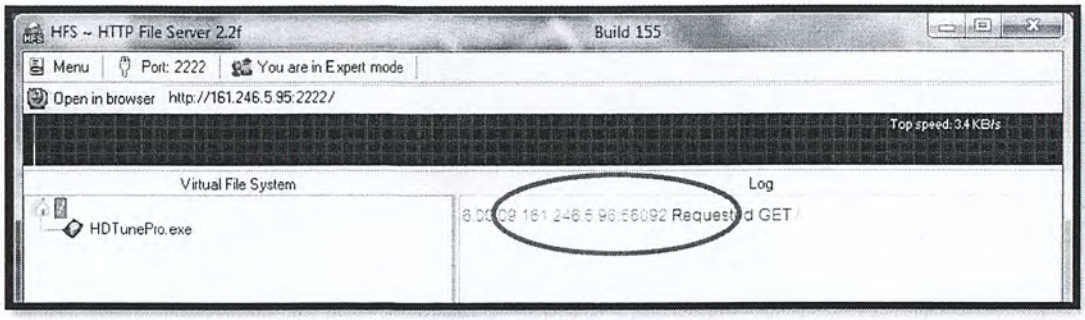
> ip rule add from **0.0.0.0/0** fwmark **20** table **ANRM**



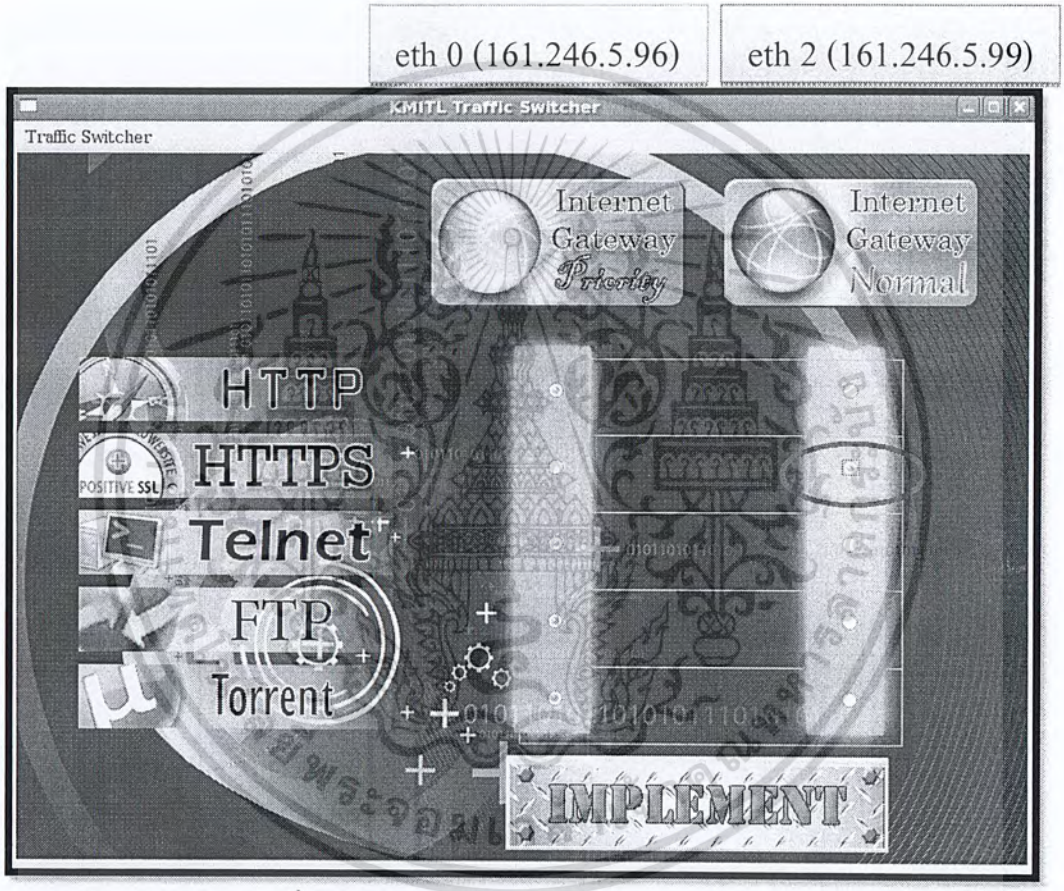
รูป 4.5 จากเครื่องเซิร์ฟเวอร์ ใช้ HFS ในการทดสอบว่า IP Tables ทำงานได้ตามต้องการ



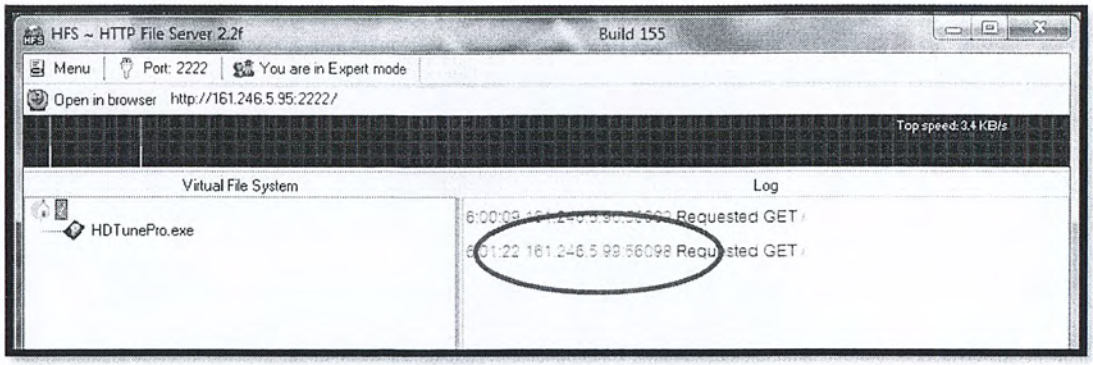
รูป 4.6 กำหนดจากเครื่อง Gateway ให้ออก Interface eth0 (IP Address 161.246.5.96)



รูป 4.7 มีการ Request จากเครื่องที่เป็น Gateway ด้วย IPaddress 161.246.5.96



รูป 4.8 กำหนดจากเครื่อง Gateway ให้ออก Interface eth2 (IPaddress 161.246.5.99)



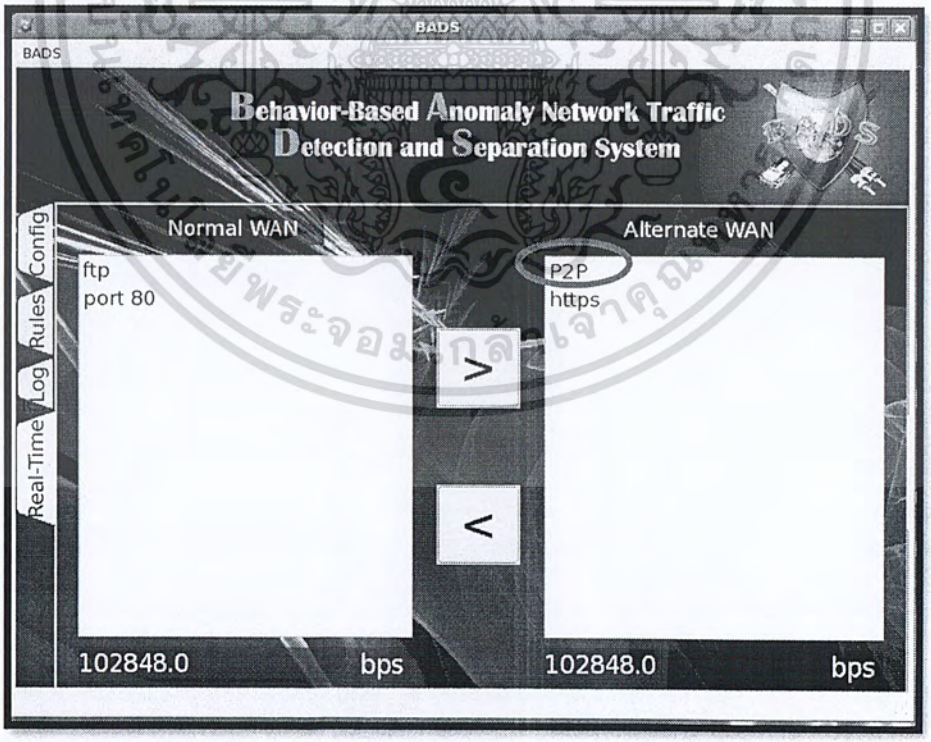
รูป 4.9 มีการ Request จากเครื่องที่เป็น Gateway

มีการ Request จากเครื่องที่เป็นเกตเวย์ด้วยไอพีแอดเดรส 161.246.5.99 (Interface eth2) สรุปได้ว่าสามารถทำให้ไอพีเทเบิลเร้าท์แพคเกตไปยังเส้นทางที่ต้องการได้ด้วยวิธีดังกล่าว

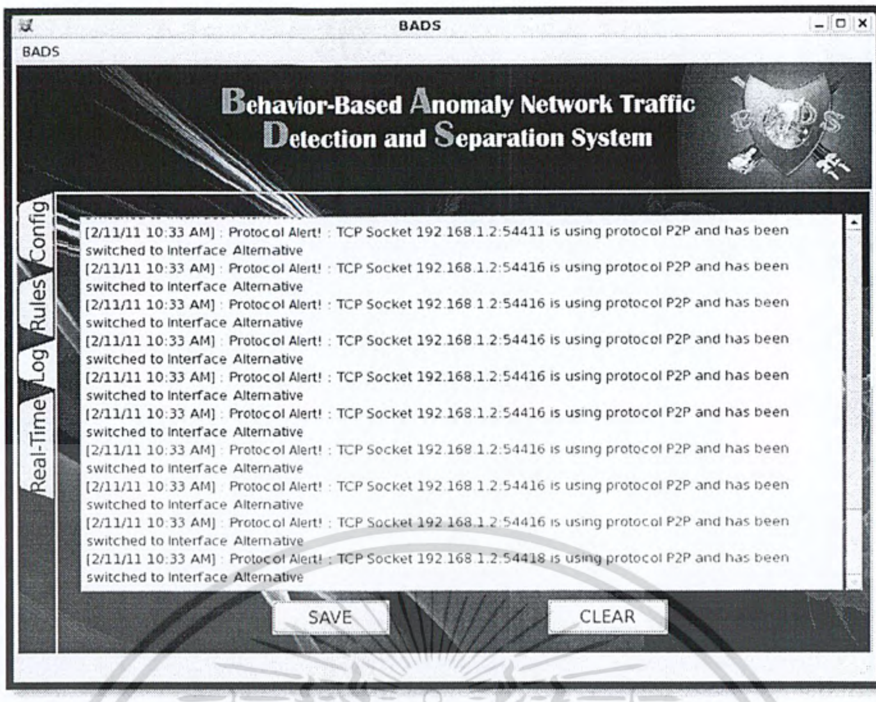
4.4 การทดสอบระบบจริง

4.4.1 การทดสอบแยกทราฟฟิกตามโปรโตคอลที่กำหนด

ในการทดลองคัดแยกทราฟฟิกตามโปรโตคอลที่กำหนดในที่นี้กำหนดให้โปรโตคอลเป็นบิททอเรนท์ (interface eth 2 IP 161.246.5.99) ดังรูป 4.10



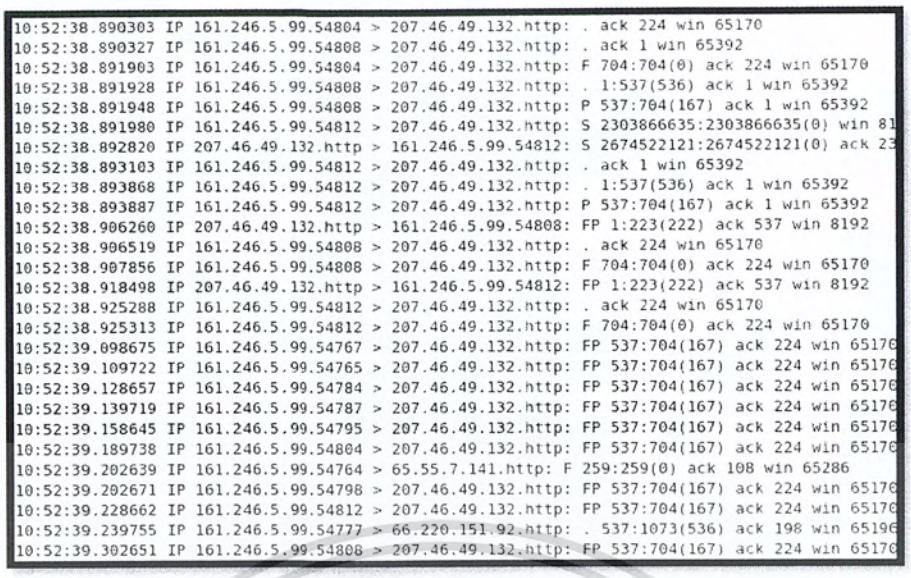
รูป 4.10 กำหนด rule Protocol P2P ให้กับ Alternate WAN



รูป 4.11 Log File ที่ Alert

จากรูป 4.11 เมื่อมีการดาวน์โหลดทราฟฟิกประเภท P2P จะมีการแสดงผลขึ้นมาที่หน้าลอคและเมื่อเปิดดูเราท์ติงเทเบิลก็แสดงให้เห็นว่ามีการสั่งให้เราท์แพคเกตออกตามอินเตอร์เฟสที่กำหนด ดังรูป 4.12

รูป 4.12 เราท์ติ้งเทเบิล



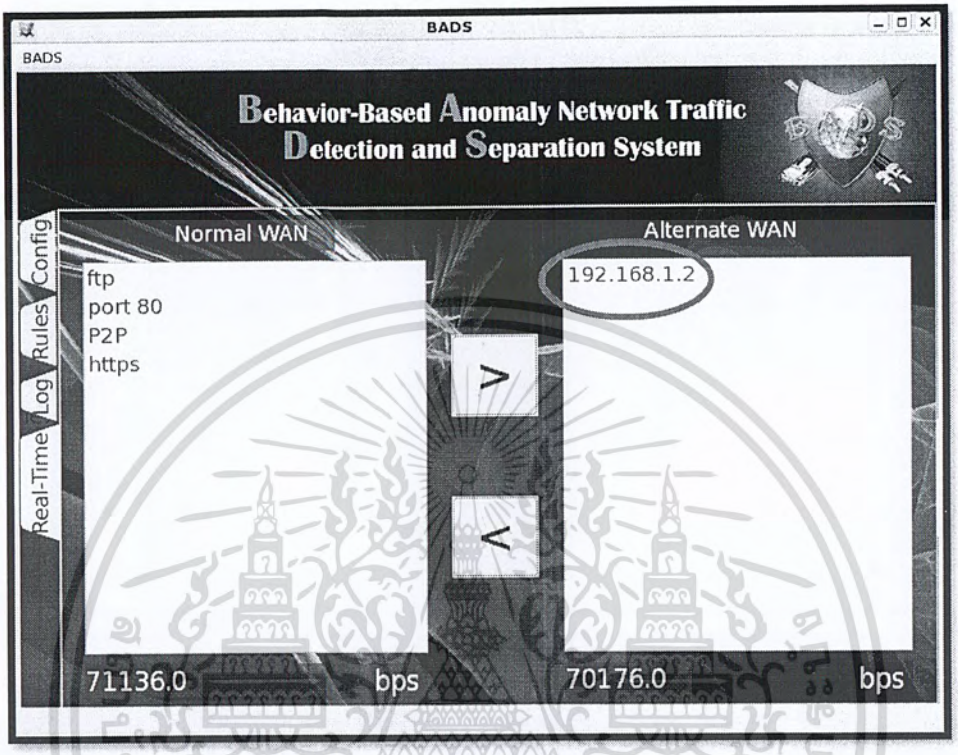
```
10:52:38.890303 IP 161.246.5.99.54804 > 207.46.49.132.http: . ack 224 win 65170
10:52:38.890327 IP 161.246.5.99.54808 > 207.46.49.132.http: . ack 1 win 65392
10:52:38.891903 IP 161.246.5.99.54804 > 207.46.49.132.http: F 704:704(0) ack 224 win 65170
10:52:38.891928 IP 161.246.5.99.54808 > 207.46.49.132.http: . 1:537(536) ack 1 win 65392
10:52:38.891948 IP 161.246.5.99.54808 > 207.46.49.132.http: P 537:704(167) ack 1 win 65392
10:52:38.891980 IP 161.246.5.99.54812 > 207.46.49.132.http: S 2303866635:2303866635(0) win 81
10:52:38.892820 IP 207.46.49.132.http > 161.246.5.99.54812: S 2674522121:2674522121(0) ack 23
10:52:38.893103 IP 161.246.5.99.54812 > 207.46.49.132.http: . ack 1 win 65392
10:52:38.893868 IP 161.246.5.99.54812 > 207.46.49.132.http: . 1:537(536) ack 1 win 65392
10:52:38.893887 IP 161.246.5.99.54812 > 207.46.49.132.http: P 537:704(167) ack 1 win 65392
10:52:38.906260 IP 207.46.49.132.http > 161.246.5.99.54808: FP 1:223(222) ack 537 win 8192
10:52:38.906519 IP 161.246.5.99.54808 > 207.46.49.132.http: . ack 224 win 65170
10:52:38.907856 IP 161.246.5.99.54808 > 207.46.49.132.http: F 704:704(0) ack 224 win 65170
10:52:38.918498 IP 207.46.49.132.http > 161.246.5.99.54812: FP 1:223(222) ack 537 win 8192
10:52:38.925288 IP 161.246.5.99.54812 > 207.46.49.132.http: . ack 224 win 65170
10:52:38.925313 IP 161.246.5.99.54812 > 207.46.49.132.http: F 704:704(0) ack 224 win 65170
10:52:39.098675 IP 161.246.5.99.54767 > 207.46.49.132.http: FP 537:704(167) ack 224 win 65170
10:52:39.109722 IP 161.246.5.99.54765 > 207.46.49.132.http: FP 537:704(167) ack 224 win 65170
10:52:39.128657 IP 161.246.5.99.54784 > 207.46.49.132.http: FP 537:704(167) ack 224 win 65170
10:52:39.139719 IP 161.246.5.99.54787 > 207.46.49.132.http: FP 537:704(167) ack 224 win 65170
10:52:39.158645 IP 161.246.5.99.54795 > 207.46.49.132.http: FP 537:704(167) ack 224 win 65170
10:52:39.189738 IP 161.246.5.99.54804 > 207.46.49.132.http: FP 537:704(167) ack 224 win 65170
10:52:39.202639 IP 161.246.5.99.54764 > 65.55.7.141.http: F 259:259(0) ack 108 win 65286
10:52:39.202671 IP 161.246.5.99.54798 > 207.46.49.132.http: FP 537:704(167) ack 224 win 65170
10:52:39.228662 IP 161.246.5.99.54812 > 207.46.49.132.http: FP 537:704(167) ack 224 win 65170
10:52:39.239755 IP 161.246.5.99.54777 > 66.220.151.92.http: . 537:1073(536) ack 198 win 65196
10:52:39.302651 IP 161.246.5.99.54808 > 207.46.49.132.http: FP 537:704(167) ack 224 win 65170
```

รูป 4.13 ทราฟฟิก P2P ที่วิ่งผ่าน Alternate WAN

จากรูป 4.13 จะเห็นได้ว่ามีทราฟฟิก P2P วิ่งผ่าน Alternate WAN โดยใช้ TCP Dump ดูทราฟฟิกที่วิ่งผ่านระหว่าง Interface eth2 IP : 161.246.5.99

4.4.2 การทดสอบแยกทราฟฟิกตามไอพีแอดเดรส

ในการทดลองคัดแยกทราฟฟิกตามไอพีแอดเดรสในที่นี้กำหนดให้ไอพีแอดเดรสที่สนใจเป็น 192.168.1.2 (interface eth 2 IP 161.246.5.99) ดังรูป 4.14



รูป 4.14 กำหนด rule IP address ให้กับ Alternate WAN

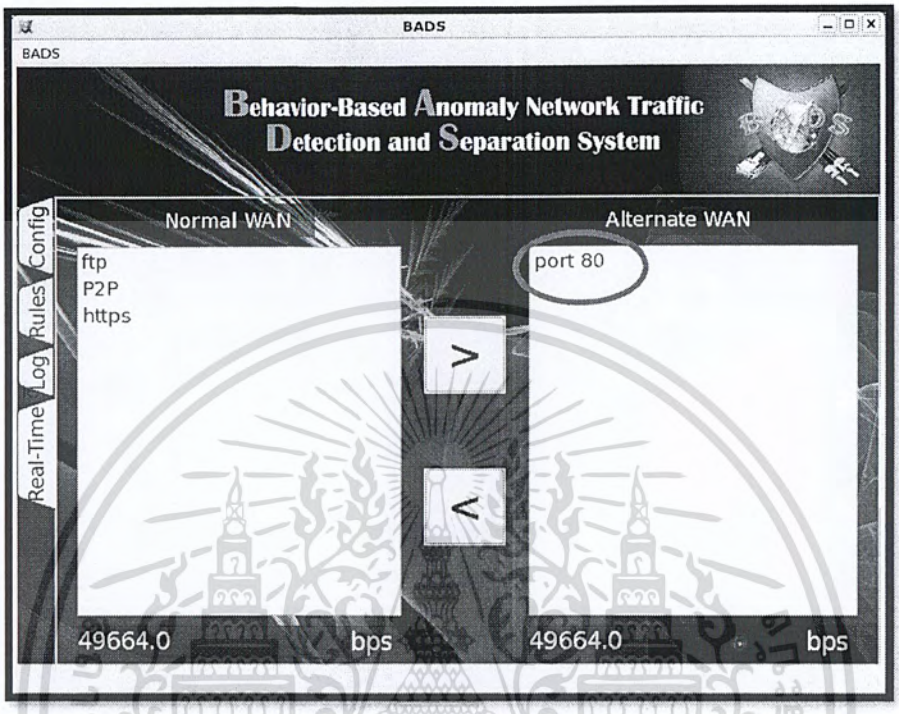
```
[root@network06 ~]# tcpdump -i eth2 icmp && src host 161.246.5.99
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on eth2, link-type EN10MB (Ethernet), capture size 96 bytes
11:09:04.650794 IP network09.ce.kmitl.ac.th > network05.ce.kmitl.ac.th: ICMP echo request, id 1, seq 264, length 40
11:09:04.651649 IP network05.ce.kmitl.ac.th > network09.ce.kmitl.ac.th: ICMP echo reply, id 1, seq 264, length 40
11:09:05.651326 IP network09.ce.kmitl.ac.th > network05.ce.kmitl.ac.th: ICMP echo request, id 1, seq 265, length 40
11:09:05.651530 IP network05.ce.kmitl.ac.th > network09.ce.kmitl.ac.th: ICMP echo reply, id 1, seq 265, length 40
11:09:06.650785 IP network09.ce.kmitl.ac.th > network05.ce.kmitl.ac.th: ICMP echo request, id 1, seq 266, length 40
11:09:06.650974 IP network05.ce.kmitl.ac.th > network09.ce.kmitl.ac.th: ICMP echo reply, id 1, seq 266, length 40
11:09:07.652321 IP network09.ce.kmitl.ac.th > network05.ce.kmitl.ac.th: ICMP echo request, id 1, seq 267, length 40
11:09:07.652514 IP network05.ce.kmitl.ac.th > network09.ce.kmitl.ac.th: ICMP echo reply, id 1, seq 267, length 40

8 packets captured
8 packets received by filter
0 packets dropped by kernel
```

รูป 4.15 ทราฟฟิกของ IP 192.168.1.2 ที่วิ่งผ่าน Alternate WAN

จากรูป 4.15 แสดงไอพีที่วิ่งผ่าน Alternate WAN eth2 โดยสังเกตจากไอพี network09.ce.kmitl.ac.th ซึ่งเป็นไอพี 192.168.1.2 มีการเปลี่ยนเส้นทางไปยัง Alternate WAN eth2 จริง

4.4.3 การทดสอบแยกทราฟฟิกตามพอร์ตที่กำหนด

ในการทดลองคัดแยกทราฟฟิกตามพอร์ตที่กำหนดในที่นี้กำหนดให้พอร์ตที่สนใจเป็น Port 80 (http) (interface eth 2 IP 161.246.5.99) ดังรูป 4.16



รูป 4.16 กำหนด rule Port ให้กับ Alternate WAN

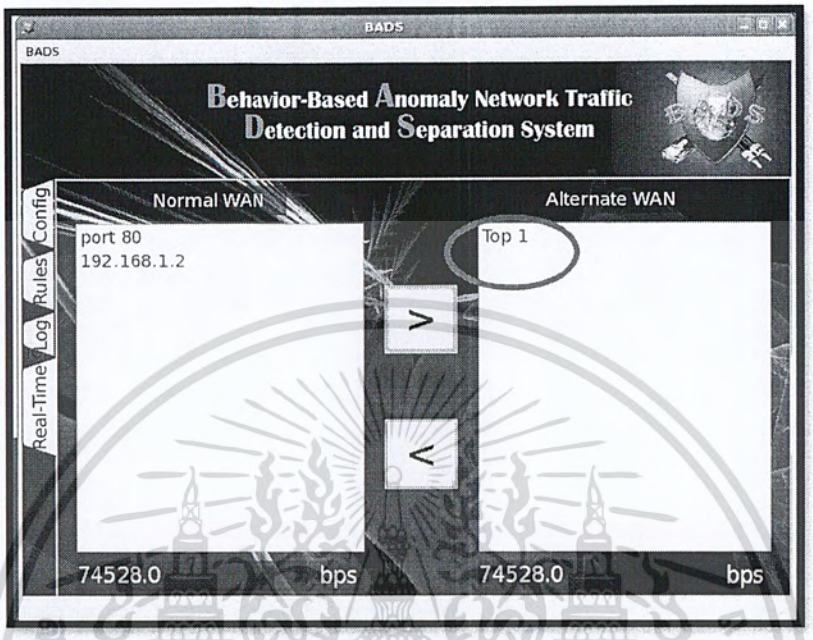
```
[root@network06 ~]# tcpdump -i eth2 src host 161.246.5.99 && dst port 80
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on eth2, link-type EN10MB (Ethernet), capture size 96 bytes
11:13:37.766274 arp who-has emerald.ce.kmitl.ac.th tell network09.ce.kmitl.ac.th
11:13:37.766900 IP network09.ce.kmitl.ac.th.55037 > 65.55.7.141.http: S 1729848308:1729848308(0) win 8192 <m
11:13:37.767860 IP network09.ce.kmitl.ac.th.55037 > 65.55.7.141.http: . ack 1024784443 win 65392
11:13:37.770619 IP network09.ce.kmitl.ac.th.55037 > 65.55.7.141.http: . 0:536(536) ack 1 win 65392
11:13:37.770643 IP network09.ce.kmitl.ac.th.55037 > 65.55.7.141.http: P 536:570(34) ack 1 win 65392
11:13:37.773798 IP network09.ce.kmitl.ac.th.55037 > 65.55.7.141.http: . ack 112 win 65282
11:13:37.779715 IP network09.ce.kmitl.ac.th.55037 > 65.55.7.141.http: F 570:570(0) ack 112 win 65282
11:13:37.896960 IP network09.ce.kmitl.ac.th.55041 > 207.46.49.133.http: S 2535422302:2535422302(0) win 8192
11:13:37.899341 IP network09.ce.kmitl.ac.th.55041 > 207.46.49.133.http: . ack 1251693679 win 65392
11:13:37.902437 IP network09.ce.kmitl.ac.th.55041 > 207.46.49.133.http: . 0:536(536) ack 1 win 65392
11:13:37.902463 IP network09.ce.kmitl.ac.th.55041 > 207.46.49.133.http: P 536:703(167) ack 1 win 65392
11:13:37.912033 IP network09.ce.kmitl.ac.th.55043 > 207.46.49.133.http: S 369196465:369196465(0) win 8192 <m
11:13:37.918498 IP network09.ce.kmitl.ac.th.55041 > 207.46.49.133.http: . ack 224 win 65170
11:13:37.918726 IP network09.ce.kmitl.ac.th.55041 > 207.46.49.133.http: F 703:703(0) ack 224 win 65170
```

รูป 4.17 ทราฟฟิกที่ใช้งาน port 80 วิ่งผ่าน Alternate WAN

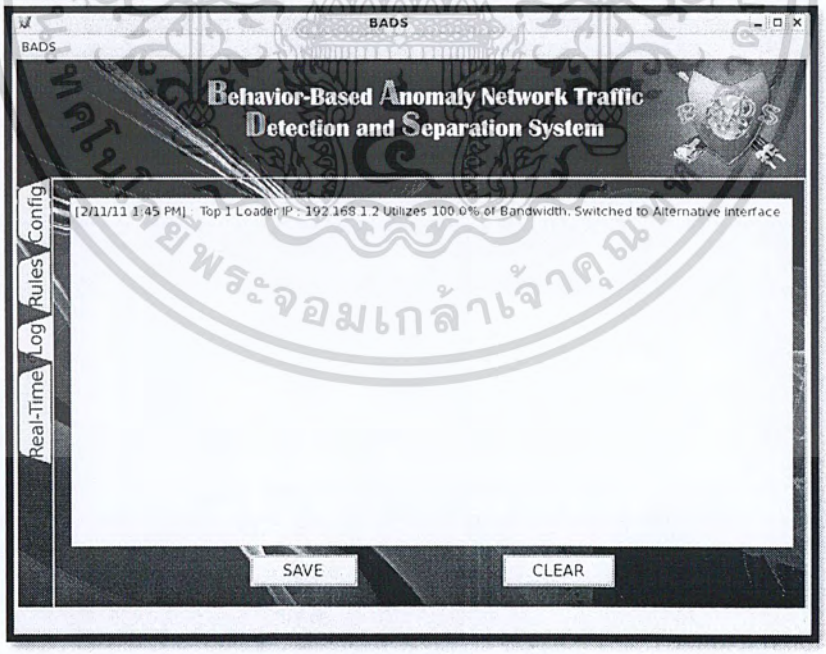
จากรูป 4.17 แสดงทราฟฟิกที่ใช้งานพอร์ต 80 ได้ทำการวิ่งผ่าน Alternate WAN จริง

4.4.3 การทดสอบแยกทราฟฟิกตามปริมาณการใช้งานแบนวิดธ์

ในการทดลองคัดแยกทราฟฟิกตามปริมาณการใช้งานในที่นี้กำหนดให้การใช้งานแบนด์วิดธ์อันดับ 1 ทำการ Forward ออก Alternate WAN ดังรูป 4.18



รูป 4.18 กำหนด rule Most Bandwidth Usage ให้กับ Alternate WAN



รูป 4.19 log IP ที่ใช้งานสูงสุด

จากรูป 4.19 ลอคจะแสดงไอพีแอดเดรสที่มีการใช้งานปริมาณแบนวิดธ์สูงสุดตามจำนวนที่เลือกไว้

บทที่ 5
สรุปผลการทดลอง ปัญหาและแนวทางการพัฒนา

5.1 สรุปผลการดำเนินโครงการ

จากการทำการทดลองสรุปได้ว่า ระบบสามารถคัดแยกทราฟฟิกไปยังคนละเส้นทางได้ตามกฎที่ผู้ดูแลระบบสร้างขึ้นและเลือกใช้เมื่อแพคเกตเข้ามาในระบบจะสามารถแบ่งออกเป็น 4 กรณีคือ คัดแยกตามโปรโตคอลที่ใช้งาน คัดแยกตามไอพีแอดเดรส คัดแยกตามพอร์ท คัดแยกตามปริมาณการใช้งาน จากนั้นส่งต่อให้ไอพีเทเบิลทำการมาร์คแพคเกตนั้นแล้วส่งต่อให้ไอพีเราท์ทำการส่งต่อแพคเกตไปยังอินเตอร์เฟสตรงกับกฎในการใช้งานที่ตั้งไว้ ระบบสามารถแสดงปริมาณการใช้งานของแต่ละอินเตอร์เฟส และแสดงไฟล์ลอคที่สามารถบันทึกเป็นไฟล์ข้อความได้ โดยทั้งหมดนี้ผู้ใช้ควบคุมโดยผ่านหน้าต่างโปรแกรมซึ่งใช้ภาษาไพธอนในการพัฒนาใช้ฐานข้อมูลมายเอสคิวแอล ในการเก็บกฎและลอคของสนอร์ท

5.2 ปัญหาและอุปสรรค

1) ผู้พัฒนายังขาดความรู้และประสบการณ์ในการใช้งานระบบปฏิบัติการลีนุกซ์
2) คุณสมบัติของไลบราลี่พีวายคิวที 4 บางคุณสมบัติไม่สามารถใช้งานบนระบบปฏิบัติการลีนุกซ์ได้ทำให้ยากต่อการออกแบบส่วนแสดงผลข้อมูล
3) การตรวจจับการรับ-ส่งข้อมูลแบบเพียร์ทูเพียร์ สามารถทำได้ยากเนื่องจากโปรโตคอลมีความซับซ้อนสูง
4) การแสดงผลกราฟข้อมูลเข้า-ออกแต่ละอินเตอร์เฟสสามารถทำได้ยาก
5) มีความหน่วงในการอ่าน-เขียนไฟล์บนฐานข้อมูลเล็กน้อย

5.3 แนวทางการพัฒนาโครงการและประยุกต์ใช้กับงานอื่นๆ

1) พัฒนาระบบให้สามารถเซตค่าความสำคัญของกฎได้ในหน้าตั้งค่ากฎ
2) พัฒนาระบบให้มีโหมดการทำงานแบบอัตโนมัติ เพื่อความสะดวกในการใช้งานของผู้ดูแลระบบ
3) พัฒนาผลปริมาณการใช้งานทราฟฟิกของแต่ละอินเตอร์เฟสเป็นกราฟแบบตามเวลาจริงได้
4) แสดงลอคและรายงานผลให้เข้าใจง่ายขึ้น

บรรณานุกรม

รศ. ดร.ศุภมิตร จิตตะยโศธร. **เอกสารประกอบการอบรม Database Design & SQL**. ศูนย์ฝึกอบรมเนคเทค

ผศ. เกียรติกูล เจียรนัยธนะกิจ. 2551. **ทฤษฎีการคำนวณ Theory of Computation**. กรุงเทพฯ : บริษัท ซัคเซส มีเดีย.

ชมภูนุช สันธนะผล และ ชาญกฤษณ์ มากมี. 2552. **รายงานวิชา Computer Project หัวข้อเรื่อง "Firewall.** : คณะวิศวกรรมศาสตร์ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

Linux Advanced Routing Mini HOWTO [Online] Available :
http://www.linuxhorizon.ro/iproute2.html

Configure Snort to log packets to MySQL.2008 [Online] Availables:
http://www.zdnetasia.com/configure-snort-to-log-packets-to-mysql-62039525.htm

ภูวดล ด่านระหาญ.2544. **การติดตั้ง Snort ร่วมกับ ACID (+MySQL)** [Online] Availables:
http://www.thaicert.org/paper/ids/snort2.php

ACID: Database (v100-103) ER Diagram.2010 [Online] Availables:
http://www.andrew.cmu.edu/user/rdanyliw/snort/acid_db_er_v102.html#schema

TCPDUMP Protocol Analyzer.2010 [Online] Availables:
http://www.kunawut.com/blog/?p=164

Python - Linux: Parse Network Stats From ifconfig [Online] Availables:
http://coreygoldberg.blogspot.com/2010/09/python-linux-parse-network-stats-from.html

Python v2.7.1 documentation [Online] Availables:
http://docs.python.org/

Emerging Threats.net Open rulesets.2010 [Online] Availables:

http://rules.emergingthreats.net/

Jialong He.**TCPDUMP Quick Reference** [Online] Availables:

http://taviso.decsystem.org/files/tcpdump_quickref.pdf

Snort Team. 2010.**Snort Users Manual 2.8.6.** :

ภาคผนวก ก
คู่มือการติดตั้ง

ก.1 ความต้องการทางด้านซอร์ฟแวร์และการติดตั้งซอร์ฟแวร์

1) ทำการติดตั้งระบบปฏิบัติการ CentOS ซึ่งสามารถดาวน์โหลดได้จาก

http://www.centos.org/modules/tinycontent/index.php?id=15

จากนั้นใช้คำสั่ง

#yum –y update

เพื่ออัพเดต Tools ต่างๆ

ใช้คำสั่ง

#yum groupinstall "Development Tools"

เพื่อติดตั้งไลบรารี่และเครื่องมือต่างๆสำหรับนักพัฒนา

2) ติดตั้ง iptables1.3.5, MySQL 5.0.77, MySQL-python1.2.1-1

ผ่านการใช้คำสั่ง yum install ดังนี้

#yum install iptables

#yum install MySQL

#yum install MySQL-python

3) ติดตั้ง python 2.7 โดย ดาวน์โหลดจาก http://www.python.org

จะได้ไฟล์ .tar.bz2 มาให้ทำการแตกไฟล์จะได้โฟลเดอร์ Python-2.7

เข้าไปยังโฟลเดอร์ดังกล่าว แล้วใช้คำสั่ง

#./configure

#make

#make install

4) ติดตั้ง Qt โดยใช้คำสั่งดังนี้

#yum install qt4

#yum install qt4-devel

#yum install qt4-doc

#yum install qt4-postgresql

#yum install qt4-odbc

#yum install qt4-sqlite

#yum install qt4-creator

#rpm -ivh http://software.freivald.com/centos/software.freivald.com-1.0.0-1.noarch.rpm

#yum update fontconfig fontconfig-devel qt4 qt4-devel qt4-doc qt4-postgresql qt4-odbc qt4-sqlite qt-creator

5) ทำการติดตั้ง PyQt

ดาวน์โหลด PyQt4.8.2 จากลิงค์

http://www.riverbankcomputing.co.uk/software/pyqt/download

ดาวน์โหลด sip-4.12 จาก

http://www.riverbankcomputing.co.uk/software/sip/download

ทำการแตกไฟล์ ทั้ง 2 ไฟล์

ไปยังโฟลเดอร์ sip-4.12 แล้วใช้คำสั่งดังนี้

#python configure.py

#make

#make install

ไปยังโฟลเดอร์ PyQt4.8.2 แล้วใช้คำสั่งดังนี้

#python configure.py -q /usr/lib/qt4/bin/qmake

#make

#make install

6) ติดตั้ง Snort และตั้งค่าให้เก็บ log ลงฐานข้อมูล

โดยทำตามขั้นตอนดังต่อไปนี้เปิดหน้าเทอร์มินอลขึ้นมาจากนั้นพิมพ์คำสั่ง

#cd /root

#mkdir snortinstall

#cd /root/snortinstall

ดาวน์โหลด Snort เวอร์ชั่น 2.8.6 จาก www.snort.org จากนั้นให้ขยายไฟล์ออก ดังนี้

#tar xvzf snort-2.8.6.tar.gz

#cd snort-2.8.6

#./configure --with-mysql –enable-dynamicplugin

#make

#make install

ทั้งนี้ในขณะที่โปรแกรมกำลังคอมไพล์อยู่นั้น ควรจะสังเกตเห็นคำว่า checking for mysql... yes ปรากฏขึ้น จากนั้นให้Copyข้อมูล configuration และ rules files จาก source ของ Snort ไปยัง /etc/snort เพื่อความเป็นระเบียบ สร้างไดเร็คทอรีเพื่อเก็บลอคไฟล์ของ Snort ทั้งหมด แยกต่างหาก

#groupadd snort

#useradd –g snort snort –s /sbin/nologin

#mkdir /etc/snort

#mkdir /etc/snort/rules

#mkdir /etc/snort/so_rules

#mkdir /var/log/snort

chown root:snort /etc/snort/snort.conf

chmod 0640 /etc/snort/snort.conf

#cp * /etc/snort

ทำการแก้ไขไฟล์ snort.conf ซึ่งอยู่ในไดเรคทอรี /etc/snort โดยใช้ text editer ทำการแก้ไขดังนี้

“var RULE_PATH ../rules” เปลี่ยนบรรทัดนี้เป็น “var RULE_PATH rules”

“var SO_RULE_PATH ../rules” เปลี่ยนบรรทัดนี้เป็น “var SO_RULE_PATH so_rules”

ทำการตั้งค่าให้ snort เก็บ log ลงในฐานข้อมูล

mysql -u root -p

#mysql> create database snort;

#mysql> grant INSERT,SELECT,UPDATE,CREATE,DELETE,EXECUTE on snort.*

to snort@snort.host;

#mysql> set password for snort@snort.host=PASSWORD('snortpass');

#mysql> flush privileges;

#mysql> q

ทำการสร้าง Schemas ของสนอร์ทในฐานข้อมูล โดยใช้คำสั่ง

mysql -u root -p snort </root/snortinstall/snort2.8.6/schemas/create_mysql

ทำการแก้ไขไฟล์ snort.conf โดยเพิ่มบรรทัดต่อไปนี้เข้าไปในส่วนของ output

output database: log, mysql, user=snort password=snortpass dbname=snort host=127.0.0.1

ทดสอบการทำงานของ snort ด้วยคำสั่งต่อไปนี้

#/usr/local/bin/snort –u snort –g snort –c /etc/snort/snort.conf/ -i eth0

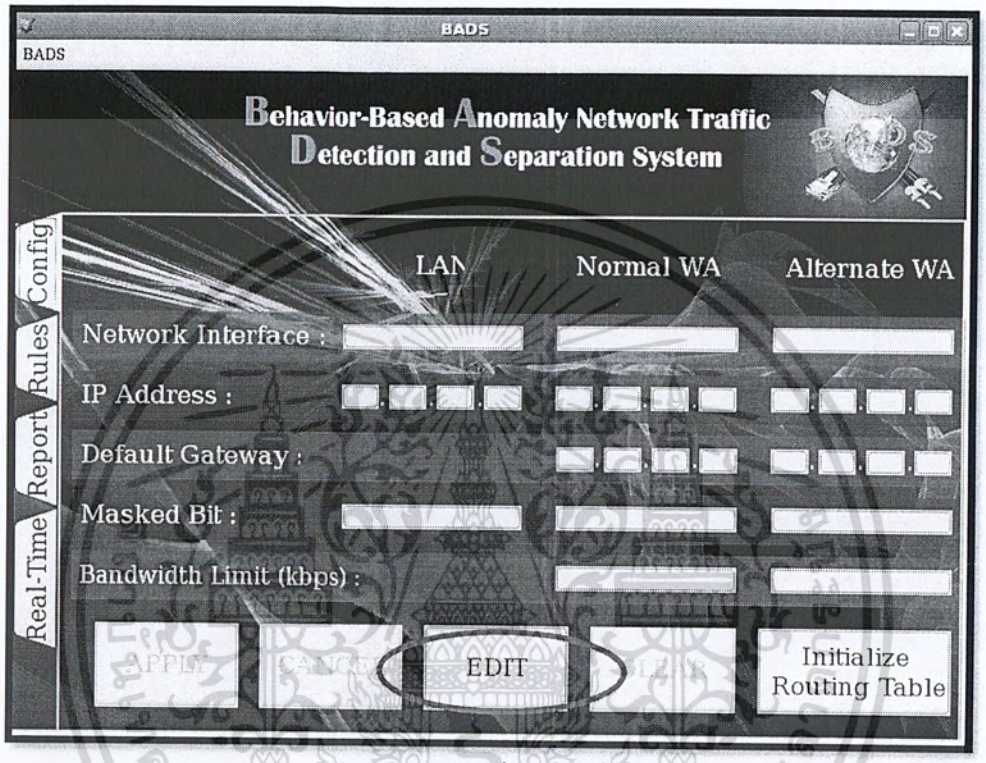
ก.2 ความต้องการของระบบด้านฮาร์ดแวร์

1) Network Interface Card อย่างต่ำ 3 Card

ภาคผนวก ข

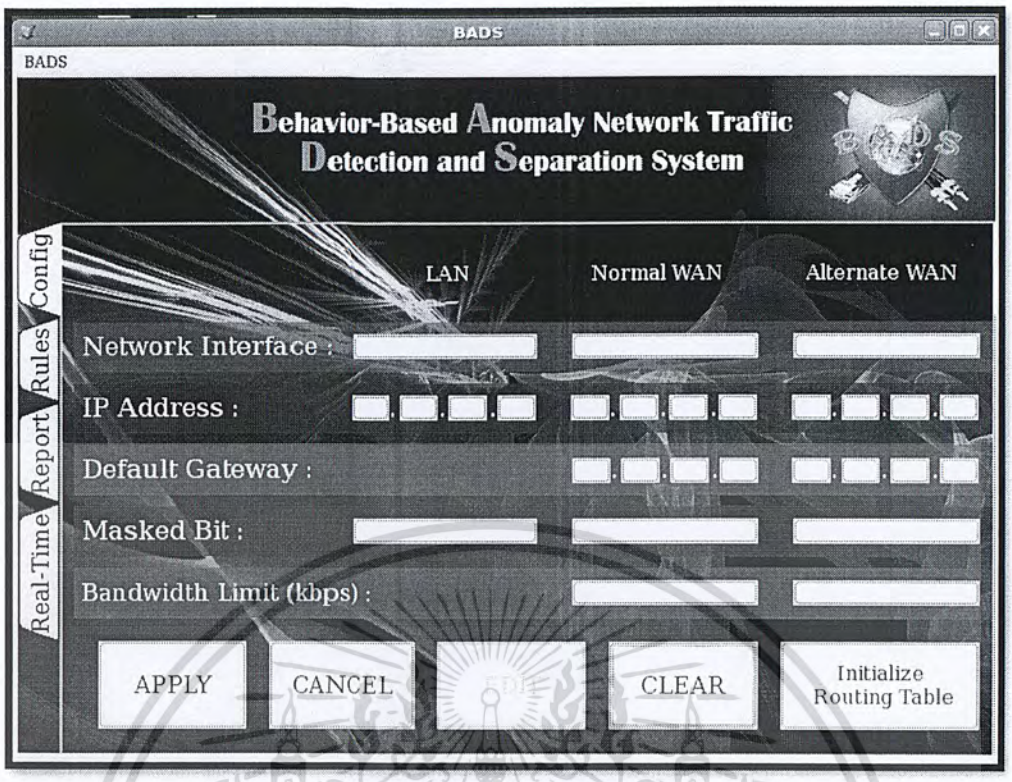
คู่มือการใช้งานโปรแกรม

หน้าต่าง Config : เริ่มเปิดโปรแกรมขึ้นมาจะพบกับหน้าต่างดังรูป ข.1



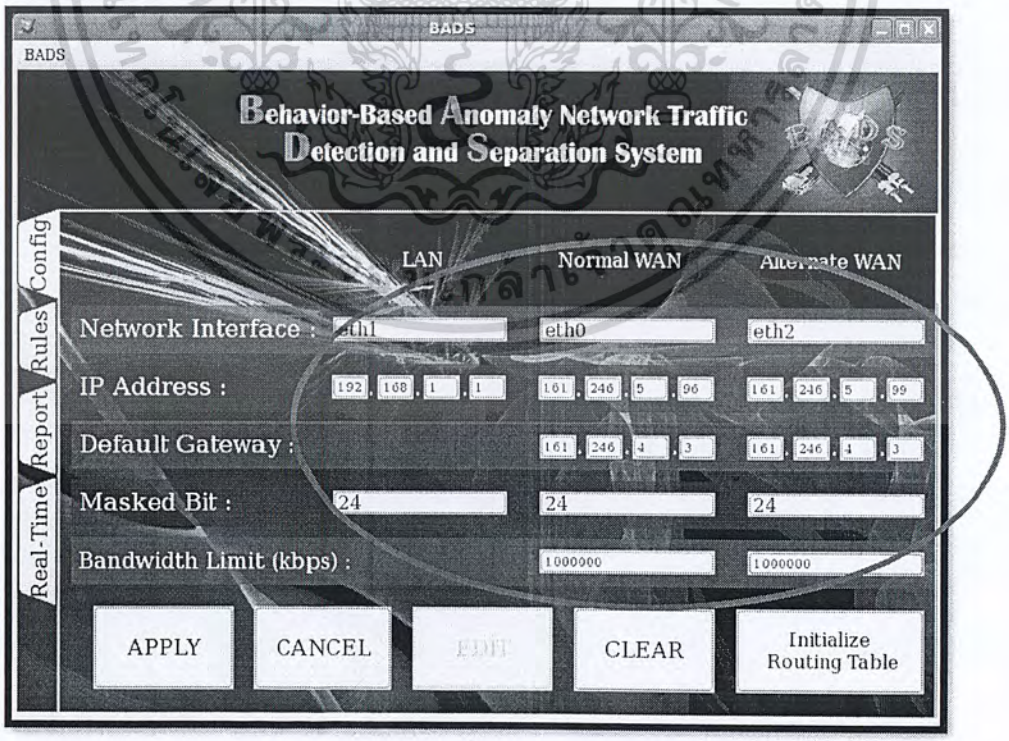
รูป ข.1 หน้าต่างเริ่มต้นโปรแกรม

เมื่อต้องการเพิ่มข้อมูลเนตเวิร์กในหน้า Config ให้คลิกที่ปุ่ม EDIT เพื่อกรอกข้อมูลให้ครบในช่องสีขาวที่กำหนดไว้ ดังรูป ข.2



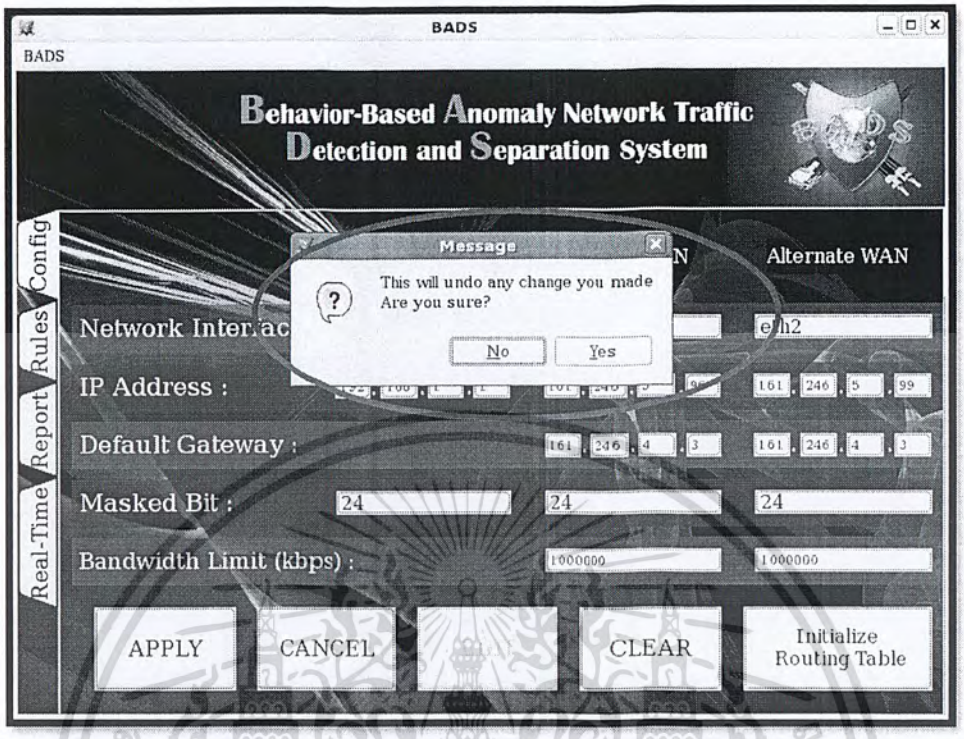
รูป ข.2 หน้าต่าง ตั้งค่าอินเตอร์เฟส

เมื่อกรอกข้อมูลในรูป ข.3 จนครบตามกำหนดแล้วให้คลิกปุ่ม APPLY เพื่อตั้งค่าใช้งาน



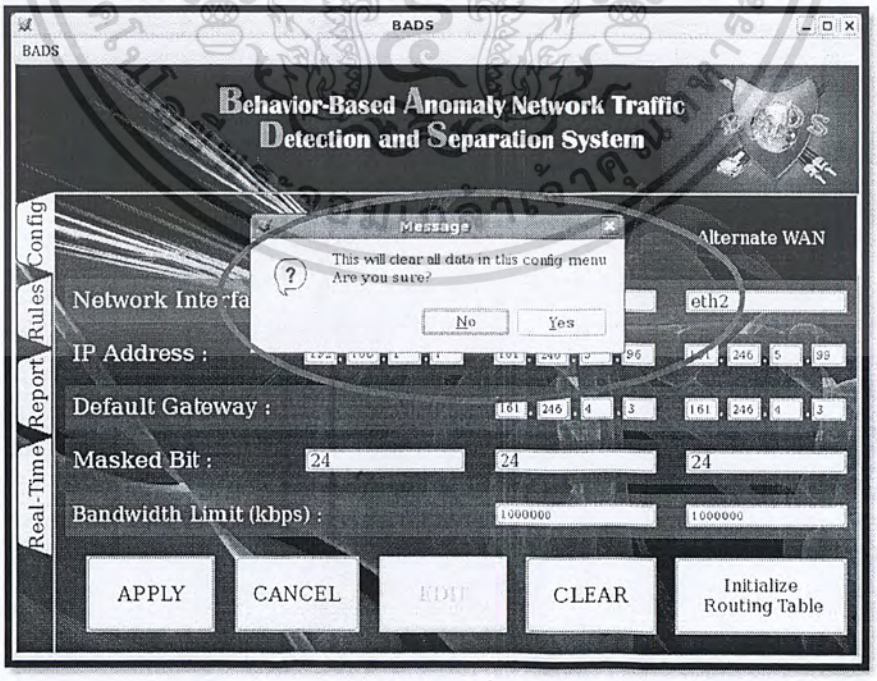
รูป ข.3 ทำการใส่ค่าเริ่มต้นของอินเตอร์เฟส

หากต้องการยกเลิก ให้คลิกปุ่ม CANCLE จะปรากฏหน้าต่างดังรูป ข.4



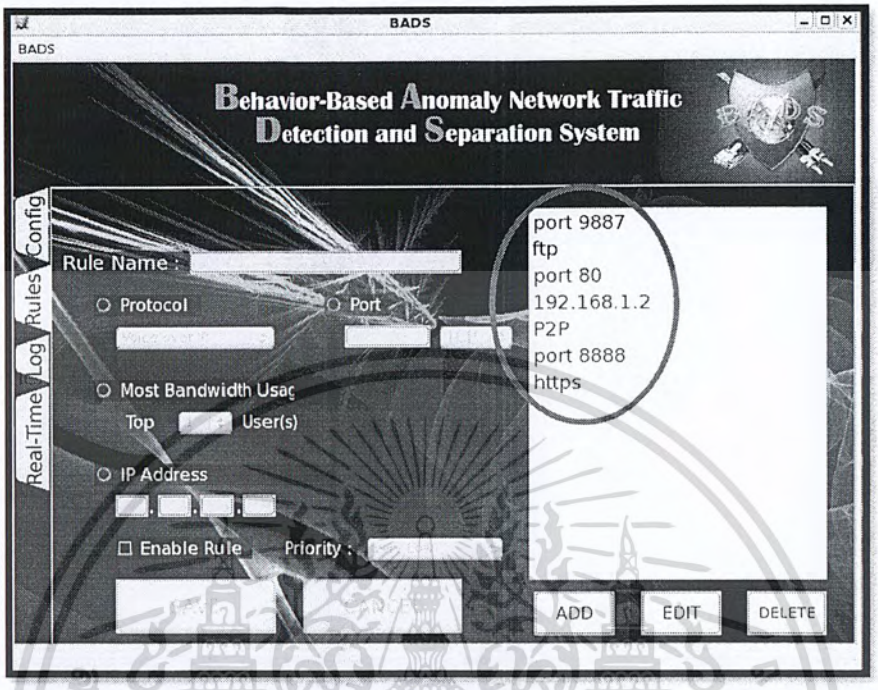
รูป ข.4 ข้อความแจ้งเตือนเมื่อคลิกปุ่ม CANCLE

หากต้องการล้างข้อมูลและกรอกใหม่ให้คลิกปุ่ม CLEAR จะปรากฏหน้าต่างดังรูป ข.5



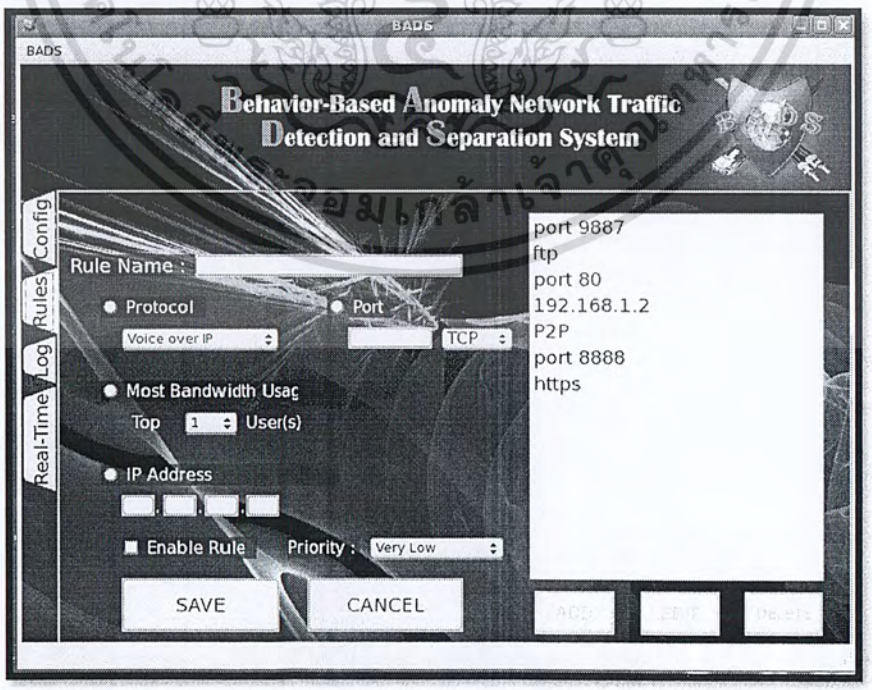
รูป ข.5 ข้อความแจ้งเตือนเมื่อคลิกปุ่ม CLEAR

หน้าต่าง Rules : คลิกที่แท็ป Rules จะพบกับหน้าต่างดังรูปภายในวงกลมสีแดงคือ กฎที่ผู้ใช้เคยตั้งไว้แล้วก่อนหน้านี้ ดังรูป ข.6



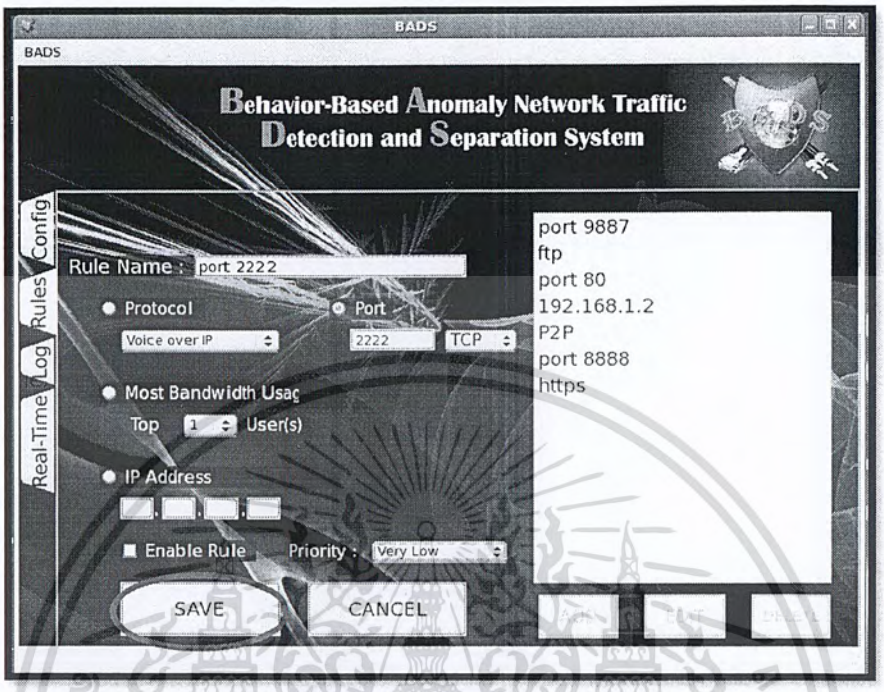
รูป ข.6 หน้าต่างปรับแต่งกฎ

คลิกที่ ADD เพื่อเพิ่ม rule โดยกรอกข้อมูลและเลือกเกณฑ์ที่ใช้ในการคัดแยกดังรูป ข.7



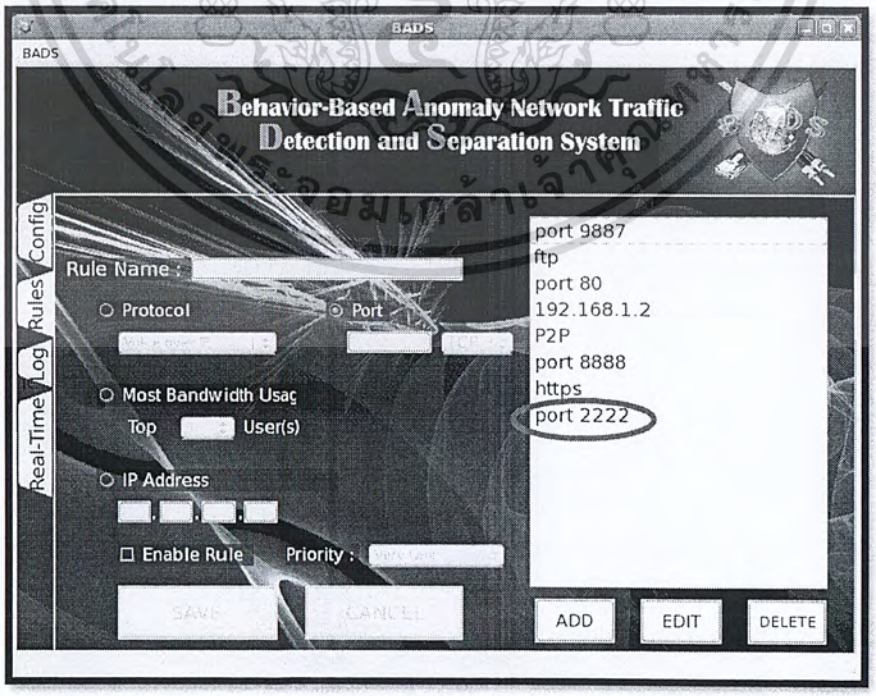
รูป ข.7 การเพิ่มกฎ

เมื่อกรอกข้อมูลทั้งหมดแล้วให้คลิกที่ปุ่ม SAVE เพื่อบันทึกกฎลงในฐานข้อมูล หากต้องการใช้งานกฎที่สร้างขึ้นให้คลิกที่ช่อง Enable Rule ดังรูป ข.8



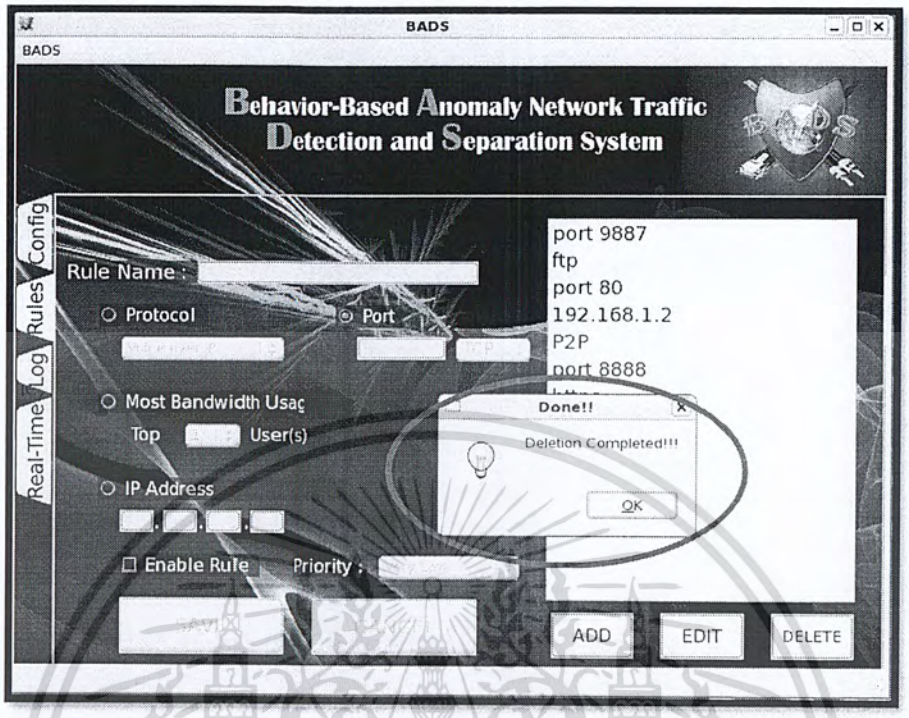
รูป ข.8 คลิกปุ่ม SAVE เพื่อนบันทึกกฎ

จะปรากฏชื่อกฎที่ได้ตั้งไว้ใน List Box ทางด้านขวาของโปรแกรม ดังรูป ข.9



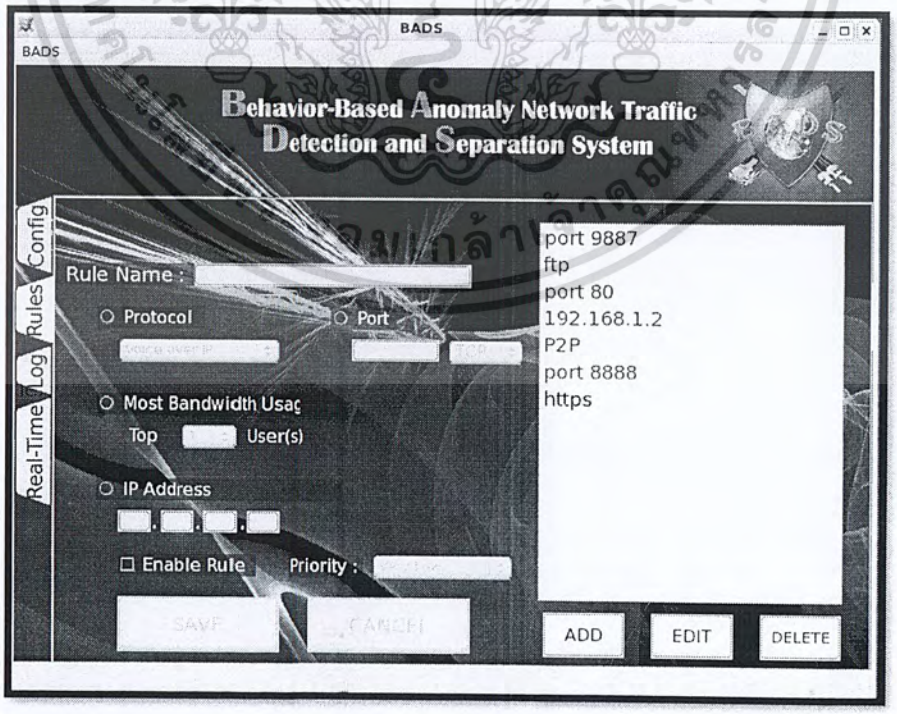
รูป ข.9 รายชื่อกฎ ใน List Box

หากต้องการลบกฎที่ตั้งขึ้นให้คลิกที่ปุ่ม DELETE จะปรากฏหน้าต่างดังรูป ข.10



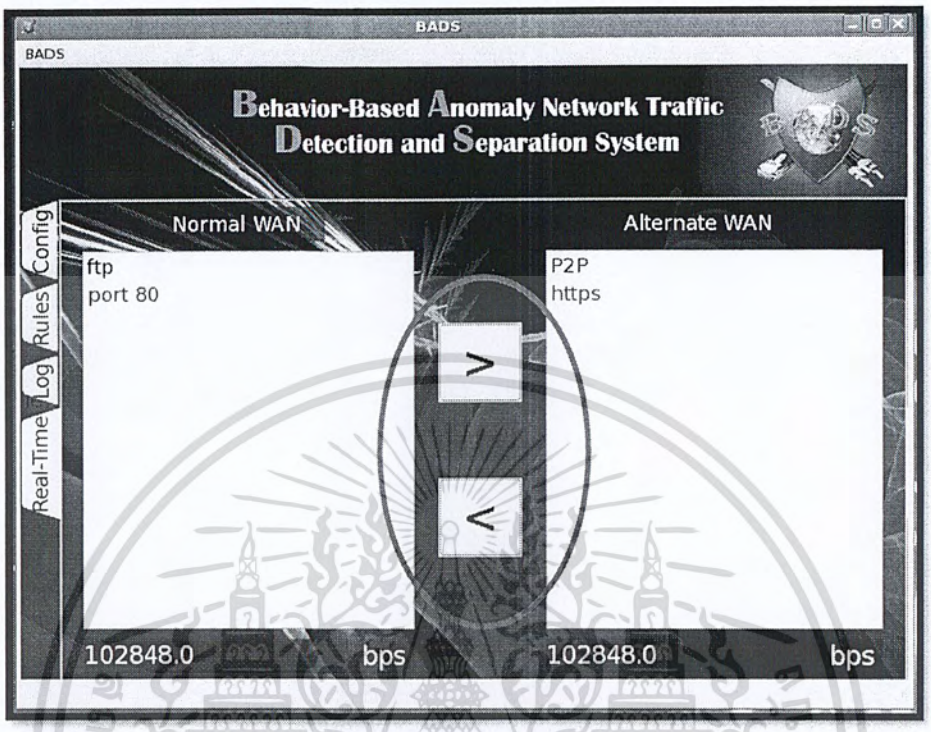
รูป ข.10 ข้อความแจ้งเตือนเมื่อคลิกปุ่ม DELETE

สังเกตได้ว่ากฎที่ต้องการลบ ถูกลบออกไปแล้ว ดังรูป ข.11

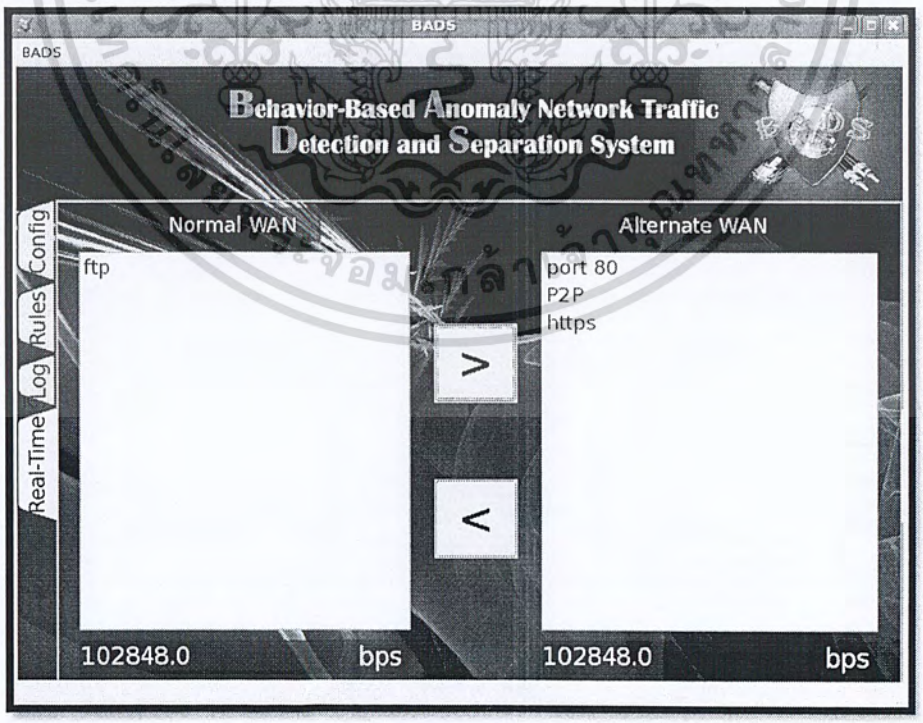


รูป ข.11 การลบกฎ

หน้าต่าง Real-Time : เป็นหน้าต่างที่ใช้สำหรับเลือกกฎให้ใช้กับอินเตอร์เฟสที่ต้องการ โดยคลิกที่ปุ่มทิศทางตรงกลาง ดังรูป ข.12 และ ข.13

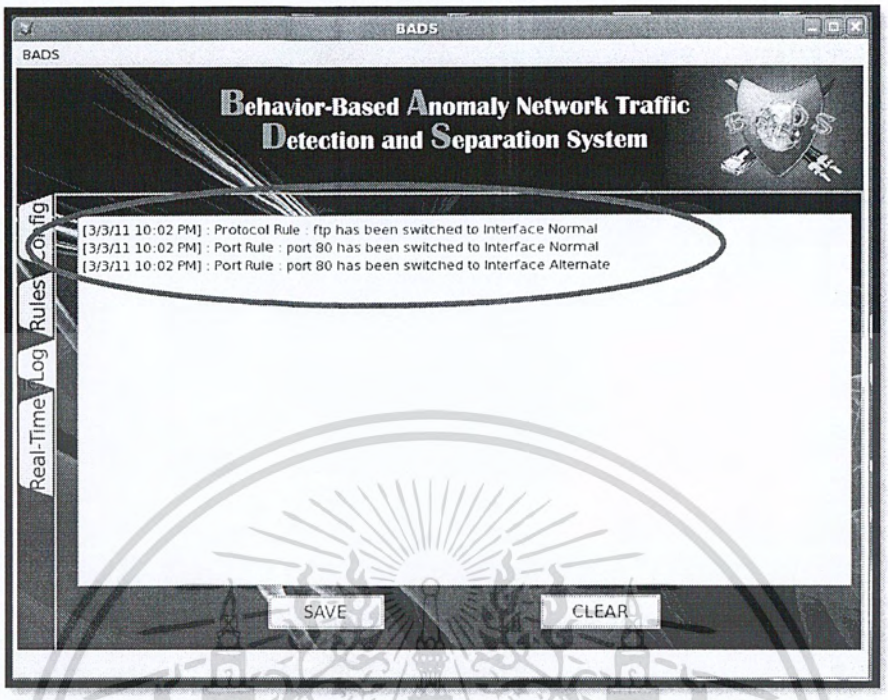


รูป ข.12 หน้าต่าง Real-Time



รูป ข.13 การสวิตช์กฎให้กับอินเตอร์เฟส

หน้าต่าง Log : โดยแสดงในรูปของลอคไฟล์และสามารถบันทึกเป็น Text File ได้ดังรูป ข.14



รูป ข.14 หน้าต่างลอค