พัฒนาอุปกรณ์ในการจัดการและควบคุมข้อมูลในเครือข่าย

DEVELOPMENT OF TRAFFIC MANAGEMENT APPLIANCE

โดย

นุชา เสาสีอ่อน

NUCHA SAOSEEON

อาจารย์ที่ปรึกษา

รศ.ดร.โชติพัชร์ ภรณวลัย

รายงานนี้เป็นส่วนหนึ่งของวิชาโครงการพัฒนาระบบงาน
หลักสูตรวิทยาศาสตรมหาบัณฑิต สาขาวิชาเทคโนโลยีสารสนเทศ
คณะเทคโนโลยีสารสนเทศ
สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง
ภาคเรียนที่ 2 ปีการศึกษา 2552

DEVELOPMENT OF TRAFFIC MANAGEMENT APPLIANCE

NUCHA SAOSEEON

A REPORT SUBMITTED IN PARTIAL FULFILLMENT OF THE

REQUIREMENTS OF THE COURSE

SYSTEM DEVELOPMENT PROJECT

MASTER OF SCIENCE PROGRAM IN INFORMATION TECHNOLOGY

FACULTY OF INFORMATION TECHNOLOGY

KING MONGKUT'S INSTITUTE OF TECHNOLOGY LADKRABANG

2/2009

ใบรับรองโครงการพัฒนาระบบงาน (System Development Project)

เรื่อง

พัฒนาอุปกรณ์ในการจัดการและควบคุมข้อมูลในเครือข่าย

Development of Traffic Management Appliance

นายนุชา เสาสีอ่อน

รหัสประจำตัว 50066545

ขอรับรองว่ารายงานฉบับนี้ข้าพเจ้าไม่ได้คัดลอกมาจากที่ใด
รายงานฉบับนี้ได้รับการตรวจสอบและอนุมัติให้เป็นส่วนหนึ่งของ
การศึกษาวิชาโครงการพัฒนาระบบงาน หลักสูตรวิทยาศาสตรมหาบัณฑิต (เทคโนโลยีสารสนเทศ)
ภาคเรียนที่ 2 ปีการศึกษา 2552

.....อาจารย์ที่ปรึกษา
(รศ.ดร.โชติพัชร์ ภรณวลัย)

.....กรรมการสอบ
(รศ.ดร.นพพร โชติกกำธร)

.....กรรมการสอบ
(รศ.ดร. จันทร์บูรณ์ สถิตวิริยวงศ์)

หัวข้อ	พัฒนาอุปกรณ์ในการจัดการและควบคุมข้อมูลในเครือข่าย
นักศึกษา	นายนุชา เสาสีอ่อน
รหัสนักศึกษา	50066545
ปริญญา	วิทยาศาสตรมหาบัณฑิต
สาขาวิชา	เทคโนโลยีสารสนเทศ
แขนงวิชา	วิทยาการสารสนเทศ
ปีการศึกษา	2552
อาจารย์ที่ปรึกษา	รศ.ดร.โชติพัชร์ ภรณวลัย

บทคัดย่อ

โครงงานพัฒนาอุปกรณ์ในการจัดการและควบคุมข้อมูลในเครือข่าย ได้พัฒนาระบบขึ้นบน Embedded System (iBord) ซึ่งใช้ส่วนประมวลผล RISC 100MHz, หน่วยความจำ 32MB, ส่วนเก็บข้อมูล (Flash) 8MB และ 2 อีเทอร์เน็ต โดยมีระบบปฏิบัติการเป็นลินุกซ์เคอร์แนล 2.6 และมีแอพพลิเคชันหลักที่ใช้พัฒนาระบบคือ tc (Traffic Control) และ iptables โดยที่ tc จะจัดการในส่วนของการควบคุมการใช้แบนด์วิดท์ซึ่งเป็นแอพพลิเคชันอยู่ในชุดของ iproute2 โดยอัลกอริทึมการจัดการแบนด์วิดท์ใช้ HTB (Hierarchy Token Bucket) ซึ่งเป็นหนึ่งในความสามารถของ tc โดย iptables ทำหน้าที่จัดการควบคุมเส้นทางของแพคเกตให้เป็นไปตามที่กำหนด สำหรับส่วนติดต่อผู้ใช้นั้นพัฒนาโดยใช้ภาษา PHP และมี SQLite ในการจัดการฐานข้อมูลของระบบ ด้านความสามารถของระบบนั้นสามารถกำหนดการใช้งานแบนด์วิดท์ตามผู้ใช้, กลุ่มผู้ใช้, ไอพีและ แอพพลิเคชั่นได้ โดยสามารถระบุขนาดแบนด์วิดท์ ต่ำสุด, สูงสุด และความสำคัญของข้อมูลได้

Title	Development of Traffic Management Appliance
Student	Mr. Nucha Saoseeon
Student ID.	50066545
Degree	Master of Science
Program	Information Technology
Major	Information Science
Academic Year	2009
Advisor	Assoc. Prof. Dr. Chotipat Pornavalai

ABSTRACT

The development of traffic management appliance project has been developed, based on the Embedded System (iBord). The iBord is composed of: 100MHz RISC CPU, 32MB RAM, 8Mbps Flash Memory, 2 Ethernet ports and LINUX Kernel version 2.6. The main applications used to develop the system are tc (Traffic Control) and iptables. The tc is a part of iproute2 application that is used to control bandwidth usage. The algorithm used to control bandwidth is HTB (Hierarchy Token Bucket), which is a part of tc capability. The iptables are used to define the routes of each packet. The User Interface is developed using PHP, and SQLite is used for system database management. The system has capabilities to define bandwidth usage based on number of users, group of users, IP address, and applications. Furthermore, the system can define the minimum and maximum bandwidth. It also can perform data classification.

กิตติกรรมประกาศ

โครงงานพัฒนาระบบงานนี้สำเร็จได้ ด้วยคำแนะนำ และคำปรึกษาจาก รศ.ดร.โชติพัชร์ ภรณวลัย ซึ่งเป็นอาจารย์ที่ปรึกษา ข้าพเจ้าขอขอบพระคุณอย่างยิ่งที่ท่านได้ให้ความอนุเคราะห์ด้วยดีเสมอมา จนกระทั่งพัฒนาโครงการนี้ให้สำเร็จลุล่วงไปได้ด้วยดี

ขอกราบขอบพระคุณอาจารย์คณะเทคโนโลยีสารสนเทศ และอาจารย์ต่างคณะ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบังที่เข้ามาให้ความรู้ ตลอดจนอาจารย์ท่านอื่นที่มาจากต่างสถาบัน ที่ได้ให้ความรู้กับข้าพเจ้า

ขอขอบคุณบริษัท ทรู คอร์ปอเรชั่น จำกัด (มหาชน) ที่อนุญาตให้ลาหยุดงานในช่วงสอบ และให้ออกก่อนเวลาเลิกงานเพื่อเรียนในตอนเย็น รวมถึงเพื่อนร่วมงานที่สนับสนุนกันตลอดมา

ขอขอบคุณบริษัท เอ็มเบสเทคโนโลยี (ประเทศไทย) จำกัด และคุณณัฐนันท์ ศรีสะอาด ที่ให้คำแนะนำต่างๆ ที่เป็นประโยชน์อย่างมากในการพัฒนาระบบบน iBoard

ขอบคุณเพื่อนๆ พี่ๆ รุ่น 23.2 ที่มีส่วนสนับสนุน ให้กำลังใจและเป็นที่ปรึกษาในทุกๆเรื่อง

สุดท้ายนี้ข้าพเจ้าขอกราบขอบพระคุณ บิดา มารดา และครอบครัวของข้าพเจ้าที่เป็นกำลังใจ และให้การสนับสนุนในทุกเรื่องๆ ทำให้ข้าพเจ้าสามารถทำโครงงานพัฒนาระบบงานฉบับนี้สำเร็จลุล่วงด้วยดี ข้าพเจ้าขอระลึกในพระคุณและขอกราบขอบพระคุณมา ณ ที่นี้

คุณค่าและประโยชน์อันพึงมาจากวิทยานิพนธ์ฉบับนี้ ข้าพเจ้าขอมอบแด่ผู้มีพระคุณทุกท่าน

นุชา เสาสีอ่อน

สารบัญ

หน้า

สารบัญ (ต่อ)

หน้า

สารบัญ (ต่อ)

หน้า

สารบัญ (ต่อ)

หน้า

สารบัญตาราง

สารบัญรูป

รูปที่ หน้า

สารบัญรูป (ต่อ)

รูปที่ หน้า

สารบัญรูป (ต่อ)

รูปที่ หน้า

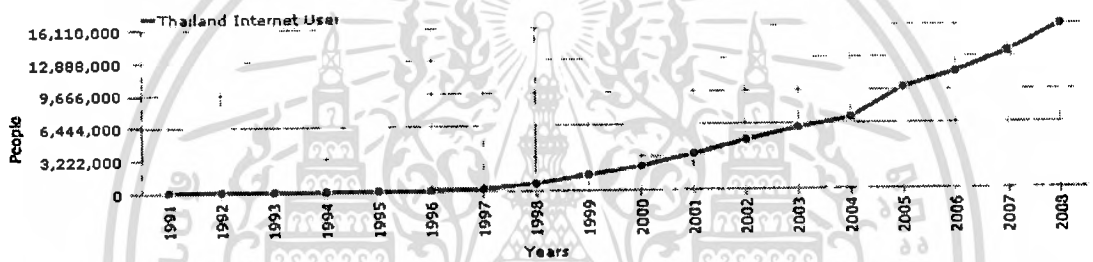
สารบัญรูป (ต่อ)

รูปที่ หน้า

บทที่ 1
บทนำ

1.1 ความเป็นมาและความสำคัญของปัญหา

ปัจจุบันอินเทอร์เน็ตเป็นที่รู้จักอย่างแพร่หลาย โดยมีการใช้งานกันอย่างกว้างขวางมากยิ่งขึ้นทำให้องค์กรต่างๆ ทั้งภาครัฐและเอกชน ซึ่งในประเทศไทยนั้นยอดผู้ใช้อินเทอร์เน็ตมีเพิ่มสูงขึ้นทุกปี จากข้อมูลของศูนย์เทคโนโลยีอิเล็กทรอนิกส์และคอมพิวเตอร์แห่งชาติ (NECTEC) จะเห็นได้ว่าตั้งแต่ปีพ.ศ. 2541 – 2551 ยอดผู้ใช้อินเทอร์เน็ตเพิ่มขึ้นอย่างต่อเนื่อง จนปัจจุบันมียอดผู้ใช้มากว่า 16 ล้านคน

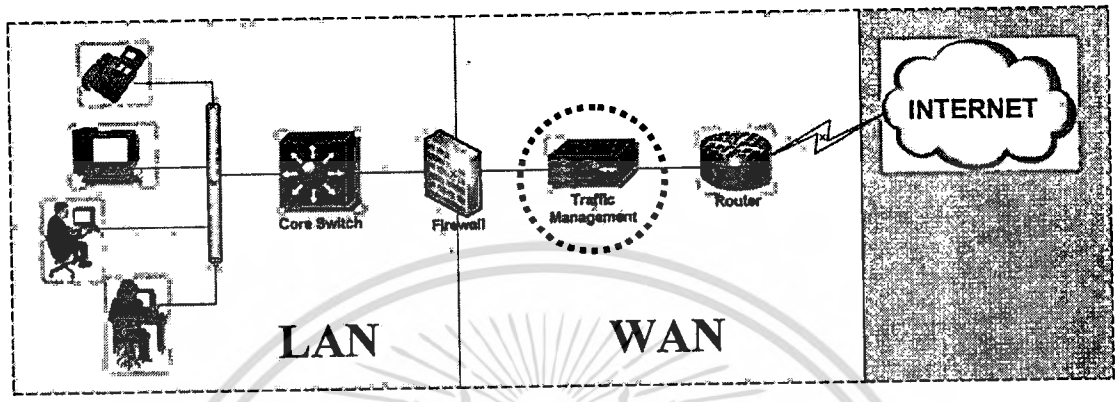


รูปที่ 1.1 จำนวนผู้ใช้อินเทอร์เน็ตในประเทศไทย

เนื่องจากการที่อินเทอร์เน็ตเป็นโครงข่ายที่มีขนาดใหญ่เชื่อมต่อทั่วโลก จึงทำให้อินเทอร์เน็ตเป็นเสมือนชุมชนขนาดใหญ่ไม่มีข้อจำกัดเรื่องของระยะทาง ดังนั้นการที่องค์กรใดก็ตามที่เชื่อมต่อเข้าอินเทอร์เน็ตเมื่อเชื่อมต่อเข้ามาก็จะได้รับประโยชน์ต่างจากอินเทอร์เน็ตด้วยเช่นกัน ตัวอย่างที่เห็นได้ชัดเจน การที่อินเทอร์เน็ตเป็นแหล่งข้อมูลขนาดใหญ่ช่วยให้สามารถค้นคว้าข้อมูลที่เป็นประโยชน์ในการศึกษาวิจัยต่าง หรืออินเทอร์เน็ตทำให้โลกไร้พรมแดนจากการที่สามารถสื่อสารกันได้อย่างรวดเร็วแม้ว่าจะอยู่ไกลกันก็สามารถสื่อสารกันได้อย่างรวดเร็วโดยการใช้ E-Mail, Chat, VoIP, Video Conference และอื่นๆ สำหรับองค์กรที่เกี่ยวกับการทำธุรกิจก็สามารถที่จะใช้ปรโยชน์จากอินเทอร์เน็ตได้ด้วยเช่นกัน ไม่ว่าจะเป็นการขยายกลุ่มลูกค้าซึ่งสามารถมีลูกค้าได้ทั่วโลก ลดต้นทุนและระยะเวลาในการสื่อสาร ทำให้ธุรกิจสามารถแข่งขันได้ดียิ่งขึ้น

ถึงแม้ว่าอินเทอร์เน็ตมีประโยชน์ต่างๆมากมาย อินเทอร์เน็ตก็ยังมีข้อเสียด้วยเช่นกัน การที่องค์เชื่อมต่อเครือข่ายของตัวเองเข้ากับอินเทอร์เน็ตนั้นก็ย่อมที่จะทำให้ ผู้อื่นสามารถเข้าถึงเครือข่ายขององค์กรได้ด้วยเช่นกัน ปัญหาที่พบอยู่บ่อยเช่น ถูกขโมยข้อมูล, ถูกโจมตีเครือข่ายทำให้ไม่สามารถใช้งานเครือข่ายได้, ติดไวรัสซึ่งบางครั้งร้ายแรงจนทำให้ข้อมูลขององค์กรเสียหายได้ ดังนั้นองค์กรควรระวังและมีแนวทางการป้องกันก่อนที่จะเชื่อมต่อเครือข่ายกับอินเทอร์เน็ต

ในการเชื่อมต่อเครือข่ายกับอินเทอร์เน็ตนั้นสามารถแบ่งออกเป็น 3 ส่วนหลักๆ คือ เครือข่ายขององค์กร (LAN), ลิงค์ที่เชื่อมต่อไปยังอินเทอร์เน็ต (WAN), อินเทอร์เน็ต ซึ่งในโครงการพัฒนาระบบนี้จะสนใจในส่วนของการเชื่อมต่ออินเทอร์เน็ต (WAN) ซึ่งในส่วนนี้นั้นมีส่วนประกอบหลายอย่างที่ทำงานร่วมกัน เช่น Link, Router, Traffic Management, Firewall



รูปที่ 1.2 ลักษณะการเชื่อมต่ออินเทอร์เน็ต

โดยจะสนใจในส่วนของ Traffic Management เป็นหลักเนื่องจากค่าใช้จ่ายในการเชื่อมต่อลิงค์ไปยังอินเทอร์เน็ตนั้นมีราคาต่อแบนด์วิดท์ที่ค่อนข้างสูง การที่จะใช้แบนด์วิดท์ที่มีอยู่อย่างคุ้มค่าที่สุดนั้นถือว่ามีความสำคัญ เมื่อเทียบกับการบริหารแบนด์วิดท์ที่อยู่ให้ได้ประโยชน์สูงสุดกับการที่ต้องเสียค่าใช้จ่ายในการขยายแบนด์วิดท์ เพื่อให้รองรับข้อมูลบางอย่างที่ไม่เป็นประโยชน์ต่อองค์กร เช่น การโหลดบิต (BitTorrent) หรือ ไวรัส ที่ทำให้แบนด์วิดท์เต็ม การที่จะจัดการบริหารข้อมูลในเครือข่ายที่จะออกไปยังอินเทอร์เน็ตนี้เรียกว่า Traffic Management ซึ่งจะใช้เทคนิคของการทำ QoS-(คิวโอเอส) เข้ามาช่วยซึ่งในตลาดนั้นก็มีอุปกรณ์กลุ่มนี้ขายอยู่แล้วเช่น InnGate/AntLabs, AG3100/Nomadix, BBSM/Cisco ซึ่งอุปกรณ์เหล่านี้มีราคาค่อนข้างสูงและเหมาะสำหรับองค์กรขนาดใหญ่เป็นหลัก สำหรับองค์กรขนาดกลางหรือขนาดเล็ก จะไม่คุ้มค่าสำหรับการลงทุน ดังนั้นจึงเป็นที่มาของการพัฒนาอุปกรณ์ในการจัดการและควบคุมข้อมูลในเครือข่ายนี้ขึ้นมา ซึ่งมีความสามารถในการจัดการบริหารแบนด์วิดท์ และมีค่าใช้จ่ายไม่แพง

1.2 ความมุ่งหมายและวัตถุประสงค์ของการศึกษา

วัตถุประสงค์ของการพัฒนาอุปกรณ์ในการจัดการและควบคุมข้อมูลในเครือข่าย มีดังนี้

1. เพื่อพัฒนาระบบบน Embedded System ที่มีลินุกซ์เป็นระบบปฏิบัติการ ให้มีความสามารถในการจัดการและควบคุมข้อมูลในเครือข่าย

2. เพื่อพัฒนาส่วนติดต่อผู้ใช้ของระบบจากเดิมที่เป็น CLI (Command Line Interface) ให้เป็นเว็บอินเตอร์เฟส

3. เพื่อพัฒนาระบบที่เป็นทางเลือกในการจัดการแบนด์วิดท์ สำหรับองค์กรที่มีขนาดกลางหรือเล็ก ที่ต้องการระบบที่งานต่อการใช้งานและดูแลรักษา

4. เพื่อช่วยลดภาระค่าใช้จ่ายในการที่จะต้องเพิ่มแบนด์วิดท์ให้กับองค์กร และทำให้การใช้แบนด์วิดท์ที่มีอยู่ได้ประสิทธิภาพสูงสุด ตามเป้าหมาย

1.3 ขอบเขตของการศึกษา

ในการพัฒนาอุปกรณ์ในการจัดการและควบคุมข้อมูลในเครือข่ายนี้ ได้ศึกษา วิเคราะห์และพัฒนาระบบขึ้นโดยตั้งอยู่บนสมมติฐานขององค์กรที่มีการเชื่อมต่อเครือข่ายขององค์กรไปยังอินเทอร์เน็ต โดยจะใช้อุปกรณ์ในการจัดการและควบคุมข้อมูลในเครือข่าย มาจัดการบริหารแบนด์วิดท์ให้มีประสิทธิภาพ และเป็นประโยชน์ต่อองค์กรสูงสุด ซึ่งจะมีขอบเขตการพัฒนาระบบให้ระบบมีความสามารถครอบคลุมการทำงาน ดังนี้

1. ระบบพัฒนามาจาก Embedded System เดิมที่มีอยู่แล้ว (iBoard) โดยนำมาพัฒนาให้มีความสามารถในการจัดการและควบคุมข้อมูลในเครือข่าย

2. ระบบที่พัฒนาขึ้นสามารถมีส่วนติดต่อผู้ใช้ได้ทั้งที่เป็นแบบ Command Line Interface (CLI) สำหรับผู้ใช้ที่ต้องการการจัดการในเชิงลึก และแบบ Web Base Interface เพื่อง่ายต่อการใช้งาน

3. ระบบที่พัฒนาขึ้นสามารถมีการพิสูจน์ตัวตน (Login) ของผู้ดูแลระบบ เพื่อเข้าจัดการระบบ

4. ระบบที่พัฒนาขึ้นสามารถให้มีการพิสูจน์ตัวตน (Login) ของผู้ใช้ที่ต้องการใช้งานแบนด์วิดท์

5. ระบบที่พัฒนาขึ้นสามารถที่ควบคุมการใช้งานแบนด์วิดท์ตาม ผู้ใช้ (User), กลุ่มผู้ใช้ (Group), เครื่องคอมพิวเตอร์ (Host), กลุ่มเครือข่าย (Network)

6. ระบบที่พัฒนาขึ้นสามารถที่ควบคุมการใช้งานแบนด์วิดท์ตามแอพพลิเคชั่นโพรโทคอล เช่น เว็บ, เอฟทีพี, อีเมล

1.4 ขั้นตอนของการศึกษา

ขั้นตอนในการดำเนินโครงการพัฒนาอุปกรณ์ในการจัดการและควบคุมข้อมูลในเครือข่าย มีขั้นตอน ดังนี้

1. กำหนดหัวข้อในการศึกษา เป้าหมายของการศึกษา ขอบเขต วัตถุประสงค์ และ ผลที่คาดว่าจะได้รับ

2. ศึกษาความสามารถ ข้อจำกัดของอุปกรณ์ (iBoard) และวิธีการปรับปรุงความสามารถของอุปกรณ์เพิ่มเติม ให้สามารถรองรับการทำงานได้ตามที่ต้องการ

3. ศึกษาเทคโนโลโยการจัดการและควบคุมข้อมูลในเครือข่าย โดยมีพื้นฐานอยู่บนระบบปฏิบัติการลินุกซ์

4. พัฒนาเทคโนโลโยการจัดการและควบคุมข้อมูลในเครือข่าย ให้สามารถทำงานได้บนอุปกรณ์

5. ออกแบบระบบงาน ฐานข้อมูล และหน้าจอให้เหมาะสมสำหรับการใช้งานของผู้ใช้ระบบ

6. พัฒนาระบบตามที่ได้ออกแบบไว้ และทดสอบระบบร่วมกับผู้ใช้ รวมถึงนำระบบไปทดลองใช้งานจริง

7. สรุปผลจากข้อคิดเห็นและคำเสนอแนะในการทดสอบและทดลองใช้งานระบบตลอดจนจัดทำเอกสารประกอบการพัฒนาระบบงาน

1.5 ประโยชน์ที่คาดว่าจะได้รับ

ประโยชน์ที่คาดว่าจะได้รับจากการพัฒนาระบบช่วยประมาณภาระงานในการพัฒนาซอฟท์แวร์ มีดังนี้

1. ระบบสามารถทำการควบคุมการใช้งานแบนด์วิดท์ได้อย่างมีประสิทธิภาพ
2. ระบบสามารถใช้งานได้ง่าย ไม่ยุ่งยาก ไม่ต้องการผู้ดูแลที่มีความรู้เป็นพิเศษ
3. ระบบมีการพิสูจน์ตัวตน (Login) ของผู้ดูแลระบบ ก่อนที่จะเข้าจัดการระบบ
4. ระบบให้มีการพิสูจน์ตัวตน (Login) ของผู้ใช้ที่ต้องการใช้งานแบนด์วิดท์
5. ระบบมีส่วนติดต่อผู้ใช้เป็นแบบเว็บแอพพลิเคชันที่สามารถควบคุมการใช้งานแบนด์วิดท์ตาม ผู้ใช้ (User), กลุ่มผู้ใช้ (Group), เครื่องคอมพิวเตอร์ (Host), กลุ่มเครือข่าย (Network)
6. ระบบมีส่วนติดต่อผู้ใช้เป็นแบบเว็บแอพพลิเคชันที่ควบคุมการใช้งานแบนด์วิดท์ตามแอพพลิเคชั่นโพรโทคอล เช่น เว็บ, เอฟทีพี, อีเมล

บทที่ 2
ระบบสมองกลฝังตัว

ในโครงการพัฒนาอุปกรณ์ในการจัดการและควบคุมข้อมูลในเครือข่าย นี้ได้ใช้ทฤษฎีที่เกี่ยวข้องหลายเรื่องด้วยกัน ทั้งที่เป็นเทคโนโลยีของฮาร์ดแวร์, ซอฟต์แวร์, เครื่องมือต่างที่นำมาใช้ในพัฒนา, วิเคราะห์และออกแบบระบบงาน ดังนั้นในบทนี้จึงขอกล่าวถึงรายละเอียดของเทคโนโลยีที่ใช้ในโครงการ ซึ่งมีรายละเอียด ดังนี้

2.1 ระบบสมองกลฝังตัว (Embedded System)

ระบบสมองกลฝังตัว หรือ Embedded Systems คือระบบที่มีการทำงานร่วมกันระหว่างฮาร์ดแวร์และซอฟต์แวร์ต่างๆ ใช้สำหรับการควบคุมการทำงานของอุปกรณ์ โดยมีไมโครโพรเซสเซอร์หรือไมโครคอนโทรลเลอร์เป็นหัวใจหลักในการประมวลผลการทำงานมักพบอยู่ในรูปของส่วนควบคุมการทำงานของอุปกรณ์ต่างๆ ทั่วไป เช่น อุปกรณ์เครื่องใช้ไฟฟ้าประจำบ้าน เครื่องจักรกลต่างๆ เครื่องมือทางการแพทย์ โทรศัพท์มือถือ เครื่องเล่นเกมต่างๆ ได้แก่ เครื่องซักผ้า เตาอบไมโครเวฟ วิทยุ โทรทัศน์ เครื่องควบคุมความเร็วของมอเตอร์ เป็นต้น

ในปัจจุบันเทคโนโลยีระบบสมองกลฝังตัวได้เข้ามามีบทบาทในชีวิตเรามากขึ้น และเทคโนโลยีดังกล่าวสามารถนำไปประยุกต์ใช้ในหลายด้าน อีกทั้งยังมีความต้องการอย่างมากจากภาคอุตสาหกรรมเพื่อนำไปใช้ในการปรับปรุงผลิตภัณฑ์ ซึ่งระบบสมองกลฝังตัวนี้แบ่งได้เป็น 2 ชนิดดังนี้

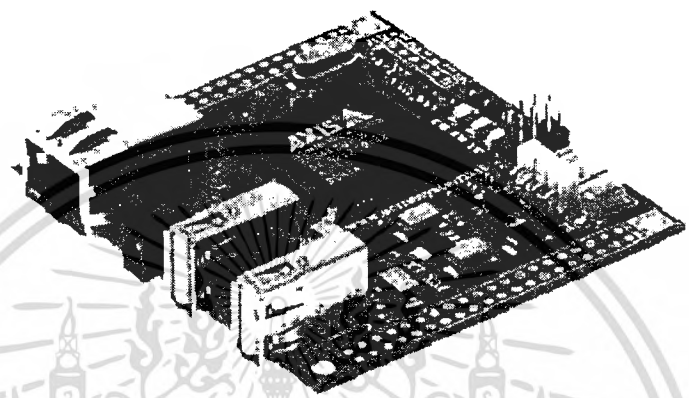
- แบบไมโครโพรเซสเซอร์เดี่ยว ซึ่งเป็นระบบใช้อยู่ในอุปกรณ์ขนาดเล็ก เช่นอุปกรณ์วงจรไฟฟ้า และเครื่องตรวจจับต่าง ๆ
- แบบไมโครโพรเซสเซอร์หลายตัวรวมกัน ซึ่งเป็นระบบที่ใช้อยู่ในอุปกรณ์ควบคุมที่ซับซ้อน เช่น อุปกรณ์ควบคุมกระแสไฟฟ้า, อุปกรณ์ขยายสัญญาณต่าง, เครื่องควบคุมเครื่องจักรในโรงงาน หรืออุปกรณ์เครือข่าย อย่างเราท์เตอร์, ไฟร์วอลล์

2.2 iBoard

iBoard เป็นระบบสมองกลฝังตัวที่ถูกพัฒนาขึ้นโดย บริษัท EMBES Technology (Thailand) Co., Ltd. ซึ่งจะมีคุณสมบัติทางฮาร์ดแวร์ดังนี้

- Microprocessor ETRAX 100LX RISC Architecture 100 MIPS 32-bit
- 8 MB Flash up to 32 MB

- 32 MB SDRAM up to 64 MB
- 1 x Ethernet 10/100 on board RJ45 header
- 2 x USB port (one onboard USB type A and transceiver chip)
- 2 x 20 ping header for 48 DIO, 4 Serial UARTs and/or Parallel port
- Extreme Low Power 300mA @ 5V

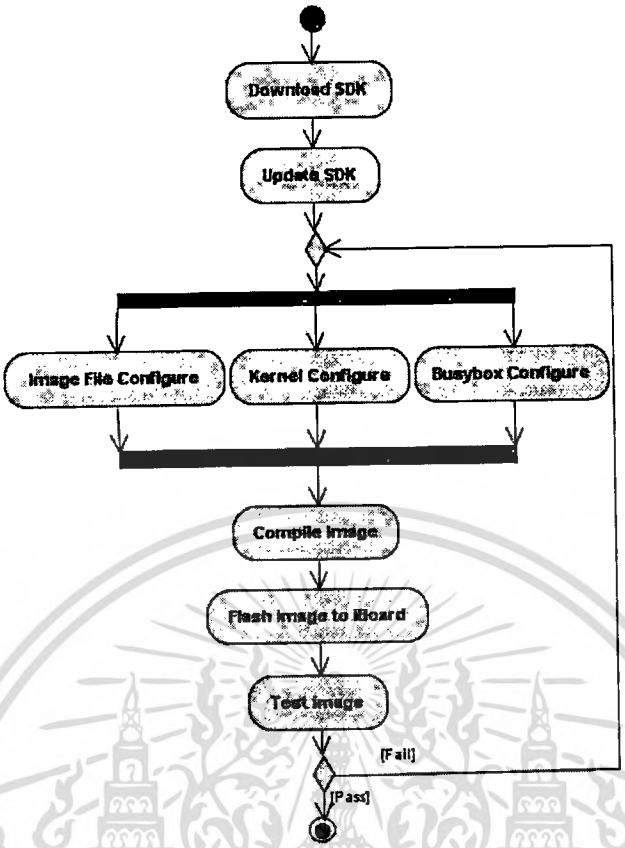


รูปที่ 2.1 iBoard

ในส่วนของซอฟต์แวร์นั้น iBoard ใช้ระบบปฏิบัติการลินุกซ์เป็นระบบปฏิบัตการหลักซึ่งมีรายละเอียดดังนี้

- Linux Operating System Kernel 2.4.x or 2.6.x
- Build in Web server
- Full suit of TCP/IP ipv4, ipv6
  - PPP for asynchronous modem (GPRS/EDGE, Analog,ISDN)
  - PPPoE for ADSL, IPSec, VPN
  - FTP, TFTP, SNMP
- IPTables software packet
- Fast startup < 5 sec boot
- Firmware Upgradeable though FTP, SSH and Telnet with Lan Interface
- Development Kit installed under GNU License

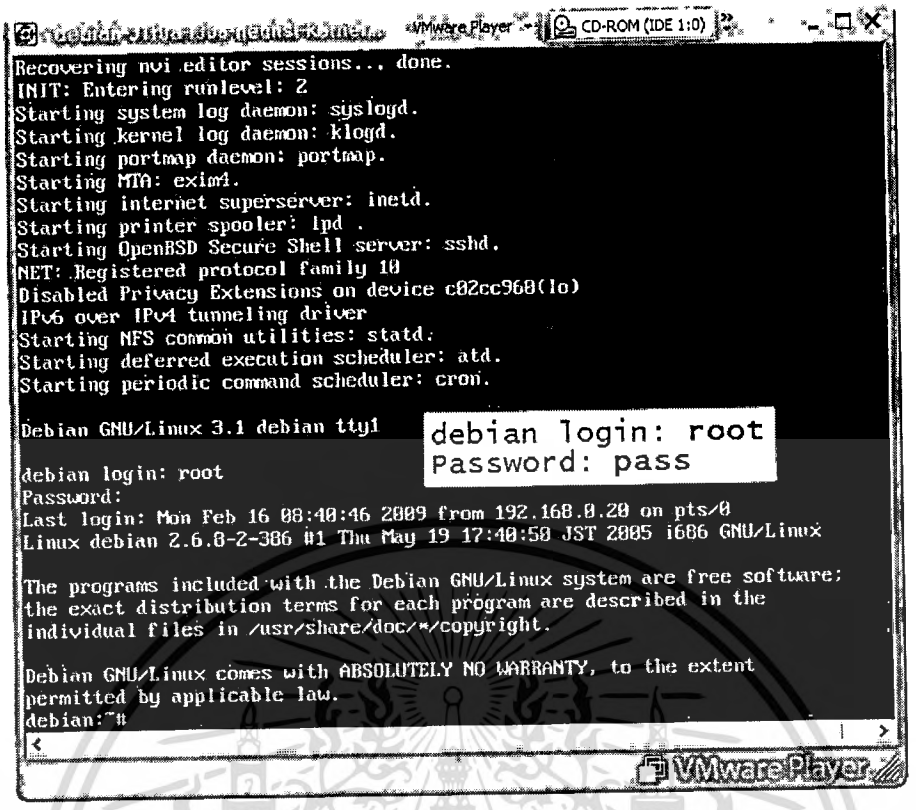
ความสามารถทางด้านซอฟแวร์ของ iBoard นั้นสามารถปรับเปลี่ยนได้ซึ่ง iBoard มี SDK (Software Development Kit) ให้ สามารถที่จะดาวน์โหลดมาติดตั้งและทำการปรับเปลี่ยนค่าได้ ซึ่งขั้นตอนต่างนั้นแสดงดังรูปที่ 2.2 และจะได้กล่าวถึงโดยละเอียดในหัวข้อถัดไป



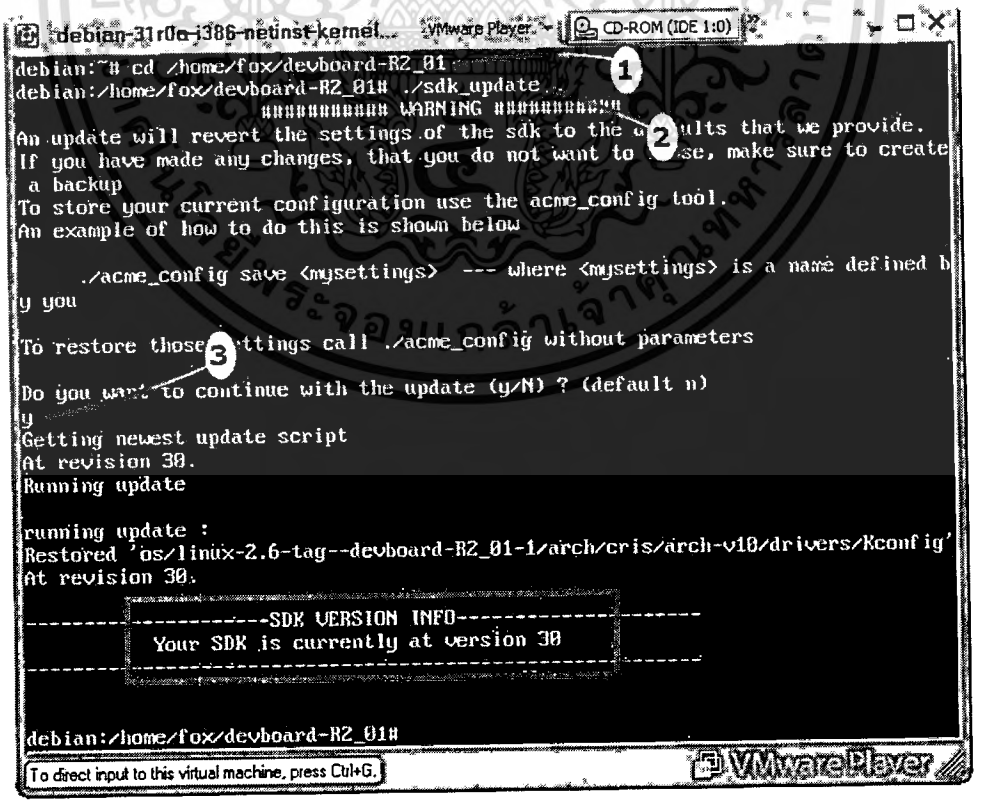
รูปที่ 2.2 ขั้นตอนการเปลี่ยนแปลงซอฟต์แวร์ของ iBoard

2.2.1 การดาวน์โหลดและติดตั้ง SDK ของ iBoard

การจะพัฒนาความสามารถด้านซอฟต์แวร์ของ iBoard นั้นต้องใช้ SDK ของ Phrozen ซึ่งสามารถดาวน์โหลด ได้จาก http://foxlx.acmesystems.it/download/foxsdk.zip ซึ่งเมื่อดาวน์โหลดมาสมบูรณ์แล้วทำการแตกไฟล์ ซึ่งจะได้มาเป็น อิมเมจไฟล์ของ VMware โดยสามารถใช้ VMware Player เปิดใช้งานได้เลย (สามารถดาวน์โหลดได้จาก http://www.vmware.com/download/player/) โดย SDK นี้จะทำงานอยู่บนระบบปฏิบัติการ Dabain Linux ซึ่งต้องทำการล็อกอินดังรูปที่ 2.2 เมื่อสามารถล็อกอินได้แล้ว ขั้นตอนแรกสุดของการใช้งานครั้งแรกคือต้องอับเดท SDK ให้เป็นรุ่นที่ใหม่ที่สุดเสียก่อนซึ่ง SDK นั้นจะมีไดเร็คทอรีหลักที่ใช้ทำงานคือ **/home/fox/devboard-R2_01** ซึ่งขั้นตอนการอับเดท SDK คือเข้าไปยังไดเร็คทอรีหลัก cd /home/fox/devboard-R2_01 แล้วสั่ง ./sdk_update เพื่อสั่งอับเดท SDK รุ่นที่มีการปรับปรุงล่าสุด โดยการทำงานทั้งหมดจะเป็นไปอัตโนมัติเมื่อเสร็จสิ้นจะมีการแจ้งให้ทราบถึง SDK รุ่นที่ได้รับการอับเดทมา ดังรูปที่ 2.3 หลังจากที่ได้ทำการอัพเดจสมบูรณ์แล้วขั้นตอนต่อไปคือการเปลี่ยนแปลงคุณสมบัติต่างๆ ของซอฟต์แวร์ที่จะให้ทำงานบน iBoard ซึ่งจะมีขั้นตอนและรายละเอียดจะกล่าวถึงในหัวข้อถัดไป



รูปที่ 2.3 การล็อกอินเข้าสู่ SDK ของ iBoard



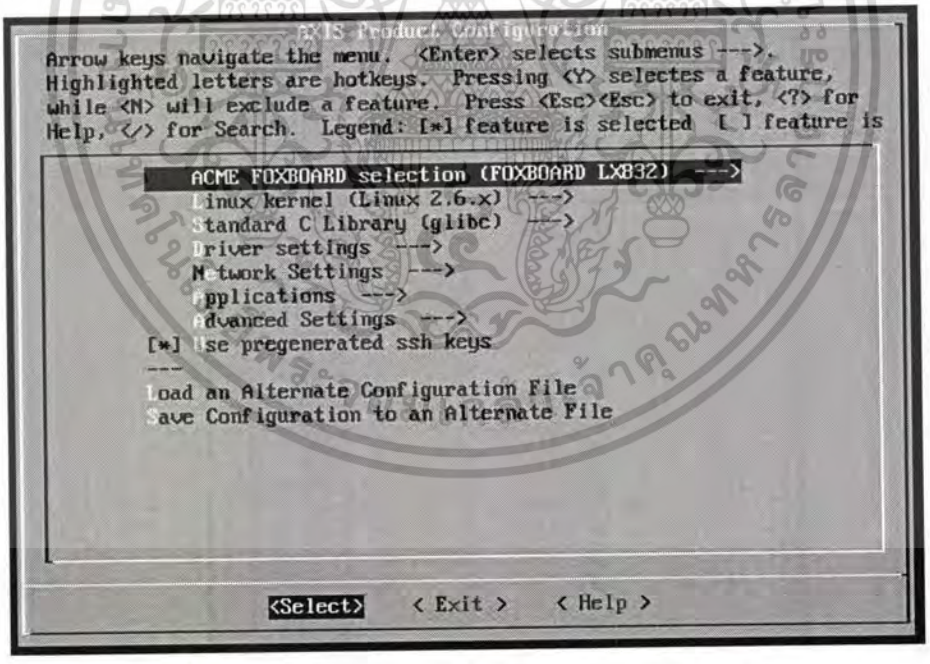
รูปที่ 2.4 การอับเดท SDK ของ iBoard

2.2.2 การปรับเปลี่ยนโปรแกรมประยุกต์ (Image File Configure)

การปรับเปลี่ยนโปรแกรมประยุกต์นั้นจะเป็นสิ่งแรกที่จะทำหลังจากที่ SDK พร้อมใช้งาน เพราะในส่วนนี้นั้นจะเป็นการกำหนดว่าซอฟต์แวร์ (iBoard Image) ที่จะมีการสร้างขึ้นนั้นนำไปใช้กับ iBoard รุ่นที่มีชิปเซตเป็นรุ่นไหน, ใช้ระบบปฏิบัติการลินุกซ์ที่มีเคอร์เนลเป็น 2.4 หรือ 2.6, จะต้องการติดตั้งไดร์เวอร์เพิ่มเติมสำหรับฮาร์ดแวร์อื่นหรือไม่, ต้องการให้มีโปรแกรมประยุกต์อะไรติดตั้งลงไปบ้าง, กำหนดหมายเลขไอพีของการ์ดแลน และที่สำคัญที่สุดคือการแบ่งพื้นที่ของการใช้ Flash Memory ซึ่งอยู่เพียง 8MB ว่าจะให้เป็นส่วน Read Only สำหรับระบบปฏิบัติการ และ Read Write สำหรับการเก็บค่าคอนฟิคต่างๆเท่าไหร่ ซึ่งขั้นตอนต่างๆ ของคำสั่งมีดังนี้

```
# cd /home/fox/devboard-R2_01
# source init_env
# make menuconfig
```

เมื่อทำตามคำสั่งข้างต้นจะมีหน้าจอ *AXIS Product Configuration* ขึ้นมาเพื่อสามารถปรับเปลี่ยนค่าต่างที่ได้กล่าวมาแล้วข้างต้นดังรูปที่ 2.5 ซึ่งรายละเอียดการเลือกค่าต่างๆนั้น เป็นไปตามตารางที่ 2.1



รูปที่ 2.5 เมนูหลักของการแก้ไขค่าของ iBoard Image File

ตารางที่ 2.1 รายละเอียดการเลือกค่าต่างของคอนฟิก Image File

```
ACME FOXBOARD selection (FOXBOARD LX832) -->
       (X) FOXBOARD LX832
Linux kernel (Linux 2.6.x)  --->
       (X) Linux 2.6.x
Standard C Library (glibc)  --->
       (X) glibc
Driver settings  --->
       NOT Select any
Network Settings  --->
       Ethernet Settings  --->
              [X] Use custom IP settings for eth0
                    Proto (Static IP)  --->
              (192.168.0.254) IP
              (192.168.0.255) Broadcast (NEW)
              (192.168.0.1) Gateway (NEW)
              (255.255.255.0) Subnet (NEW)
       Nameserver Settings  --->
              [X] Use custom nameserver
              (203.144.207.29) Nameserver
       MAC -Address  --->
              [ ] Set a MAC
       Firewall Script  --->
              [ ] Enable firewall script
Applications  --->
       Libraries  --->
              [X] Enable libusb
       Applications  --->
              [X] Enable PHP 5.0.5 (might not fit on small boards)
       Networking  --->
              [X] Enable web server
              [X] Enable Point to Point Protocol (PPP) support
              [X] Enable IPtables support
              [X] Enable tcpdump
              --- Enable to include libpcap
              [X] Enable SMTP client support
              [X]   Enable SMTP Authentication
              [X] Enable FTP client support
              --- Enable OpenSSL support
              [X] Enable DHCP support
              [X] Enable HTTPS support
              [X] Enable SSH support
              [X] Enable TELNETD support
       System Tools  --->
              [X] Enable EasyEdit support
              [X] Enable BusyBox support
       Device Nodes  --->
              [X] Create SCSI device nodes
              [X] Create TTYUSB device nodes
Advanced Settings  --->
           Debug Configuration  --->
              [ ] Enable GDB server support
       [X] Partition table
       [X]   USB Device Filesystem
       ()  Kernel configuration file
       [ ] Enable strace
           Manually partition FOX (No)  --->
              (X) No
           Partition the FOX for 8MB Flash to ro - rw (5,0MB-3,0MB) --->
              (X) 0x500000 - 0x2F0000 (5,0MB - 3,0MB)

[]Use pregenerated ssh keys
```

หลังจากที่ กำหนดค่าทุกอย่างเรียบร้อยแล้วเลือก Exit และยืนยันการแก้ไข หลังจากนั้นก็ทำการสั่งคำสั่ง

```
# ./configure
```

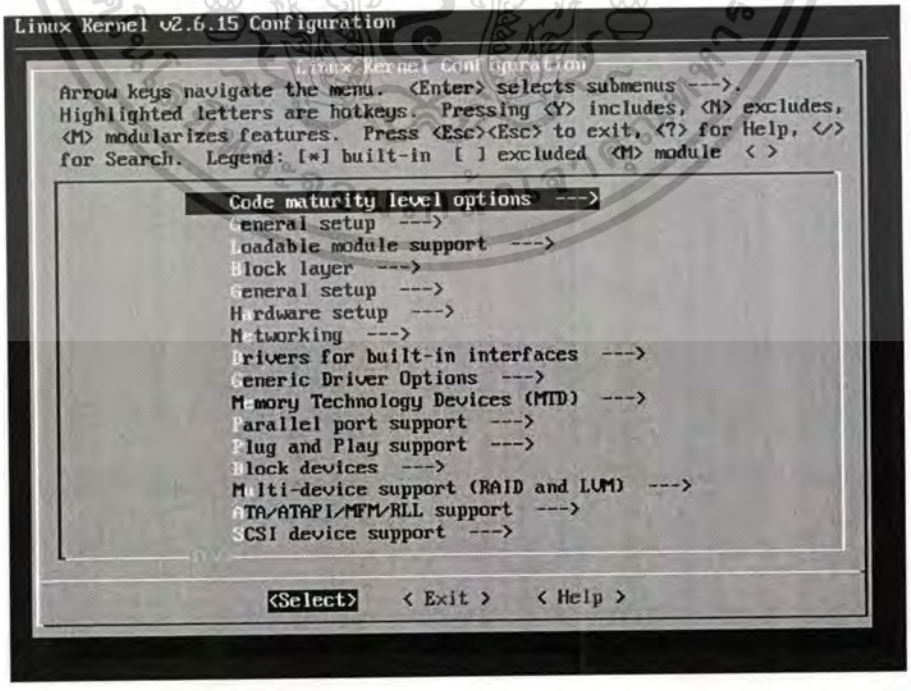
เพื่อทำการอับเดทค่าคอนฟิคไปยัง Makefile ให้พร้อมสำหรับการคอมไพล์ ขั้นตอนต่อไปก็จะเป็นการปรับค่าเคอร์เนลของลินุกซ์เพื่อให้รองรับการทำงาน

2.2.3 การปรับเปลี่ยนเคอร์เนลของลินุกซ์ (Linux Kernel Configure)

เป็นการกำหนดรายละเอียดของ Kernel ที่จะคอมไพล์ซึ่งโดยหลักๆแล้วก็จะใช้ค่ามาตราฐานที่มีมาให้อยู่แล้ว แต่บางส่วนที่ต้องกำหนดเพิ่มเช่น กำหนดรายละเอียดของ iptables, กำหนดให้รองรับการทำ QoS, กำหนดรายละเอียดไดร์เวอร์ของอุปกรณ์ย่อยบนบอร์ด ซึ่งใช้คำสั่ง

```
# make kernelconfig
```

ก็จะมีเมนูแสดงการปรับเปลี่ยนค่าของเคอร์เนลดังรูปที่ 2.6 ซึ่งรายละเอียดการคอนฟิคแสดงในตารางที่ 2.2 โดยเลือกแสดงเฉพาะค่าที่ทำจำเป็นเท่านั้น ส่วนค่าอื่นๆ จะใช้เป็นค่าเดิม และเมื่อทำการปรับแก้ไขค่าต่างๆครบถ้วนแล้วเลือก Exit ออกจากเมนู และยืนยันการแก้ไข



รูปที่ 2.6 เมนูหลักของการคอนฟิคเคอร์เนล

ตารางที่ 2.2 รายละเอียดการเลือกค่าต่างของคอนฟิกเคอร์เนล

```
Code maturity level options  --->
      [*] Prompt for development and/or incomplete code/drivers
      [*]   Select only drivers expected to compile cleanly
General setup  --->
      ()  Local version - append to kernel release
      [*] Automatically append version information to the version string
      [*] System V IPC
      [*] Sysctl support
      [*] Kernel Userspace Events
      [*] Kernel .config support
      [*]   Enable access to .config through /proc/config.gz
      [*] Configure standard kernel features (for small systems) --->
            --- Configure standard kernel features (for small systems)
            [*]   Enable support for printk
            [*]   BUG() support
            [*]   Enable full-sized data structures for core
            [*]   Enable futex support
            [*]   Enable eventpoll support
            [*]   Use full shmem filesystem
            (0)   Function alignment
            (0)   Label alignment
            (0)   Loop alignment
            (0)   Jump alignment
Loadable module support  --->
      [*] Enable loadable module support
      [*]   Module unloading
      [*]   Module versioning support (EXPERIMENTAL)
Block layer  --->
General setup  --->
      [*] Kernel support for ELF binaries
      <*> Kernel support for MISC binaries
      (root=/dev/mtdblock3 init=/linuxrc) Kernel command line
      [*] Enable ETRAX watchdog
      [*]Disable watchdog during Oops printouts
      [*] Enable ETRAX fast timer API
       Preemption Model (Preemptible Kernel (Low-Latency Desktop))  --->
      [*] Preempt The Big Kernel Lock
       Memory model (Flat Memory)  --->
Hardware setup  --->
          Processor type (ETRAX-100LX-v2)  --->
      (32) DRAM size (dec, in MB)
      (2) Buswidth of NOR flash in bytes
      (1) Buswidth of NAND flash in bytes
      (0) FLASH1 size (dec, in MB. 0 = Unknown)
          Product LED port (Port-PA-LEDs)  --->
      (2) First green LED bit
      (2) First red LED bit
      (3) Second green LED bit
      (3) Second red LED bit
      (2) Third green LED bit
      (2) Third red LED bit
          Product debug-port (Serial-0)  --->
          Product rescue-port (Serial-0)  --->
      (0x95f8) R_WAITSTATES
      (0x004) R_BUS_CONFIG
      [*] SDRAM support
      (0x09603737) R_SDRAM_CONFIG
      (0x80008002) R_SDRAM_TIMING
      (0x1c) R_PORT_PA_DIR
      (0xf0) R_PORT_PA_DATA
      (0x00) R_PORT_PB_CONFIG
      (0xce) R_PORT_PB_DIR
      (0x03) R_PORT_PB_DATA
      [ ] Software Shutdown Support
```

ตารางที่ 2.2 (ต่อ)

```
Networking  --->
    [*] Networking support
        Networking options  --->
          <*> Packet socket
          <*> Unix domain sockets
          [*] TCP/IP networking
          <*>   INET: socket monitoring interface
          [ ]   TCP: advanced congestion control
            IP: Virtual Server Configuration  --->
          < >   The IPv6 protocol
          [*] Network packet filtering (replaces ipchains)  --->
              --- Network packet filtering (replaces ipchains)
             [ ]   Network packet filtering debugging
                    Core Netfilter Configuration  --->
                       <*> Netfilter netlink interface
                       <*>   Netfilter NFQUEUE over NFNETLINK interface
                       <*>   Netfilter LOG over NFNETLINK interface
                    IP: Netfilter Configuration  --->
                       <*> Connection tracking (required for masq/NAT)
                       <*>   FTP protocol support
                       <*> IP tables support (required for
                       filtering/masq/NAT)
                       <*>   IP range match support
                       <*>   Packet type match support
                       <*>   Packet filtering
                       <*> REJECT target support
                       <*>   LOG target support
                       <*>   Full NAT
                       <*> MASQUERADE target support
                       <*> REDIRECT target support
                       <*> NETMAP target support
                       <*> SAME target support
                       <*> Basic SNMP-ALG support (EXPERIMENTAL)
          QoS and/or fair queueing  --->
             [*] QoS and/or fair queueing
                    Packet scheduler clock source (Timer interrupt)  --->
                     ---   Queueing/Scheduling
                     <*>   Class Based Queueing (CBQ)
                     <*>   Hierarchical Token Bucket (HTB)
                     <*>   Hierarchical Fair Service Curve (HFSC)
                     <*>   Multi Band Priority Queueing (PRIO)
                     <*>   Random Early Detection (RED)
                     <*>   Stochastic Fairness Queueing (SFQ)
                     <*>   True Link Equalizer (TEQL)
                     <*>   Token Bucket Filter (TBF)
                     <*>   Generic Random Early Detection (GRED)
                     <*>   Differentiated Services marker (DSMARK)
                     <*>   Network emulator (NETEM)
                     <*>   Ingress Qdisc
                     ---   Classification
                     <*>   Elementary classification (BASIC)
                     <*>   Traffic-Control Index (TCINDEX)
                     <*>   Routing decision (ROUTE)
                     <*>   Netfilter mark (FW)
                     <*>   Universal 32bit comparisons w/ hashing (U32)
                     [*] Performance counters support
                     [*] Netfilter marks support
                     <*>   IPv4 Resource Reservation Protocol (RSVP)
                     <*>   IPv6 Resource Reservation Protocol (RSVP6)
                     [*]   Extended Matches
                     (32)Stack size
                     <*> Simple packet data comparison
                     <*> Multi byte comparison
                     <*> U32 key
```

ตารางที่ 2.2 (ต่อ)

```
                    <*> Multi byte comparison
                    <*> U32 key
                    <*> Metadata
                    <*> Textsearch
                    [*]   Actions
                    <*> Traffic Policing
                    <*> Generic actions
                    [*]   Probability support
                    <*> Redirecting and Mirroring
                    <*> IPtables targets
                    <*> Packet Editing
                    <*> Simple Example (Debug)
                    [*]   Incoming device classification
                    ---   Rate estimator
                    Network testing  --->
Drivers for built-in interfaces  --->
      [*] Ethernet support
        Network LED behavior (LED_on_when_activity)  --->
      [*] Serial-port support
      [*] Fast serial port DMA flush
      [*]   Serial port 0 enabled
      [ ]   Serial port 1 enabled
      [*]   Serial port 2 enabled
      (4) Ser2 DTR on PA bit (-1 = not used)
      (5) Ser2 RI  on PA bit (-1 = not used)
      (6) Ser2 DSR on PA bit (-1 = not used)
      (7) Ser2 CD  on PA bit (-1 = not used)
      [*]   Serial port 3 enabled
      [*] USB host
      [*]   USB port 1 enabled
      [*]   USB port 2 enabled
      [*] Axis flash-map support
      [*] I2C support
      [*]   I2C uses PB not PB-I2C
      (0) I2C SDA bit number
      (1) I2C SCL bit number
      [ ]   I2C EEPROM (non-volatile RAM) support
      [*] GPIO support
      (0xFF) PA user changeable dir mask
      (0xFF) PA user changeable bits mask
      (0xFF) PB user changeable dir mask
      (0xFF) PB user changeable bits mask
      [ ] Port G Output
      [*] Real Time Clock support
        RTC chip (DS1302)  --->
      [ ] DS1302 RST on Generic Port (NEW)
      (2) DS1302 RST bit number (NEW)
      (1) DS1302 SCL bit number (NEW)
      (0) DS1302 SDA bit number (NEW)
      (0) DS1302 Trickle charger value (NEW)
Generic Driver Options  --->
      [*] Select only drivers that don't need compile-time external
      firmware
      [*] Prevent firmware from being built
      <*> Hotplug firmware loading support
Memory Technology Devices (MTD)  --->
Parallel port support  --->
Plug and Play support  --->
Block devices  --->
      <*> RAM disk support
      (16)  Default number of RAM disks
      (4096) Default RAM disk size (kbytes)
Multi-device support (RAID and LVM)  --->
ATA/ATAPI/MFM/RLL support  --->
```

ตารางที่ 2.2 (ต่อ)

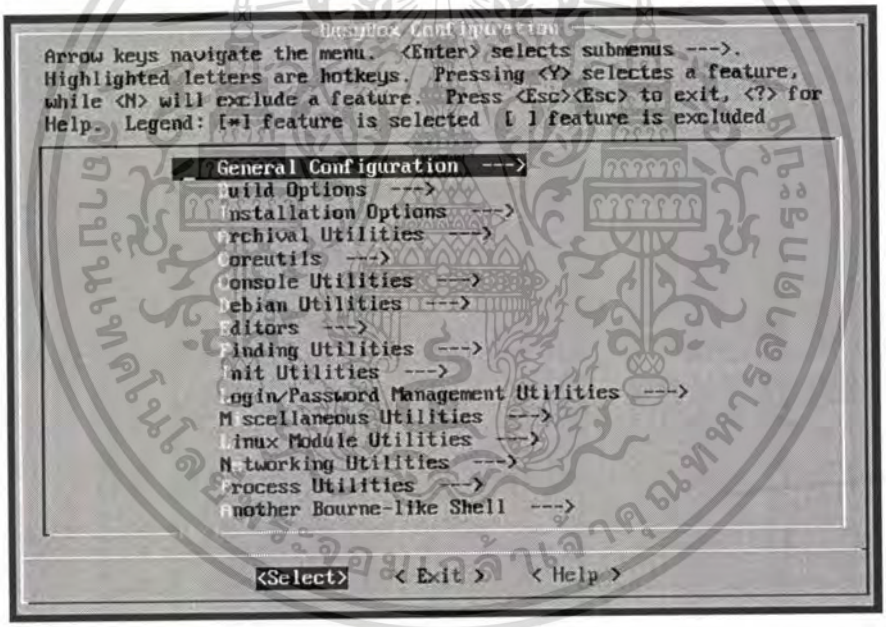
```
SCSI device support  --->
        [*]   legacy /proc/scsi/ support
       ---   SCSI support type (disk, tape, CD-ROM)
       <*>   SCSI disk support
IEEE 1394 (FireWire) support  --->
I2O device support  --->
Network device support  --->
       [*] Network device support
       <*>   PPP (point-to-point protocol) support
       <*>     PPP support for async serial ports
       <*>     PPP Deflate compression
ISDN subsystem  --->
Telephony Support  --->
Input device support  --->
Character devices  --->
       [*] Virtual terminal
       [*]   Support for console on virtual terminal (NEW)
       [*] Unix98 PTY support
       [*] Legacy (BSD) PTY support
Multimedia devices  --->
       [*] Inotify file change notification support
       [*] Dnotify support
       < > Filesystem in Userspace support
           CD-ROM/DVD Filesystems  --->
           DOS/FAT/NT Filesystems  --->
             <*> MSDOS fs support
             <*> VFAT (Windows-95) fs support
             (437) Default codepage for FAT
             (iso8859-1) Default iocharset for FAT
             < > NTFS file system support
           Pseudo filesystems  --->
             [*] /proc file system support
             [*]   /proc/kcore support
             [*] sysfs file system support
             [*] Virtual memory file system support (former shm fs)
             < > Relayfs file system support
           Miscellaneous filesystems  --->
           Network File Systems  --->
           Partition Types  --->
             [*] Advanced partition selection
             [*]   PC BIOS (MSDOS partition tables) support
           Native Language Support  --->
Sound  --->
PCCARD (PCMCIA/CardBus) support  --->
USB support  --->
       <*> USB Mass Storage support
       --- USB port drivers
           USB Serial Converter support  --->
             <*> USB Serial Converter support
             <*>   USB FTDI Single Port Serial Driver (EXPERIMENTAL)
             <*>   USB Prolific 2303 Single Port Serial Driver
Kernel hacking  --->
Security options  --->
Cryptographic options  --->
Library routines  --->
Acmesystems  --->
```

2.2.4 การปรับเปลี่ยนคอนฟิคของ Busybox (Busybox Configure)

Busybox เป็นชุดของโปรแกรมอัตถประโยชน์ที่มารวมกันเป็นไฟล์ ไฟล์เดียวโดยที่ Busybox นั้นได้ถูกพัฒนามาเพื่อรวมเอาคำสั่งต่างที่เป็นประโยชน์ของยูนิกซ์ เช่นโปรแกรมอิดิเตอร์, udhcp server, และโปรแกรมประยุกต์อื่นๆ ไว้เพื่อให้คำสั่งทั้งหมดรวมเป็นไฟล์ที่มีขนาดเล็ก เหมาะสำหรับ Embedded System แต่การที่จะให้ Busybox เป็นส่วนหนึ่งของ image ไฟล์ตอนคอมไพล์ได้นั้น ในหัวข้อ make menuconfig ต้องเลือก “Enable Busybox support” ด้วย ซึ่งการจะเข้าไปปรับแก้ไขจะต้องใช้คำสั่ง

```
# make busybox
```

ซึ่งจะมีเมนูแสดงการปรับเปลี่ยนค่า Busybox ดังรูปที่ 2.7 และเมื่อทำการปรับแก้ไขค่าต่างๆ ครบถ้วนแล้วเลือก Exit ออกจากเมนู และยืนยันการแก้ไข



รูปที่ 2.7 เมนูหลักของการคอนฟิค Busybox

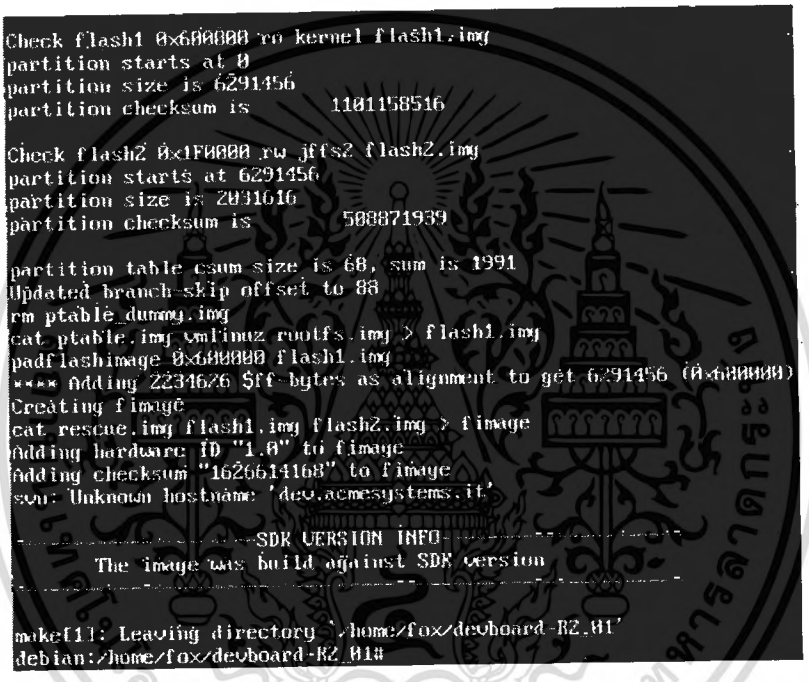
และเมื่อการเตรียมการคอนฟิคทั้ง 3 ขั้นตอนเสร็จสิ้นแล้ว ก็พร้อมที่จะทำการคอมไพล์ image ไฟล์ของ iBoard และทำการ flash image ไฟล์เพื่อให้ iBoard รองรับการทำงานต่างตามที่ได้กำหนดไว้

2.2.5 การสร้างและติดตั้งระบบปฏิบัติการให้ iBoard (Make and Flash iBoard image)

เมื่อกำหนดค่าทุกอย่างของระบบเรียบร้อยแล้ว ต่อไปก็ถึงขั้นตอนที่สำคัญคือการคอมไพล์สร้าง image ไฟล์เพื่อนำไปติดตั้งให้กับ iBoard โดยใช้คำสั่ง

```
# make
```

ขั้นตอนนี้ใช้เวลานานตั้งแต่ 30-50 นาทีขึ้นอยู่กับเครื่องคอมพิวเตอร์ที่ใช้คอมไพล์ และเมื่อเสร็จแล้วจะได้ไฟล์ที่มีชื่อว่า fimage ซึ่งจะเป็น image ที่จะนำไปติดตั้งให้กับ iBoard

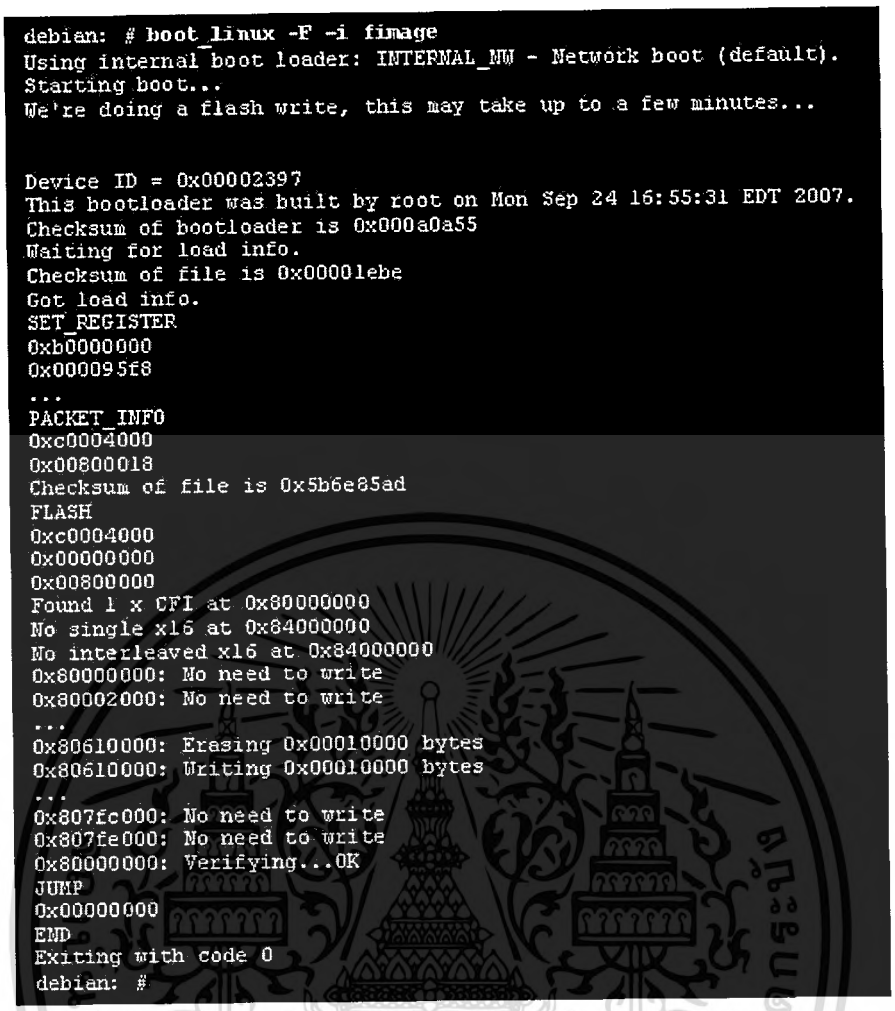


รูปที่ 2.8 แสดงการเสร็จสิ้นการคอมไพล์

ขั้นตอนต่อไปคือการ flash image ไฟล์ที่ได้ให้กับ iBoard การที่จะสามารถ flash image ให้ iBoard ได้นั้นต้องทำการปรับ DIP Switch 2 ของ iBoard ให้อยู่ในตำแหน่ง ON (DIP Switch อื่นๆ อยู่ตำแหน่ง OFF) และเปิด iBoard ซึ่งขณะนี้ iBoard จะเปลี่ยนสถานะจากการเปิดทำงานปกติ มาอยู่ในสถานะการเปิดเพื่อรอการ flash image ขั้นตอนต่อมาที่ SDK ซึ่งตอนนี้อยู่ในไดเร็คทอรีที่เก็บไฟล์ fimage ที่ได้จากการคอมไพล์ ทำการสั่งคำสั่ง

```
# boot_linux -F -i fimage
```

ซึ่งผลการ flash image แสดงดังรูปที่ 2.9



รูปที่ 2.9 แสดงการ flash image ให้ iBoard

2.2.6 การเพิ่มอุปกรณ์ใหม่ (Add new hardware)

เนื่อง iBoard นั้นมีเน็ตเวิร์คอินเตอร์เฟส (LAN Card) มาให้เพียง 1 อินเตอร์เฟสแต่สำหรับการพัฒนาโครงการนี้จำเป็นต้องมีเน็ตเวิร์คอินเตอร์เฟส อย่างน้อย 2 อินเตอร์เฟส ดังนั้นจึงต้องเพิ่มเข้าไปโดยใช้เน็ตเวิร์คอินเตอร์เฟสแบบที่เป็น USB-to-LAN เมื่อมีฮาร์ดแวร์ใหม่เพิ่มเข้ามาจึงจำเป็นอย่างยิ่งที่ต้องทำให้ iBoard รู้จักและสามารถใช้งานได้ ซึ่งขั้นตอนในการปฏิบัติมีอยู่ 3 ขั้นตอนคือ คอมไพล์ไดร์เวอร์บน SDK, เอฟทีพีไดร์เวอร์เข้า iBoard, โหลดไดร์เวอร์เข้าทำงาน ซึ่งขั้นตอนต่างๆ จะแสดงดังนี้

- คอมไพล์ไดร์เวอร์: หาต้นฉบับของไดร์เวอร์นั้น ซึ่งในกรณีนี้ได้ให้มาพร้อมแผ่นซีดี ซึ่งจะมีไฟล์อยู่ 4 ไฟล์ ทำการแก้ไขไฟล์ Makefile โดยแก้ค่าตัวแปร "KERNELDIR=/home/fox/devboard-R2_01 /os/linux-2.6" ทำการบันทึก และสั่ง make เพื่อคอมไพล์ไดร์เวอร์ เมื่อเสร็จสิ้นการคอมไพล์ไดร์จะได้ไฟล์ "dm9601.ko" ซึ่งจะเป็นไดร์เวอร์

```
# ls
dm9601.c  dm9601.h  Makefile  readme.txt

# vi Makefile

Edit "KERNELDIR=/home/fox/devboard-R2_01/
os/linux-2.6"

# make
```

- ทำการเอฟทีพีไฟล์ dm9601.ko เข้า iBoard เก็บไว้ในไดเร็คทอรี "/mnt/flash/lib"
- ทำการโหลดไดร์เวอร์เข้าทำงานโดยการสร้างเป็นShell Script เก็บไว้ที่ "/etc/init.d/boottime" ดังนี้

```
# cd /etc/init.d/boottime

# echo "insmod /mnt/flash/lib/dm9601.ko"
> ins_lan_driver.sh

# chmod +x ins_lan_driver.sh
```

เมื่อทำการเปิดระบบทุกครั้งไดร์เวอร์จะถูกโหลดเข้าทำงานโดยอัตโนมัติ

2.3 การเพิ่มโปรแกรมประยุกต์ (Add other application)

เนื่องจาก SDK ที่ใช้ในการพัฒนาระบบไม่ได้มีเครื่องมือหรือโปรแกรมอัตถประโยชน์ให้เพียงพอหรือตรงกับความต้องการ ดังนั้นก็ต้องหาเข้ามาเพิ่ม และจะต้องมีการคอมไพล์เพื่อเพิ่มเข้าไปในระบบให้สามารถทำงานได้ตามที่ต้องการ ในการทำโครงการนี้จำเป็นอย่างยิ่งที่จะต้องมีโปรแกรมที่ช่วยควบคุมข้อมูลจราจรในเครือข่าย (Traffic Control) โดยชุดโปรแกรมที่ทำหน้าที่นี้มีอยู่ในชุดโปรแกรมอัตถประโยชน์ที่ชื่อว่า iproute2 ซึ่งภายในชุดโปรแกรม iproute2 นี้มีอยู่ 2 โปรแกรมหลักคือ tc และ ip ขั้นตอนต่างๆ ในการคอมไพล์มีดังนี้

- สามารถดาวน์โหลด iproute2 มาจาก http://devresources.linux-foundation.org/dev/iproute2/download/
- ทำการคัดลอกเข้าไปไว้ในไดเร็คทอรี่หลักของ SDK ("/home/fox/devboard-R2_01") และทำการแตกไฟล์

- ทำการแก้ไข Makefile โดยเพิ่ม "KERNEL _INCLUDE=/home/fox/devboard-R2_01/os /linux-2.6/include/linux" เข้าไป และแก้ไขค่าของ CC กับ HOSTCC ให้เป็น "cris-gcc -mlinux"
- ทำการคอมไพล์ด้วยคำสั่ง make แล้วจะได้ไฟล์ tc ซึ่งอยู่ในไดเร็คทอรี tc และไฟล์ ip ซึ่งอยู่ในไดเร็คทอรี ip แล้วทำการเอฟทีพี 2 ไฟล์นี้เข้าไปเก็บไว้ iBoard "/mnt/flash/bin" และกำหนด PATH เพิ่มเติมให้ชี้มายัง "/mnt/flash/bin" ก็จะสามารถใช้ tc, ip ในการจัดการข้อมูลได้

```
# cp iproute2-2.6.15.tar.gz /home/fox/devboard-
R2_10/.

# tar -xzvf iproute2-2.6.15. tar.gz

# cd iproute2-2.6.15
# vi Makefile

{
Add: "KERNEL_INCLUDE=/home/fox /devboard-
R2_01/os/linux-2.6 /include/linux"

Edit: "CC=cris-gcc -mlinux", "HOSTCC=cris-gcc -
mlinux"
}

# make

ftp tc ip => iBoard /mnt/flash/bin

#chmod +x tc ip

#export PATH='/mnt/flash/bin:'$PATH
```

และเมื่อขั้นตอนของการเตรียมความพร้อมของอุปกรณ์เสร็จสิ้นสมบูรณ์แล้วก็พร้อมที่นำไปใช้งานได้แล้ว ซึ่งในบทต่อไปจะกล่าวถึงทฤษฎีหลักการจัดการข้อมูลในระบบเครือข่ายและรายละเอียดของคำสั่งที่จำเป็นสำหรับการทำงาน

บทที่ 3
การจัดการทราฟฟิกของลินุกซ์

ในโครงการพัฒนาอุปกรณ์ในการจัดการและควบคุมข้อมูลในเครือข่ายนี้ได้พัฒนาขึ้นบนระบบปฏิบัติการลินุกซ์ จึงมีความจำเป็นอย่างยิ่งที่จะต้องทราบถึงวิธีการที่ใช้นำมาใช้จัดการและควบคุมทราฟฟิกโดยใช้ความสามารถของระบบปฏิบัติการลินุกซ์ ซึ่งความสามารถที่จะกล่าวถึงในบทนี้นั้นมีอยู่ 2 ส่วนคือ ความสามารถทางด้านไฟร์วอลล์ โดยใช้โปรแกรมประยุกต์ iptables เพื่อควบคุมทราฟฟิกชนิดที่อนุญาตให้ผ่านได้, ไม่ได้ หรือส่งต่อไปยังส่วนอื่น และความสามารถทางด้าน QoS โดยใช้โปรแกรมประยุกต์ tc เพื่อจัดลำดับความสำคัญของข้อมูล, ควบคุมการใช้แบนด์วิดท์ตามข้อกำหนดที่ตั้งไว้ ซึ่งโดยสรุปในบทนี้จะกล่าวถึงวิธีการทำงานของ iptables ตัวอย่างการสั่งงานที่จำเป็น และในส่วนของ QoS นั้นจะกล่าวถึง QoS คืออะไร ตัวอย่างของ QoS อัลกอริทึมที่ใช้ในโครงการพัฒนานี้ และวิธีการ ตัวอย่างคำสั่งที่ใช้สั่งงาน tc ให้ทำงานตามอัลกอริทึมที่กำหนด ซึ่งรายละเอียดดังที่จะได้กล่าวต่อไป

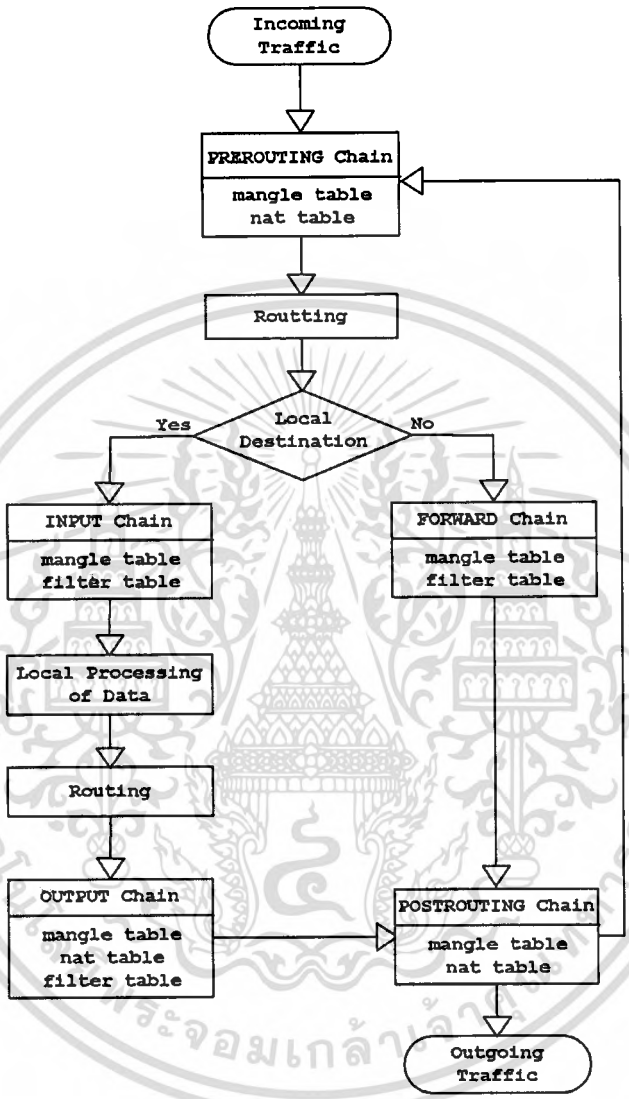
3.1 ลินุกซ์ไฟร์วอล์ (Linux Firewall)

ลินุกซ์สามารถใช้งานเป็นไฟร์วอลล์ได้ตั้งแต่เคอร์เนล 1.1 ซึ่งเป็นเวอร์ชันแรก โดย Alan Cox ใช้ชื่อว่า ipfw ต่อมาลินุกซ์ 2.0 ก็ได้ถูกพัฒนาและปรับปรุงได้เครื่องมือที่มีชื่อว่า ipfwadm โดยเครื่องมือชิ้นนี้อนุญาตให้ผู้ใช้สามารถควบคุมกฎการกรอง (filtering rule) ได้ มีการพัฒนาต่อในลินุกซ์ 2.2 ได้สร้างเครื่องมือตัวใหม่ชื่อ ipchains ซึ่งเผยแพร่ในปี 1998 โดย Rusty Russel และทีมงาน ทั้งนี้ ipchains นี้ถือได้ว่าเป็นพัฒนาการขั้นที่สามของลินุกซ์ไฟร์วอลล์จนกระทั่งในปัจจุบัน ก็มี netfilter และ iptables ซึ่งถือว่าเป็นพัฒนาการขั้นที่สี่ของลินุกซ์ไฟร์วอลล์ netfilter นั้นเป็นชื่อใหม่ของโค้ดที่ทำหน้าที่เป็น packet handler(stateful inspection) ในลินุกซ์เคอร์เนล 2.4 ซึ่งได้ถูกออกแบบและปรับปรุงใหม่จากเวอร์ชันก่อนหน้านี้ netfilter นั้นสามารถทำงานย้อนหลังร่วมกับ ipchains และ ipfwadm ได้ ก่อนที่จะให้ลินุกซ์ทำหน้าที่เป็นไฟร์วอลล์ได้นั้นเคอร์เนลต้องรองรับการทำงานของ iptables ซึ่งถ้าเคอร์เนลเป็นรุ่น 2.4 ขึ้นไปจะรองรับการทำงานของ iptables แล้ว ถ้าต่ำการนั้นต้องทำการคอมไพล์เคอร์เนลใหม่ เพื่อให้ใช้งาน iptables ได้

3.1.1 กระบวนการทำงานของ iptables

ก่อนที่จะเริ่มสั่งงาน iptables ได้นั้นจะอธิบายถึงลำดับการไหลของข้อมูลที่จะต้องผ่านกระบวนการต่างๆ ของ iptables จากรูปที่ 3.1 ทำให้ได้เห็นถึงลักษณะการไหลของข้อมูลใน

iptables ซึ่งเมื่อข้อมูลเข้ามาจะเข้าสู่ PREROUTING Chain ซึ่งจะเกี่ยวข้องกับ 2 ตารางคือ Mangle และ NAT (โดย Chain และ Table จะอธิบายรายละเอียดต่อไป) ผ่านไปสู่ Routing Process ซึ่งจะหาเส้นทางในการส่งข้อมูล จากนั้นตรวจสอบปลายทางข้อมูลว่าเป็นตัวระบบเองหรือไม่



รูปที่ 3.1 ลำดับการส่งผ่านข้อมูลของ iptables

- ใช่ ส่งผ่านให้ INPUT Chain เพื่อตรวจสอบการเข้าถึงระบบ ว่าอนุญาตข้อมูลอะไรบ้าง สามารถเข้าระบบได้ เมื่อผ่านก็จะเข้าสู่ Local Processing of Data คือประมวลผลข้อมูลที่เข้ามา และเมื่อระบบต้องการส่งข้อมูลออกก็จะทำการหาเส้นทางการส่งข้อมูลออกจาก Routing Process แล้วส่งออกไปยัง OUTPUT Chain เพื่อตรวจสอบว่าอนุญาตให้ออกไปหรือไม่

- ไม่ใช่ ส่งผ่านออกไปสู่ FORWARD Chain เพื่อจะตรวจสอบว่าข้อมูลนั้นสามารถให้ผ่านต่อออกไปได้หรือไม่

เมื่อข้อมูลได้รับอนุญาตส่งออกแล้ว ก็เข้าสู่ POSTROUTING Chain เพื่อปรับปรุงหรือแก้ไขข้อมูลก่อนส่งข้อมูลออกไป จากลำดับการไหลของข้อมูลที่ผ่านมาจะเห็นได้ว่าข้อมูลที่ผ่านแต่ละ Chain จะเกี่ยวข้องกับ Table ต่างๆ ซึ่งสามารถแสดงความสัมพันธ์ระหว่าง Chain กับ Table ได้ดังรูปที่ 3.2

RELATION with		Chain				
		[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
Table	MANGLE	X	X	X	X	X
	NAT	X		X		X
	FILTER		X	X	X	

รูปที่ 3.2 ความสัมพันธ์ระหว่าง Chain กับ Table

3.1.2 รูปแบบคำสั่ง iptables

ก่อนหน้าได้ทราบถึงภาพรวมของข้อมูลที่ผ่านกระบวนการต่างของ iptables มาแล้วในส่วนนี้จะมาทราบถึงรายละเอียดของแต่ละกระบวนการว่าทำงานอย่างไร และมีกระบวนการย่อยอะไรบ้าง ซึ่งรูปแบบการสั่งงานของ iptables นั้นจะมีรูปแบบการสั่งงานดังนี้

```
iptables [Table] <Command> <Match> <Target/Jump>
```

- **Table :** คือตารางที่ต้องการให้ iptables ทำงานด้วยในกรณีที่ไม่ได้ระบุนั้นจะถือว่าทำงานอยู่กับ Filter Table
- **Command :** คือคำสั่งที่ต้องการสั่งให้ iptables ทำอะไร
- **Match :** คือการตรวจสอบข้อมูลที่ผ่านเข้ามาใน iptables ว่าตรงตามที่ระบุหรือไม่เพื่อจะได้กระทำกับข้อมูลนั้นต่อไป
- **Target/Jump :** คือการกำหนดการกระทำกับข้อมูลเมื่อข้อมูลนั้น Match เช่นส่งผ่านข้อมูลต่อไป หรือทิ้งข้อมูลนั้น

3.1.2.1 Table

ใน iptables นั้นมีอยู่ด้วยกัน 3 Table หลักคือ Filter Table, NAT Table และ Mangle Table ซึ่งต้องเรียกใช้โดยออปชัน –t และตามด้วยชื่อ Table หากไม่ระบุออปชัน –t ก็จะถือว่าทำงานกับ Filter Table รายละเอียดของแต่ละ Table มีดังนี้

- **Filter Table :** ทำหน้าที่ในการกลั่นกรองข้อมูลที่เข้ามาหรือออกจากโดยที่จะมี chain หลักอยู่ 3 chain คือ FORWORD จะกรองข้อมูลที่ผ่านเข้ามาหาไฟร์วอลล์เพื่อที่ผ่านไปยังปลายทางอื่น, INPUT จะกรองข้อมูลที่เขามาหาไฟร์วอลล์ และ OUTPUT จะกรองข้อมูลที่ออกจากไฟร์วอลล์
- **NAT Table :** ทำหน้าที่ในการเปลี่ยน ไอพีต้นทาง หรือไอพีปลายทางของข้อมูล ซึ่งจะมีอยู่ 2 chain หลักด้วยกันคือ PREROUTING Chain และPOSTROUTING Chain
- **Mangle Table :** ทำหน้าที่ในการแก้ไขค่าต่างของ TCP header เพื่อจัดการในส่วนของ QoS โดยจะมี chain ที่เกี่ยวข้องอยู่ดังนี้ PREROUTING, POSTROUTING, OUTPUT, INPUT และ FORWORD

3.1.2.2 Command

ใน iptables นั้น Command อยู่หลาย Command ซึ่งจะมีรายละเอียดดังนี้

- **-A** *chain rule* : เป็นการเพิ่มกฎเข้าต่อท้าย Chain
- **-D** *chain rule* หรือ **-D** *chain rulenum* : เป็นการลบกฎออกจาก Chain ซึ่งทำได้ 2 วิธี แบบแรกทำการลบโดยพิมพ์กฎที่ต้องการลบ แบบที่สองคือการลบโดยระบุเลขบรรทัดของกฎ
- **-I** *chain [rulenum] rule* : เป็นการเพิ่มกฎโดยการแทรกเข้าไปใน Chain ตามบรรทัดที่ระบุ ถ้าระบุบรรทัดเป็น 1 จะเป็นการแทรกเข้าต้น Chain
- **-R** *chain rulenum rule* : เป็นการแทนกฎเดิมตามบรรทัดที่ระบุด้วยกฎใหม่
- **-L** *[chain]* : เป็นการแสดงกฎทั้งหมดที่มีอยู่ใน Chain ที่ระบุ ถ้าไม่ระบุ Chain จะเป็นการแสดงทุกกฎ ในทุกๆ Chain
- **-F** *[chain]* : เป็นการลบกฎทั้งหมดที่มีอยู่ทั้งหมดใน Chain ที่ระบุ ถ้าไม่ระบุเป็นการลบทุกกฎในทุก Chain ดังนั้นการสั่งออปชันนี่ควรระวังอย่างยิ่ง
- **-Z** *[chain]* : เป็นการปรับค่าตัวนับจำนวนข้อมูลของ Chain ให้เป็นศูนย์
- **-N** *chain* : เป็นการเปิดให้ผู้ใช้สามารถสร้าง Chain ใหม่ได้แต่ชื่อห้ามซ้ำกับ Chain เดิมที่มีอยู่

- **-X** *[chain]* : เป็นการลบ Chain ที่ผู้ใช้สร้างขึ้น แต่ไม่สามารถลบ Chain ที่ระบบสร้างไว้ได้

3.1.2.3 Match

ใช้เพื่อตรวจสอบข้อมูลว่าตรงตามที่ระบุไว้หรือไม่ เสมือนเป็นการตรวจสอบข้อมูลโดยมีออปชันดังนี้

- **p** *[!] protocol* : เป็นการระบุโพรโตคอลโดยสามารถระบุได้ดังนี้คือ TCP, UDP, ICMP ถ้าใช้ ! เป็นการบอกว่าทุกโพรโตคอลยกเว้นที่ระบุ
- **-s** *[!] address[/mask]* : เป็นการระบุหมายเลขต้นทางที่เข้ามาของข้อมูล สามารถกำหนดเป็นชื่อ, ไอพี, ช่วงของไอพี หรือช่วงเน็ตเวิร์คโดยใช้ /mask ระบุ
- **-i** *[!] name* : เป็นการระบุอินเทอร์เฟซขาเข้าของข้อมูลว่าเข้ามาทางอินเทอร์เฟซไหน
- **-o** *[!] name* : เป็นการระบุอินเทอร์เฟซขาออกของข้อมูลว่าจะออกทางอินเทอร์เฟซไหน

3.1.2.4 Target/Jump

เป็นการกระทำกับข้อมูลที่ Match ตามกฎที่ตั้งไว้ว่าจะต้องหารให้ทำอะไรกับข้อมูลนั้น ซึ่งออปชันเหล่านี้จะทำงานตามหลังออปชัน -j

- **ACCEPT** : เป็นการยอมรับข้อมูลนั้น ถือเป็นการจบกระบวนการของ iptables และทำการส่งข้อมูลนั้นให้ระบบปฏิบัติการหรือโปรแกรมอื่นทำงานต่อ
- **DROP** : เป็นการลบทิ้งข้อมูลนั้น และเป็นการจบกระบวนการของ iptables
- **REJECT** : เหมือนกับการ DROP แต่จะทำการส่งข้อความกลับไปบอกผู้ส่งว่าข้อมูลถูกทิ้งไป
- **LOG** : เป็นการบอกให้ส่งข้อมูลของ Packet ไปเก็บยัง Syslog และ iptables จะนำ Packet ตรวจสอบกับกฎอื่นๆต่อไป

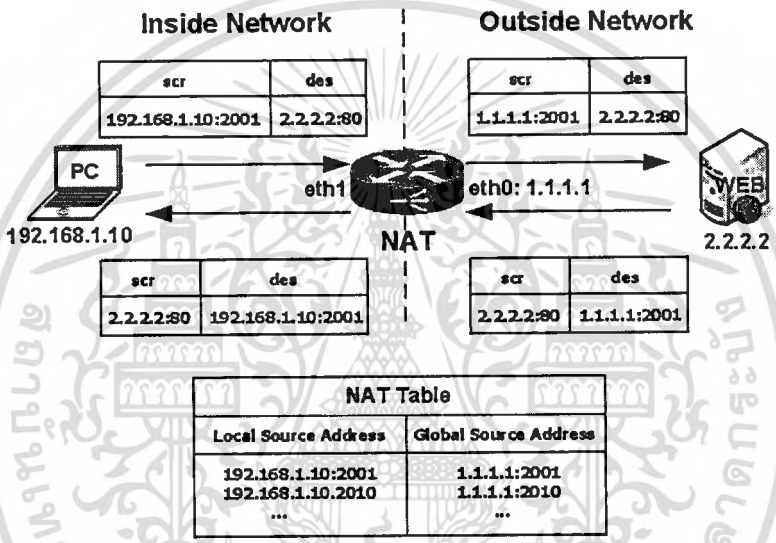
3.1.3 NAT (Network Address Translation)

NAT เป็นวิธีการเปลี่ยนแปลงไอพีต้นทาง (Source IP) หรือ ไอพีปลายทาง (Destination IP) ของข้อมูลที่ส่งไปในเครือข่าย สาเหตุที่ต้องทำเช่นนั้น เนื่องมาจากในปัจจุบันที่ใช้ระบบไอพีรุ่นที่สี่ (IPv4) นั้นจำนวนไอพีมีไม่เพียงต่อต่อการใช้งาน ดังนั้นเพื่อชลอการหมดไปของไอพี ก่อนที่จะได้ใช้ไอพีรุ่นที่หก (IPv6) จึงทำให้มีการใช้ NAT เกิดขึ้นเช่น บางองค์กรมีผู้ที่ต้องการใช้อินเทอร์เน็ต จำนวน 100 คน แต่ได้รับไอพีจริง (Public IP) มาเพียง 1 ไอพีเท่านั้น การที่จะทำให้ 1

ไอพีรองรับการใช้อินเทอร์เน็ตของคนจำนวน 100 คนได้ต้องใช้ NAT ช่วย รูปแบบของการทำ NAT นั้น มีอยู่ด้วยกัน 2 แบบ คือ NAT ขาออก (Source NAT: SNAT) และ NAT ขาเข้า (Destination NAT: DNAT) หรือเรียกอีกอย่างว่า PAT (Port Address Translation)

3.1.3.1 NAT ขาออก (Source NAT: SNAT)

ลักษณะการทำ NAT ขาออกนั้นจะทำในกรณีที่เครื่องลูกข่ายที่อยู่ในเครื่องข่ายที่เป็นเครือข่ายภายในที่ใช้ไอพีปลอม (Private IP) ออกไปยังเครือข่าย อินเทอร์เน็ตได้ โดยการเปลี่ยนไอพีขาออกของข้อมูลให้เป็นไอพีจริง ดังรูปที่ 3.3



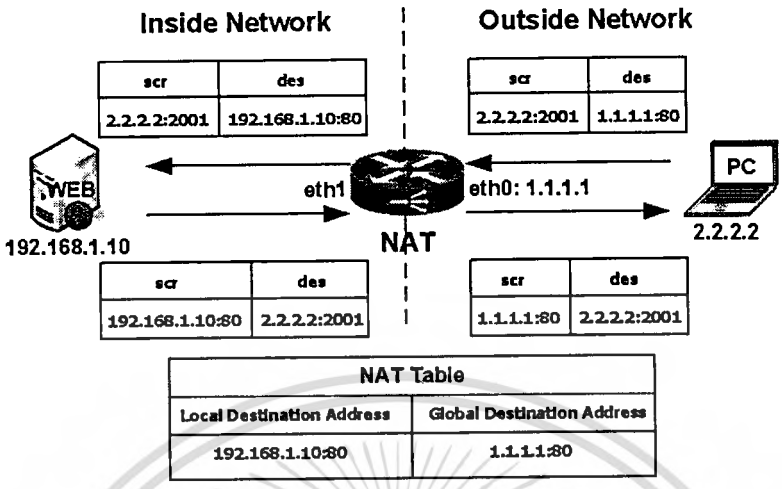
รูปที่ 3.3 NAT ขาออก

จะเห็นได้ว่าเมื่อเครื่องลูกข่ายซึ่งมีไอพีเป็น 192.168.1.10 ต้องการร้องขอการเปิดเว็บจากที่มีไอพีเป็น 2.2.2.2 ข้อมูลที่ส่งผ่าน NAT ออกไปจะถูกเปลี่ยนไอพีขาออกจาก 192.168.1.10:2001 (:2001 คือหมายเลขพอร์ท) เป็น 1.1.1.1:2001 เพื่อให้ข้อมูลถูกส่งผ่านไปในเครือข่ายอินเทอร์เน็ตได้ และอุปกรณ์ที่ทำ NAT จะเก็บค่านี้ใช้ไว้ใน NAT Table เมื่อข้อมูลส่งกลับมาก็จะทำการแปลงไอพีกลับคืนเพื่อส่งกลังไปยังเครื่องลูกข่ายภายในต่อไป

3.1.3.2 NAT ขาเข้า (Destination NAT: DNAT)

ลักษณะการทำ NAT ขาเข้านั้นมักจะใช้ในกรณีที่ได้มีการวางเครื่องแม่ข่าย เช่นเครื่องบริการเว็บ ไว้ในเครือข่ายภายในและต้องการให้ผู้ใช้บริการจากภายนอกเข้ามาใช้งานได้ แต่ไม่ต้องการให้ผู้ใช้ทราบไอพีจริงๆ ของเครื่องให้บริการแต่จะรู้จักเพียงอุปกรณ์ NAT เท่านั้น โดยที่เอ็น

เอที่สามารถจำกัดการเข้าถึงเครื่องให้บริการเพียงบางพอร์ท หรือบางโพรโทคอลที่ได้ทำการอนุญาติไว้ก่อนเท่านั้นดังรูปที่ 3.4



รูปที่ 3.4 NAT ขาเข้า

จะเห็นได้ว่า NAT มีการกำหนดไว้ก่อนในตาราง NAT ว่าถ้ามีการเรียกมาที่ไอพี 1.1.1.1 พอร์ท 80 ให้ทำการแปลงไอพีปลายทางเป็น 192.168.1.10:80 เพื่อส่งต่อให้เครื่องบริการเว็บต่อ เมื่อมีเครื่อง 2.2.2.2 ร้องขอการเปิดเว็บมาที่ 1.1.1.1:80 NAT ก็จะทำการแปลงไอพีปลายทางตามที่กำหนดไว้แล้วส่งต่อไปให้เครื่องบริการเว็บ และเมื่อการเปิดเว็บตอบกลับมา NAT ก็จะแปลงกลับไปให้เครื่อง 2.2.2.2 ต่อไป

3.1.3.3 NAT บนลินุกซ์

การทำ NAT บนลินุกซ์นั้นใช้ iptables ช่วยในการทำ ซึ่งก่อนหน้านี้ได้กล่าวถึงการนำ iptables ทำเป็นไฟร์วอลล์ไปแล้ว ต่อจากนี้จะกล่าวถึงการนำ iptables มาทำเป็น NAT บ้าง ในการทำ NAT โดยใช้ iptables มาช่วยนั้นจำเป็นจะต้องเกี่ยวข้องอยู่กัน 2 Chain นั้นคือ

- PREROUTING : เป็นกระบวนการเปลี่ยนไอพีก่อนที่จะเข้าสู่การกระบวนการเราท์ติง ซึ่งจะทำการเปลี่ยนไอพีของปลายทางที่จะไปให้เหมาะสมกับตารางเราท์ติงที่มี โดยกระบวนการนี้เรียกว่า Destination NAT หรือ DNAT
- POSTROUTING : เป็นกระบวนการเปลี่ยนไอพีหลังจากที่ได้ผ่านกระบวนการเราท์ติงมาแล้ว ซึ่งจะทำการเปลี่ยนไอพีต้นปลายทาง โดยกระบวนการนี้เรียกว่า Source NAT หรือ SNAT

3.1.4 ตัวอย่างการใช้คำสั่ง iptables

จากหัวข้อที่ผ่านมาได้ทราบถึงหลักการทำงานของ iptables และรายละเอียดของคำสั่งมาแล้วในหัวข้อนี้จะเป็นการแสดงวิธีการสั่งงาน iptables ในรูปแบบต่าง เช่นทำไฟร์วอลล์, การเปลี่ยนเส้นทางของทราฟฟิก และการทำ NAT เพื่อให้เข้าใจมากยิ่งขึ้น

ตัวอย่างที่ 1

```
# iptables -A INPUT -i eth0 -p icmp -s 192.168.1.0/24 -j ACCEPT
# iptables -A INPUT -p icmp -j DROP
# iptables -A FORWARD -p icmp -j DROP
```

อธิบาย

- *iptables -A INPUT* เป็นการเพิ่มกฎเข้าใน table filter / INPUT chain โดยเพิ่มเข้าต่อทายกฎเดิมที่มีอยู่, *-i eth0* เข้ามาทางอินเทอร์เฟซ eth0, *-p icmp* โพรโทคอล icmp, *-s 192.168.1.0/24* มาจากไอพี 192.168.1.0/24, *-j ACCEPT* อนุญาติ โดยสรุปคืออนุญาติให้เครื่องลูกข่ายที่มีไอพีอยู่ในเน็ตเวิร์ค 192.168.1.0/24 ที่เรียกใช้งานโพรโทคอล icmp เข้ามาหาไฟร์วอลล์ผ่านทางอินเทอร์เฟซ eth0 ผ่านเข้ามาได้
- *-j DROP* ทำการ Drop ทราฟฟิกทิ้ง สรุปคือไม่อนุญาติให้ทราฟฟิกที่เข้ามาหาไฟร์วอลล์ด้วยโพรโทคอล icmp ผ่านได้
- *iptables -A FORWARD* เป็นการเพิ่มกฎเข้าใน table filter / FORWARD chain โดยเพิ่มเข้าต่อทายกฎเดิมที่มีอยู่ สรุปคือไม่อนุญาติให้ทราฟฟิกที่จะผ่านไฟร์วอลล์ด้วยโพรโทคอล icmp ผ่านได้

ตัวอย่างที่ 2

```
# iptables -A FORWARD -p udp -i eth0 -d 0/0 --dport 53 -j ACCEPT
# iptables -A FORWARD -p tcp -i eth0 -d 0/0 --dport 80 -j ACCEPT
```

อธิบาย

- *-p udp* โพรโทคอล udp, *-d 0/0 --dpors 53* ไปหาปลายทางที่ไหนก็ได้ ที่มี port ปลายทางเท่ากับ 53 (DNS) โดยสรุปคือ อนุญาติให้ทราฟฟิกที่เข้ามาทางอินเทอร์เฟซ eth0 โพรโตคอล udp ซึ่งปลายทางไอพีอะไรก็ได้แต่ port ปลายทางต้องเป็น 53 ผ่านได้

- *-p tcp* โพรโทคอล tcp, *-d 0/0 --dport 80* ไปหาปลายทางที่ไหนก็ได้ ที่มี port ปลายทางเท่ากับ 80 (Web) โดยสรุปคือ อนุญาติให้ทราฟฟิกที่เข้ามาทางอินเทอร์เฟซ eth0 โพรโตคอล tcp ซึ่งปลายทางไอพีอะไรก็ได้แต่ port ปลายทางต้องเป็น 80 ผ่านได้

ตัวอย่างที่ 3

```
# iptables -t nat -A PREROUTING -i eth0 -p tcp --dport 80 -j REDIRECT
  --to-port 8080
```

อธิบาย

- *iptables -t nat -A PREROUTING* เป็นการเพิ่มกฎเข้าใน table nat / PREROUTING chain โดยเพิ่มเข้าต่อทายกฎเดิมที่มีอยู่, *-i eth0* เข้ามาทางอินเทอร์เฟซ eth0, *-p tcp* โพรโทคอล tcp, *--dports 80* port ปลายทางเท่ากับ 80, *-j REDIRECT --to-port 8080* เปลี่ยน port ปลายทางเป็น 8080 สรุปคือให้ทำการเปลี่ยนข้อมูลของทราฟฟิกที่เข้ามาทางอินเทอร์เฟซ eth0 จาก port ปลายทางเดิม 80 เป็น 8080 และส่งให้ไฟร์วอลล์

ตัวอย่างที่ 4

```
# iptables -t nat -A POSTROUTING -o eth1 -j MASQUERADE
```

อธิบาย

- *iptables -t nat -A POSTROUTING* เป็นการเพิ่มกฎเข้าใน table nat / POSTROUTING chain โดยเพิ่มเข้าต่อทายกฎเดิมที่มีอยู่, *-o eth1* ออกทางอินเทอร์เฟซ eth1, *-j MASQUERADE* ทำการ NAT ขาออก สรุปคือให้ทำการเปลี่ยนข้อมูลของทราฟฟิกที่จะออกทางอินเทอร์เฟซ eth1 จาก source ip เดิมให้เอา ip จากอินเทอร์เฟซ eth1 ไปแทนแล้วส่งต่อออกไป

ตัวอย่างที่ 5

```
# iptables -t nat -A PREROUTING -p tcp --dport 80 -i eth1 -j DNAT --to
  192.168.1.10 :8000
```

อธิบาย

- *iptables -t nat -A PREROUTING* เป็นการเพิ่มกฎเข้าใน table nat / PREROUTING chain โดยเพิ่มเข้าต่อทายกฎเดิมที่มีอยู่, *-p tcp --dport 80* โพรโตคอล tcp ที่มี port ปลายทางเท่ากับ 80, *-i eth1* เข้ามาทางอินเทอร์เฟซ eth1, *-j DNAT --to 192.168.1.10:80* ทำการ NAT ขาเข้าโดยเปลี่ยน destination ip จากเดิมเป็น 192.168.1.10 และ port ปลายทางเป็น 8000 สรุปคือให้ทำการเปลี่ยนข้อมูลของทราฟฟิกที่เข้ามาทางอินเทอร์เฟซ eth1 โดยมี port ปลายทางเป็น 80 และโพรโทคอล เป็น tcp จาก destination ip และ destination port เดิมให้เป็น ip 192.168.1.10 port 8000 แล้วส่งต่อไป

ตัวอย่างที่ 6

```
# iptables -L
# iptables -L -t nat
# iptables -L INPUT
# iptables -F INPUT
# iptables -Z
# iptables -L OUTPUT -v -x
```

อธิบาย

- แสดงกฎทั้งหมดในที่มีอยู่ใน filter table
- แสดงกฎทั้งหมดในที่มีอยู่ใน nat table
- แสดงกฎทั้งหมดในที่มีอยู่ใน filter table / INPUT chain
- ลบกฎทั้งหมดที่อยู่ใน filter table / INPUT chain
- ทำการ reset ค่าการนับทั้งหมดในทุก chain ของ filter table ให้เป็น 0
- ให้แสดงผลของจำนวน byte และ จำนวน packet ที่ตรงกับกฎนั้นออกมาโดยที่ไม่ต้องมีหน่วยเป็น K, M, G

3.2 QoS (Quality of Service)

คุณภาพการให้บริการการส่งข้อมูลบนเครือข่าย โดยมีการรับประกันว่าการส่งข้อมูลของบริการต่างจะเป็นไปตามคุณภาพ และเงื่อนไขทางเทคนิคที่บริการนั้นต้องการ ตัวอย่างเช่น การใช้งานแอพพลิเคชันต่างๆ ที่มีความต้องการการบริการจากเครือข่ายในรูปแบบที่แตกต่างกันเช่น VoIP,

Streaming Video ต้องการการส่งที่รวดเร็ว โดยสามารถยอมรับได้ถ้าเกิดการหาสูญหายของแพคเกตบางส่วน แต่ถ้าเป็นที่ต้องการความถูกต้องของข้อมูลเช่น email, ftp, โปรแกรมการเงิน หรืออื่น จะไม่สามารถยอมรับได้ถ้าข้อมูลผิดพลาดแต่จะยอมรับความล่าช้าในการส่งได้ จากความที่ต้องการที่แตกต่างกันนั้นทำให้การใช้งานเครือข่ายได้คุณภาพที่ดีที่สุดจึงต้องหาสิ่งที่วัดได้ในเครือข่ายที่มีผลต่อคุณภาพดังกล่าว ซึ่งสำหรับเครือข่ายการสื่อสารแบบแพคเกตนั้น สิ่งต่างๆ ที่จะกล่าวถึงได้แก่ Throughput, Packet loss , Delay, และ Jitter ซึ่งความต้องการต่างๆในสิ่งเหล่านี้ก็จะแตกต่างกันไปตามลักษณะของแอพพลิเคชันหรือบริการที่ใช้งานนั้น

- **Throughput** หรือแบนด์วิดท์ ของเครือข่ายสื่อสารที่ให้บริการนับว่าเป็นสิ่งสำคัญมากสิ่งหนึ่งของคุณภาพของการให้บริการเครือข่าย หรือใช้ในการเปรียบเทียบเครือข่ายต่างๆ หากว่าแบนด์วิดท์นั้นไม่เพียงพอ อย่างอื่นก็คงไม่ต้องพูดถึง แต่หากว่าเราจะเพิ่มแบนด์วิดท์เข้าไปอย่างเดียวโดยที่ไม่ได้มีการปรับแต่งให้เข้ากับความหลากหลายของการใช้งาน ก็จะสิ้นเปลืองเกินไป ดังนั้นหากจะกล่าวว่าเรื่องของ QoS ก็ถือว่าเป็นเรื่องของการใช้ทรัพยากรในเครือข่ายให้คุ้มค่าได้ด้วยเช่นกัน
- **Packet loss** หรือการเกิดการสูญหายของข้อมูลระหว่างการส่ง ไม่ว่าจะเกิดขึ้นจากความผิดพลาดในส่งของระบบเครือข่าย, ข้อมูลหายจากปัญหาภายในเครือข่ายเช่น การล่ม หรือแม้กระทั่งความหนาแน่นของข้อมูลในเครือข่ายที่สูงจนทำให้เกิดบัฟเฟอร์เต็ม หรือเครือข่ายเต็มเป็นต้น สิ่งเหล่านี้จะเป็นตัวสะท้อนคุณภาพเครือข่ายได้ว่ามีคุณภาพดีเพียงใด หากการส่งข้อมูลใดไปก็ตามปลายทางก็ควรจะได้รับข้อมูลนั้นได้ครบถ้วนถูกต้อง ยิ่งทางด้านการสื่อสารความผิดพลาดหรือการแปลงสารที่เกิดขึ้น เป็นเรื่องที่ไม่ควรจะเกิดขึ้นอย่างยิ่ง
- **Delay** หรือระยะเวลาที่ใช้ในการรับส่งข้อมูลจากต้นทางไปยังปลายทางหรือเรียกอีกอย่างว่า Latency ซึ่งอาจจะเกิดได้ทั้งจากระยะทางที่ข้อมูลใช้ในการเดินทาง, ความเร็ว, แบนด์วิดท์ของเครือข่าย หรือแม้กระทั่งสิ่งต่างๆที่เกิดขึ้น แล้วทำให้เวลาที่ใช้รับส่งมากขึ้น สำหรับการรับส่งข้อมูลบางชนิด เช่น การใช้งาน Voice หรือ VoIP เวลาที่ใช้ในการรับส่งถือเป็นเรื่องสำคัญ เพราะหากมี Delay มากๆ ก็จะรู้สึกรำคาญ บางครั้งอาจจะถึงขั้นสื่อสารกันไม่รู้เรื่อง เพราะการสนทนาที่มีการโต้ตอบนั้นผิดเพี้ยนไปหมด Delay สามารถแยกย่อยออกได้เป็น 4 ชนิดคือ
    1. Queuing Delay คือ delay ที่เกิดจากการรอคิวในการส่งข้อมูล
    2. Processing Delay คือ delay ที่เกิดจากการประมวลผล
    3. Transmission Delay คือจำนวนของเวลาในการส่ง bit ทั้งหมดของ packet ลงไปในสื่อนำสัญญาณ

4. Propagation Delay คือ delay ของสื่อที่ใช้ส่งข้อมูลขึ้นอยู่กับระยะทางและชนิดของสื่อนำสัญญาณ

- **Jitter** จะเกิดขึ้นได้กับเครือข่ายแบบ packet switching หรือ Packet Network เพราะข้อมูลที่ทำการส่งไปจะถูกแบ่งย่อยเป็นแพคเกตแล้วส่งออกไป ซึ่งอาจจะมีการใช้เส้นทางการส่งที่ต่างกันได้ และเนื่องจากเส้นทางที่ต่างกันนั้นจะทำให้เกิดการ Delay ที่ต่างกันการที่มี Delay ของส่วนต่างๆ ของข้อมูลที่ต่างกันนี่เรียกว่า Jitter ซึ่งก็จะส่งผลต่อการรับข้อมูลของผู้รับ เช่น การรอคอยข้อมูลให้ครบเพื่อที่จะประกอบเข้าด้วยกัน หรืออาจจะทำให้ข้อมูลนั้นเสียหายจาก Time out ของข้อมูล หรือ Queue รอรับข้อมูลเต็ม เป็นต้น

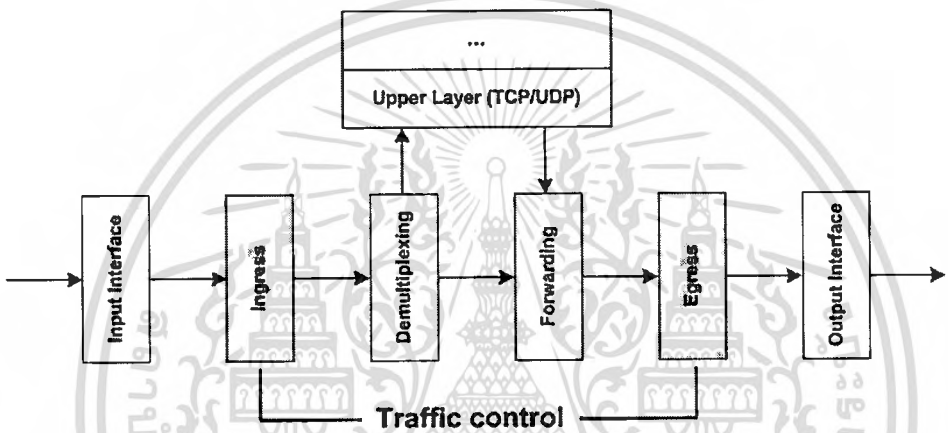
ตารางที่ 3.1 ตัวอย่างแอพพลิเคชั่นและความต้องการจากเครือข่าย

Parameter	**Interactive Gaming**	**Voice**	**Streaming Media**	**Data**	**Video**
Data Rate	50-80 kbps	4-64 kbps	5-384 kbps	0.01-100 Mbps	>1 Mbps
Traffic Flow	Real-time	Real-time continuous	Continuous, bursty	Non-realtime, bursty	Continuous
Packet loss	Zero	<1%	<1% audio, <2% video	Zero	$<10^{-8}$
Delay Variation	N/A	< 20ms	< 2 sec	N/A	< 2 sec
Delay	<50 -150 ms	< 100 ms	< 250 ms	Flexible	< 100 ms

3.3 อัลกอริทึม QoS ในลินุกซ์

การจะควบคุมและจัดการข้อมูลในเครือข่ายได้นั้นจะต้องมีวิธีการที่จะจัดการกับแพคเกตที่มีอยู่ในเครือข่ายซึ่งในลินุกซ์นั้นมีอัลกอริทึมในการควบคุมและจัดการข้อมูลในเครือข่ายซึ่งเรียกว่า Queuing discipline - qdisc โดยที่อัลกอริทึม qdisc นั้นมีอยู่ด้วยกัน 2 รูปแบบหลักคือ Classless qdisc และ Classful qdisc โดยจะมีอัลกอริทึมที่ทำงานแบบ Classless qdisc เช่น fifo, pfifo_fast, TBF, SFQ ส่วนการทำงานแบบ Classful qdisc เช่น PRIO, CBQ, HTB ซึ่งรายละเอียดจะ

กล่าวถึงในหัวข้อต่อไป อัลกอริทึมต่างๆที่กล่าวมาข้างต้นสามารถทำงานได้บนลินุกซ์ โดยอาศัยชุดคำสั่ง Traffic control - tc ซึ่งเป็นชุดคำสั่งที่ใช้ในการบริหารจัดการแบนด์วิดท์ในเครือข่ายซึ่งถูกติดตั้งมากับลีนุกซ์โดย tc เป็นตัวควบคุมกลุ่มของคิว และระบบกลไกการทำงานของการรับส่งแพคเกต โดยรูปแบบของระบบคิวทั่วไปนั้นถูกกำหนดให้เป็นแบบ fifo (First-In First-Out) ซึ่งเมื่อมีแพคเกตเข้ามายังคิวก็จะทำการส่งข้อมูลออกไปอย่างเร็วที่สุดโดยยึดหลัก เข้าก่อน-ออกก่อน แต่สำหรับการจัดการคิวที่ซับซ้อนนั้นจะมีอัลกอริทึมจัดการคิวสำหรับแพคเกตจะใช้เพื่อหาเส้นทาง, วิธีการส่งข้อมูล และยังสามารถกำหนดเงื่อนไขต่างๆ ในการส่งได้อีกด้วย เช่น อนุญาตให้เฉพาะบางโพรโทคอล หรือบางไอพี เท่านั้นที่สามารถส่งผ่านข้อมูลได้ ซึ่งมีรายละเอียดดังนี้

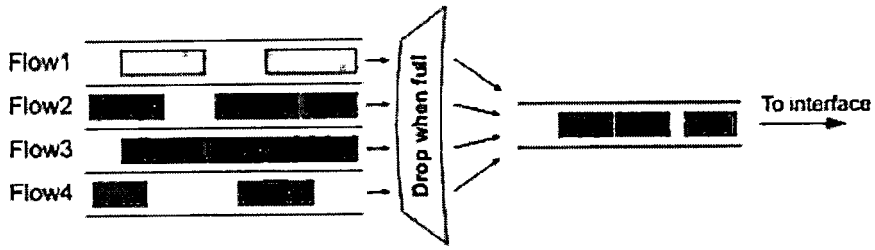


รูปที่ 3.5 การจัดการแพคเกตในลินุกซ์เคอร์เนล

3.3.1 Classless qdisc

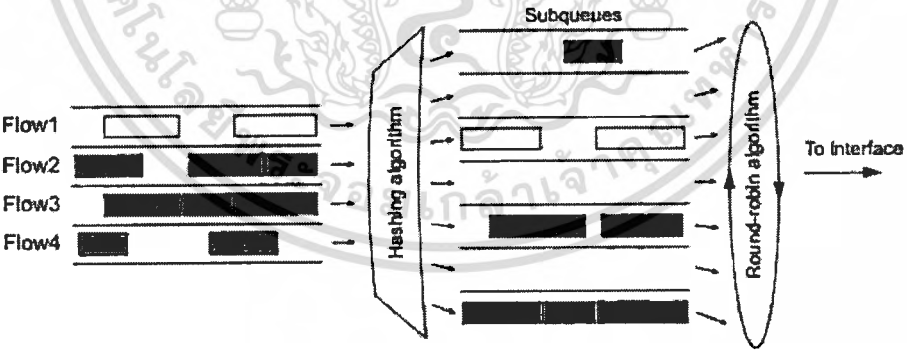
เป็นคลาสอย่างง่ายที่สุดที่ไม่สามารถสร้างคลาสย่อยทำหน้าที่เพียงแค่ accept, drop, delay or reschedule ข้อมูลในเครือข่ายซึ่งในที่นี้หมายถึงว่าเป็น นคลาสเดียว หรือมีเพียงหนึ่งอินเทอร์เฟซ (root qdisc) ที่จะสามารถถูกจำกัดแบนด์วิดท์ ซึ่งมีตัวอย่างเช่น

- **Packet First-In First-Out / Bytes First-In First Out (PFIFO/BFIFO) :** เป็นอัลกอริทึมที่ทำงานอยู่บนพื้นฐานของ FIFO (Fisrt-In First-Out) คือ มาก่อนออกก่อน แต่ต่างกันตรงที่การกำหนดขนาดของคิวที่ให้ข้อมูลสามารถรออยู่ได้ โดย PFIFO จะกำหนดขนาดของคิวเป็นจำนวนแพคเกต ส่วน BFIFO จะกำหนดขนาดของคิวเป็นจำนวนไบต์ของข้อมูล ถ้าหากคิวเต็มอยู่ข้อมูลที่เข้ามาจะถูกโยนทิ้ง (Drop) แต่ถ้าขยายขนาดของคิวให้ใหญ่เกินไปก็จะทำให้เกิดการรอในคิวนานทำให้ค่า Latency สูง การใช้ลักษณะคิวแบบนี้เหมาะสำหรับลิงค์ที่ไม่เต็ม



รูปที่ 3.6 การทำคิวแบบ FIFO

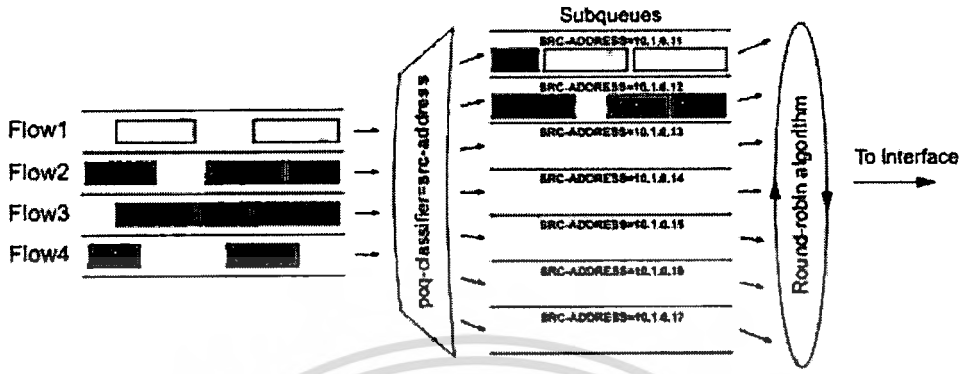
- **Stochastic Fairness Queuing (SFQ) :** จะใช้หลักการพยายามแบ่งทราฟฟิกให้เท่าๆกันทุก session เมื่อช่องทางการส่งข้อมูลเต็ม โดย session ที่ใช้ในการแบ่งทราฟฟิกดแบ่งตาม TCP session หรือ UDP streams ซึ่ง SFQ จะให้หลักการทำ Hashing session ซึ่งจะใช้ค่า ไอพีต้นทาง (source address), ไอพีปลายทาง (destination address), พอร์ทต้ทาง (source port), พอร์ทปลาย (destination port) มาทำการคำนวนเพื่อจะได้ระบุ session ให้แยกกันได้ แล้วทำการส่งข้อมูลแบบ round-robin คือสลับกับส่งไปเป็นรอบเท่าๆกัน SFQ จะสามารถรองรับได้ 128 แพคเกตเป็นเสมือนบัฟเฟอร์ และจะมีคิวย่อย (sub queues) 1024 คิว เพื่อรองรับข้อมูลที่ทำการ hashing แล้ว และจะทำการส่งข้อมูลออกโดยใช้อัลกอริทึมแบบ round-robin คิวแบบ SFQ นั้นเหมาะสำหรับลิงค์ที่คับคั่งเพราะสามารถแน่ใจได้ว่าทุก session สามารถส่งข้อมูลได้เท่าเทียมกัน เหมาะอย่างยิ่งสำหรับเครือข่ายไร้สาย (Wireless)



รูปที่ 3.7 การทำคิวแบบ SFQ

- **Per Connection Queuing (PCQ) :** จะมีหลักการทำงานเหมือนกับ SFQ แต่จะแตกต่างกันที่ PCQ จะใช้การตัวกรอง (filter) เพื่อแยกทราฟฟิกที่จะเข้าไปยังแต่ละคิวย่อย โดยจะไม่ใช้การ Hashing เหมือน SFQ ส่วนการกรองนั้นสามารถระบุได้จาก ไอ

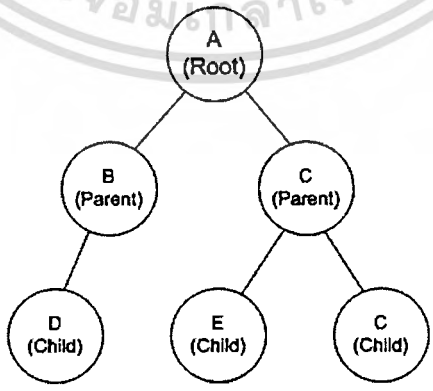
พีต้นทาง (source address), ไอพีปลายทาง (destination address) และในแต่ละคิวย่อย สามารถระบุขนาดเพื่อจำกัดการส่งข้อมูลได้ด้วย



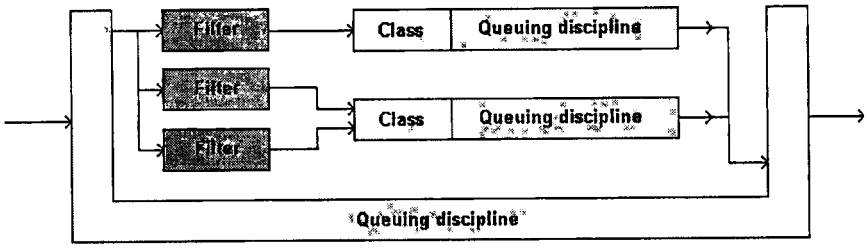
รูปที่ 3.8 การทำคิวแบบ PCQ

3.3.2 Classful qdisc

การทำงานของ Classful qdisc คือการที่ ระบบคิวสามารถที่จะมีคลาสแตกย่อยออกไปได้อีกทั้งยังสามารถกำหนดเรื่อนไขต่างๆได้ละเอียดเพิ่มขึ้นไม่ว่าจะเป็นความเร็วในแต่ละคลาสย่อย และการคัดแยกแพคเกตเข้าไปยังคลาสย่อยเหล่านั้น ซึ่งคลาสย่อย เรียกอีกอย่างว่า คลาสลูก (Child Class) ส่วนคลาสที่มีคลาสลูกแยกออกมาเรียกว่าคลาสแม่ (Parent Class) ซึ่งคลาสแม่อาจจะมีคลาสลูกหรือไม่ก็ได้ขึ้นอยู่กับการออกแบบ และคลาสแม่ที่อยู่ด้านบนสุดเรียกว่ารูทคลาส (Root Class) โดยรูปแบบความสัมพันธ์ดังรูปที่ 3.9 และนอกจากนี้แล้วความสัมพันธ์ระหว่างคลาสลูกที่อยู่ในคลาสแม่เดียวกันสามารถที่จะยืมแบนด์วิดท์ของคลาสลูกอื่นที่ไม่ได้ใช้งานได้ ซึ่งรูปแบบการทำงานของอัลกอริธึม Classful qdisc นั้นสามารถแสดงได้ดังรูปที่ 3.10

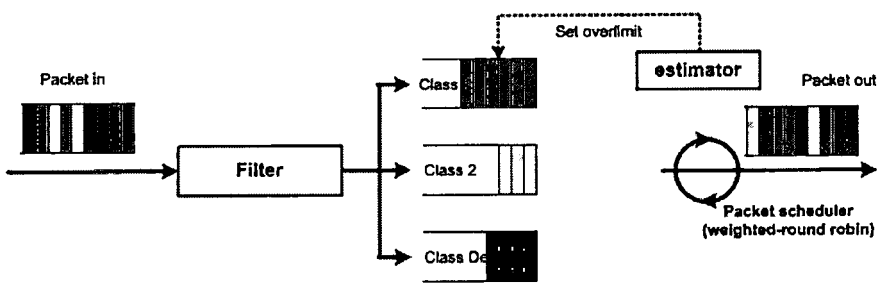


รูปที่ 3.9 ลักษณะคลาสต่างแบบ Classful



รูปที่ 3.10 รูปแบบการทำงานแบบ Classful

- **Class-Based Queueing (CBQ) :** CBQ จะมีคลาสหลายคลาส โดยจะมีการกำหนดคลาสที่มีค่าโดยปริยาย (default class) ไว้ให้ในกรณีที่ข้อมูลที่เข้ามานั้นไม่ตรงกับคลาสใดๆ ที่กำหนดไว้ก็จะมาทำตามกฎที่กำหนดไว้ใน class default โดยในแต่ละคลาสจะมีคิวของคลาสเอง และมีการกำหนดค่าการแชร์แบนด์วิดท์ให้กับแต่ละคลาสด้วย โดยที่คลาสที่เป็นคลาสลูกนั้นสามารถขอยืมใช้แบนด์วิดท์ของคลาสที่เป็นคลาสพ่อแม่ได้ จากรูปที่ 3.6 จะมีตัวกรองคัดกรองแพคเกต (filter) จะคอยจำแนกแพคเกตที่มาถึงไปยังคลาสต่างที่เหมาะสม โดย estimator จะเป็นตัวตรวจสอบว่าคลาสมีการใช้แบนด์วิดท์เกินกว่าค่าที่กำหนดไว้หรือไม่ ถ้าหากเกินกว่าที่กำหนด estimator จะทำการ กำหนด overlimit ไว้ที่คลาสนั้นๆ ส่วน scheduler จะพิจารณาแพคเกตต่อไปที่จะถูกส่งจากคลาสซึ่งขึ้นอยู่กับระดับความสำคัญ และสถานะของคลาสโดยจะใช้หลักการ weighted-round robin scheduling คือในแต่ละรอบการทำงานนั้นจะวนไปทุกคลาสตามลำดับ โดยที่เมื่อถึงคลาสนั้นจะมีการตรวจสอบค่าถ่วงน้ำหนัก (weighted) ถ้ามีค่ามากก็จะส่งแพคเกตออกไปได้มากกว่าคลาสที่มีค่าถ่วงน้ำหนักน้อย จากรูปที่ 3.11 มาดูในส่วนของ Packet out สมมุติว่าใน 1 รอบของการส่งข้อมูลคือ 5 แพคเกต Class 1 ได้รับการกำหนดค่าค่าถ่วงน้ำหนักไว้มากจึงสามารถส่งได้ถึง 3 แพคเกต แต่ในขณะที่ Class 2 และ Class Default ไม่ได้มีการกำหนดค่าถ่วงน้ำหนักไว้ใน 1 รอบจึงส่งได้เพียง 1 แพคเกต



รูปที่ 3.11 รูปแบบการทำงานแบบ CBQ

- **Hierarchical Token Bucket (HTB) :** เป็นอีกรูปแบบ Classful qdisc ที่มีความสามารถในการจัดการทราฟฟิกได้ดี ซึ่งในโครงงานนี้ใช้ HTB มาช่วยในการควบคุมและจัดการแบนด์วิดท์ ดั้งนั้นเพื่อให้เกิดประโยชน์ในการทำความเข้าใจให้ได้มาที่สุดจึงแยก HTB ออกมาเป็นหัวข้อหลักในการอธิบายในหัวข้อถัดไป

3.4 Hierarchical Token Bucket (HTB)

HTB เป็นอัลกอริทึมหนึ่งที่ช่วยในการจัดการและควบคุมทราฟฟิกโดยการควบคุมทราฟฟิกขาออก (egress) โดยหลักการคือการทำให้อินเทอร์เฟซจริงที่มีอยู่แค่ 1 อินเทอร์เฟซนั้นเสมือนว่ามีอินเทอร์เฟซย่อย (class) หลายๆ อินเทอร์เฟซอยู่ในอินเทอร์เฟซจริง ซึ่งการที่จะจัดการก็จะสามารถทำได้ที่อินเทอร์เฟซย่อยนั้นได้ โดยสามารถกำหนดค่าต่าง ได้เช่น ขนาดของแบนด์วิดท์ (rate), การอนุญาติให้ยืมแบนด์วิดท์ (ceil), การอนุญาติให้เพิ่มการส่งอย่างฉับพลัน (burst) และการจัดลำดับความสำคัญ (priority) ซึ่งในรายละเอียดจะได้กล่าวต่อไป

3.4.1 หลักการขอยืมใช้แบนด์วิดท์ (Borrowing)

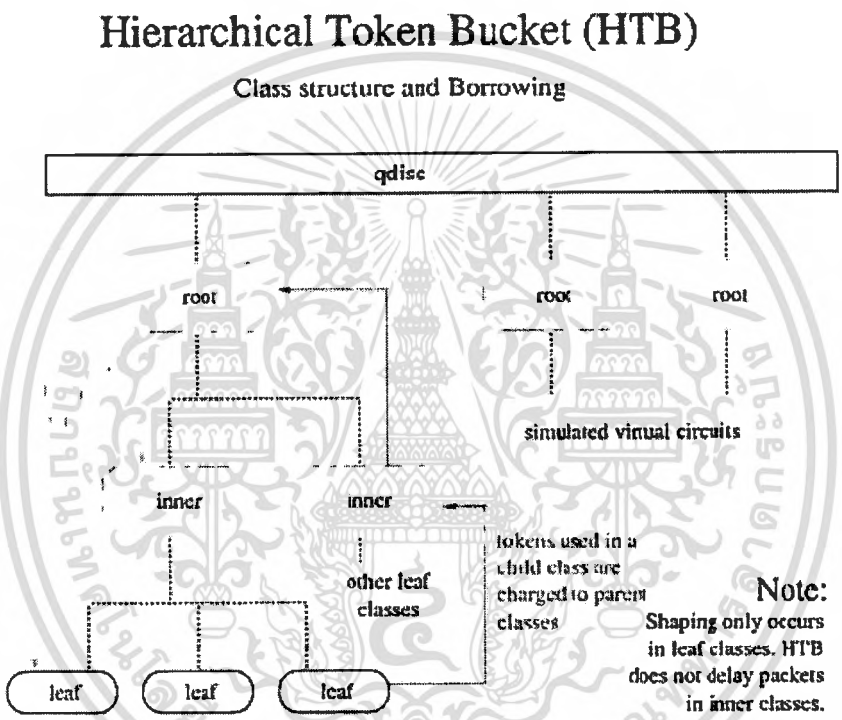
โดยหลักการของ Classful qdisc นั้นจะสามารถมีคลาส (class) ได้เป็นคลาสแม่ (parent class) และคลาสลูก (child class) ดังรูปที่ 3.9 ซึ่งทุกคลาสจะมีการระบุแบนด์วิดท์ที่ได้โดยการกำหนดค่า rate และจะมีการกำหนดให้คลาสลูกสามารถยืมใช้แบนด์วิดท์จากคลาสแม่โดยการกำหนดค่า ceil ซึ่งรายละเอียดสถานะต่างๆของการยืมมีดังตารางที่ 3.2

ตารางที่ 3.2 กฎการยืมแบนด์วิดท์ของ HTB

ชนิดคลาส	สถานะคลาส	สถานะภายใน HTB	ผลการยืม
leaf	< rate	HTB_CAN_SEND	คลาสลูกสามารถส่งได้จนถึงค่า rate ที่กำหนดไว้
leaf	> rate, < ceil	HTB_MAY_BORROW	คลาสลูกสามารถจะยืมจากคลาสแม่ได้ถ้าคลาสแม่มีเหลือ โดยจะแบ่งเป็นอัตราส่วนกับคลาสลูกอื่นๆ
leaf	> ceil	HTB_CANT_SEND	จะไม่สามารถส่งได้เกินค่า ceil และจะทำให้เกินการ delay ขึ้นและค่า latency สูง
inner, root	< rate	HTB_CAN_SEND	คลาสแม่สามารถให้ลูกยืมได้

ตารางที่ 3.2 (ต่อ)

ชนิดคลาส	สถานะคลาส	สถานะภายใน HTB	ผลการยืม
inner, root	> rate, < ceil	HTB_MAY_BORROW	คลาสแม่สามารถยืมคลาสแม่ของตัวเองได้ และสามารถให้ลูกยืมได้ตามอัตตราส่วนกับคลาสลูกอื่นๆ
inner, root	> ceil	HTB_CANT_SEND	คลาสแม่สามารถยืมแม่ได้ และสามารถให้ลูกยืมได้ตามอัตตราส่วน



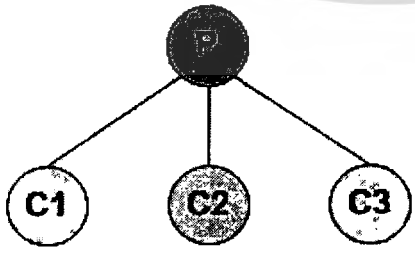
รูปที่ 3.12 แสดงโครงสร้างและการยืมแบนด์วิดท์ของ HTB

3.4.2 พารามิเตอร์ของ HTB (HTB parameter)

ก่อนหน้านี้ได้กล่าวถึงไปแล้วบ้างสำหรับบางพารามิเตอร์ในหัวข้อนี้จะกล่าวถึงรายละเอียดเพื่อจะทำให้เข้าใจและสามารถนำไปใช้งานกับ tc ได้อย่างมีประสิทธิภาพมากที่สุด

- **default :** เป็นพารามิเตอร์ทางเลือกของ HTB โดยมีค่าปริยาย (default) เท่ากับ0 ใช้สำหรับในกรณีที่มีแพคเกตเข้ามายัง HTB และไม่ตรงกับคลาสใดๆเลย แพคเกตนั้นจะตกเข้าไปยัง default แต่ถ้าไม่ได้กำหนดค่าให้ default การส่งแพคเกตนั้นจะส่งตรงไปยังอินเทอร์เฟซตรงซึ่งค่าแบนด์วิดท์ที่ได้ในกรณีนี้จะเป็นค่าสูงสุดของอินเทอร์เฟซนั้นๆ

- **rate :** เป็นค่าที่ใช้ระบุแบนด์วิดท์น้อยที่สุดที่คลาสจะได้รับ และรับประกันว่าคลาสจะได้รับแบนด์วิดท์อย่างน้อยที่สุดตามค่า rate ที่กำหนด
- **ceil :** เป็นค่าที่ใช้ระบุแบนด์วิดท์มากที่สุดที่คลาสจะได้รับ โดยค่าที่จะได้เพิ่มมาเป็นค่าสูงสุดนั้นจะได้มาจากการยืมคลาสแม่ดังที่ได้กล่าวมาแล้วก่อนหน้าในหัวข้อ 3.4.1
- **burst :** เป็นการกำหนดค่าที่อนุญาติให้คลาสสามารถแทรกการส่งข้อมูลออกไปได้ทันทีก่อนในกรณีที่แบนด์วิดท์คับคั่ง เหมาะสำหรับกำหนดให้คลาสที่ข้อมูลที่ส่งไม่มากแต่ต้องการความรวดเร็วในการส่งโดยไม่ต้องรอคิวนาน สิ่งที่ควรระวังในการกำหนดค่า burst ของ HTB นั้นค่า burst ของคลาสแม่นั้นต้องสูงกว่าคลาสลูกเสมอ มิฉะนั้นจะทำให้คลาสแม่มีปัญหาในการส่งเพราะค่าที่คลาสลูกต้องการมีมากกว่าค่าที่คลาสแม่จะให้ได้
- **priority :** เป็นการระบุให้ HTB ทราบว่าคลาสใดมีความสำคัญกว่า โดยคลาสที่มีความสำคัญสูงสามารถที่จะใช้แบนด์วิดท์ได้ก่อน ซึ่งค่าที่สามารถระบุได้คือ 0 – 7 โดยการที่ต้องการระบุให้คลาสมีความสำคัญมากต้องกำหนดค่าตัวเลขให้น้อย
- **quantum :** เป็นค่าที่ใช้สำหรับการควบคุมการยืมแบนด์วิดท์ของ HTB โดยปรกตินั้นค่า quantum จะคำนวนโดย HTB โดยผู้ใช้ไม่สามารถแก้ได้ โดยค่านี้มีความสำคัญมากในการยืมใช้แบนด์วิดท์ระหว่างคลาสลูกหลายๆ คลาสที่มายืมใช้แบนด์วิดท์จากคลาสแม่เดียวกัน เช่น มีคลาสดังรูปที่ 3.13 เมื่อในช่วงเวลาหนึ่งกำหนดให้คลาสลูกทั้ง 3 (C1, C2, C3) ส่งข้อมูลที่ใช้แบนด์วิดท์ 90 ผลที่ได้ทุกคลาสจะได้แบนด์วิดท์เท่ากับค่าน้อยสุดที่จะได้รับ (rate) เพราะคลาส P มีแบนด์วิดท์เพียง 100 เมื่อเอาค่า rate คลาสลูกทั้ง 3 รวมกันจะได้เท่ากับที่ค่าคลาสแม่ให้ได้พอดี แต่เมื่อต่อมา C1 หยุดส่งก็จะมีแบนด์วิดท์เหลือ ดังนั้นคลาส C2 และ C3 จึงใช้แบนด์วิดท์ที่เหลือได้โดยแบ่งให้ C2 1 ส่วน และ C3 6 ส่วน (ตามอัตตราส่วนจากค่า rate)



class	rate	ceil
P	100	100
C1	30	100
C2	10	100
C3	60	100

รูปที่ 3.13 ตัวอย่างคลาสในการคิดค่า quantum

3.5 การจัดการทราฟฟิกของลินุกซ์ (Linux Traffic Control)

การจัดที่จะสามารถคัดแยกข้อมูลได้นั้นสามารถทำได้หลายวิธี แต่โดยทั่วไปแล้วจใช้วิธีการคัดแยกจากการดูข้อมูลส่วนหัว (Header) ของข้อมูล โดยในแต่ละเลเยอร์ที่แตกต่างกันก็จะมีชนิดหรือรูปแบบของข้อมูลส่วนหัวที่แตกต่างกันไป เช่นในลำดับชั้นของเน็ตเวิร์คก็จะดูในส่วนข้อมูลส่วนหัวของไอพี (IP Header) แต่ถ้าพูดถึงภาพรวมทั้งหมดโดยเรียกข้อมูลที่ส่งไปว่าแพคเกตแล้วเรียกรวมข้อมูลส่วนหัวของแพคเกตว่า Packet Header แล้ว ส่วนของข้อมูลส่วนหัวเหล่านี้ เช่น ไอพีต้นทาง (Source IP), ไอพีปลายทาง (Destination IP), หมายเลขพอร์ตต้นทาง (Source Port) และหมายเลขพอร์ตปลายทาง (Destination Port) จะช่วยให้ การคัดแยกแพคเกตสามารถทำได้โดยไม่ยาก ในส่วนลินุกซ์นั้นจะมีเครื่องมือที่ช่วยในการจัดการทราฟฟิคโดยใช้ tc (Traffic Control) โดยพื้นฐานการคัดแยกแพ็คเก็ตนั้นจะนำมาคัดแยกที่คิวขาเข้า (Ingress queue) หรือคิวขาออก (Egress queue) ที่ใดที่หนึ่ง ในการกำหนดโดยปกติจะกำหนด 1 คิวหลักสำหรับ 1 อินเทอร์เฟซ และข้อมูลจะถูกกำหนดคิวเป็น FIFO (First-in First-out) โดยมาตรฐานการทำงานร่วมกันของคิว และอัลกอริทึมในการส่งข้อมูลออกนั้นเรียกว่า qdisc ซึ่งได้กล่าวมาแล้วในบทที่ 2

ในส่วนนี้จะอธิบายถึงวิธีการจัดการแพคเกตโดยใช้ tc และ qdisc อัลกอริทึมที่ฝั่งขาออกของอินเทอร์เฟซ (Egress Interface) โดยการจะคัดแยกแพคเกตให้วิ่งไปตามเส้นทางที่กำหนด (flow) และอธิบายวิธีการจัดการข้อมูลของ qdisc ว่าสามารถทำได้อย่างไร

3.5.1 Traffic Control (tc)

Traffic Control (tc) เป็นส่วนหนึ่งของชุดคำสั่งที่รวมไว้อยู่ใน iproute2 ของลินุกซ์ เพื่อให้ผู้ใช้สามารถจัดการกับความสามารถด้านเครือข่ายได้โดยชุดของคำสั่ง tc นั้นแบ่งส่วนการทำงานออกเป็น 2 ส่วนคือ การกัดการกับอัลกอริทึม qdisc โดยสามารถกำหนดได้ว่าจะใช้อัลกอริทึม qdis อะไรในการจักการข้อมูล และส่วนถัดมาคือการที่จะคัแยกแพคเกตเพื่อที่จะส่งให้เข้าไปยัง qdis ต่างที่ได้สร้างรอไว้แล้วในการจัดการแพคเกต ก่อนที่จะกล่าวรายละเอียดการใช้คำสั่ง tc จะขออธิบายคำศัพท์ที่เกียวข้องดังนี้

- **Queueing Disciipline (qdisc) :** เป็นอัลกอรึทึมที่ใช้ในการจัดการคิวของแพ็คเก็ตขณะที่จะทำการส่งแพ็คเก็ตออกไป
- **Classless qdisc :** คือ qdisc ที่ไม่สามารถมี qdisc ย่อยภายในได้
- **Classful qdisc :** คือ qdisc ที่สามารถมี qdisc ย่อยภายในได้ และสามารถจัดแบ่งหมวดหมู่ย่อย (Classes) ภายใน เพื่อการจัดการแพ็คเก็ตได้
- **Root qdisc :** คือ qdisc หลักที่อยู่บนอินเทอร์เฟซ โดยมีทั้งแบบ classful และ classless

- **egress qdisc :** เป็นการทำงานกับข้อมูลในขณะที่ข้อมูลจะออกจากอินเทอร์เฟซ
- **ingress qdisc :** เป็นการทำงานกับข้อมูลในขณะที่ข้อมูลจะเข้าสู่อินเทอร์เฟซ
- **Class :** เป็นการกำหนดกลุ่มของข้อมูลที่ต้องการจัดการ โดยสามารถมีคลาสย่อยหรือไม่มีก็ได้ หรือจะเป็นที่ที่ qdisc ทำงานอยู่
- **Filter :** เป็นการคัดแยกข้อมูลเพื่อจะส่งไปยังคลาสต่างๆที่กำหนดไว้ก่อนหน้านี้

3.5.2 รูปแบบคำสั่ง tc

รูปแบบคำสั่งของ tc นั้นจะสามารถแบ่งออกเป็นชุดคำสั่งได้เป็น การสร้าง root qdisc, การสร้าง qdisc ที่ไม่ใช่ root, การสร้าง class, การสร้าง filter ซึ่งชุดคำสั่งจะมีรายละเอียดดังนี้

การสร้าง root qdisc:

```
tc qdisc add dev DEV root handle 1: QDISC [PARAMETER]
```

การสร้าง qdisc ที่ไม่ใช่ root:

```
tc qdisc add dev DEV parent PARENTID handle HANDLE QDISC
                [PARAMETER]
```

การสร้าง class:

```
tc class add dev DEV parent PARENTID classid CLASSID QDISC
                [PARAMETER]
```

การสร้าง filter:

```
tc filter [add|del|change|get] dev DEV [pref PRIO]
          [protocol PROTO] [root|class CLASSID]
          [handle FILTERID] [[FILTER_TYPE][help|OPTION]]
```

DEV: อินเทอร์เฟซที่ต้องการคอนฟิค เช่น eth0, eth1

PARENTID: เป็นเลข ID ของคลาสแม่ที่ qdisc ทำงานอยู่ภายใต้

HANDLEID: เป็นค่าเลข ID ที่ไม่ซ้ำ เพื่อใช้ในการแยกว่าจะทำงานกับ qdisc ไหน

CLASSID: เป็นค่าเลข ID ที่ไม่ซ้ำ เพื่อใช้ในการแบ่งแยกคลาสต่างๆกัน

QDISC: คือชนิดของ qdisc algorithm เช่น HTB, CBQ, PFIFO, SFQ

PARAMETER: คือตัวแปรปลีกย่อย ซึ่งจะขึ้นอยู่กับ qdisc แต่ละชนิด

add|delete|change|get: เป็นการระบุว่าต้องการที่จะกระทำอะไรกับ filter

PRIO: เป็นการกำหนดค่า priority ที่กำหนดให้ filter นั้น

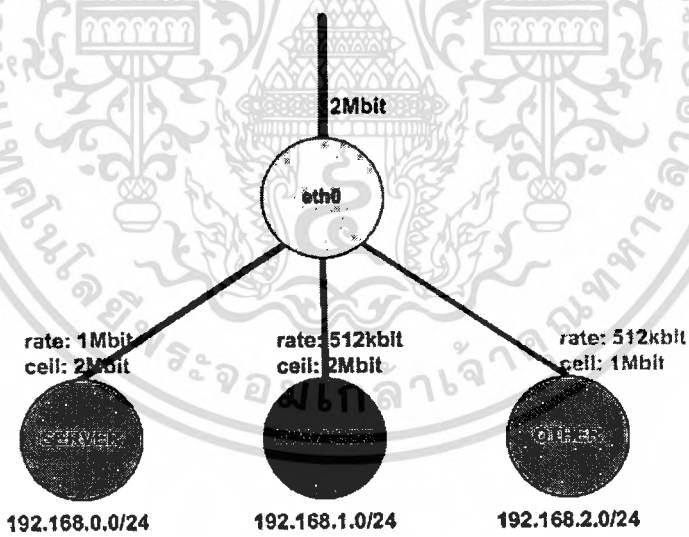
PROTO: เป็นการระบุโพรโตคอลที่ต้องการจะทำการ filter

FILTERID: เป็นค่าตัวเลขที่ใช้ระบุถึง filter โดยจะเป็นค่าที่ไม่ซ้ำกัน

FILTER_TYPE: จะเป็นการระบุชนิดการ filter เช่น rsvp, u32, fw, route อื่นๆ

3.5.3 ตัวอย่างคำสั่ง tc

ก่อนหน้าได้กล่าวรายละเอียดต่างของคำสั่งต่างๆ ไปแล้วในหัวข้อนี้จะทำการแสดงการนำคำสั่งเหล่านั้นมาร่วมกันทำงานเพื่อจัดการแบนด์วิดท์ที่มี โดยตัวอย่างนี้มีรายละเอียดคือ มีแบนด์วิดท์อยู่ทั้งหมด 2Mbit โดยที่จัดแบ่งกสนใช้งานแบนด์วิดท์เป็น 3 กลุ่มคือ Server, Manger, Other ซึ่งแต่ละกลุ่มได้รับจัดสรรแบนด์วิดท์ (rate, ceil) ตามรูปที่ 3.14 และแต่ละกลุ่มมีไอพีดังรูป โดยการจัดสรรนั้นจะคัดแยกตามกลุ่มไอพี



รูปที่ 3.14 ตัวอย่างการจัดสรรแบนด์วิดท์

คำสั่ง

```
1# tc qdisc add dev eth0 root handle 1:0 htb

2# tc class add dev eth0 parent 1:0 classid 1:1 htb rate 2048kbit
3# tc class add dev eth0 parent 1:1 classid 1:10 htb rate 1024kbit \
     ceil 2048kbit
4# tc class add dev eth0 parent 1:1 classid 1:11 htb rate 512kbit \
     ceil 2048kbit
5# tc class add dev eth0 parent 1:1 classid 1:12 htb rate 512kbit \
     ceil 1048kbit

6# tc filter add dev eth0 parent 1:0 protocol ip u32 match ip \
     dst 192.168.0.0/24 flowid 1:10
7# tc filter add dev eth0 parent 1: protocol ip u32 match ip \
     dst 192.168.1.0/24 flowid 1:11
8# tc filter add dev eth0 parent 1: protocol ip u32 match ip \
     dst 192.168.2.0/24 flowid 1:12
```

อธิบายคำสั่ง

- 1# ประกาศให้ eth0 เป็น root qdisc โดยใช้ htb อัลกอริธึม มีเลอ้างอิงเป็น 1:0
- 2# สร้างคลาสหลัก 1:1 โดยมีแบนด์วิดท์ 2048kbit โดยไม่สามารถ burst ได้
- 3# สร้างคลาส 1:10 ให้กับ SERVER โดยมีคลาส 1:1 เป็นคลาสแม่ มีแบนด์วิดท์ 1Mbit และสามารถ burst ได้ถึง 2048kbit
- 4# สร้างคลาส 1:11 ให้กับ MANAGER โดยมีคลาส 1:1 เป็นคลาสแม่ มีแบนด์วิดท์ 512kbit และสามารถ burst ได้ถึง 2048kbit
- 5# สร้างคลาส 1:12 ให้กับ OTHER โดยมีคลาส 1:1 เป็นคลาสแม่ มีแบนด์วิดท์ 512kbit และสามารถ burst ได้ถึง 1Mbit
- 6# การกฎการจับทราฟฟิกที่อยู่ในกลุ่ม SERVER (192.168.0.0/24) เมื่อพบให้ส่งไปยัง คลาส 1:10 (การที่ให้ match ip dst : ไอพีปลายทางเพราะเป็นการควบคุมแบนด์วิดท์ขาเข้า ดังนั้นไอพีของ SERVER จึงเป็น ip dst)
- 7# การกฎการจับทราฟฟิกที่อยู่ในกลุ่ม MANAGER (192.168.1.0/24) เมื่อพบให้ส่งไปยัง คลาส 1:11
- 8# การกฎการจับทราฟฟิกที่อยู่ในกลุ่ม OTHER (192.168.2.0/24) เมื่อพบให้ส่งไปยัง คลาส 1:12
- ถ้ามีทราฟฟิกอื่นๆที่ไม่ตรงกับค่า filter ที่ระบุทราฟฟิกนั้นจะถูกส่งตรงไปยัง eth0

บทที่ 4
การวิเคราะห์และออกแบบระบบ

จากปัญหาเรื่องของการจะจัดสรรแบนด์วิดท์ที่มีอยู่อย่างจำกัดในการใช้งานให้มีประสิทธิภาพสูงสุด เนื่องจากปัจจุบันอินเทอร์เน็ตได้รับความนิยมอย่างสูงทำให้มีแอพพลิเคชันต่างๆ ที่ทำงานอยู่บนอินเทอร์เน็ตมากมาย ไม่ว่าจะเป็น เว็บ, เอฟทีพี, อีเมล, วีดีโอออนไลน์, เกมออนไลน์, บิททอร์เรนท์ หรือแม้นแต่ไวรัส ซึ่งตัวอย่างแอพพลิเคชันเหล่านี้มีสิ่งที่เป็นประโยชน์แกองค์กรและไม่เป็นประโยชน์ ซึ่งในการที่ปล่อยให้การใช้แบนด์วิดท์ เป็นไปโดยอิสระทำให้องค์ไม่ได้ประโยชน์เท่าที่ควร จึงต้องมีการดูแลควบการใช้แบนด์วิดท์ ให้เกิดประโยชน์สูงสุด แต่การจะทำได้นั้นต้องมีอุปกรณ์หรือเครื่องมือมาช่วยซึ่งค่อยข้างมีราคาสูง หรือถ้าเป็นฟรีแวร์ (Freeware) อย่างลินุกซ์ก็ต้องการคนที่มีความชำนาญในการติดตั้งและคอนฟิค ซึ่งไม่คุ้มค่าที่จะลงทุนสำหรับองค์กร ดังนั้นโครงงานนี้จึงได้มีการพัฒนาอุปกรณ์ในการควบคุมข้อมูลในเครือข่ายขึ้นมาทำให้การใช้งานแบนด์วิดท์มีประสิทธิภาพสูงสุด, การใช้งานง่าย และราคาไม่แพง ซึ่งมีขั้นการออกแบบและพัฒนาระบบดังที่จะได้กล่าวต่อไป

4.1 การวิเคราะห์และออกแบบระบบงาน

จากปัญหาที่กล่าวมาข้างต้น จึงได้มีการออกแบบระบบเพื่อเข้ามาช่วยในการจัดการข้อมูลในเครือข่าย ซึ่งในขั้นตอนของการออกแบบระบบนั้นจำเป็นต้องใช้เครื่องมือที่ช่วยในการออกแบบ ซึ่งได้เลือกใช้ยูเอ็มแอล (UML) เป็นเครื่องมือในการออกแบบระบบ และใช้แผนภาพความสัมพันธ์ระหว่างเอนทิตี (E-R Diagram) ในการออกแบบฐานข้อมูล ซึ่งมีแบบจำลองที่นำเสนอ ดังนี้

- ยูสเคสไดอะแกรม (Usecase Diagram)
- รายละเอียดยูสเคส (Usecase Description)
- แผนภาพกิจกรรม (Activity Diagram)
- ซีเควนซ์ไดอะแกรม (Sequence Diagram)
- แผนภาพความสัมพันธ์ระหว่างเอนทิตี (E-R Diagram)

4.1.1 ยูสเคสไดอะแกรม (Usecase Diagram)

ยูสเคสไดอะแกรม เป็นเครื่องมือที่ใช้แสดงขอบเขตการทำงานของระบบ และการปฏิสัมพันธ์ระหว่างแอกเตอร์และฟังก์ชันการทำงานของระบบ โดยยูสเคสไดอะแกรมของระบบช่วยประมาณภาระงานในการพัฒนาระบบ ซึ่งได้แสดงดังรูปที่ 4.1

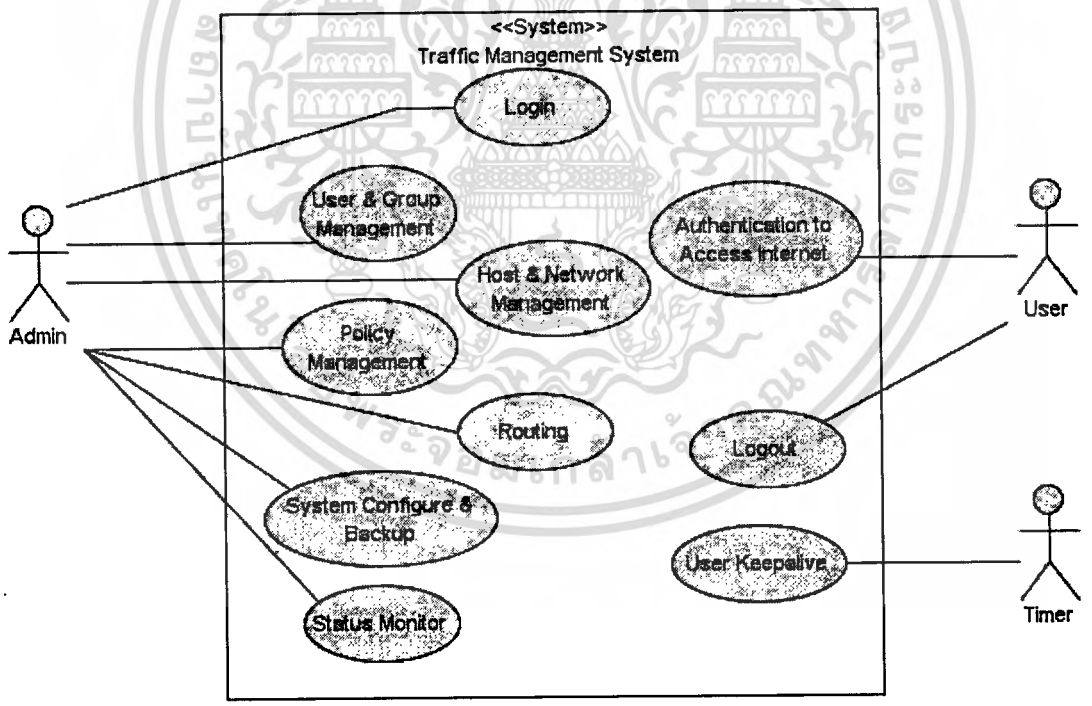
ยูสเคสไดอะแกรมดังที่ปรากฏในรูปที่ 4.1 สามารถอธิบายได้ถึงผู้ใช้ระบบที่มีส่วนร่วมกับระบบได้ทั้งสิ้น 3 แอกเตอร์ ดังนี้

1. Admin หรือ ผู้ดูแลระบบมีหน้าที่ในการจัดการข้อมูลที่เป็นข้อมูลตั้งต้นในระบบ และรับผิดชอบการปรับปรุงค่าพารามิเตอร์ที่ใช้ในการควบคุมรูปแบบการกำหนดสิทธิในการใช้แบนด์วิดท์ และการให้สิทธิแก่ผู้ใช้ในการใช้แบนด์วิดท์

2. User หรือ ผู้ใช้ คือผู้ที่ต้องการอินเทอร์เน็ตแบนด์วิดท์ โดยจะถูกกำหนดสิทธิการใช้งานแบนด์วิดท์โดยผู้ดูแลระบบ และถูกควบคุมการใช้โดยระบบ

3. Timer คือเวลาที่ถูกกำหนดให้ระบบทำงานใดๆโดยอัตโนมัติ ซึ่งช่วงเวลานี้ถูกกำหนดโดยผู้ดูแลระบบ

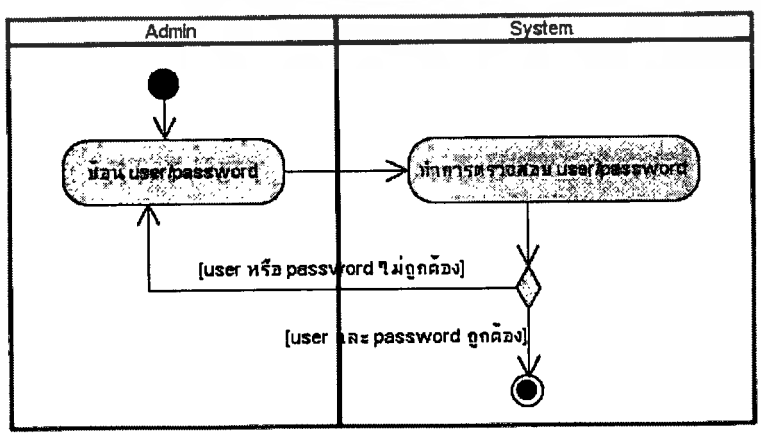
จากยูสเคสไดอะแกรมใน รูปที่ 4.1 ได้แสดงถึงฟังก์ชันการทำงานของระบบช่วยประมาณภาระงานในการพัฒนาซอฟท์แวร์ ได้ทั้งสิ้น 10 ยูสเคส โดยอธิบายรายละเอียดของแต่ละฟังก์ชันได้โดยมีรายละเอียดยูสเคส (Usecase Description) และแผนภาพกิจกรรม (Activity Diagram) เพื่อให้ให้เข้าใจได้ง่ายขึ้นดังนี้



รูปที่ 4.1 ยูสเคสไดอะแกรมของระบบการจัดการและควบคุมข้อมูลในเครือข่าย

ตารางที่ 4.1 รายละเอียดของผู้ดูแลระบบล็อกอิน

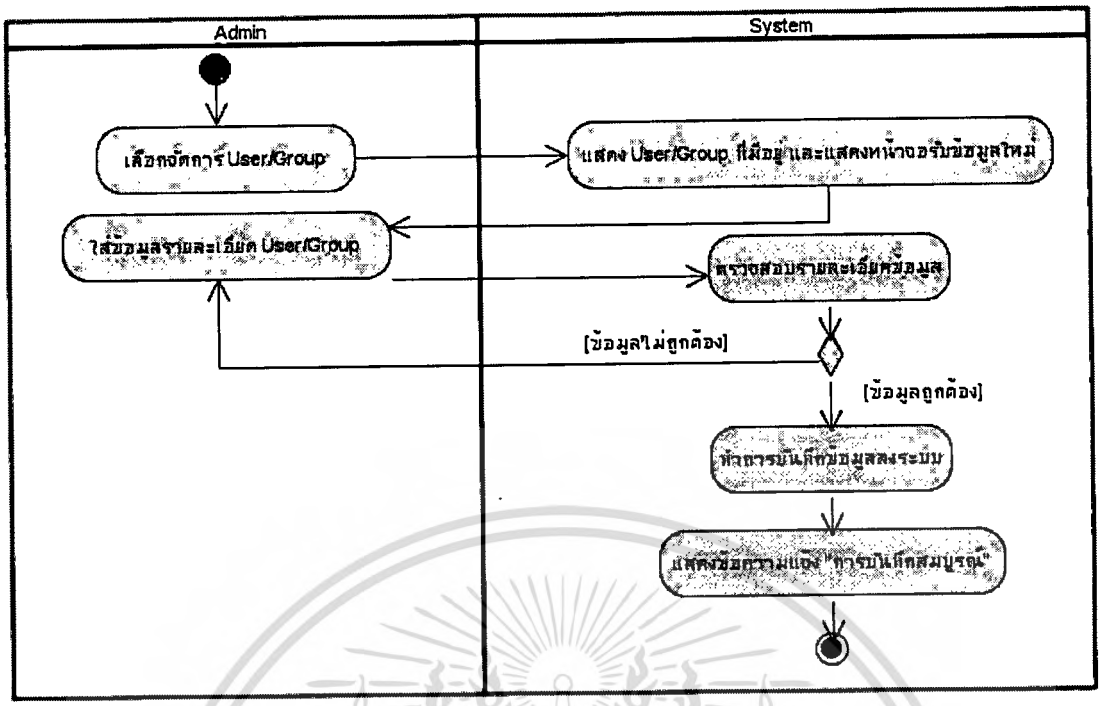
หมายเลขยูสเคส :	Usecase01	
ชื่อยูสเคส :	Login	
รายละเอียดโดยย่อ :	ฟังก์ชันสำหรับการตรวจสอบสิทธิของการเข้าจัดการระบบของผู้ดูแลระบบ	
ทริกเกอร์ :	เมื่อผู้ดูแลระบบมีความต้องการเข้าจัดการระบบ	
ผู้ใช้งานระบบ :	ผู้ดูและระบบ (Admin)	
เงื่อนไขขั้นต้น :	ผู้ดูแลระบบจะต้องมีชื่อและรหัสผ่านเข้าระบบอยู่ก่อนแล้ว	
การทำงานปกติ :	**แอกเตอร์**	**ระบบ**
	1. ผู้ดูแลระบบพิมพ์ยูอาร์แอลของหน้าหลักเข้าจัดการระบบ	2. ระบบแสดงหน้าจอรับการใส่ชื่อและรหัสผ่าน
	3. ผู้ดูแลระบบพิมพ์ชื่อและรหัสผ่านของตนเอง	4. ระบบทำการตรวจสอบชื่อและรหัสผ่าน ถูกต้อง 5. ระบบแสดงหน้าจอหลักในการจัดการระบบ
เงื่อนไขหลังจบงาน :	ผู้ดูแลระบบสามารถเข้าปรับแก้พารามิเตอร์ของระบบได้	
กรณีผิดปกติ :	4a. ชื่อหรือรหัสผ่านไม่ถูกต้อง : ระบบแสดงข้อความแจ้ง "Invalid username or password" และกลับไปยังขั้นตอนที่ 3	



รูปที่ 4.2 แผนภาพกิจกรรมผู้ดูแลระบบล็อกอิน

ตารางที่ 4.2 รายละเอียดของการจัดการผู้ใช้/กลุ่มผู้ใช้

หมายเลขยูสเคส :	Usecase02	
ชื่อยูสเคส :	User & Group Management	
รายละเอียดโดยย่อ :	ฟังก์ชันสำหรับผู้ดูแลระบบเข้าไปจัดการเพิ่ม, แก้ไขและลบ ผู้ใช้หรือกลุ่มผู้ใช้ โดยสำหรับผู้ใช้สามารถกำหนดค่าเช่น ชื่อล็อกอิน, รหัสผ่าน, จำนวนแบนด์วิดท์ที่สามารถใช้ได้, จัดเข้ากลุ่ม และในส่วนของกลุ่มนั้นสามารถกำหนดชื่อกลุ่ม, จำนวนแบนด์วิดท์ที่สามารถใช้สูงสุดของกลุ่ม	
ทริกเกอร์ :	เมื่อผู้ดูแลระบบมีความต้องการเข้าจัดการระบบ	
ผู้ใช้งานระบบ :	ผู้ดูและระบบ (Admin)	
เงื่อนไขขั้นต้น :	ผู้ดูแลระบบจะต้องผ่านการล็อกอินเข้าระบบก่อนเท่านั้น	
การทำงานปกติ :	**แอกเตอร์**	**ระบบ**
	1. ผู้ดูแลระบบเลือกเมนูจัดการ “ผู้ใช้/กลุ่มผู้ใช้”	2. ระบบแสดง “ผู้ใช้/กลุ่มผู้ใช้” ที่มีอยู่ในระบบพร้อมรายละเอียดบางส่วน และหน้าจอการรับข้อมูล “ผู้ใช้/กลุ่มผู้ใช้” ใหม่
	3. ผู้ดูแลระบบทำการเพิ่มข้อมูล “ผู้ใช้/กลุ่มผู้ใช้” ใหม่เข้าสู่ระบบและยืนยันการเพิ่ม	4. ระบบทำการตรวจสอบข้อมูลใหม่ที่รับเข้ามา ถูกต้องครบสมบูรณ์ 5. ระบบทำการบันทึกข้อมูลของ “ผู้ใช้/กลุ่มผู้ใช้” ลงฐานข้อมูลระบบ 6. ระบบแสดงข้อความแจ้ง “ได้ทำการบันทึกข้อมูลสมบูรณ์”
เงื่อนไขหลังจบงาน :	ข้อมูลของ “ผู้ใช้/กลุ่มผู้ใช้” สามารถเพิ่มเข้าฐานข้อมูลระบบสมบูรณ์	
กรณีผิดปกติ :	4a. ข้อมูลใหม่ที่รับเข้ามาไม่ถูกต้อง : ระบบแจ้งให้ทราบถึงข้อมูลที่ผิด และกลับไปยังขั้นตอนที่ 3	



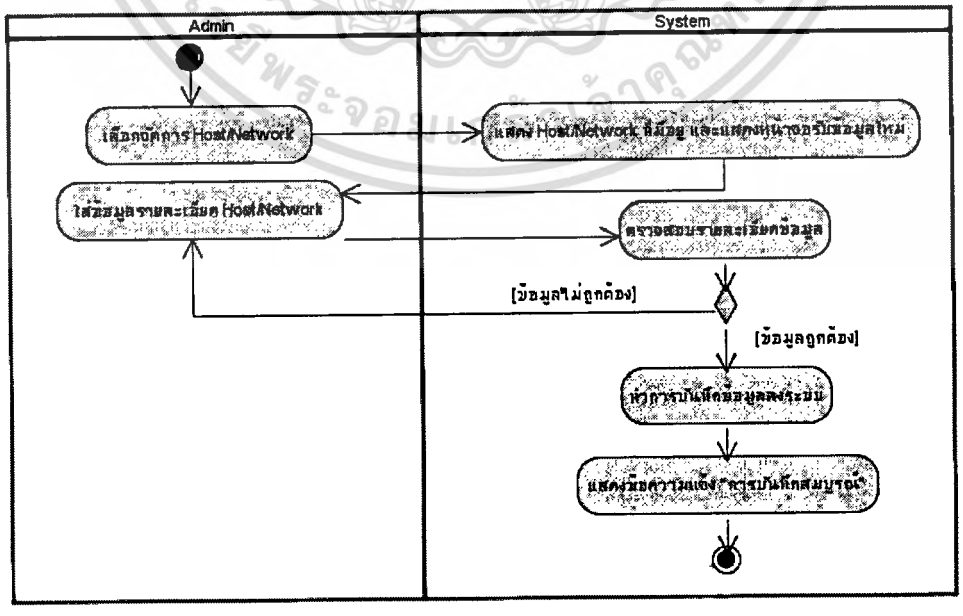
รูปที่ 4.3 แผนภาพกิจกรรมผู้ดูแลระบบจัดการผู้ใช้/กลุ่มผู้ใช้

ตารางที่ 4.3 รายละเอียดของการจัดการโฮสท์และเน็ตเวิร์ค

หมายเลขยูสเคส :	Usecase03	
ชื่อยูสเคส :	Host & Network Management	
รายละเอียดโดยย่อ :	ฟังก์ชันสำหรับผู้ดูแลระบบเข้าไปจัดการเพิ่ม, แก้ไขและลบ โฮสท์และเน็ตเวิร์ค โดยสำหรับโฮสท์และเน็ตเวิร์คสามารถกำหนดค่าเช่น ชื่อ,ไอพี, เน็ตมาร์ค, จำนวนแบนด์วิดท์ที่สามารถใช้ได้ โดยเครื่องที่มีไอพี หรือมีไอพีอยู่ในกลุ่มเน็ตเวิร์คจะไม่ต้อง มีการตรวจสอบสิทธิ โดยจะสามารถใช้แบนด์วิดท์ตามที่กำหนดให้ได้เลย	
ทริกเกอร์ :	เมื่อผู้ดูแลระบบมีความต้องการเข้าจัดการเพิ่ม, แก้ไข, ลบ โฮสท์และเน็ตเวิร์ค	
ผู้ใช้งานระบบ :	ผู้ดูแลระบบ (Admin)	
เงื่อนไขขั้นต้น :	ผู้ดูแลระบบจะต้องผ่านการล๊อกอินเข้าระบบก่อนเท่านั้น	
การทำงานปกติ :	**แอกเตอร์**	**ระบบ**
	1. ผู้ดูแลระบบเลือกเมนูจัดการ “โฮสท์และเน็ตเวิร์ค”	

ตารางที่ 4.3 (ต่อ)

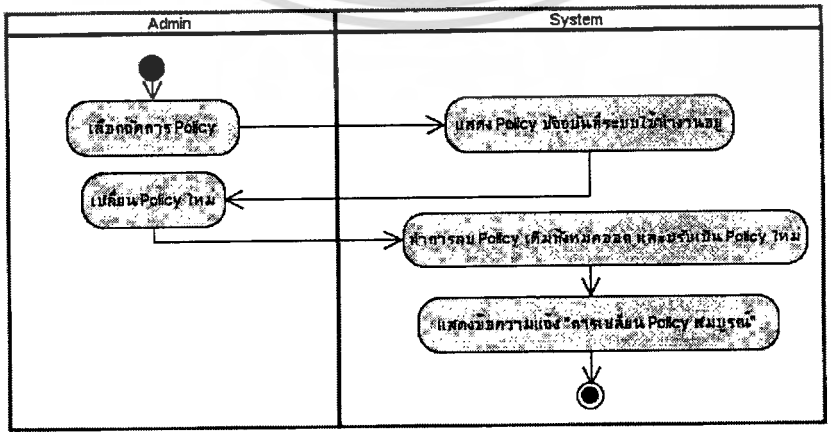
การทำงานปกติ :	**แอกเตอร์**	**ระบบ**
		2. ระบบแสดง “โฮสท์และเน็ตเวิร์ค” ที่มีอยู่ในระบบพร้อมรายละเอียดบางส่วน และหน้าจอการรับข้อมูล “โฮสท์และเน็ตเวิร์ค” ใหม่
	3. ผู้ดูแลระบบทำการเพิ่มข้อมูล “โฮสท์และเน็ตเวิร์ค” ใหม่เข้าสู่ระบบ และยืนยันการเพิ่ม	
		4. ระบบทำการตรวจสอบข้อมูลใหม่ที่รับเข้ามา ถูกต้องครบสมบูรณ์
		5. ระบบทำการบันทึกข้อมูลของ “โฮสท์และเน็ตเวิร์ค” ลงฐานข้อมูลระบบ
		6. ระบบแสดงข้อความแจ้ง “ได้ทำการบันทึกข้อมูลสมบูรณ์”
เงื่อนไขหลังจบงาน :	ข้อมูลของ “โฮสท์และเน็ตเวิร์ค” สามารถเพิ่มเข้าฐานข้อมูลระบบสมบูรณ์	
กรณีผิดปกติ :	4a. ข้อมูลใหม่ที่รับเข้ามาไม่ถูกต้อง : ระบบแจ้งให้ทราบถึงข้อมูลที่ผิด และกลับไปยังขั้นตอนที่ 3	



รูปที่ 4.4 แผนภาพกิจกรรมผู้ดูแลระบบจัดการโฮสท์และเน็ตเวิร์ค

ตารางที่ 4.4 รายละเอียดของการจัดการนโยบายการใช้แบนด์วิดท์

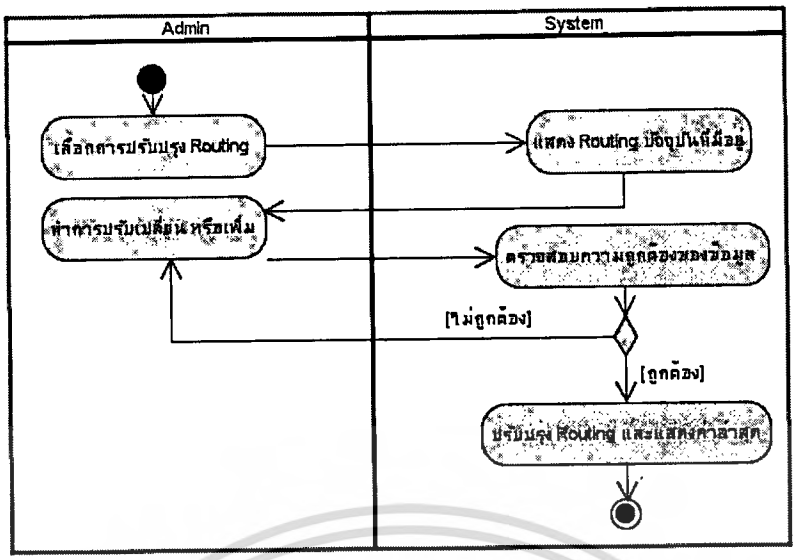
หมายเลขยูสเคส :	Usecase04	
ชื่อยูสเคส :	Policy Management	
รายละเอียดโดยย่อ :	ฟังก์ชันสำหรับผู้ดูแลระบบเข้ากำหนดว่าการจัดการบริหารแบนด์วิดท์นั้นจะให้แบ่งแยกตาม ผู้ใช้ หรือแอพพลิเคชัน หรือแบบกำหนดเองทั้งหมด	
ทริกเกอร์ :	เมื่อผู้ดูแลระบบมีความต้องเปลี่ยนการนโยบายการใช้แบนด์วิดท์	
ผู้ใช้งานระบบ :	ผู้ดูและระบบ (Admin)	
เงื่อนไขขั้นต้น :	ผู้ดูแลระบบจะต้องผ่านการล๊อกอินเข้าระบบก่อนเท่านั้น	
การทำงานปกติ :	**แอกเตอร์**	**ระบบ**
	1. ผู้ดูแลระบบเลือกเมนูจัดการ “Policy” 3. ผู้ดูแลระบบเลือกเปลี่ยน Policy ใหม่	2. ระบบแสดง Policy ที่ระบบใช้ทำงานอยู่ปัจจุบัน 4. ระบบทำการลบ Policy เดิมออกจากระบบทั้งหมดและปรับเปลี่ยนค่าต่างๆ ตาม Policy ใหม่ 5. ระบบแสดงข้อความแจ้ง “ได้ทำการเปลี่ยน Policy สมบูรณ์”
เงื่อนไขหลังจบงาน :	ระบบเปลี่ยนการทำงานตาม Policy ใหม่สมบูรณ์	
กรณีผิดปกติ :	-	



รูปที่ 4.5 แผนภาพกิจกรรมผู้ดูแลระบบเปลี่ยน Policy ระบบ

ตารางที่ 4.5 รายละเอียดของการจัดการเราท์ติง (Routing)

หมายเลขยูสเคส :	Usecase05	
ชื่อยูสเคส :	Routing Management	
รายละเอียดโดยย่อ :	ฟังก์ชันสำหรับผู้ดูแลระบบทำการแก้ไขปรับเปลี่ยนตารางเราท์ติงของระบบ โดยสามารถเพิ่ม และลบ เราท์ติงระบบ	
ทริกเกอร์ :	เมื่อผู้ดูแลระบบมีความต้องเปลี่ยนตารางเราท์ติง	
ผู้ใช้งานระบบ :	ผู้ดูและระบบ (Admin)	
เงื่อนไขขั้นต้น :	ผู้ดูแลระบบจะต้องผ่านการล๊อกอินเข้าระบบก่อนเท่านั้น	
การทำงานปกติ :	**แอกเตอร์**	**ระบบ**
	1. ผู้ดูแลระบบเลือกเมนูจัดการ "เราท์ติง" 3. ผู้ดูแลระบบทำการใส่ค่า เราท์ติงใหม่เข้าระบบ และยืนยันการเพิ่มเราท์ติง	2. ระบบแสดงตารางเราท์ติงที่ใช้ทำงานอยู่ปัจจุบัน 4. ระบบทำการตรวจสอบค่าของเราท์ติงใหม่ที่ใส่เข้ามาว่าถูกต้องตามรูปแบบ ที่กำหนด 5. ระบบทำการเพิ่มเราท์ติงใหม่เข้าระบบ 6. ระบบแสดงข้อความแจ้ง "ได้ทำเพิ่มเราท์ติงสมบูรณ์"
เงื่อนไขหลังจบงาน :	ระบบทำการเพิ่มเราท์ติงใหม่สมบูรณ์ และสามารถส่งแพคเกตไปยังปลายทางใหม่ได้ถูกต้อง	
กรณีผิดปกติ :	4a. ค่าเราท์ติงใหม่ที่ใส่เข้ามาไม่ถูกต้อง : แสดงข้อความแจ้ง "ค่าเราท์ติงผิดพลาด" และกลับไปยังขั้นตอนที่ 3	



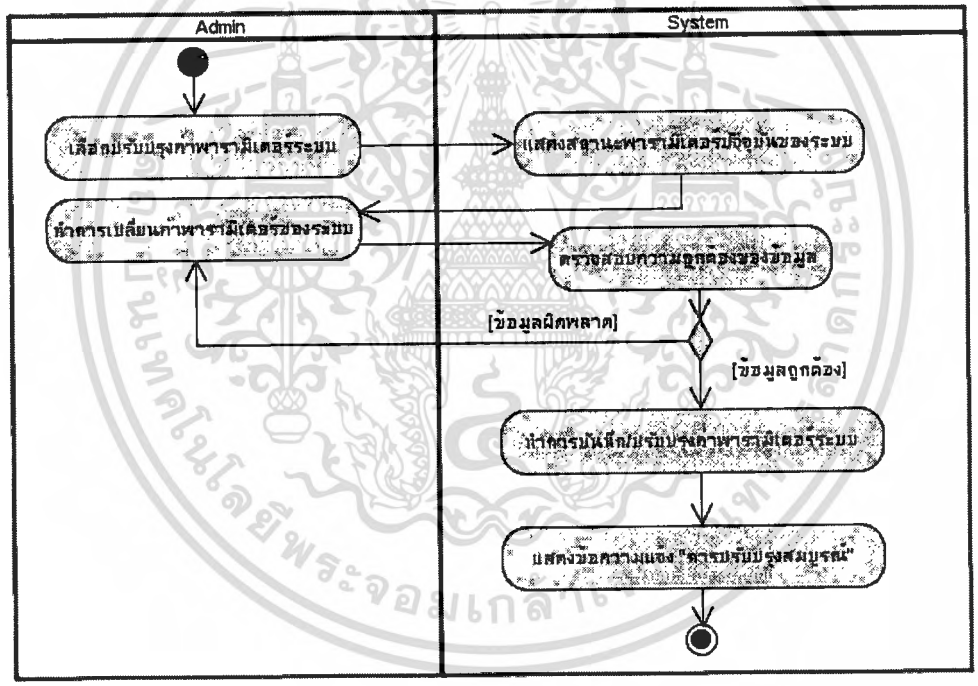
รูปที่ 4.6 แผนภาพกิจกรรมผู้ดูแลระบบเปลี่ยนเราท์ติงระบบ

ตารางที่ 4.6 รายละเอียดของการปรับเปลี่ยนพารามิเตอร์ของระบบ

หมายเลขยูสเคส :	Usecase06	
ชื่อยูสเคส :	System Configure & Backup	
รายละเอียดโดยย่อ :	ฟังก์ชันสำหรับผู้ดูแลระบบทำการแก้ไขค่าพารามิเตอร์หลักของระบบเช่น ไอพีอินเทอร์เฟสขานอก, ขาใน, ดีเอ็นเอส, โดเมน, ดีเฮสซีพี (DHCP), การสำรองคอนฟิค (Backup), รีสตาร์ทระบบและอื่นๆ	
ทริกเกอร์ :	เมื่อผู้ดูแลระบบมีความต้องเปลี่ยนค่าพารามิเตอร์หลักของระบบ	
ผู้ใช้งานระบบ :	ผู้ดูแลระบบ (Admin)	
เงื่อนไขขั้นต้น :	ผู้ดูแลระบบจะต้องผ่านการล็อกอินเข้าระบบก่อนเท่านั้น	
การทำงานปกติ :	**แอกเตอร์**	**ระบบ**
	1. ผู้ดูแลระบบเลือกเมนูจัดการ “System Configure”	
		2. ระบบแสดงค่าพารามิเตอร์ต่างๆ ปัจจุบันที่ระบบใช้ทำงานอยู่
	3. ผู้ดูแลระบบทำการแก้ไขค่าพารามิเตอร์ และยืนยันการแก้ไข	
		4. ระบบทำการตรวจสอบค่าของพารามิเตอร์ ที่ได้รับเข้ามา ถูกต้อง

ตารางที่ 4.6 (ต่อ)

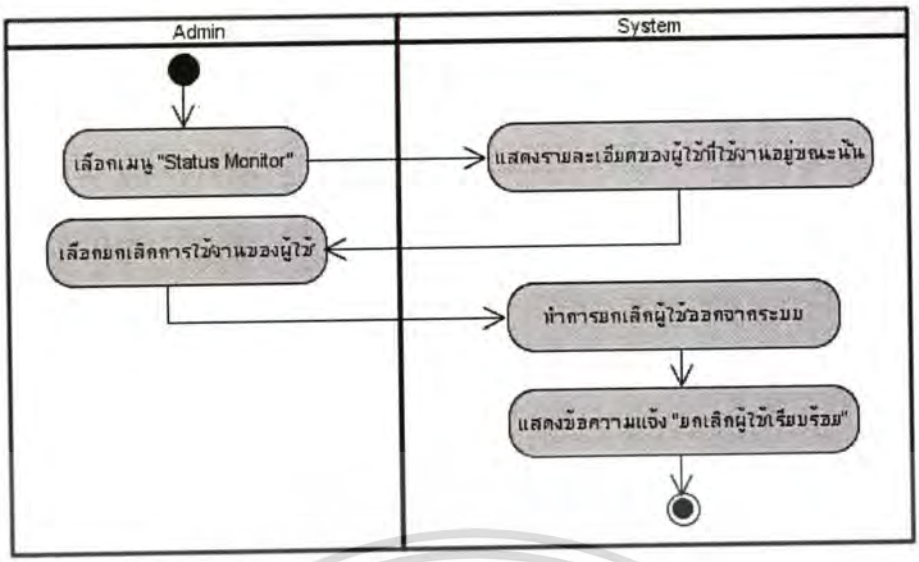
การทำงานปกติ :	**แอกเตอร์**	**ระบบ**
		5. ระบบทำการปรับปรุงค่าพารามิเตอร์ ตามค่าใหม่ที่ได้รับ และแสดงข้อความแจ้ง “ได้ทำการเปลี่ยนพารามิเตอร์สมบูรณ์”
เงื่อนไขหลังจบงาน :	ระบบปรับเปลี่ยนค่าพารามิเตอร์ ตามค่าที่ได้รับเข้ามาใหม่ และทำงานตามค่าใหม่ได้ถูกต้องสมบูรณ์เหมือนเดิม	
กรณีผิดปกติ :	4a. ค่าพารามิเตอร์ที่ใส่เข้ามาไม่ถูกต้อง : แสดงข้อความแจ้ง “ค่าพารามิเตอร์ ผิดพลาด” และกลับไปยังขั้นตอนที่ 3	



รูปที่ 4.7 แผนภาพกิจกรรมผู้ดูแลระบบแก้ไขพารามิเตอร์ระบบ

ตารางที่ 4.7 รายละเอียดของการแสดงสถานะปัจจุบันของระบบ

หมายเลขยูสเคส :	Usecase07	
ชื่อยูสเคส :	Status Monitor	
รายละเอียดโดยย่อ :	ฟังก์ชันสำหรับผู้ดูแลระบบใช้ดูสถานะปัจจุบันของระบบ เช่น ผู้ใช้ที่ใช้งานผ่านระบบอยู่ ณ ปัจจุบัน, ดีเฮสซีพี (DHCP) ได้แจกไอพีอะไรออกไปแล้วบ้าง	
ทริกเกอร์ :	เมื่อผู้ดูแลระบบมีความต้องทราบสถานะปัจจุบันของระบบ	
ผู้ใช้งานระบบ :	ผู้ดูแลระบบ (Admin)	
เงื่อนไขขั้นต้น :	ผู้ดูแลระบบจะต้องผ่านการล๊อกอินเข้าระบบก่อนเท่านั้น	
การทำงานปกติ :	**แอกเตอร์**	**ระบบ**
	1. ผู้ดูแลระบบเลือกเมนู "แสดงสถานะระบบ" 3. ผู้ดูแลระบบเลือก ยกเลิกการใช้งานของผู้ใช้ (Disconnect)	2. ระบบแสดงผู้ใช้ที่มีอยู่ในระบบ ณ ปัจจุบัน และแสดงไอพี ของเครื่องที่ผู้ใช้งาน 4. ระบบทำการลบข้อมูลผู้ใช้ออกจากฐานข้อมูล "Online User" และลบข้อมูลไอพีของผู้ใช้ออกจากระบบ 5. แสดงข้อความแจ้ง "ได้ทำการยกเลิกการใช้งานผู้ใช้ เรียบร้อยแล้ว"
เงื่อนไขหลังจบงาน :	ระบบสามารถแสดงข้อมูลปัจจุบันของผู้ใช้ที่ใช้ระบบ และสามารถยกเลิกการใช้ของผู้ใช้ได้	
กรณีผิดปกติ :	-	



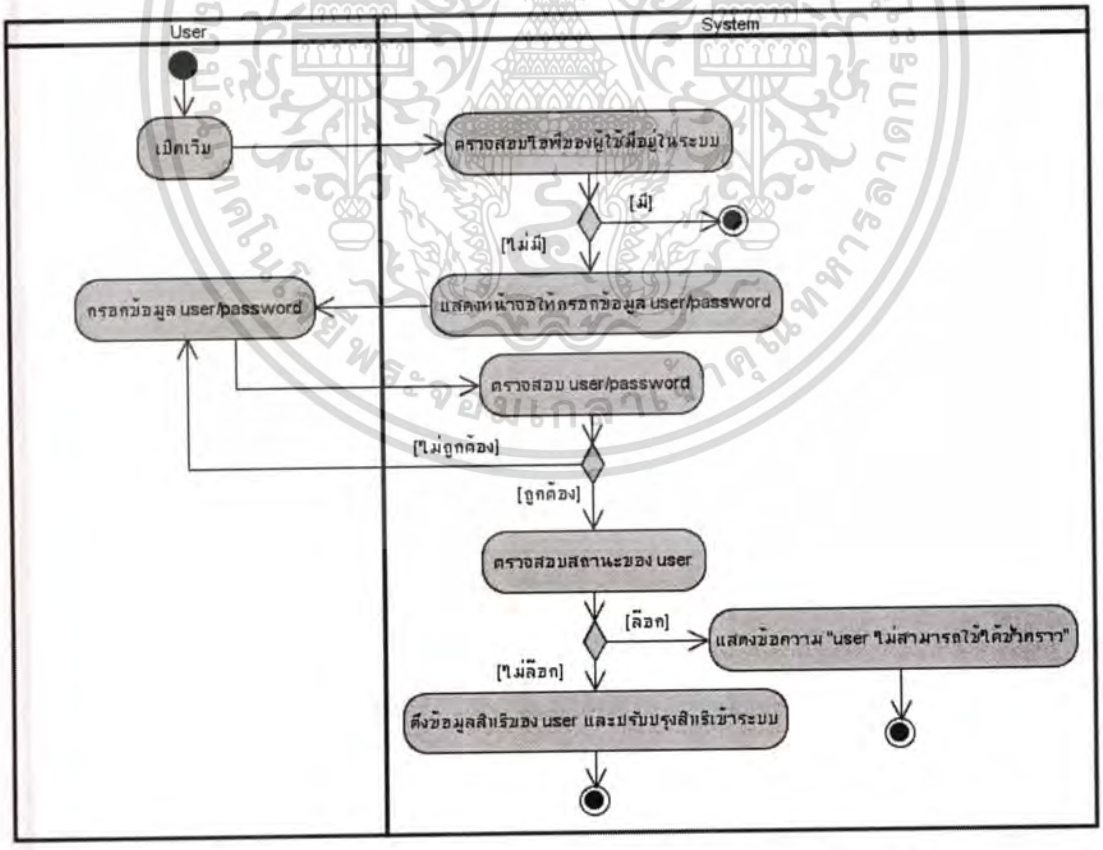
รูปที่ 4.8 แผนภาพกิจกรรมผู้ดูแลทำการตรวจดูสถานะระบบ

ตารางที่ 4.8 รายละเอียดของการตรวจสอบสิทธิผู้ใช้ก่อนใช้งานอินเทอร์เน็ต

หมายเลขยูสเคส :	Usecase08	
ชื่อยูสเคส :	Authentication to access internet	
รายละเอียดโดยย่อ :	ฟังก์ชันสำหรับผู้ใช้ เพื่อบังคับให้ผู้ใช้ทำการยืนยันตัวตนก่อนที่จะอนุญาติให้สามารถใช้งานอินเทอร์เน็ตได้	
ทริกเกอร์ :	เมื่อผู้ใช้พิมพ์ยูอาร์แอล (url) เพื่อใช้งานอินเทอร์เน็ต	
ผู้ใช้งานระบบ :	ผู้ใช้ (User)	
เงื่อนไขขั้นต้น :	ผู้ใช้ต้องมีการลงทะเบียนไว้ในระบบโดยผู้ดูแลระบบไว้ก่อนที่จะสามารถใช้งานได้	
การทำงานปกติ :	**แอกเตอร์**	**ระบบ**
	1. ผู้ใช้พิมพ์ยูอาร์แอล (url) เพื่อใช้งานอินเทอร์เน็ต	2. ระบบตรวจสอบยังไม่พบไอพีของผู้ใช้มีอยู่ในระบบ 3. ระบบแสดงหน้าจอให้ใส่ ชื่อผู้ใช้และรหัสผ่าน
	4. ผู้ใช้ใส่ชื่อผู้ใช้และรหัสผ่าน	5. ระบบตรวจสอบชื่อผู้ใช้และรหัสผ่านถูกต้อง

ตารางที่ 4.8 (ต่อ)

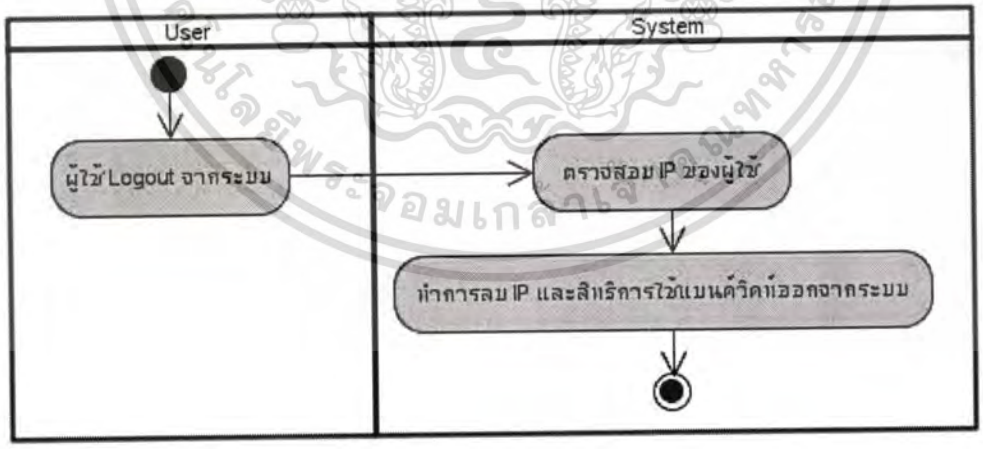
การทำงานปกติ :	**แอกเตอร์**	**ระบบ**
		6. ระบบตรวจสอบสถานะของผู้ใช้ไม่ถูกยกเลิกการใช้งานชั่วคราว 7. ระบบดึงข้อมูลรายละเอียดสิทธิของผู้ใช้ และทำการเพิ่มเข้าระบบ
เงื่อนไขหลังจบงาน :	ผู้ใช้สามารถใช้งานอินเทอร์เน็ตได้และได้สิทธิการใชงานแบนด์วิดท์ตามที่ได้กำหนดไว้	
กรณีผิดปกติ :	2a. ระบบตรวจสอบพบไอพีของผู้ใช้มีอยู่ในระบบแล้ว : ปล่อยผ่านแพคเกตให้สามารถใช้งานอินเทอร์เน็ตได้ 5a. ชื่อผู้ใช้หรือรหัสผ่านไม่ถูกต้อง : แสดงข้อความ “Invalided user or password ” และกลับไปยังขั้นตอนที่ 4 6a. ผู้ใช้ถูกยกเลิกการใช้งานชั่วคราว : แสดงข้อความ “User ไม่สามารถใช้งานได้ชั่วคราว โปรดติดต่อผู้ดูแลระบบ”	



รูปที่ 4.9 แผนภาพกิจกรรมตรวจสอบสิทธิผู้ใช้ก่อนใช้งานอินเทอร์เน็ต

ตารางที่ 4.9 รายละเอียดของการออกจากระบบของผู้ใช้

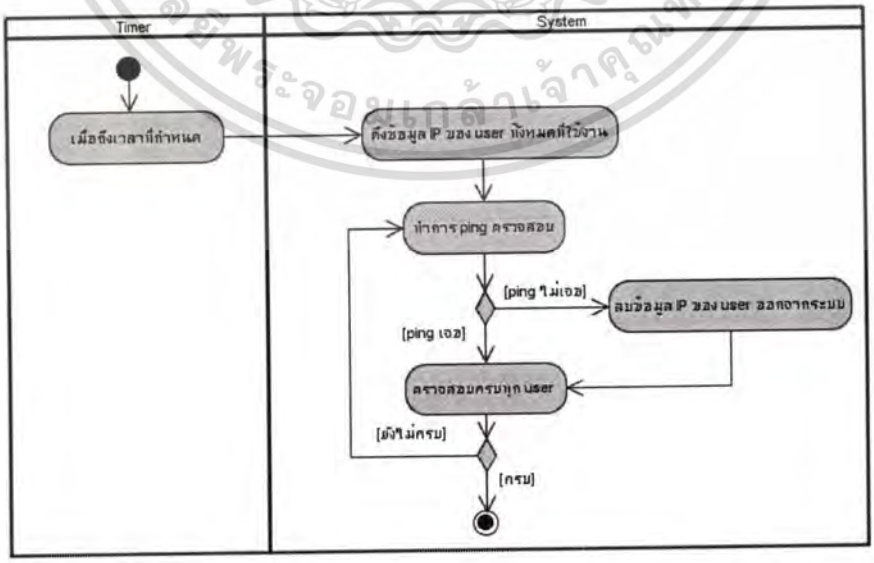
หมายเลขยูสเคส :	Usecase09	
ชื่อยูสเคส :	User Logout	
รายละเอียดโดยย่อ :	ฟังก์ชันสำหรับผู้ใช้ เมื่อต้องการออกจากระบบ โดยระบบจะทำการลบไอพีและสิทธิการใช้งานแบนด์วิดท์ของผู้ใช้ออกจากระบบ	
ทริกเกอร์ :	เมื่อผู้ใช้ต้องการยกเลิกการใช้งานอินเทอร์เน็ต	
ผู้ใช้งานระบบ :	ผู้ใช้ (User)	
เงื่อนไขขั้นต้น :	ผู้ใช้ต้องผ่านการล๊อกอินเข้าใช้ระบบแล้ว	
การทำงานปกติ :	**แอกเตอร์**	**ระบบ**
	1. ผู้ใช้เลือก “ออกจากระบบ”	2. ระบบตรวจสอบไอพีของผู้ใช้ 3. ระบบทำการลบไอพี และสิทธิการใช้งานแบนด์วิดท์ของผู้ใช้ออกจากระบบ
เงื่อนไขหลังจบงาน :	ระบบสามารถยกเลิกการใช้งานของผู้ใช้ได้	
กรณีผิดปกติ :	-	



รูปที่ 4.10 แผนภาพกิจกรรมการออกจากระบบของผู้ใช้

ตารางที่ 4.10 รายละเอียดของการตรวจสอบการอยู่ในระบบของผู้ใช้

หมายเลขยูสเคส :	Usecase10	
ชื่อยูสเคส :	User Keepalive	
รายละเอียดโดยย่อ :	ฟังก์ชันระบบใช้สำหรับตรวจสอบว่าผู้ใช้ยังใช้ระบบอยู่หรือไม่ โดยเมื่อถึงระยะเวลาที่กำหนด ระบบจะทำการ ping เช็คไอพีของผู้ใช้ทุกคนที่ออนไลน์อยู่ในระบบ ถ้าไม่เจอก็จะทำการลบไอพีและสิทธิของผู้ใช้นั้นออกจากระบบ	
ทริกเกอร์ :	เมื่อถึงระยะเวลาที่ระบบกำหนดไว้	
ผู้ใช้งานระบบ :	Timer	
เงื่อนไขขั้นต้น :	ระบบจะต้องมีผู้ใช้อยู่ในระบบอย่างน้อย 1 คน	
การทำงานปกติ :	**แอกเตอร์**	**ระบบ**
	1. เมื่อถึงระยะเวลาที่ระบบกำหนด	2. ดึงไอพีผู้ใช้ที่มีอยู่ในระบบ 3. ทำการ ping ตรวจสอบไม่พบ 4. ทำการลบไอพีและสิทธิผู้ใช้ออกจากระบบ
เงื่อนไขหลังจบงาน :	ระบบสามารถยกเลิกการใช้งานของผู้ใช้ได้ที่ไม่ได้ใช้งานแล้วได้	
กรณีผิดปกติ :	2a. ไม่มีไอพีผู้ใช้แล้ว : จบการทำงาน 3a. ping ตรวจสอบพบ : ไปทำขั้นตอนที่ 2 ใหม่เพื่อตรวจสอบคนต่อไป	



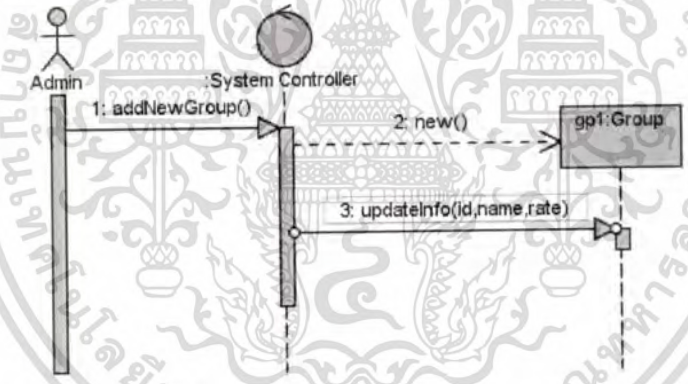
รูปที่ 4.11 แผนภาพกิจกรรมตรวจสอบการอยู่ในระบบของผู้ใช้

4.1.2 ซีเควนซ์ไดอะแกรม (Sequence Diagram)

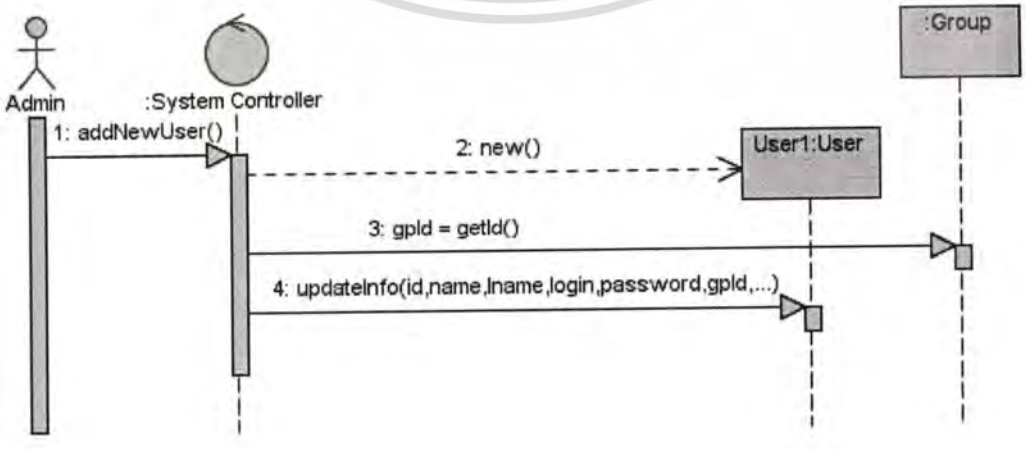
ซีเควนซ์ไดอะแกรมเป็นแผนภาพสำหรับแสดงการปฏิสัมพันธ์ (Interaction) กันระหว่างอ็อบเจกต์ ตามลำดับการทำงานของเหตุการณ์ที่เกิดขึ้น โดยแต่ละอ็อบเจกต์จะถูกกระตุ้นให้ทำงานผ่านทางข้อความ (Message) ซึ่งในโครงงานการจัดการและควบคุมข้อมูลในเครือข่ายนี้ ได้นำเอาซีเควนซ์ไดอะแกรมมาช่วยในการอธิบายลักษณะการทำงานร่วมกันของระบบย่อย ต่างๆที่ช่วยทำให้ระบบการจัดการและควบคุมข้อมูลในเครือข่ายสามารถทำงานได้ ซึ่งซีเควนซ์ไดอะแกรมนี้จะอธิบายการทำงานของระบบย่อยที่สำคัญดังนี้

1. การจัดการกลุ่มผู้ใช้ (Group) และผู้ใช้ (User)

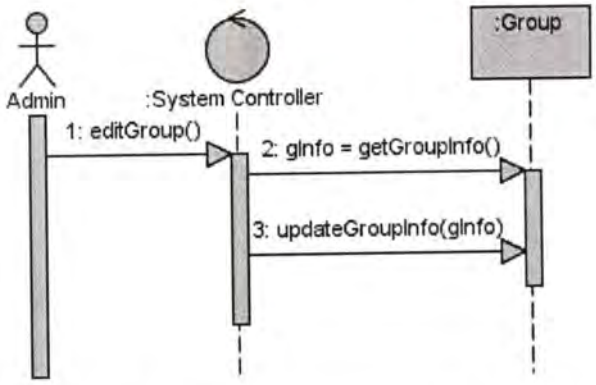
การทำงานของระบบในส่วนของการจัดการกลุ่มผู้ใช้และผู้ใช้ จะเป็นการสร้างกลุ่มผู้ใช้และผู้ใช้ขึ้นมาโดยมีการกำหนดสิทธิในการที่จะสามารถใช้แบนด์วิดท์ต่อคนได้เท่าไหร่ และผู้ใช้ทุกคนในกลุ่มผู้ใช้จะสามารถใช้ได้ไม่เกินเท่าไหร่ ซึ่งในส่วนจัดการนี้มีอยู่ 2 ส่วนคือการสร้างและการแก้ไข ซึ่งการสร้างและการแก้ไขนั้นทำโดยผู้ดูแลระบบ (Admin) โดยจะแสดงในรูปที่ 4.12 - 4.15



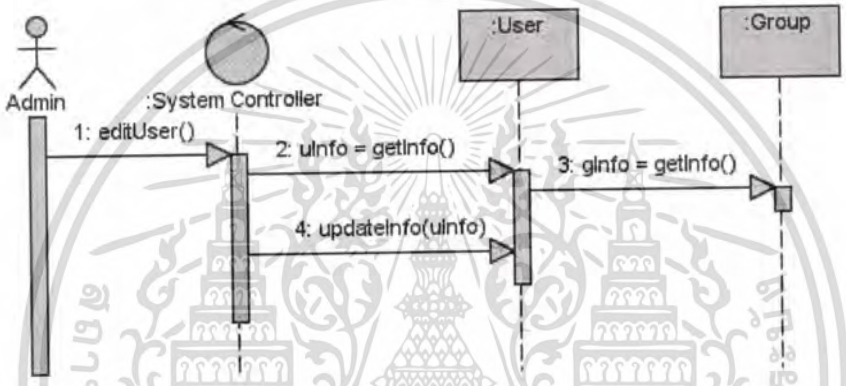
รูปที่ 4.12 ซีเควนซ์ไดอะแกรมของการสร้างกลุ่มผู้ใช้



รูปที่ 4.13 ซีเควนซ์ไดอะแกรมของการสร้างผู้ใช้



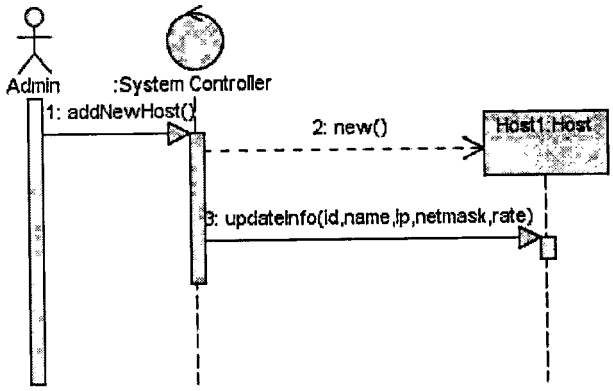
รูปที่ 4.14 ซีเควนซ์ไดอะแกรมของการแก้ไขกลุ่มผู้ใช้



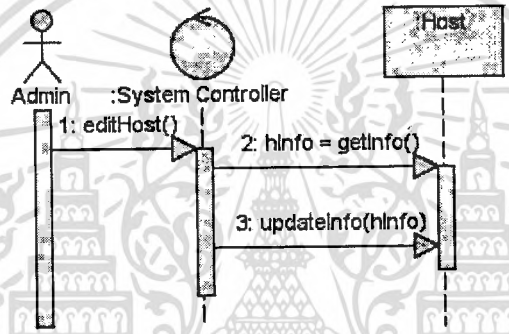
รูปที่ 4.15 ซีเควนซ์ไดอะแกรมของการแก้ไขผู้ใช้

2. การจัดการโฮสท์ (Host) และเน็ตเวิร์ค (Network)

การทำงานของระบบในส่วนของการจัดการโฮสท์และเน็ตเวิร์คนั้น จะเป็นการสร้างโฮสท์และเน็ตเวิร์ค ซึ่งโฮสท์เป็นการเจาะจงลงไปว่าเป็นเครื่องคอมพิวเตอร์ เครื่องใดเครื่องหนึ่งซึ่งค่าเน็ตมาร์คจะมีค่าเป็น /32 และเน็ตเวิร์คจะเป็นกลุ่มของเครื่องคอมพิวเตอร์ ซึ่งจะมีค่าเน็ตมาร์คน้อยกว่า 32 ซึ่งโฮสท์และเน็ตเวิร์คนั้นที่สร้างขึ้นนั้นจะถูกเพิ่มเข้า Policy Controller อัตโนมัติเมื่อระบบเริ่มทำงาน จึงทำให้โฮสท์และเน็ตเวิร์ค สามารถใช้งานอินเทอร์เน็ตได้โดยที่ไม่ต้องมีการตรวจสอบสิทธิ ซึ่งโดยทั่วไปแล้วกลุ่มของโฮสท์และเน็ตเวิร์ค นั้นควรเป็นเซิร์ฟเวอร์ เท่านั้น ซึ่งในส่วนจัดการนี้มีอยู่ 2 ส่วนคือการสร้างและการแก้ไข ซึ่งการสร้างและการแก้ไขนั้นทำโดยผู้ดูแลระบบ (Admin) โดยจะแสดงในรูปที่ 4.16 - 4.17



รูปที่ 4.16 ซีเควนซ์ไดอะแกรมของการสร้างโฮสท์



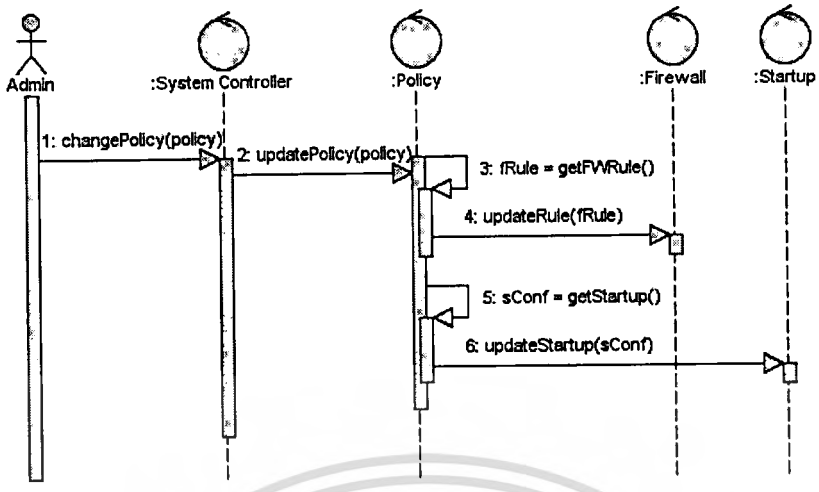
รูปที่ 4.17 ซีเควนซ์ไดอะแกรมของการแก้ไขโฮสท์

3. การจัดการนโยบาย (Policy)

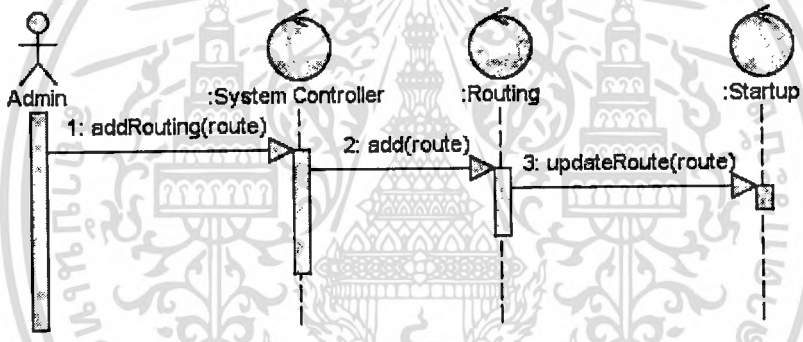
การทำงานของระบบในส่วนของการจัดการนโยบายนั้น เป็นการปรับเปลี่ยนวิธีการจัดการการใช้แบนด์วิดท์ตามนโยบายที่ได้กำหนดไว้ก่อนแล้ว จึงมีแต่ขั้นตอนของการเปลี่ยนนโยบายเท่านั้น ซึ่งการการเปลี่ยนนั้นทำโดยผู้ดูแลระบบ (Admin) โดยจะแสดงในรูปที่ 4.18

4. การจัดการเราท์ติง (Routing)

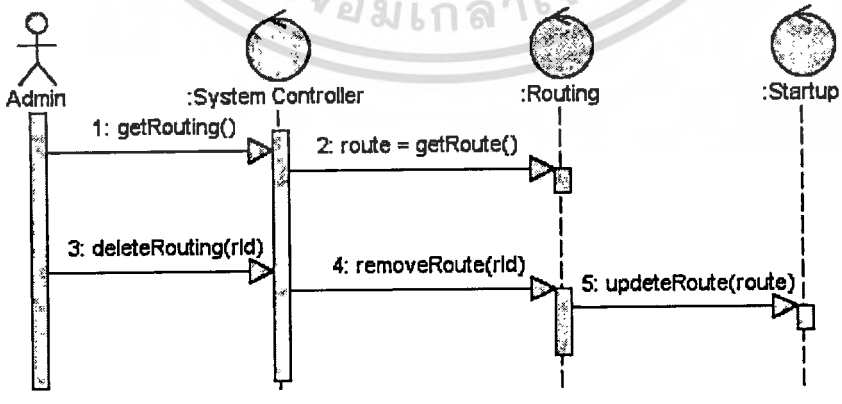
การทำงานของระบบในส่วนของการจัดการเราท์ติงนั้นจะเป็นการสร้างเราท์ติงเพื่อเพิ่มเส้นทางในการส่งแพคเกตให้เพิ่มขึ้น และการลบเราท์ติงในกรณีที่ไม่ต้องการใช้ ซึ่งการการเปลี่ยนนั้นทำโดยผู้ดูแลระบบ (Admin) โดยจะแสดงในรูปที่ 4.19 – 4.20



รูปที่ 4.18 ซีเควนซ์ไดอะแกรมของการจัดการนโยบาย (Policy)



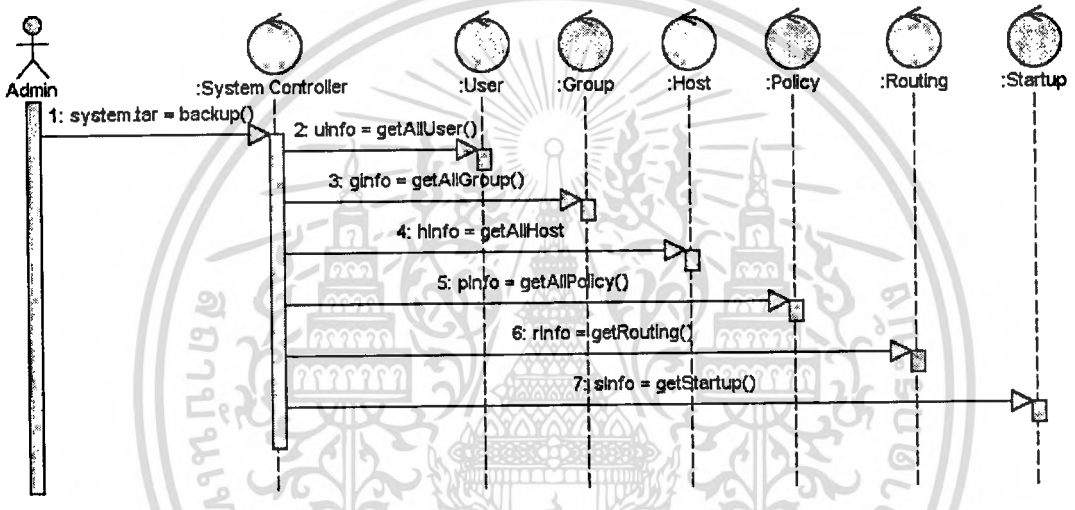
รูปที่ 4.19 ซีเควนซ์ไดอะแกรมของการเพิ่มเราท์ติง



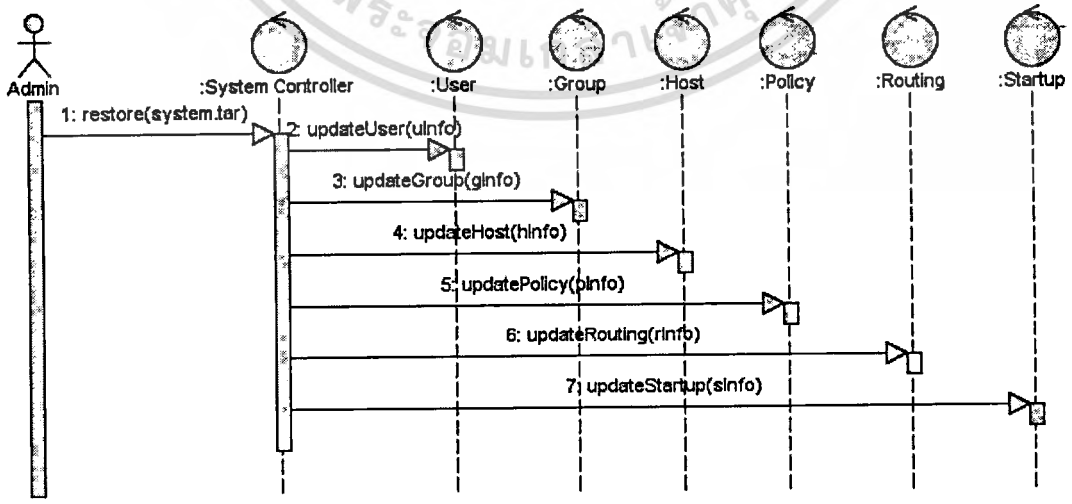
รูปที่ 4.20 ซีเควนซ์ไดอะแกรมของการลบเราท์ติง

5. การจัดการสำรองและกลับคืนคอนฟิค (Backup and Restore Configure)

การทำงานของระบบในส่วนของการสำรองและกลับคืนคอนฟิคนั้น ทำโดยผู้ดูแลระบบ (Admin) เมื่อผู้ดูแลระบบสั่งสำรองคอนฟิค (Backup Configure) System Controller จะทำการร้องขอไปยัง ระบบย่อยต่างเพื่อให้ส่งค่าที่จัดเก็บไว้ทั้งหมดออกมา แล้วทำการรวมไว้ในไฟล์ system.tar และในลักษณะเดียวกันในการกลับคืนคอนฟิค (Restore Configure) ผู้ดูแลระบบจะส่งคอนฟิคไฟล์ที่มีอยู่ให้ System Controller แล้ว System Controller จะแยกค่าคอนฟิคของแต่ละระบบย่อยออกมาและส่งไปทำการปรับค่าระบบย่อยทั้งหมด โดยจะแสดงในรูปที่ 4.21 – 4.22



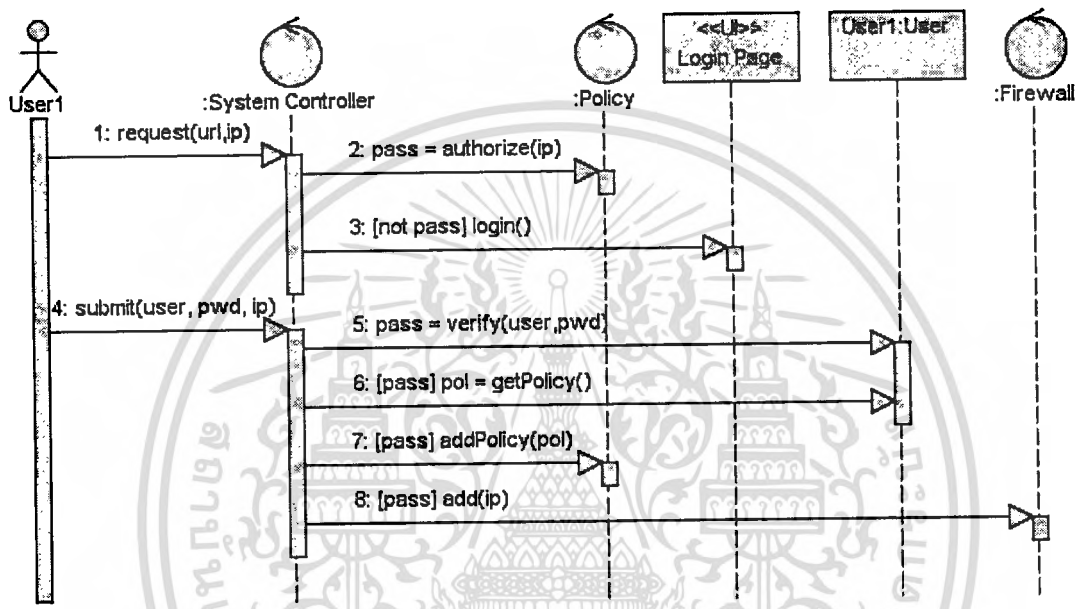
รูปที่ 4.21 ซีเควนซ์ไดอะแกรมของการสำรองคอนฟิค (Backup Configure)



รูปที่ 4.22 ซีเควนซ์ไดอะแกรมของการกลับคืนคอนฟิค (Restore Configure)

6. การตรวจสอบผู้ใช้ก่อนใช้งานอินเทอร์เน็ต (Authentication to Access Internet)

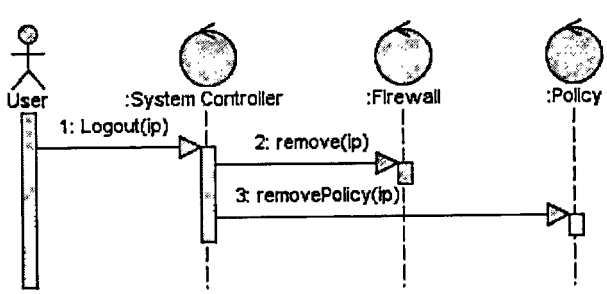
การทำงานของระบบในส่วนของการตรวจสอบผู้ใช้ก่อนใช้งานอินเทอร์เน็ต นั้นระบบจะคอยตรวจสอบการร้องขอเว็บของผู้ใช้ที่ทีซีพี (TCP) พอร์ต 80 ถ้าไอพีของผู้ใช้นั้นยังไม่มีอยู่ในระบบแสดงว่าผู้ใช้นั้นยังไม่ได้ผ่านการล็อกอิน ระบบจะทำการส่งหน้าจอล็อกอินมาให้ผู้ใช้ และเมื่อผู้ใช้ล็อกอินผ่านระบบจะทำการเพิ่มไอพีผู้ใช้เข้าระบบและเพิ่มสิทธิการใช้แบนด์วิทด์ให้กับผู้ใช้ โดยจะแสดงดังรูปที่ 4.23



รูปที่ 4.23 ซีเควนซ์ไดอะแกรมของการตรวจสอบผู้ใช้ก่อนใช้งานอินเทอร์เน็ต

7. การล็อกเอาท์ของผู้ใช้ (User Logout)

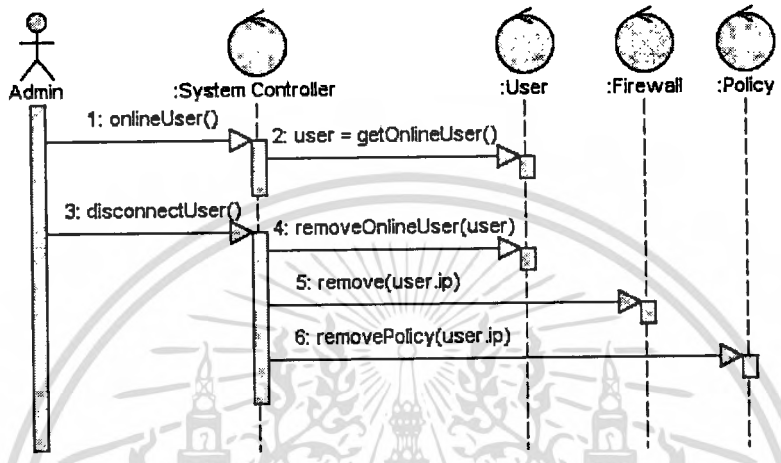
หลังจากที่ผู้ใช้ทำการล็อกอินเข้าใช้ระบบแล้ว เมื่อต้องการที่จะออกจากระบบก็จะต้องทำการล็อกเอาท์ เพื่อที่ระบบจะได้ลบข้อมูลต่างๆ ที่สามารถทำให้เครื่องของผู้ใช้ใช้งานอินเทอร์เน็ตได้ออกจากระบบ โดยข้อมูลที่ถูกลบออกมี ข้อมูลการเอาไอพีออกจากไฟร์วอลล์ และเอาสิทธิการใช้แบนด์วิดท์ออก โดยจะแสดงดังรูปที่ 4.24



รูปที่ 4.24 ซีเควนซ์ไดอะแกรมของผู้ใช้ล็อกเอาท์

8. การแสดงสถานะของระบบ (Status Monitor)

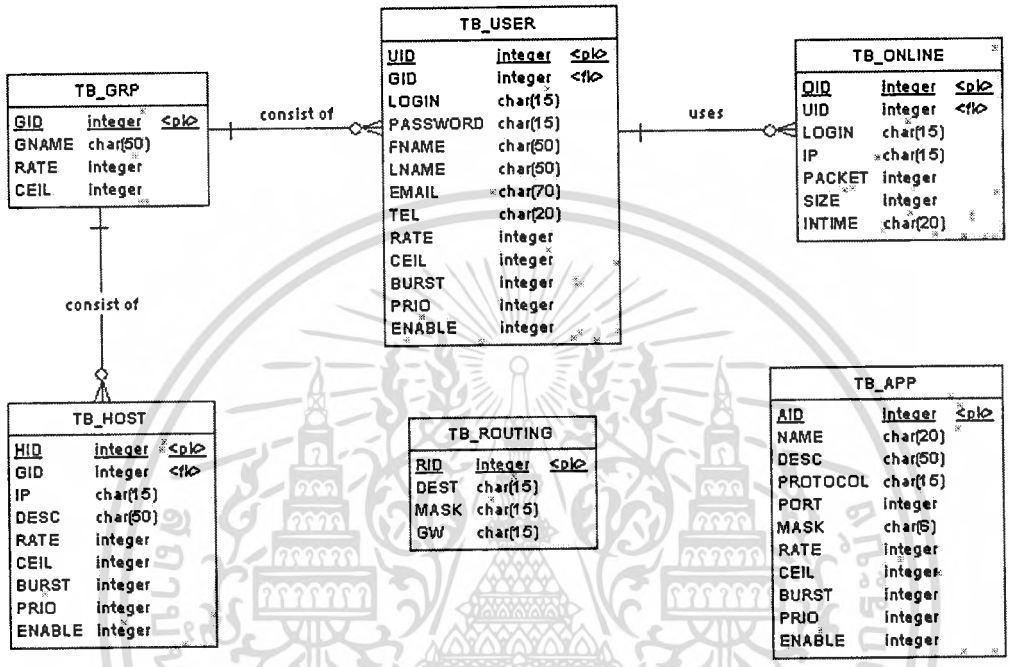
การแสดงสถานะของระบบนั้นระบบสามารถแสดงสถานะปัจจุบันเช่น ไอพีของอินเทอร์เฟส, ไอพีของเครื่องลูกข่ายที่ DHCP Server แจกไป และผู้ใช้ที่ออนไลน์อยู่ในระบบซึ่งซีเควนซ์ไดอะแกรมนี้จะเน้นที่การแสดงผู้ใช้ที่ออนไลน์ และผู้ดูแลระบบสามารถสั่งตัดผู้ใช้ออกจากระบบได้โดยตรงจากส่วนของการแสดงสถานะของระบบนี้ โดยจะแสดงดังรูปที่ 4.25



รูปที่ 4.25 ซีเควนซ์ไดอะแกรมของสั่งตัดผู้ใช้ออกจากระบบโดยผู้ดูแลระบบ

4.1.3 แผนภาพความสัมพันธ์ระหว่างเอนทิตี (E-R Diagram)

ในส่วนของการออกแบบโครงสร้างฐานข้อมูลของระบบการจัดการข้อมูลในเครือข่ายนั้น ได้ออกแบบตามหลักของฐานข้อมูลเชิงสัมพันธ์ โดยความสัมพันธ์ของเอนทิตี แสดงได้ดังแผนภาพความสัมพันธ์ระหว่างเอนทิตีโดยมีเอนทิตีในระบบทั้งสิ้น 7 เอนทิตี ซึ่งแสดงเป็นแผนภาพดังรูปที่ 4.26



รูปที่ 4.26 แผนผังแสดงความสัมพันธ์ระหว่างเอนทิตีของระบบการจัดการข้อมูลในเครือข่าย

4.1.4 พจนานุกรมข้อมูล

ตารางที่ 4.11 ตาราง TB_USER

ชื่อแอททริบิวต์	คำอธิบาย	ชนิดข้อมูล	คีย์	ตารางอ้างอิง
UID	รหัสของผู้ใช้ นำไปใช้การสร้าง class ลูก	integer	PK	
GID	รหัสของกลุ่ม นำไปใช้การสร้าง class แม่ (Parent Class)	integer	FK	TB_GRP
LOGIN	ชื่อผู้ใช้สำหรับ Login	char(50)		
PASSWORD	รหัสผ่านของผู้ใช้	char(10)		
FNAME	ชื่อของผู้ใช้	char(50)		
LNAME	นามสกุลผู้ใช้	char(50)		
TEL	เบอร์โทรติดต่อผู้ใช้	char(20)		
EMAIL	อีเมลติดต่อผู้ใช้	char(70)		
RATE	แบนด์วิดท์ที่ให้	integer		
CEIL	แบนด์วิดท์ที่สามารถใช้ได้สูงสุด	integer		
BURST	ค่าที่อนุญาติให้ส่งข้อมูลเพิ่มในช่วงเวลาสั้นๆ	integer		
PRIO	กำหนดลำดับความสำคัญของ class โดยค่าน้อย priority สูง	integer		
ENABLE	สถานะ 1=ใช้งานได้, 0=หยุดใช้งานชั่วคราว	integer		

ตารางที่ 4.12 ตาราง TB_HOST

ชื่อแอททริบิวต์	คำอธิบาย	ชนิดข้อมูล	คีย์	ตารางอ้างอิง
HID	รหัสของเครื่อง นำไปใช้การสร้าง class ลูก	integer	PK	
GID	รหัสของกลุ่ม นำไปใช้การสร้าง class แม่ (Parent Class)	integer	FK	TB_GRP

ตารางที่ 4.12 (ต่อ)

ชื่อแอททริบิวต์	คำอธิบาย	ชนิดข้อมูล	คีย์	ตารางอ้างอิง
IP	ไอพี ของเครื่อง	char(15)		
DESC	ชื่อหรือข้อมูลอธิบายถึงโฮสท์	char(50)		
RATE	แบนด์วิดท์ที่ให้	integer		
CEIL	แบนด์วิดท์ที่สามารถใช้ได้สูงสุด	integer		
BURST	ค่าที่อนุญาติให้ส่งข้อมูลเพิ่มในช่วงเวลาสั้นๆ	integer		
PRIO	กำหนดลำดับความสำคัญของ class โดยค่าน้อย priority สูง	integer		
ENABLE	สถานะ 1=ใช้งานได้, 0=หยุดใช้งานชั่วคราว	integer		

ตารางที่ 4.13 ตาราง TB_GRP

ชื่อแอททริบิวต์	คำอธิบาย	ชนิดข้อมูล	คีย์	ตารางอ้างอิง
GID	รหัสของกลุ่ม นำไปใช้การสร้าง class แม่ (Parent Class)	integer	PK	
GNAME	ชื่อของกลุ่ม	char(50)		
RATE	แบนด์วิดท์ที่ให้กลุ่ม	integer		
CEIL	แบนด์วิดท์ที่สามารถใช้ได้สูงสุด	integer		

ตารางที่ 4.14 ตารางTB_ONLINE

ชื่อแอททริบิวต์	คำอธิบาย	ชนิดข้อมูล	คีย์	ตารางอ้างอิง
ID	ลำดับของผู้ใช้	integer	PK	
UID	รหัสของผู้ใช้	integer	FK	USER
IP	ไอพีของเครื่องที่ผู้ใช้ ใช้งานอยู่	char(15)		

ตารางที่ 4.14 (ต่อ)

ชื่อแอททริบิวต์	คำอธิบาย	ชนิดข้อมูล	คีย์	ตารางอ้างอิง
LOGIN	ชื่อล็อกอินของผู้ใช้	char(15)		
PACKET	จำนวนแพคเกตที่ผู้ใช้ส่งผ่านระบบ	integer		
SIZE	จำนวนข้อมูลที่ผู้ใช้ส่งผ่านระบบหน่วยเป็น Byte	integer		
INTIME	เวลาเริ่มต้นที่ผู้ใช้ล็อกอิน	char(20)		

ตารางที่ 4.15 ตาราง TB_APP

ชื่อแอททริบิวต์	คำอธิบาย	ชนิดข้อมูล	คีย์	ตารางอ้างอิง
AID	รหัสลำดับของแอพพลิเคชัน	integer	PK	
NAME	ชื่อของแอพพลิเคชัน	char(20)		
DESC	ชื่อหรือข้อมูลที่ใช้อธิบายบรรทัดข้อมูลนี้	char(50)		
PROTOCOL	โปรโตคอลที่แอพพลิเคชันใช้ (tcp,udp,icmp)	char(15)		
PORT	พอร์ท	integer		
MASK	ค่าตัวเลขที่ใช้กัน PORT เพื่อกำหนดกลุ่มตัวเลขของ PORT	char(6)		
RATE	แบนด์วิดท์ที่ให้	integer		
CEIL	แบนด์วิดท์ที่สามารถใช้ได้สูงสุด	integer		
BURST	ค่าที่อนุญาติให้ส่งข้อมูลเพิ่มในช่วงเวลาสั้นๆ	integer		
PRIO	กำหนดลำดับความสำคัญของ class โดยค่าน้อย priority สูง	integer		
ENABLE	สถานะ 1=ใช้งานได้, 0=หยุดใช้งานชั่วคราว	integer		

ตารางที่ 4.16 ตาราง TB_ROUTING

ชื่อแอททริบิวต์	คำอธิบาย	ชนิดข้อมูล	คีย์	ตารางอ้างอิง
RID	รหัสอ้างอิงใน Routing Table	integer	PK	
DES	Network ปลายทาง	char(15)		
MASK	Subnet mask ของ Network ปลายทาง	char(15)		
GW	Gateway หมายเลข IP ที่จะส่งข้อมูลไปยัง Network ข้างต้น	char(15)		

บทที่ 5
การออกแบบหน้าจอการทำงาน

โครงการพัฒนาอุปกรณ์ในการจัดการและควบคุมข้อมูลในเครือข่ายนี้เนื่องจากระบบปฏิบัติการที่ใช้เป็นลินุกซ์ซึ่งมีลักษณะเป็น CLI (Command Line Interface) แต่เพื่อให้ผู้ใช้สามารถใช้งานได้ง่าย จึงได้พัฒนาส่วนต่อประสานกับผู้ใช้ขึ้นในรูปแบบของเว็บ แอพพลิเคชัน ซึ่งมีข้อดีอยู่ที่ใช้งานง่าย และผู้ใช้ระบบสามารถใช้งานผ่านทางเว็บเบราเซอร์ได้โดยไม่ต้องติดตั้ง ซอฟท์แวร์เพิ่มเติม ดังนั้นในส่วนของการออกแบบหน้าจอของระบบนั้นจึงเป็นการออกแบบหน้าจอเว็บเพจโดยระบบโดยโครงสร้างของเมนูของผู้ดูแลระบบ และสำหรับผู้ใช้เพื่อตรวจสอบสิทธิการใช้งานในส่วนของผู้ดูและระบบนั้นแสดงดังรูปที่ 5.1 และระบบได้แบ่งการทำงานออกเป็น 6 ดังนี้

1. **การแสดงข้อมูลระบบ** ในส่วนนี้จะเป็นหน้าแรกที่จะแสดงเพื่อผู้ดูแลระบบสามารถตรวจสอบสิทธิผ่านเข้ามาได้ โดยระบบจะแสดงข้อมูลเบื้องต้น เช่นระยะเวลาที่ระบบเปิดทำงานมา เราท์ติงของระบบ และผู้ใช้ที่ใช้งานผ่านระบบอยู่ขณะนั้น
2. **การจัดการข้อมูลของระบบ** เป็นหน้าจอการทำงานสำหรับผู้ดูแลระบบใช้กำหนดค่าเริ่มต้นต่างที่จำเป็นสำหรับระบบ เช่น ไอพีของอินเตอร์เฟส, การกำหนดค่าพารามิเตอร์ต่างๆของดีเฮสซีพีเซิร์ฟเวอร์ (DHCP Server), เปลี่ยนรหัสผ่าน, การสำรองและคืนค่าของข้อมูลระบบ
3. **การกำหนดนโยบายการใช้แบนด์วิดท์** จะเป็นส่วนหลักของระบบที่จะให้ผู้ดูแลระบบกำหนดได้ว่าการกำหนดนโยบายการใช้งานแบนด์วิดท์เป็นแบบไหน โดยจะสามารถจัดการกับรายละเอียดของพารามิเตอร์ของการกำหนดนโยบายในแต่ละแบบ เช่น การจัดการผู้ใช้ กลุ่มผู้ใช้ โฮสท์ และแอพพลิเคชัน
4. **การกำหนดผู้ดูแลระบบ** เป็นหน้าจอการทำงานสำหรับผู้ดูแลระบบสำหรับการจัดการระบบ เช่นเปลี่ยนรหัสผ่าน การกำหนดให้สามารถเข้ามาจัดการระบบผ่านทางไหนได้บ้าง การสำรองข้อมูลของระบบ และการสั่งเริ่มระบบใหม่ (Reboot)
5. **เครื่องมือที่ใช้ทดสอบการเชื่อมต่อของระบบ** เป็นส่วนของหน้าจอสำหรับผู้ดูแลระบบใช้ตรวจสอบการเชื่อมต่อระบบโดยมีการ ping และ traceroute
6. **ออกจากระบบ** เป็นส่วนสำหรับออกจากระบบเมื่อผู้ดูแลระบบทำการกำหนดค่าระบบเสร็จ

System Status
⊟ System Configure
- System
- Interface Outside
- Interface Inside
- DHCP
- Change Password
- Time
- Backup/Restore
- Factory Default
- Reboot

⊟ Policy
- Group
- User
- Host
- Application
- Policy

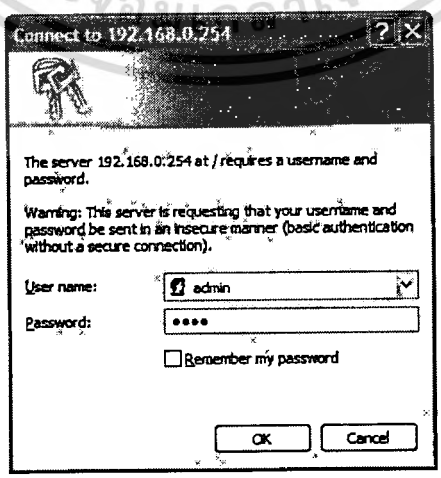
⊟ Routing/NAT
- NAT
- Static Route

⊟ Tool
- Ping
- Traceroute

Logout

รูปที่ 5.1 ผังแสดงโครงสร้างเมนูของผู้ดูแลระบบ

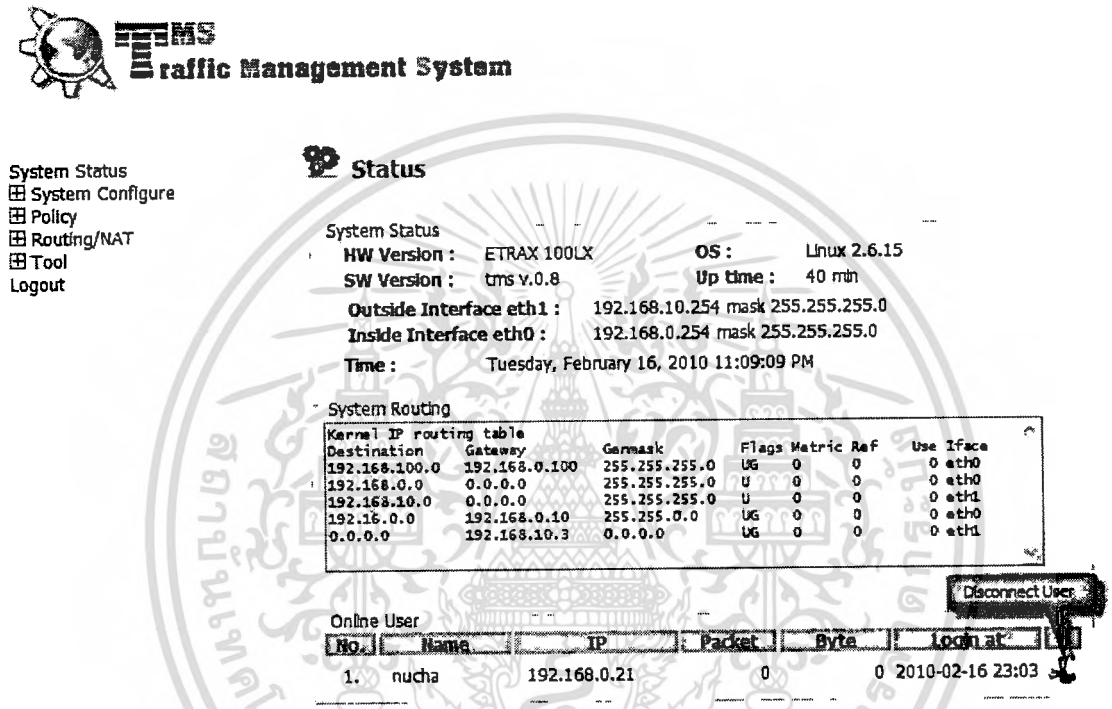
สำหรับหน้าจอการทำงานของระบบในส่วนของผู้ดูแลระบบนั้น ก่อนที่ผู้ดูแลระบบจะสามารถเข้าไปใช้งานได้นั้น จะเริ่มต้นจากหน้าจอล็อกอินเพื่อตรวจสอบสิทธิ์ในการเข้าใช้ระบบของผู้ดูแลระบบดังรูปที่ 5.2 โดยหลังจากที่ผู้ดูแลระบบผ่านการตรวจสอบสิทธิ์ก็จะสามารถเข้าไปใช้งานระบบได้โดยมีรายละเอียดของหน้าจอดังที่จะได้กล่าวต่อไป



รูปที่ 5.2 หน้าจอล็อกอินของผู้ดูแลระบบ

5.1 เมนูหลัก System Status

ในส่วนของเมนูหลัก System Status นี้หน้าจอแรกที่เมื่อผู้ดูแลระบบผ่านการตรวจสอบสิทธิเข้ามาได้แล้ว จะแสดงหน้าจอนี้ให้ผู้ดูแลระบบทราบข้อมูลเบื้องต้นของระบบ โดยจะแบ่งเป็นส่วนย่อยๆ 3 ส่วนคือ ข้อมูลระบบ, เราท์ติงของระบบ และผู้ใช้ที่ใช้งานผ่านระบบ โดยในส่วนที่แสดงข้อมูลผู้ใช้นี้ ผู้ดูแลระบบสามารถตัดการเชื่อมต่อของผู้ใช้ได้ ดังแสดงในรูปที่ 5.3



รูปที่ 5.3 หน้าจอแสดง Status ของระบบ

5.2 เมนูหลัก System Configure

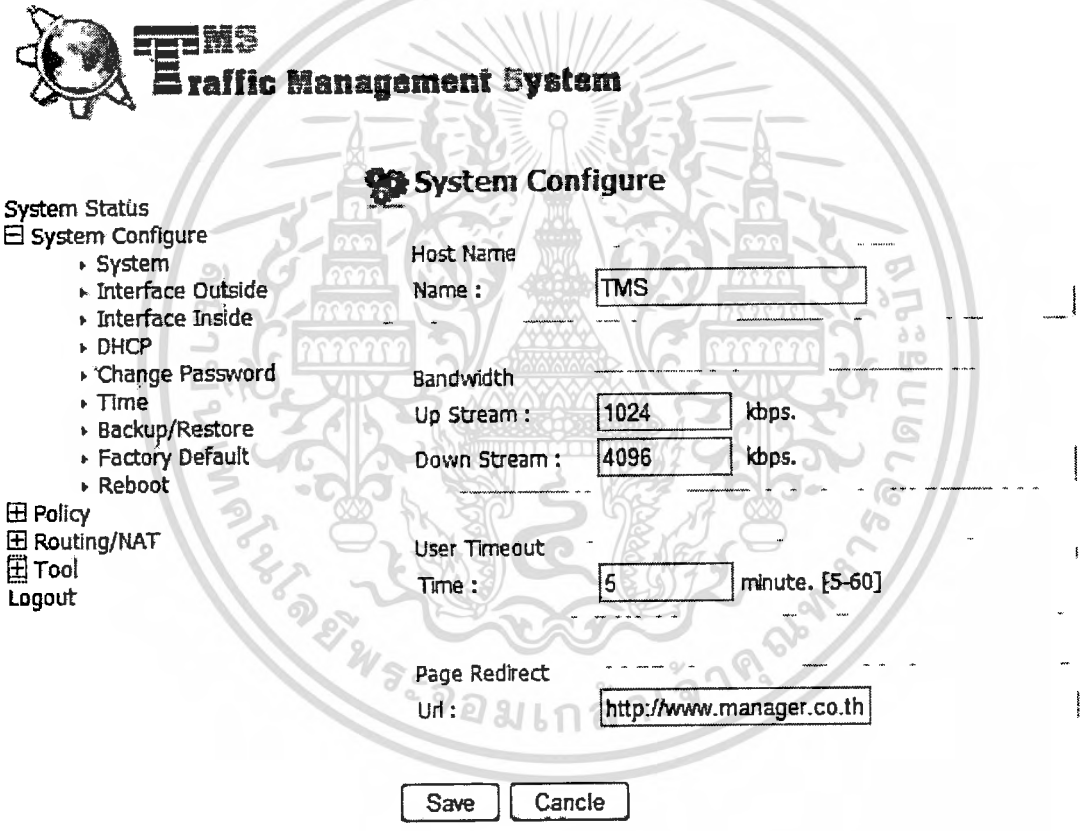
ในส่วนของเมนูหลัก System Configure นี้จะเป็นส่วนหลักในการกำหนดค่าต่างๆที่จำเป็นสำหรับระบบโดยจะมีเมนูย่อยที่ทำงานอยู่ภายใต้ทั้งหมด 9 เมนูดังนี้

5.2.1 เมนูย่อย System

ในส่วนหน้าจอของเมนูย่อย System จะแสดงดังรูปที่ 5.4 และประกอบไปด้วย 4 ส่วนดังนี้

- Host Name : จะเป็นส่วนที่กำหนดชื่อของอุปกรณ์

- Bandwidth : จะเป็นส่วนที่กำหนดค่าแบนด์วิดท์ให้กับระบบ เพื่อระบบจะได้นำไปเป็นค่าอ้างอิงในการจัดการแบนด์วิดท์ โดยจะมีการกำหนด 2 ค่าคือแบนด์วิดท์ขาขึ้น (Up Stream) และแบนด์วิดท์ขาลง (Down Stream) โดยมีหน่วยเป็น kbps (kilo bit per second)
- User Timeout : เป็นการกำหนดค่าให้ระบบทราบว่าถ้าผู้ใช้ไม่ติดต่อมายังระบบนานเท่าไหร่ ถึงจะให้ตัดการเชื่อมต่อของผู้ใช้นั้นออกไป โดยค่านั้นจะกำหนดได้ตั้งแต่ 5-60 นาที
- Page Redirect : จะเป็นการระบุค่าของเว็บเพ็จที่ต้องการให้แสดงเมื่อผู้ใช้ระบบทำการตรวจสอบสิทธิการใช้งานระบบผ่าน ระบบจะแสดงหน้าเว็บที่กำหนดออกมา



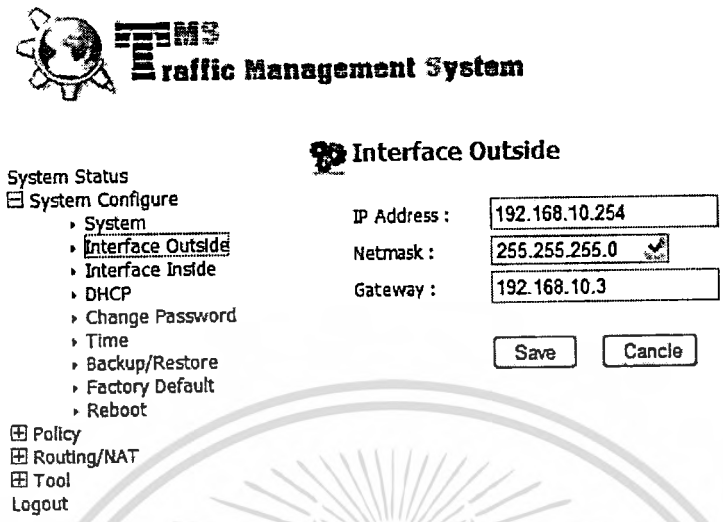
รูปที่ 5.4 หน้าจอเมนูย่อย System Configure

5.2.2 เมนูย่อย Interface Outside

ในส่วนหน้าจอของเมนูย่อย Interface Outside จะเป็นการกำหนดค่าของไอพีแอดเดรสของอินเตอร์เฟสที่เชื่อมต่อออกไปยังอินเทอร์เน็ตดังรูปที่ 5.5 โดยมีค่าพารามิเตอร์ต่างดังนี้

- IP Address : ค่าไอพีแอดเดรสที่จะกำหนดให้ระบบ
- Netmask : ค่าเน็ตมาร์ค

- Gateway : ค่าไอพีแอดเดรสของเราท์เตอร์ที่เชื่อมต่ออินเทอร์เน็ต

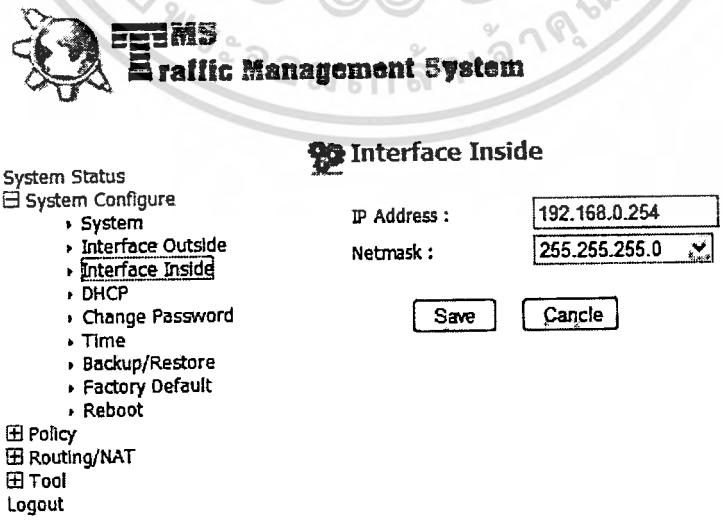


รูปที่ 5.5 หน้าจอเมนูย่อย Interface Outside

5.2.3 เมนูย่อย Interface Inside

ในส่วนหน้าจอของเมนูย่อย Interface Inside จะเป็นการกำหนดค่าของไอพีแอดเดรสของอินเตอร์เฟสที่เชื่อมต่อกับเครือข่ายภายในที่ต้องการควบคุมการใช้งานแบนด์วิดท์ดังรูปที่ 5.6 โดยมีค่าพารามิเตอร์ต่างดังนี้

- IP Address : ค่าไอพีแอดเดรสที่จะกำหนดให้ระบบ
- Netmask : ค่าเน็ตมาร์ค

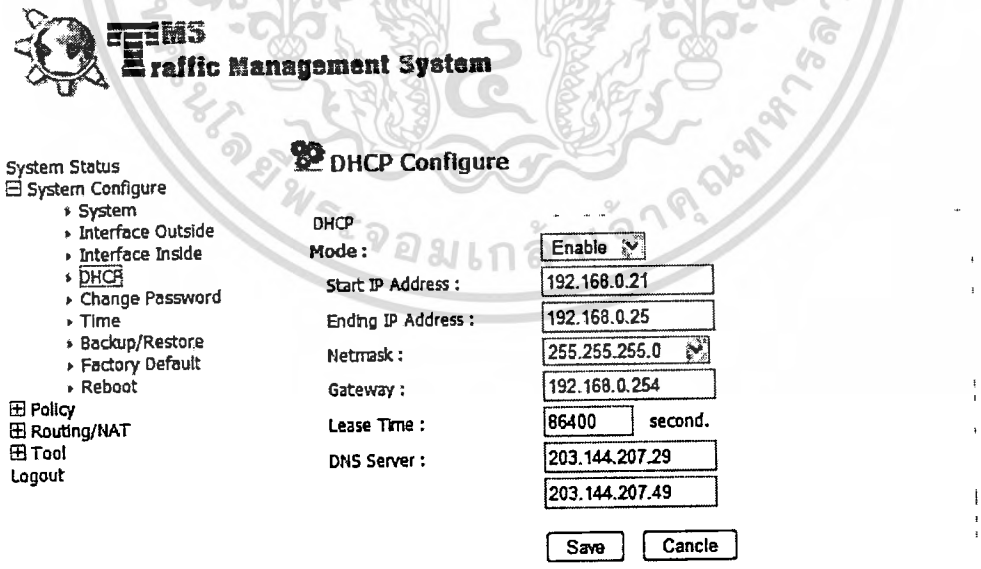


รูปที่ 5.6 หน้าจอเมนูย่อย Interface Inside

5.2.4 เมนูย่อย DHCP

ในส่วนหน้าจอของเมนูย่อย DHCP จะเป็นการกำหนดพารามิเตอร์ของบริการ DHCP Server ระบบซึ่ง DHCP จะเป็นบริการที่ช่วยกำหนดไอพีแอดเดรสให้กับเครื่องลูกข่ายที่ร้องขอไอพีโดยอัตโนมัติ ซึ่งจะอำนวยความสะดวกให้แก่ผู้ดูแลระบบที่ไม่ต้องคอยไปกำหนดไอพีให้กับเครื่องลูกข่ายดังรูปที่ 5.7 โดยมีค่าต่างดังนี้

- Enable / Disable : เป็นการกำหนดให้เปิดหรือปิดบริการแจกไอพีอัตโนมัติ
- Start IP Address : ค่าของไอพีที่จะให้เริ่มแจก
- Ending IP Address : ค่าสุดท้ายของไอพีที่จะให้แจก
- Netmask : ค่าเน็ตมาร์ค
- Gateway : ค่าของ Default Gateway ที่ให้เครื่องลูกข่ายส่งข้อมูลมาหาในกรณีที่ไอพีที่จะเชื่อมต่ออยู่ต่างเน็ตเวิร์ค
- Lease Time : ค่าของเวลาที่อนุญาติให้เครื่องลูกข่ายสามารถถือครองไอพีได้ ถ้าหมดเวลาต้องมาร้องขอจากเซิร์ฟเวอร์ใหม่ สำหรับเซิร์ฟเวอร์ถ้ายังไม่หมดเวลาไอพีนั้นจะไม่สามารแจกให้เครื่องอื่นได้ หน่วยมีค่าเป็นวินาที
- DNS Server : เป็นค่าที่ระบุถึงไอพีของ DNS Server ซึ่งจะเป็นเซิร์ฟเวอร์ที่ทำการเปลี่ยนการเรียกชื่อคอมพิวเตอร์เป็นไอพี โดยค่าของ DNS Server สามารถระบุได้ 2 ค่า

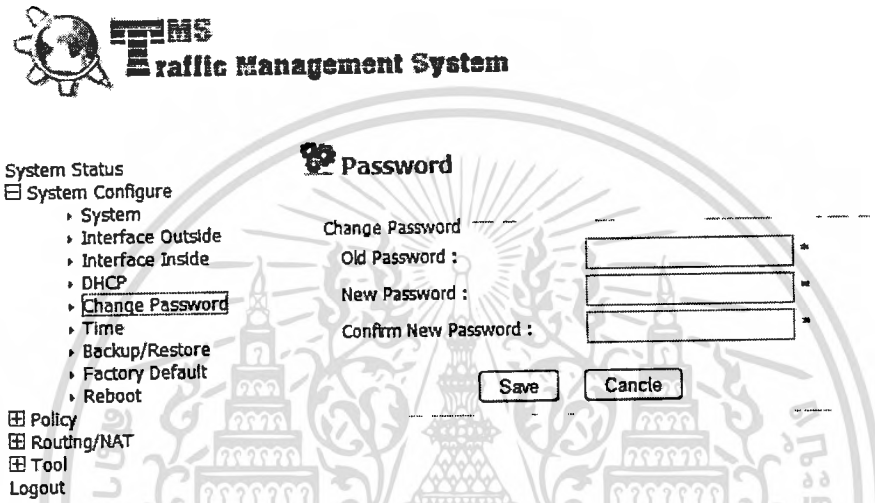


รูปที่ 5.7 หน้าจอเมนูย่อย DHCP

5.2.5 เมนูย่อย Change Password

ในส่วนหน้าจอของเมนูย่อย Password จะเป็นหน้าจอสำหรับให้ผู้ดูแลระบบสามารถเปลี่ยนรหัสผ่านได้ ซึ่งจะมีรายละเอียดดังนี้

- Old Password : รหัสผ่านเดิม
- New Password : รหัสผ่านใหม่
- Confirm Password : ยืนยันรหัสผ่านใหม่

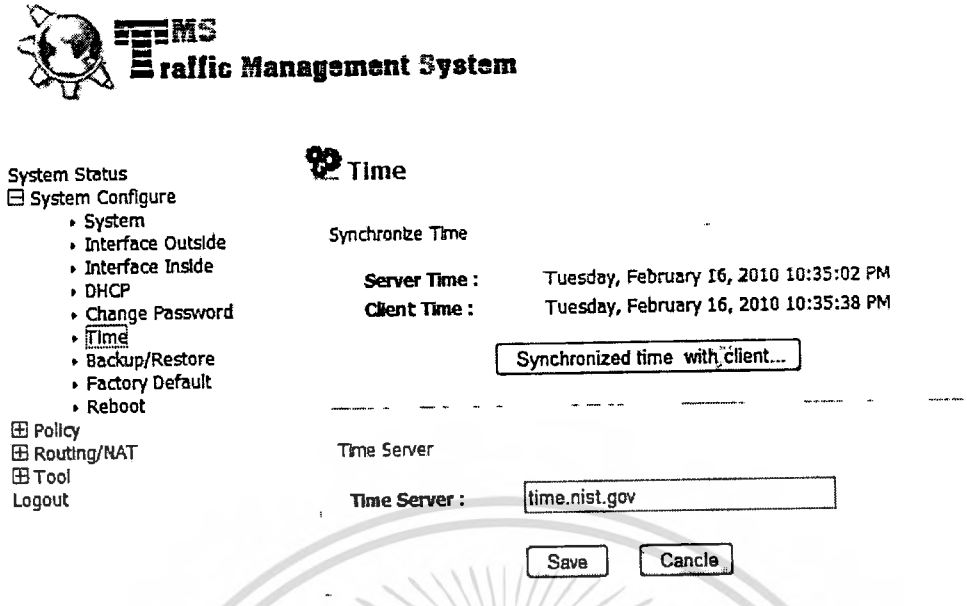


รูปที่ 5.8 หน้าจอเมนูย่อย Change Password

5.2.6 เมนูย่อย Time

ในส่วนหน้าจอของเมนูย่อย Time จะเป็นหน้าจอสำหรับให้ผู้ดูแลระบบตั้งค่าเวลาของระบบเนื่องจาก iBoard ไม่มีฮาร์ดแวร์ที่ทำงานเรื่องเวลา ดังนั้นเมื่อปิดและเปิดเครื่องขึ้นมาเวลาระบบจะเป็นวันที่ 1/1/1970 00:00 ทุกครั้งซึ่งเป็นเวลาที่ผิด ซึ่งการที่จะให้เวลาถูกต้องนั้นต้องให้ระบบสามารถประสานเวลาจาก NTP Server หรือให้ประสานเวลากับเครื่องของผู้ดูแลระบบดังรูปที่ 5.9 ซึ่งมีรายละเอียดดังนี้

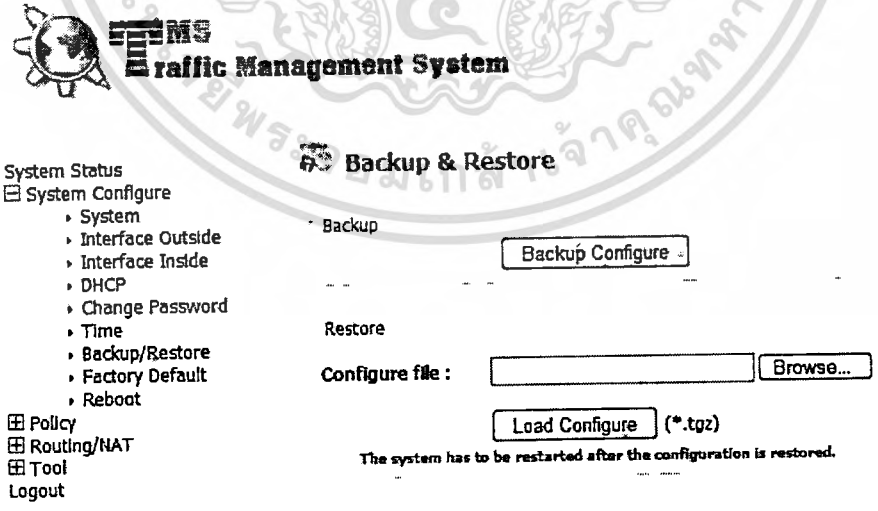
- Synchronize Time : ระบบจะแสดงค่าวันเวลาของระบบและของเครื่องผู้ดูแลระบบเปรียบเทียบกัน ถ้าต้องการให้เวลาระบบตรงกับเวลาของเครื่องผู้ดูแลระบบสามารถกด Synchronized Time with client…
- Time Server : เป็นค่าของไอพี หรือชื่อของเซิร์ฟเวอร์ ที่ให้บริการ NTP เพื่อให้ระบบสามารถเทียบเวลาได้



รูปที่ 5.9 หน้าจอเมนูย่อย Time

5.2.7 เมนูย่อย Backup/Restore

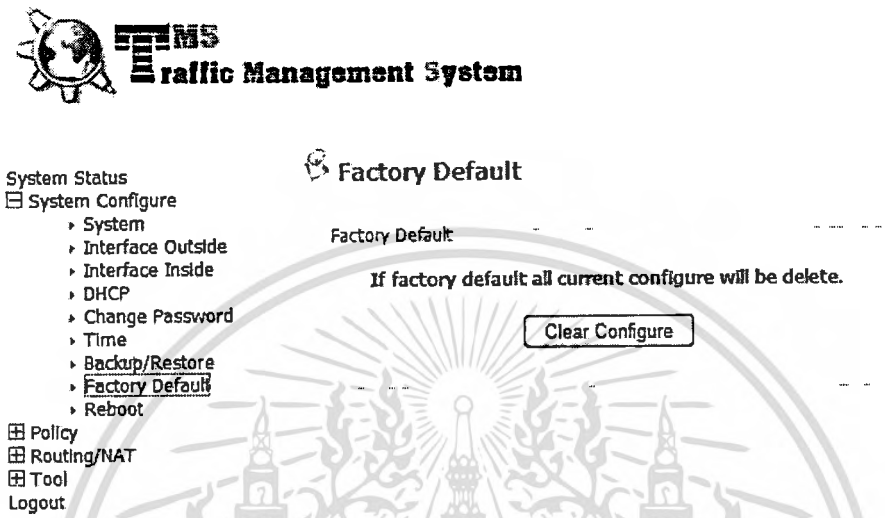
ในส่วนหน้าจอของเมนูย่อย Backup/Restore จะเป็นหน้าจอสำหรับให้ผู้ดูแลระบบทำการดึงเอาค่าคอนฟิคทั้งหมดของระบบออกมาเก็บไว้ และสามารถนำคอนฟิคที่เก็บไว้นั้นใส่กลับคืนไปให้ระบบได้ถ้าระบบมีปัญหาดังรูปที่ 5.10



รูปที่ 5.10 หน้าจอเมนูย่อย Backup & Restore

5.2.8 เมนูย่อย Factory Default

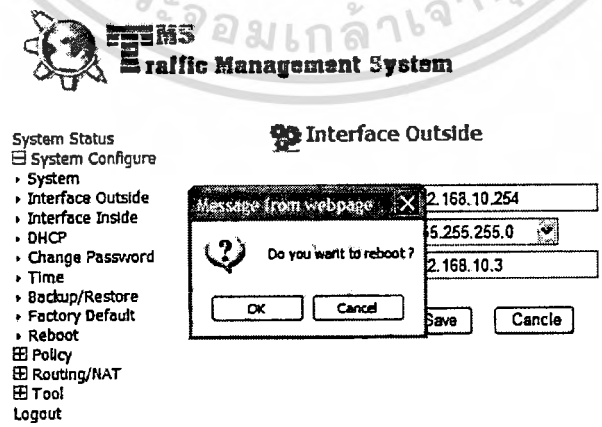
ในส่วนหน้าจอของเมนูย่อย Factory Default จะเป็นหน้าจอสำหรับให้ผู้ดูแลระบบทำการลบค่าคอนฟิคทั้งหมด และเอาค่าคอนฟิคเริ่มต้นของระบบกลับมาใช้งานแทน ซึ่งจะทำให้ข้อมูลต่างๆที่ทำไว้นั้นถูกลบทิ้งไปทั้งหมดดังรูปที่ 5.11



รูปที่ 5.11 หน้าจอเมนูย่อย Factory Default

5.2.9 เมนูย่อย Reboot

ในส่วนหน้าจอของเมนูย่อย Reboot จะเป็นหน้าจอสำหรับให้ผู้ดูแลระบบทำการสั่งให้ระบบปิด-เปิดทำงานใหม่ โดยหน้าจอนี้จะมีเพียงหน้าต่างยืนยันขึ้นมาเพื่อ ให้ผู้ดูแลระบบยืนยันความนาใจอีกครั้ง ดังรูปที่ 5.12



รูปที่ 5.12 หน้าจอเมนูย่อย Reboot

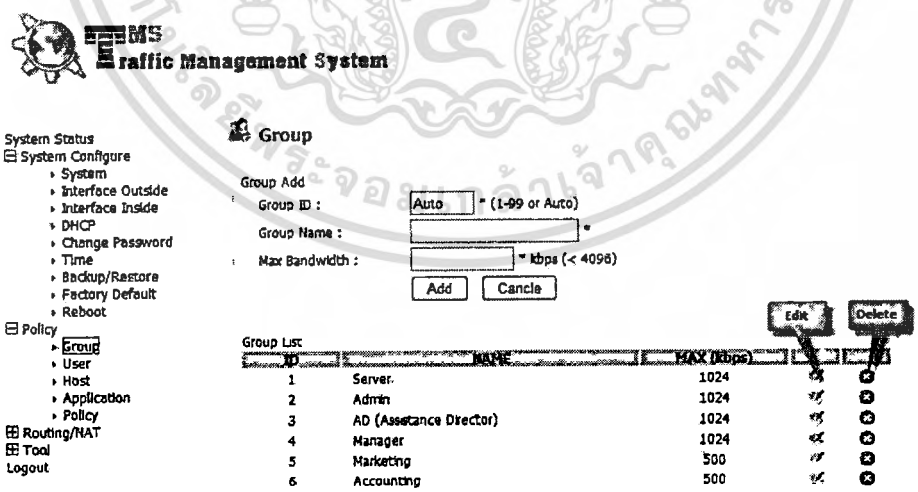
5.3 เมนูหลัก System Configure

ในส่วนของเมนูหลัก Policy นี้จะเป็นส่วนหลักในการกำหนดนโยบายการใช้งานแบนด์วิดท์ของระบบ โดยจะมีการกำหนดนโยบายให้จำกัดแบนด์วิดท์ตามผู้ใช้และโฮสท์, ตามแอพพลิเคชัน ซึ่งในแต่ละนโยบายนั้นก็ต้องพึ่งรายละเอียดอื่นในการกำหนดค่าซึ่งนะได้กล่าวในรายละเอียดต่อไป สำหรับเมนูหลัก Policy นี้ จะมีเมนูย่อยที่ทำงานอยู่ภายใต้ทั้งหมด 5 เมนูดังนี้

5.3.1 เมนูย่อย Group

ในส่วนหน้าจอของเมนูย่อย Group นี้จะแบ่งย่อยออกเป็น 2 ส่วนคือส่วนที่เพิ่มกลุ่มผู้ใช้ใหม่ แสดงส่วนที่แสดงกลุ่มผู้ใช้เดิมที่มีอยู่ ซึ่งในส่วนของการแสดงกลุ่มผู้ใช้เดิมที่มีอยู่นั้นระบบจะแสดงข้อมูลของกลุ่มผู้ใช้แต่ละกลุ่ม โดยผู้ดูแลระบบสามารถแก้ไขเป็นและลบกลุ่มผู้ใช้ออกจากระบบได้โดยด้านท้ายของข้อมูลกลุ่มผู้ใช้จะมีปุ่ม แก้ไขและลบ เมื่อกดปุ่มแก้ไขข้อมูลของกลุ่มผู้ใช้นั้นจะกลับไปแสดงในส่วนเพิ่มกลุ่มผู้ใช้ทำให้ผู้ดูแลระบบสามารถแก้ไขข้อมูลกลุ่มผู้ใช้ได้ เมื่อกดปุ่มลบระบบจะจะมีหน้าต่างยืนยันการลบเพื่อยืนยันการลบข้อมูลจากผู้ดูแลระบบ สำหรับส่วนการเพิ่มกลุ่มผู้ใช้ใหม่นั้นผู้ดูแลระบบจะต้องระบุรายละเอียดข้อมูลต่างๆของกลุ่มผู้ใช้ที่จำเป็นสำหรับระบบ และเพิ่มเข้าระบบ ซึ่งในหน้าต่างนี้ผู้ดูแลระบบมีรายละเอียดดังนี้

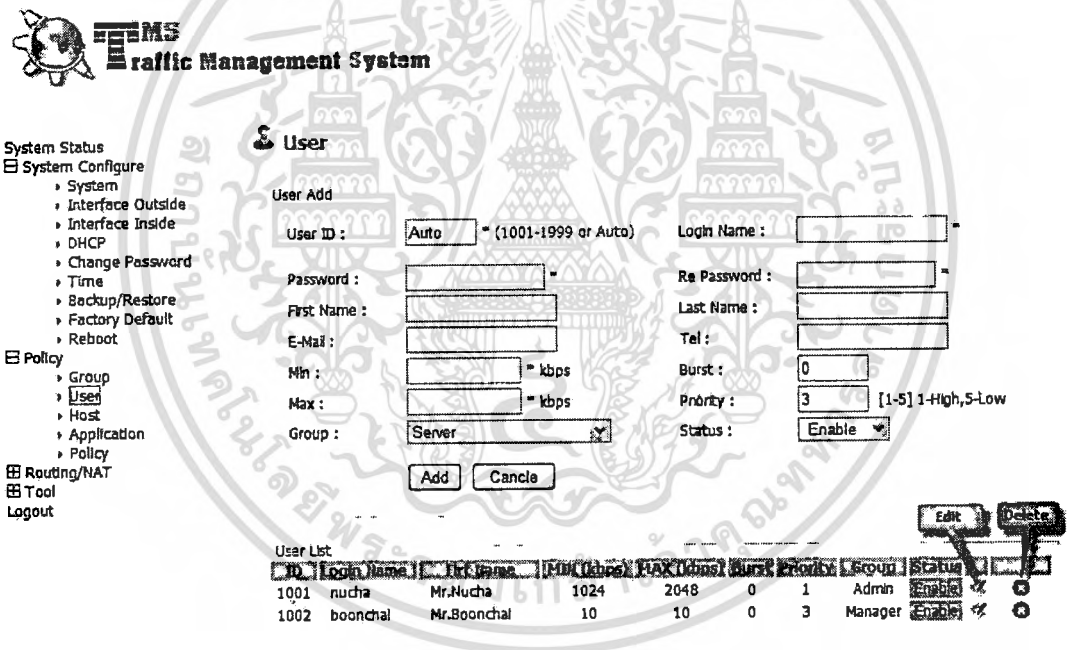
- Group ID : รหัสของกลุ่มผู้ใช้ซึ่งจะใช้สำหรับให้ผู้ใช้อ้างอิงเมื่อเป็นสมาชิกกลุ่ม
- Group Name : ชื่อของกลุ่มผู้ใช้
- Max Bandwidth : ค่าแบนด์วิดท์สูงสุดที่จะให้ผู้ใช้ในกลุ่มสามารถใช้ได้



รูปที่ 5.13 หน้าจอเมนูย่อย Group

5.3.2 เมนูย่อย User

ในส่วนหน้าจอของเมนูย่อย User นี้จะแบ่งย่อยออกเป็น 2 ส่วนคือส่วนที่เพิ่มผู้ใช้ใหม่ แสดงส่วนที่แสดงผู้ใช้เดิมที่มีอยู่ ซึ่งในส่วนของการแสดงผู้ใช้เดิมที่มีอยู่นั้นระบบจะแสดงข้อมูลบางส่วนของผู้ใช้แต่ละคน โดยผู้ดูแลระบบสามารถแก้ไขเป็นและลบผู้ใช้ออกจากระบบได้โดยด้านท้ายของข้อมูลผู้ใช้จะมีปุ่ม แก้ไขและลบ เมื่อกดปุ่มแก้ไขข้อมูลของผู้ใช้คนนั้นจะกลับไปแสดงในส่วนเพิ่มผู้ใช้ทำให้ผู้ดูแลระบบสามารถแก้ไขข้อมูลผู้ใช้ได้ เมื่อกดปุ่มลบระบบจะจะมีหน้าต่างยืนยันการลบเพื่อยืนยันการลบข้อมูลจากผู้ดูแลระบบ สำหรับส่วนการเพิ่มผู้ใช้ใหม่นั้นผู้ดูแลระบบจะต้องระบุรายละเอียดข้อมูลต่างๆของผู้ใช้ที่จำเป็นสำหรับระบบ และเพิ่มเข้าระบบ การสร้างผู้ใช้ให้สามารถเข้ามาใช้งานระบบได้โดยข้อมูลที่จำเก็บจะแบ่งเป็นข้อมูลส่วนตัวผู้ใช้, ข้อมูลการพิสูจน์ตัวตน และข้อมูลการจัดสรรแบนด์วิดท์ที่จะให้กับผู้ใช้ และผู้ใช้จะต้องถูกกำหนดให้เป็นสมาชิกของกลุ่มผู้ใช้กลุ่มใดกลุ่มหนึ่ง ซึ่งในหน้าต่างนี้ผู้ดูแลระบบมีรายละเอียดดังนี้



รูปที่ 5.14 หน้าจอเมนูย่อย User

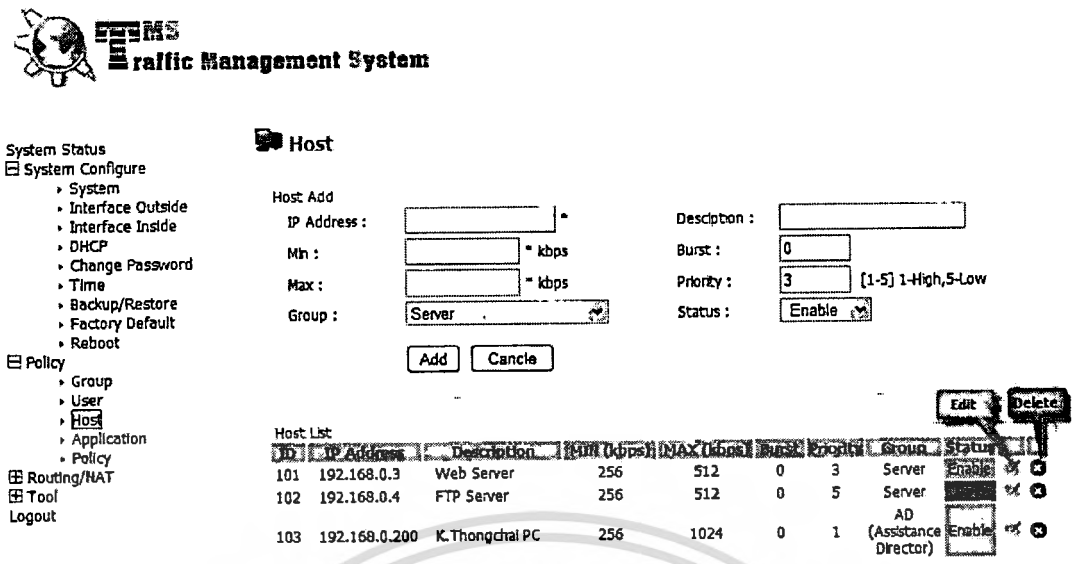
- User Id : รหัสผู้ใช้ซึ่งจะใช้สำหรับให้ระบบอ้างอิงถึงผู้ใช้มีค่าได้ตั้งแต่ 1001-1999
- Login : ชื่อผู้ใช้สำหรับใช้พิสูจน์ตัวตนเข้าระบบ
- Password : รหัสผ่านที่ใช้ยืนยันในการพิสูจน์ตัวตน
- Re Password : เป็นการยืนยันรหัสผ่านกันความผิดพลาด
- First Name : ชื่อจริงของผู้ใช้
- Last Name : นามสกุลของผู้ใช้

- E-Mail : อีเมลของผู้ใช้เพื่อให้ผูดูแลระบบสามารถติดต่อได้
- Tel : เบอร์ติดต่อผู้ใช้เผื่อผู้ดูแลระบบต้องการติดต่อผู้ใช้
- Min : แบนด์วิดท์ที่จะได้อย่างน้อยที่สุดที่ให้ผู้ใช้มีหน่วยเป็น กิโลบิตต่อวินาที (kbps)
- Max : แบนด์วิดท์ที่ให้ได้มากที่สุด ที่ให้ผู้ใช้มีหน่วยเป็น กิโลบิตต่อวินาที (kbps)
- Burst : ค่าที่อนุญาติให้ใช้แบนด์วิดท์เพิ่มขึ้นในช่วงที่จำเป็น
- Priority : การกำหนดความสำคัญของข้อมูลที่ผู้ใช้ส่งโดยถ้ากำหนดเป็นค่าน้อยจะมีความสำคัญสูง โดยค่าอยู่ระหว่าง 1-5
- Group Member : การกำหนดกลุ่มผู้ใช้เพื่อให้ผู้ใช้ไปเป็นสมาชิก
- Status : เป็นการระบุว่าผู้ใช้นี้ให้สามารถใช้งานระบบได้ (Enable) หรือห้ามใช้ชั่วคราว (Disable)

5.3.3 เมนูย่อย Host

ในส่วนหน้าจอของเมนูย่อย Host นี้จะแบ่งย่อยออกเป็น 2 ส่วนคือส่วนที่เพิ่มเครื่องคอมพิวเตอร์ใหม่ แสดงส่วนที่แสดงเครื่องคอมพิวเตอร์เดิมที่มีอยู่ ซึ่งในส่วนของการแสดงเครื่องคอมพิวเตอร์เดิมที่มีอยู่นั้นระบบจะแสดงข้อมูลของเครื่องคอมพิวเตอร์นั้น โดยผู้ดูแลระบบสามารถแก้ไขเป็นและลบเครื่องคอมพิวเตอร์ออกจากระบบได้โดยด้านท้ายของข้อมูลเครื่องคอมพิวเตอร์จะมีปุ่ม แก้ไขและลบ เมื่อกดปุ่มแก้ไขข้อมูลของเครื่องคอมพิวเตอร์นั้นจะแสดงในส่วนเพิ่มเครื่องคอมพิวเตอร์ ทำให้ผู้ดูแลระบบสามารถแก้ไขข้อมูลเครื่องคอมพิวเตอร์ได้ เมื่อกดปุ่มลบระบบจะจะมีหน้าต่างยืนยันการลบเพื่อยืนยันการลบข้อมูลจากผู้ดูแลระบบ สำหรับส่วนการเพิ่มเครื่องคอมพิวเตอร์ใหม่นั้นผู้ดูแลระบบจะต้องระบุรายละเอียดข้อมูลต่างๆของเครื่องคอมพิวเตอร์ที่จำเป็นสำหรับระบบ และเพิ่มเข้าระบบ ซึ่งในรายละเอียดของหน้าต่างมีดังนี้

- IP Address : ไอพีแอดเดรสของโฮสท์
- Description : คำอธิบายหรือชื่อของโฮสท์
- Min : แบนด์วิดท์ที่จะได้อย่างน้อยที่สุด มีหน่วยเป็นกิโลบิตต่อวินาที (kbps)
- Max : แบนด์วิดท์ที่ให้ได้มากที่สุด มีหน่วยเป็นกิโลบิตต่อวินาที (kbps)
- Burst : ค่าที่อนุญาติให้ใช้แบนด์วิดท์เพิ่มขึ้นในช่วงที่จำเป็น
- Priority : การกำหนดความสำคัญของข้อมูลที่ผู้ใช้ส่งโดยถ้ากำหนดเป็นค่าน้อยจะมีความสำคัญสูง โดยค่าอยู่ระหว่าง 1-5
- Group Member : การกำหนดกลุ่มผู้ใช้เพื่อให้ผู้ใช้ไปเป็นสมาชิก
- Status : เป็นการระบุว่าโฮสท์นี้ให้สามารถใช้งานระบบได้ (Enable) หรือห้ามใช้ชั่วคราว (Disable)



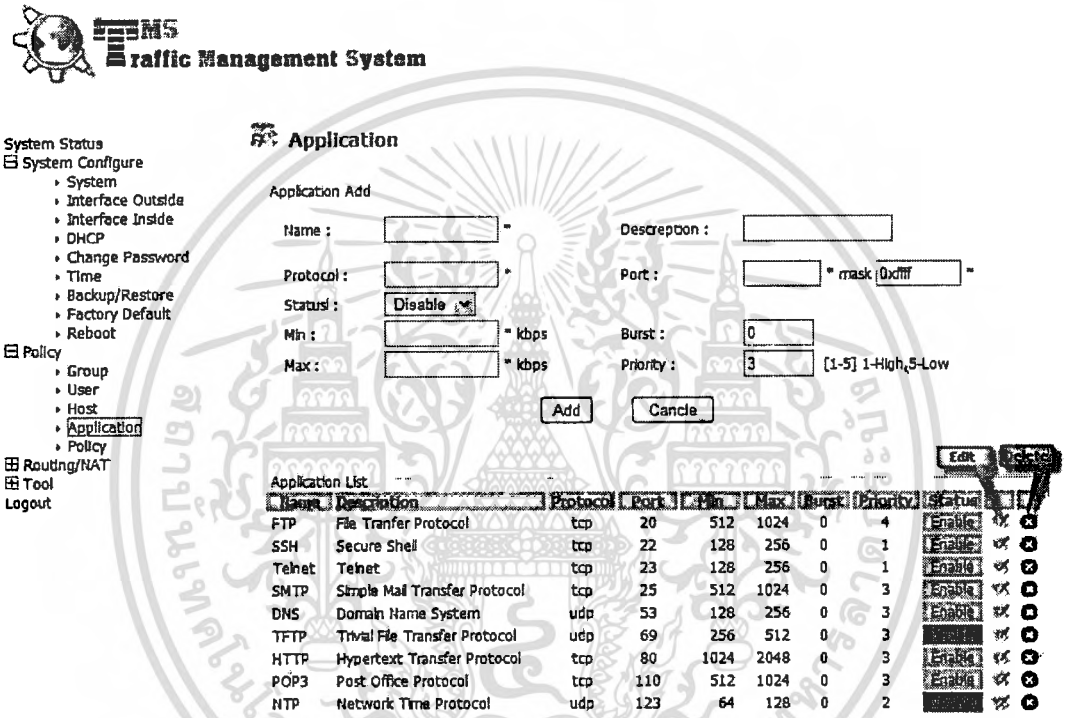
รูปที่ 5.15 หน้าจอเมนูย่อย Host

5.3.4 เมนูย่อย Application

ในส่วนหน้าจอของเมนูย่อย Application นี้จะแบ่งย่อยออกเป็น 2 ส่วนคือส่วนที่เพิ่มแอพพลิเคชันใหม่ แสดงส่วนที่แสดงแอพพลิเคชันเดิมที่มีอยู่ ซึ่งในส่วนของการแสดงแอพพลิเคชันเดิมที่มีอยู่นั้นระบบจะแสดงข้อมูลของแอพพลิเคชันนั้น โดยผู้ดูแลระบบสามารถแก้ไขเป็นและลบแอพพลิเคชันออกจากระบบได้โดยด้านท้ายของข้อมูลแอพพลิเคชันจะมีปุ่ม แก้ไขและลบ เมื่อกดปุ่มแก้ไขข้อมูลของแอพพลิเคชันนั้นจะแสดงในส่วนเพิ่มแอพพลิเคชัน ทำให้ผู้ดูแลระบบสามารถแก้ไขข้อมูลแอพพลิเคชันได้ เมื่อกดปุ่มลบระบบจะจะมีหน้าต่างยืนยันการลบเพื่อยืนยันการลบข้อมูลจากผู้ดูแลระบบ สำหรับส่วนการเพิ่มแอพพลิเคชันใหม่นั้นผู้ดูแลระบบจะต้องระบุรายละเอียดข้อมูลต่างๆของแอพพลิเคชันที่จำเป็นสำหรับระบบ และเพิ่มเข้าระบบ ซึ่งในรายละเอียดของหน้าต่างมีดังนี้

- Name : ชื่อของแอพพลิเคชัน
- Description : คำอธิบายหรือคำขยายชื่อของแอพพลิเคชัน
- Protocol : โพรโทคอลที่แอพพลิเคชันใช้ทำงาน เช่น TCP, UDP, ICMP
- Port : พอร์ตที่แอพพลิเคชันในการสื่อสาร เช่นเว็บ พอร์ตคือ 80
- Mask : มาร์คเป็นค่าตัวเลขฐาน 16 ทำหน้าที่เหมือนเน็ตมาร์คของไอพี แต่จะเป็นมาร์คของพอร์ตแทนโดยปรกติจะมีค่าเป็น 0xffffff เช่น เว็บ port = 80, mask = 0xffffff หมายถึงเว็บใช้พอร์ต 80 พอร์ตเดียว แต่ถ้าเป็น FTP ซึ่งจะใช้พอร์ท 20,21 ดังนั้นเพื่อทำให้ค่าพอร์ทครอบคลุมทั้ง 20,21 ค่า port และ mask ต้องกำหนดเป็น port = 20, mask = 0xfffffe

- Min : แบนด์วิดท์ที่จะได้อย่างน้อยที่สุด มีหน่วยเป็นกิโลบิตต่อวินาที (kbps)
- Max : แบนด์วิดท์ที่ให้ได้มากที่สุด มีหน่วยเป็นกิโลบิตต่อวินาที (kbps)
- Burst : ค่าที่อนุญาติให้ใช้แบนด์วิดท์เพิ่มขึ้นในช่วงที่จำเป็น
- Priority : การกำหนดความสำคัญของข้อมูลที่ผู้ใช้ส่งโดยถ้ากำหนดเป็นค่าน้อยจะมีความสำคัญสูง โดยค่าอยู่ระหว่าง 1-5
- Status : เป็นการระบุว่าให้ควบคุมการใช้งานแบนด์วิดท์ของแอพพลิเคชันนี้หรือไม่ ถ้าควบคุมใช้ Enable หรือถ้าไม่ควบคุมใช้ Disable



รูปที่ 5.16 หน้าจอเมนูย่อย Application

5.3.5 เมนูย่อย Policy

ในส่วนหน้าจอของเมนูย่อย Policy นี้จะเป็นส่วนที่มีความสำคัญที่สุดของระบบ เพราะจะเป็นการระบุให้ระบบทำงานในการจัดการแบนด์วิดท์อย่างไร ซึ่งการกำหนดค่าของ Policy นั้นจะมีอยู่ด้วกัน 3 โหมด เมื่อมีการเปลี่ยนโหมดการทำงานจะต้องมีการปิดแล้วเปิดระบบใหม่ทุกครั้งเพิ่มให้ระบบทำงานตาม Policy ที่กำหนด ซึ่งจะมีโหมดดังนี้คือ

- User & Host : โดยโหมดนี้ถือเป็นโหมดหลักของระบบโดยการทำงานของโหมดนี้คือ เมื่อเป็นระบบขึ้นมาระบบจะทำการโหลดข้อมูล Policy ต่างๆของโฮสท์ที่อยู่ในฐานข้อมูลและมีสถานะเป็น Enable เข้ามาในระบบโดยทำให้โฮสท์นั้นสามารถใช้

อินเทอร์เน็ตผ่านระบบได้เลยโดยได้แบนด์วิดท์ตามที่กำหนดไว้ในฐานข้อมูล ส่วนผู้ใช้นั้นเมื่อต้องการจะใช้งานอินเทอร์เน็ตผ่านระบบ ระบบจะแสดงหน้าต่างให้พิสูจน์ตัวตนเมื่อผ่านแล้วระบบจะทำการโหลดข้อมูลสิทธิการใช้งานแบนด์วิดท์จะฐานข้อมูลมาเพิ่มในระบบทำให้ผู้ใช้สามารถใช้งานแบนด์วิดท์ได้ตามสิทธิที่กำหนดไว้

- Application : โดยโหมดนี้ถือเป็นโหมดรองของระบบโดยการทำงานของโหมดนี้คือเมื่อเป็นระบบขึ้นมาระบบจะทำการโหลดข้อมูล Policy จากฐานข้อมูล Application แล้วนำค่าของแอพพลิเคชันที่มีสนานะเป็น Enable เข้ามาเพิ่มในระบบ เมื่อระบบโหลดการทำงานเสร็จสิ้น เครื่องคอมพิวเตอร์ทุกเครื่องที่ใช้งานอินเทอร์เน็ตผ่านระบบจะสามารถใช้งานได้เลยโดยที่ไม่ต้องมีการพิสูจน์ตัวตน แต่สิทธิการใช้งานแบนด์วิดท์นั้นก็จะถูกจัดสรรตามที่กำหนดไว้ตามแอพพลิเคชันนั้นๆ
- Disable : โหมดนี้ถือเป็นการปิดการจัดการแบนด์วิดท์ของระบบเครื่องคอมพิวเตอร์ทุกเครื่องจะสามารถใช้งานได้อย่างอิสระ

รูปที่ 5.17 หน้าจอเมนูย่อย Policy

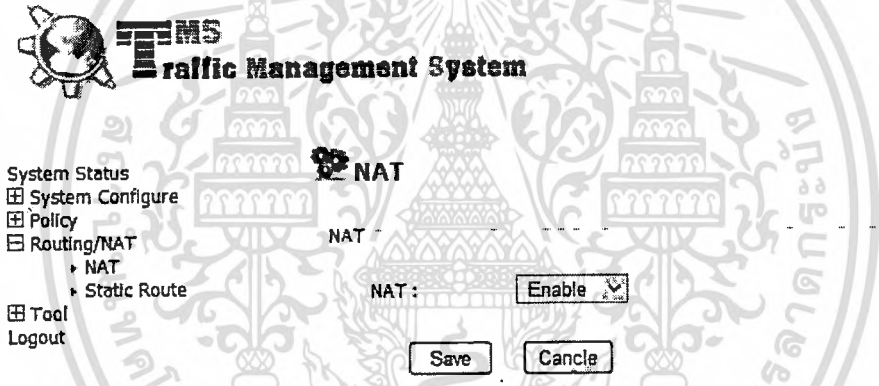
5.4 เมนูหลัก Routing/NAT

ในส่วนของเมนูหลัก Routing/NAT นี้จะเป็นส่วนที่ให้ผู้ดูแลระบบสามารถกำหนดรูปแบบการทำงานของระบบกับเครื่อข่ายที่ระบบทำงานอยู่ ว่าให้ทำงานในลักษณะแบบไหน ซึ่งสามารถกำหนดรายละเอียดต่างๆ ตามเมนูย่อยดังนี้

5.4.1 เมนูย่อย NAT

ในส่วนหน้าจอของเมนูย่อย NAT นั้นจะเป็นการกำหนดให้ระบบ จัดการกับเครื่องในเครื่อข่ายที่ต้องการควบคุมแบนด์วิดท์นั้น ให้สามารถติดต่อไปยังเครื่องที่อยู่ภายนอกเครื่อข่ายในลักษณะไหน ซึ่งสามารถกำหนดได้ 2 รูปแบบดังนี้

- Enable : กำหนดให้ติดต่อไปยังเครือข่ายภายนอกแบบ NAT โหมด คือระบบจะทำการเปลี่ยนเปลี่ยนข้อมูลของเครื่องลูกข่ายที่จะส่งไปยังเครือข่ายภายนอกในส่วนของ Source IP ใน IP Header ให้เป็นไอพีขานอกของระบบก่อนแล้วถึงจะส่งต่อข้อมูลของเครื่องลูกข่ายนั้นออกไป
- Disable : ระบบจะไม่มีการเปลี่ยนแปลงข้อมูลใดๆ ของเครื่องลูกข่ายเมื่อได้รับข้อมูลมาก็จะทำการส่งต่อให้ตามเส้นทางการส่งข้อมูลที่ระบบมีอยู่ซึ่งการทำในลักษณะนี้เรียกอีกอย่างว่า Route Mode



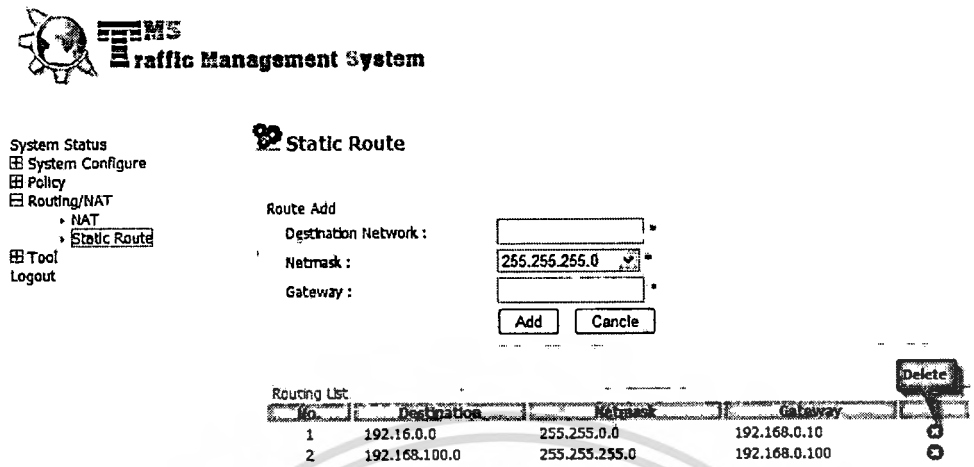
รูปที่ 5.18 หน้าจอเมนูย่อย NAT

5.4.2 เมนูย่อย Static Route

ในส่วนหน้าจอของเมนูย่อย Static Route นั้นจะเป็นการกำหนดเส้นทางหารส่งต่อข้อมูลให้ระบบได้ทราบนอกเหนือจาก Default Route ที่มีอยู่แล้ว โดยหน้าจอในส่วนนี้นั้นมีอยู่ด้วยกัน 2 ส่วนคือ ในส่วนที่จะเพิ่มเราท์ติงใหม่เข้าไป และส่วนที่แสดงเราท์ติงเดิมของระบบที่มีอยู่ ซึ่งในส่วนนี้นั้นยังสามารถที่จะสั่งลบเราท์ติงเดิมที่มีอยู่ออกได้ด้วย โดยจะมีปุ่มสำหรับลบเราท์ติงออกอยู่ทางด้านท้ายของเราท์ติง เมื่อสั่งลบจะมีหน้าต่างถามยืนยันการลบเพื่อให้ผู้ดูแลระบบยืนยันอกครั้งก็จะสามารถลบได้ ส่วนรายละเอียดของช่องการเพิ่มข้อมูลเราท์ติงนั้นมีดังนี้

- Destination Network : เป็นค่าเน็ตเวิร์คปลายทางที่ต้องการส่งข้อมูลไป
- Netmask : ค่าเน็ตมาร์คของเน็ตเวิร์คปลายทาง

- Gateway : ไอพีต่อไปที่จะส่งข้อมูลไปยังเน็ตเวิร์คปลายทางนั้น



รูปที่ 5.19 หน้าจอเมนูย่อย Static Route

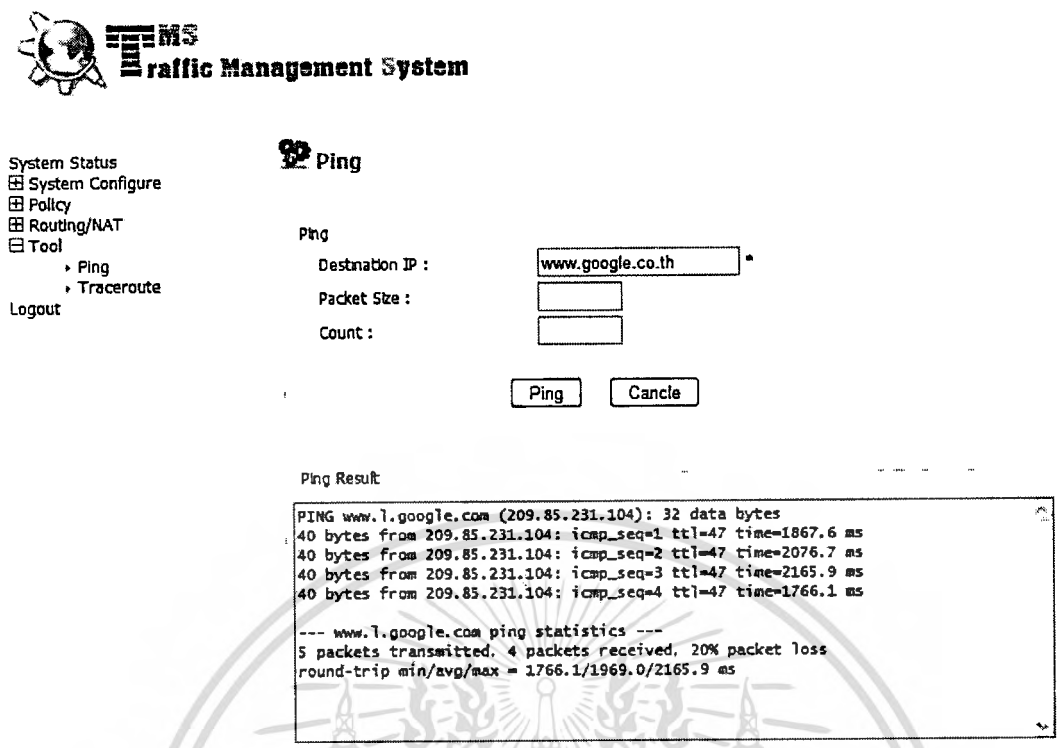
5.5 เมนูหลัก Tool

ในส่วนของเมนูหลัก Tool นี้จะเป็นจะเป็นเครื่องมือเบื้องต้นที่จะให้ผู้ดูแลระบบสามารถตรวจสอบปัญหาของเครือข่ายได้ โดยจะมีเครื่องมือให้ 2 อย่าง ตามเมนูย่อยดังนี้

5.5.1 เมนูย่อย Ping

ในส่วนหน้าจอของเมนูย่อย Ping นั้นจะเป็นเครื่องมือที่ช่วยทดสอบว่าระบบสามารถเชื่อมต่อไปยังปลายทางที่ต้องการได้หรือไม่ ถ้าได้ก็จะมีการตอบกลับมา ซึ่งค่าต่างของการ Ping มีดังนี้

- Destination IP : ชื่อหรือไอพีของเครื่องปลายทางที่ต้องการทดสอบการเชื่อมต่อด้วย
- Packet Size : ขนาดของข้อมูลที่จะส่งไป ถ้าไม่กำหนดค่าจะมีค่าโดยปริยายเท่ากับ 32
- Count : จำวนครั้งที่จะให้ทดสอบส่งข้อมูลไป ถ้าไม่กำหนดค่าจะมีค่าโดยปริยายเท่ากับ 5

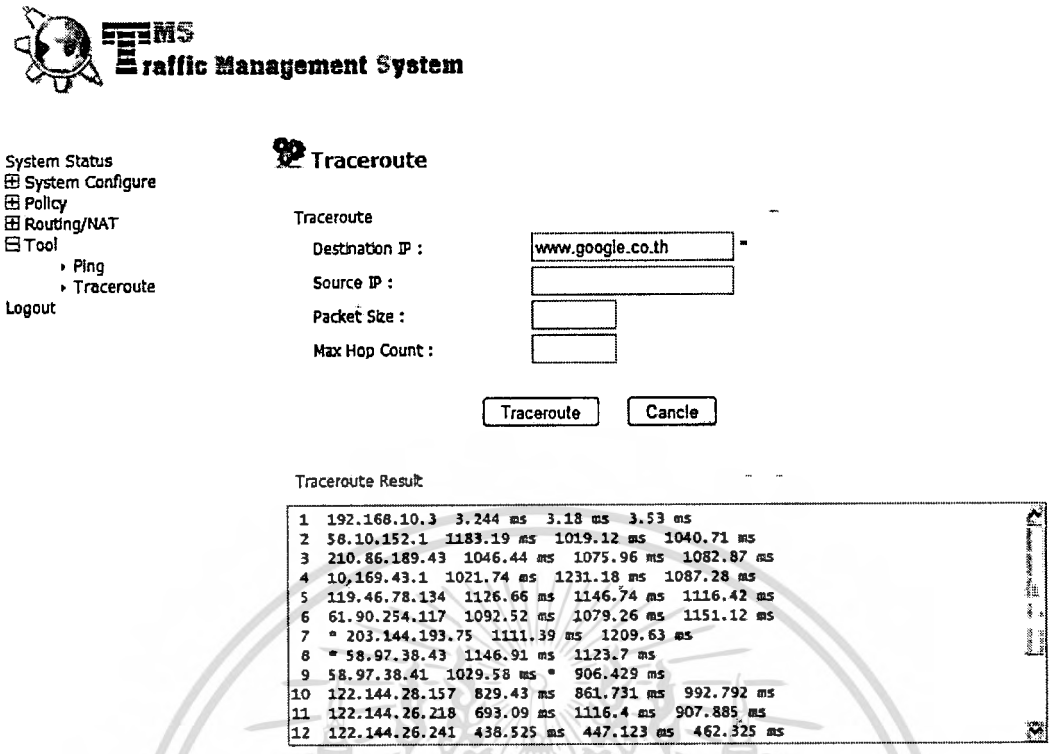


รูปที่ 5.20 หน้าจอเมนูย่อย Ping

5.5.2 เมนูย่อย Traceroute

ในส่วนหน้าจอของเมนูย่อย Traceroute จะเป็นส่วนของผู้ดูแลระบบสามารถที่จะทดสอบระบบเครือข่ายได้อีกวิธีหนึ่งนอกจากการ ping ซึ่ง traceroute จะแสดงเส้นทางการเดินทางของข้อมูลออกมาด้วย ซึ่งมีรายละเอียดได้ดังนี้

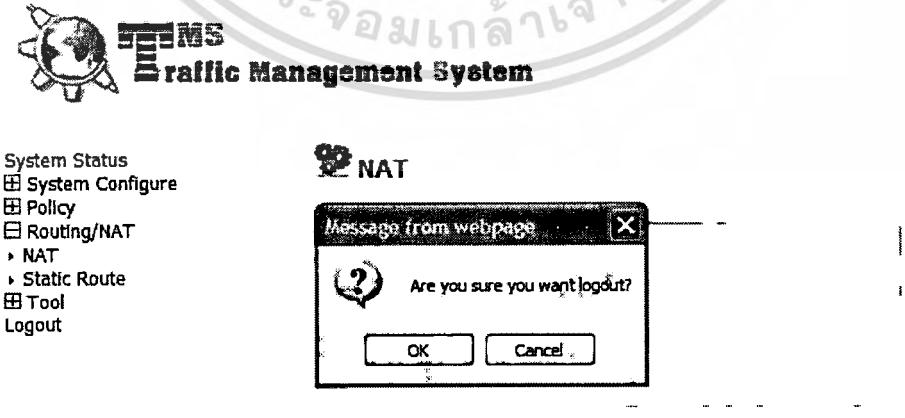
- Destination IP : ชื่อหรือไอพีปลายทางที่ต้องการทดสอบการเชื่อต่อ
- Source IP : ไอพีของระบบที่จะต้องการระบุเข้าไปในข้อมูลที่ส่งออกไปซึ่งจะเป็นค่า Source IP ใน IP Header ถ้าหากไม่ระบุ ค่าโดยปริยายที่ในถูกใส่เข้าไปคือไอพีของอินเทอร์เฟซของระบบที่ข้อมูลถูกส่งออกไป
- Packet Size : ขนาดของข้อมูลที่จะให้ส่งไปขณะทดสอบ
- Max Hop Count : จำนวนชั้นของเครือข่ายที่จะให้ข้อมูลนี้ถูกส่งออกได้ไกลที่สุด ถ้าหากไม่ระบุค่าโดยปริยายเท่ากับ 30



รูปที่ 5.21 หน้าจอเมนูย่อย Traceroute

5.6 เมนูหลัก Logout

ในส่วนของเมนูหลัก Logout เมื่อผู้ดูแลระบบต้องการที่จะออกจากระบบก็สามารถออกจากระบบโดยผ่านเมนูนี้โดยระบบจะมีหน้าต่างยืนยัน เมื่อยืนยันระบบจะสั่งปิดหน้าต่างที่ใช้ติดต่อระบบด้วย



รูปที่ 5.22 หน้าจอเมนูหลัก Logout

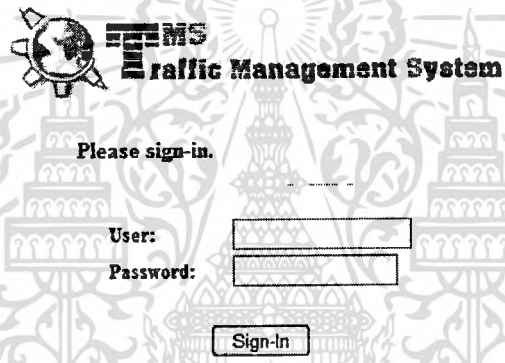
5.7 หน้าจอสำหรับผู้ใช้ระบบ

ในส่วนของหน้าจอสำหรับผู้ใช้ระบบนั้น จะมีอยู่ด้วยกัน 2 ส่วนคือ ส่วนสำหรับใช้ Login และส่วนหรับใช้ Logout ซึ่งทั้ง 2 หน้าจอนี้มีไว้ให้ผู้ใช้พิสูจน์ตัวต้นเมื่อต้องการใช้แบนด์วิดท์ และออกจากการใช้ระบบ ซึ่งจะได้ใช้เมื่อระบบถูกกำหนด Policy ให้เป็น User & Host เท่านั้น รายละเอียดต่างๆ มีดังนี้

5.7.1 หน้าจอ User Login

ในส่วนหน้าจอ User Login จะขึ้นมาให้อัตโนมัติเมื่อผู้ใช้ต้องการใช้แบนด์วิดท์ ในครั้งแรก โดยจะมีช้องสำหรับใส่ข้อมูลเพื่อพิสูจน์ตัวตนกับระบบ ดังนี้

- User : ชื่อของผู้ใช้ที่ได้รับมาเพื่อให้สามารถใช้ระบบได้
- Password : รหัสผ่านที่ต้องใช้ในการพิสูจน์ตัวต้นของผู้ใช้

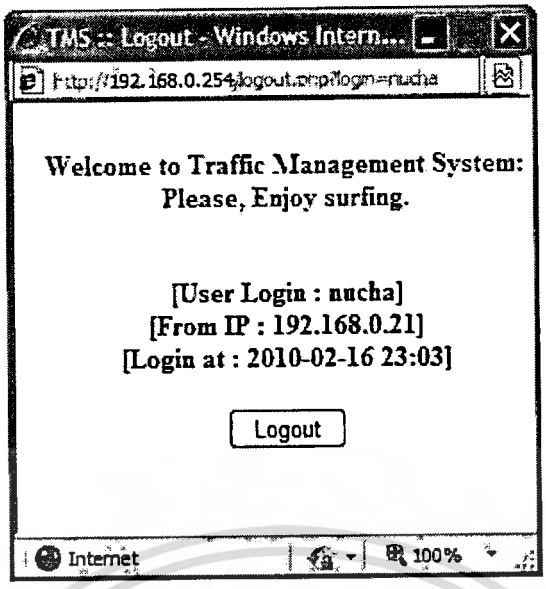


รูปที่ 5.23 หน้าจอ User Login

5.7.2 หน้าจอ User Logout

ในส่วนหน้าจอ User Logout จะขึ้นมาให้อัตโนมัติเมื่อผู้ใช้ทำการพิสูจน์ตัวตนผ่านระบบได้ ถ้าหากผู้ใช้เผลอปิดหน้าจอนี้ไป เมื่อต้องการออกจากระบบก็สามารถเรียกหน้าจอ User Logout กลับมาได้ โดยพิมพ์ http://system_ip/logout.php ซึ่งหน้าจอนี้จะแสดงข้อมูลของผู้ใช้ดังนี้

- User Login : ชื่อ Login ของผู้ใช้
- From IP : หมายเลขไอพีของเครื่องผู้ใช้
- Login at : เวลาที่ผู้ใช้เข้าใช้งานระบบ



รูปที่ 5.24 หน้าจอ User Logout

บทที่ 6
ทดสอบระบบและบทสรุป

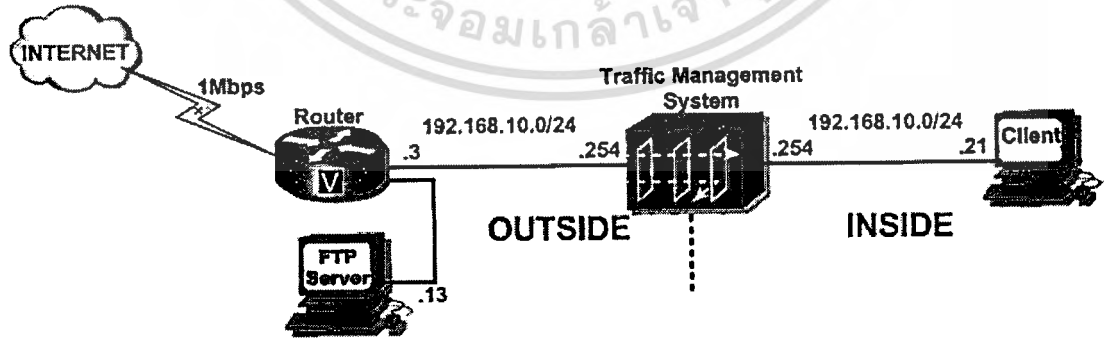
6.1 การทดสอบระบบ

ในการทดสอบการทำงานของระบบนั้นจะการเชื่อมต่อระบบในส่วนเครือข่ายภานอก (Interface Outside) เข้ากับอินเทอร์เน็ต และให้เครื่องลูกข่ายต่อเข้าเครือข่ายภายใน (Interface Inside) ของระบบ โดยขั้นตอนหลักสำหรับการทดสอบระบบนั้นจะแบ่งออกเป็น 3 การทดสอบ และทุกขั้นตอนการทดสอบจะทำการทดสอบในลักษณะเดียวกันคือ การดาวน์โหลดข้อมูลจากอินเทอร์เน็ต โดยผ่าน 2 โพรโตคอลคือ http โดยจะทดสอบดาวน์โหลด Windows-XP-SP3 จากเว็บไซต์ของ Microsoft (http://www.microsoft.com/downloads/details.aspx?displaylang=en&FamilyID=5b33b5a8-5e76-401f-be08-1e1555d4f3d4) และ ftp โดยการ ftp จากเครื่องเซิร์ฟเวอร์ไอพี 192.168.10.13 ทุกครั้งของการทดสอบจะกระทำกับเว็บไซต์และเซิร์ฟเวอร์เดิมซึ่งรูปแบบการเชื่อมต่อเป็นไปดังรูปที่ 6.1 และขั้นตอนการทดสอบมีดังนี้

1. ปิดการจัดการ Policy เพื่อทดสอบว่าถ้าระบบไม่มีการจำกัดแบนด์วิดท์แล้วสามารถที่จะดาวน์โหลดข้อมูลได้สูงสุดเท่าใด

2. เปิดการจัดการ Policy แบบ User & Host แล้วทำการจำกัดแบนด์วิดท์ของผู้ใช้ และให้ผู้ใช้ล็อกอินเข้าระบบแล้วทำการดาวน์โหลดข้อมูลตามขั้นตอนเดียวกับข้อที่ 1

3. เปิดการจัดการ Policy แบบ Application แล้วทำการจำกัดการใช้แบนด์วิดท์ของ http และ ftp แล้วให้เครื่องลูกข่ายดาวน์โหลดข้อมูลตามขั้นตอนเดียวกับข้อที่ 1

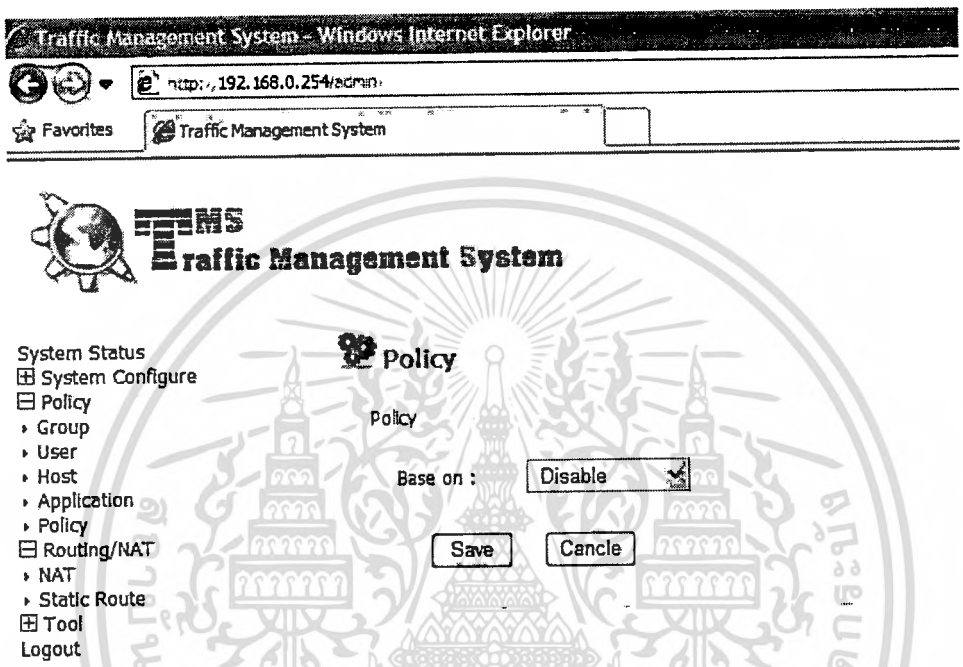


รูปที่ 6.1 แผนภาพแสดงการเชื่อมต่อระบบเพื่อการทดสอบ

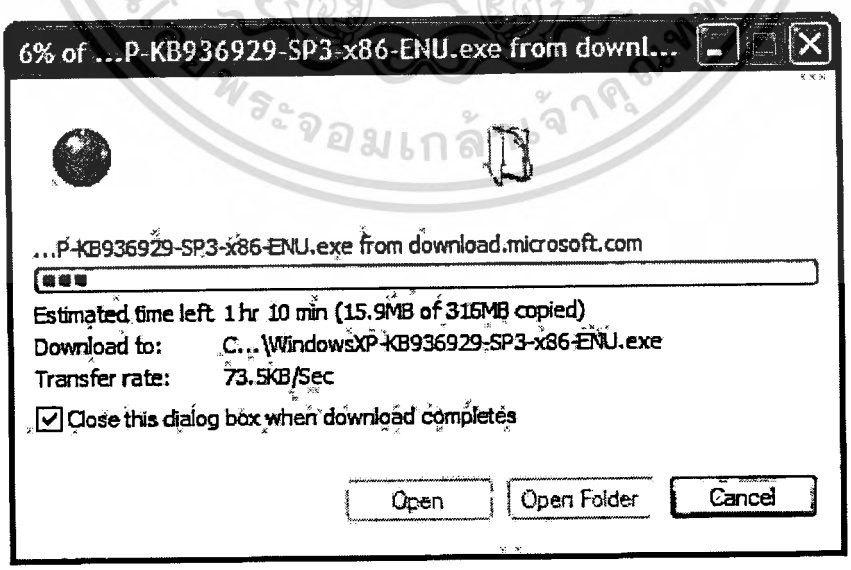
6.1.1 การทดสอบปิดการจัดการ Policy

ในการทดสอบโดยการปิด Policy จะเป็นการทดสอบว่าเมื่อไม่ได้ความคุมใดระบบ จะสามารถใช้งานแบนด์วิดท์ได้เต็มที่เท่าไหร่ เพื่อเป็นตัวตั้งในการเปรียบเทียบผลการทดสอบ ซึ่งการกำหนดค่าคอนฟิคระบบเป็นดังนี้

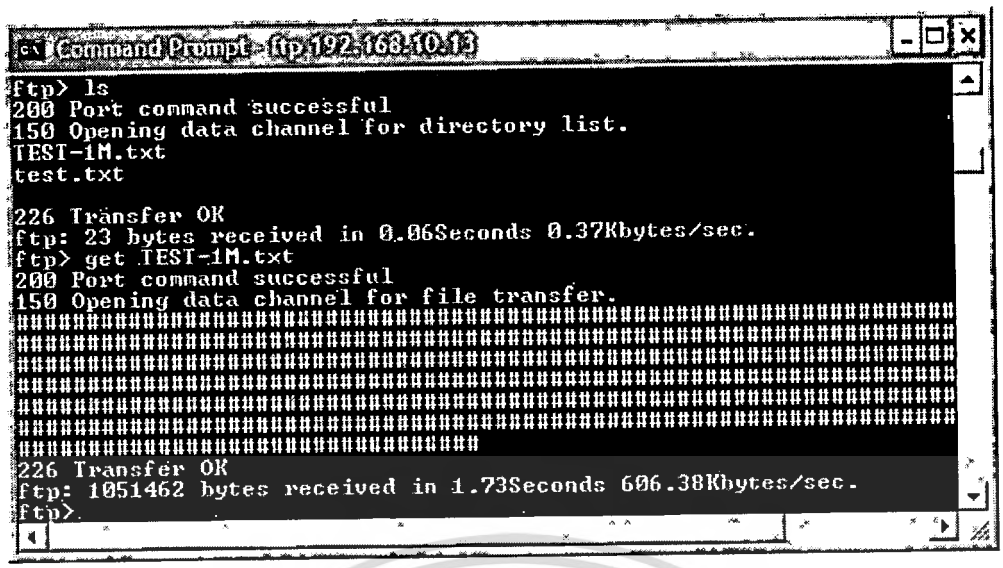
- Policy : Disable



รูปที่ 6.2 การกำหนดค่า Policy ในการทดสอบรอบที่ 1



รูปที่ 6.3 ผลการดาวน์โหลดโดยใช้ http ในการทดสอบรอบที่ 1

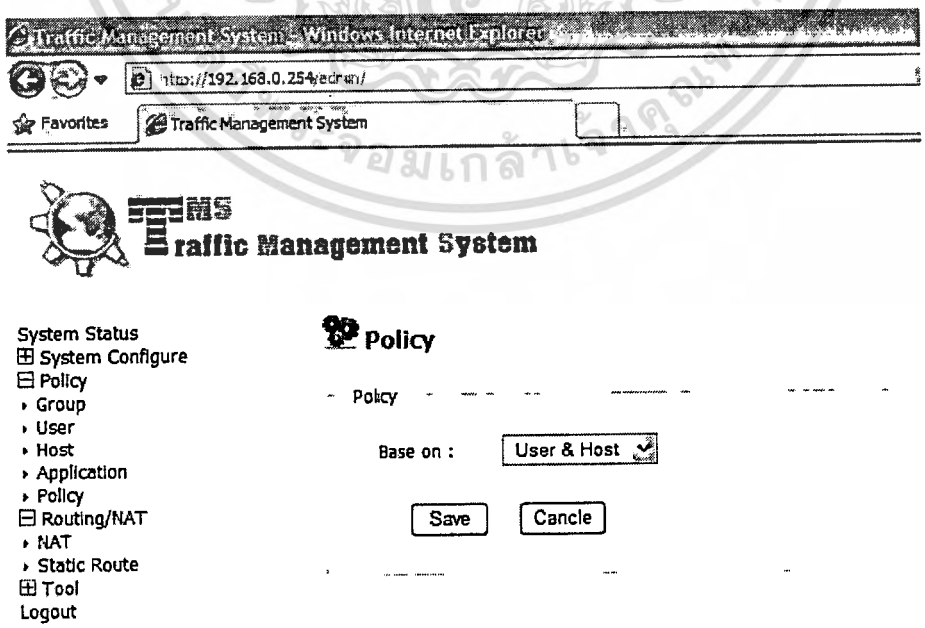


รูปที่ 6.4 ผลการดาวน์โหลดโดยใช้ ftp ในการทดสอบรอบที่ 1

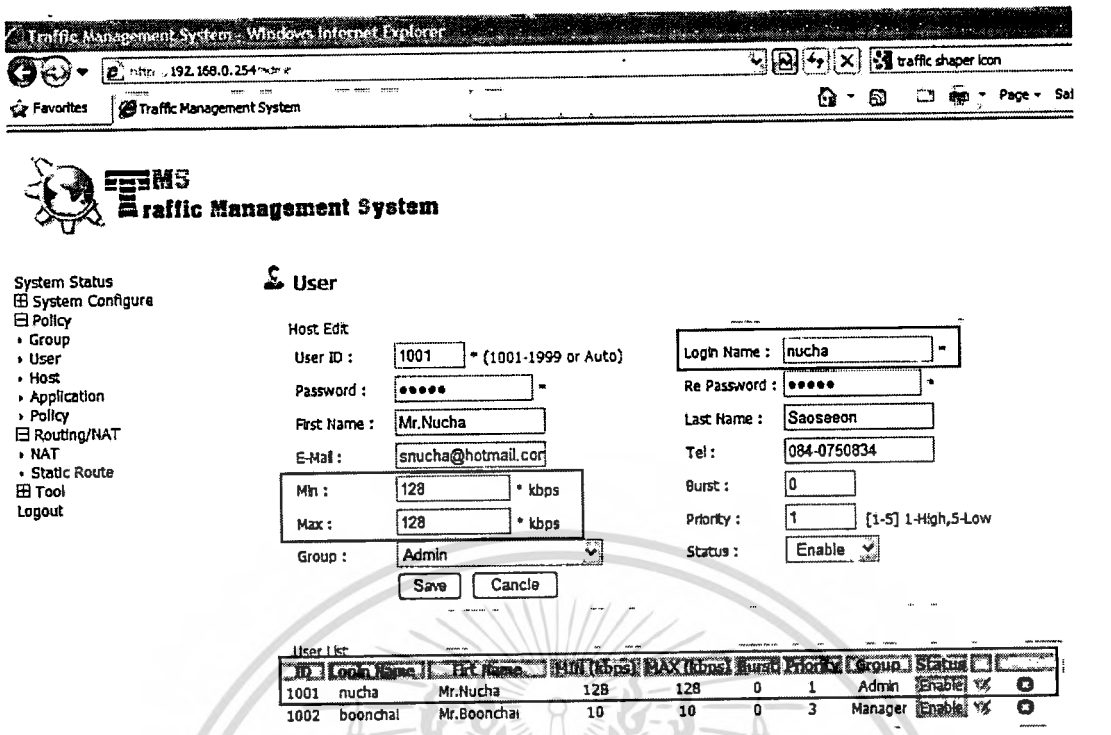
6.1.2 การทดสอบเปิดการจัดการ Policy แบบ User & Host

ในการทดสอบโดยการเปิด Policy แบบ User & Host จะเป็นการทดสอบว่าเมื่อมีการควบคุมแบนด์วิดท์ของผู้ใช้แล้ว ระบบสามารถที่จะควบคุมได้จริงตามที่ได้กำหนดไว้ ซึ่งการกำหนดค่าคอนฟิกระบบเป็นดังนี้

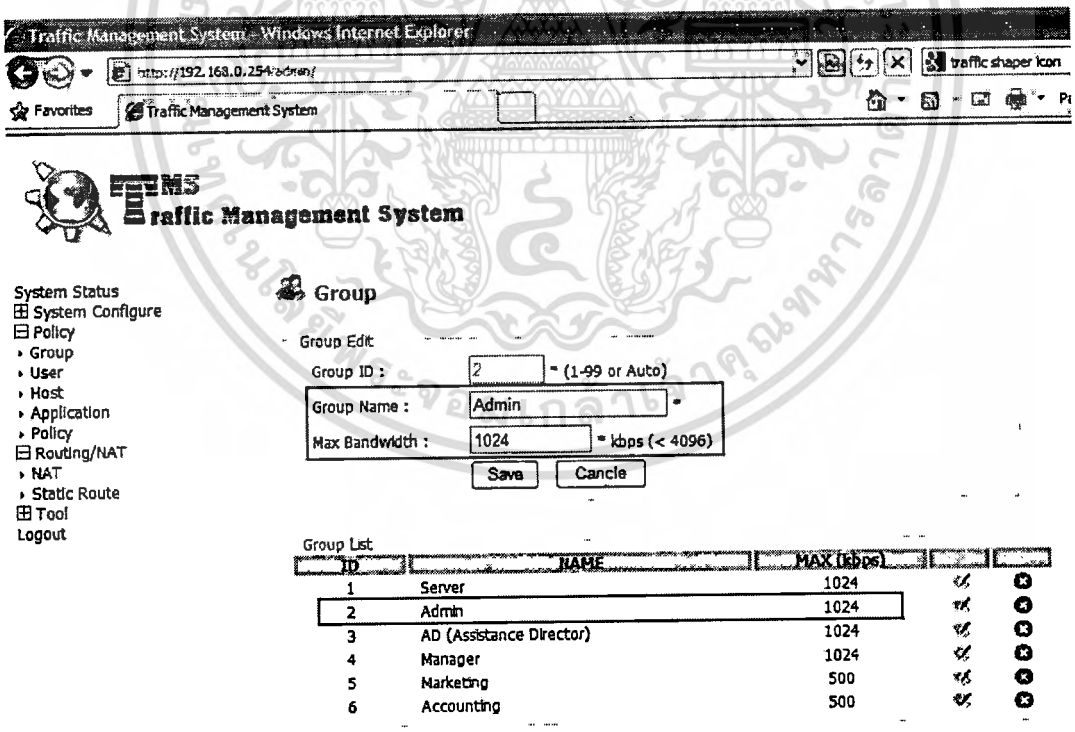
- Policy : User & Host
- User : nucha
- Max Bandwidth : 128kbps หรือ 16KB/s



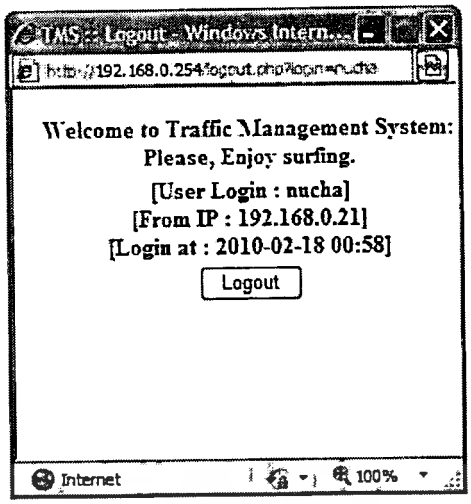
รูปที่ 6.5 การกำหนดค่า Policy ในการทดสอบรอบที่ 2



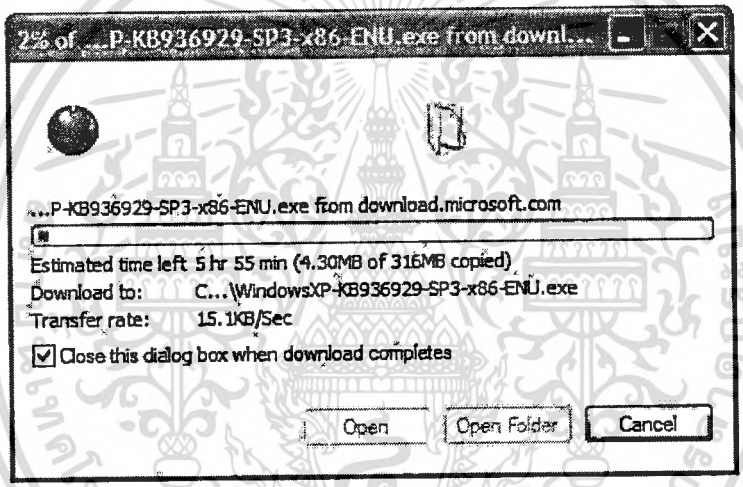
รูปที่ 6.6 การกำหนดค่า Max Bandwidth ของผู้ใช้ในการทดสอบรอบที่ 2



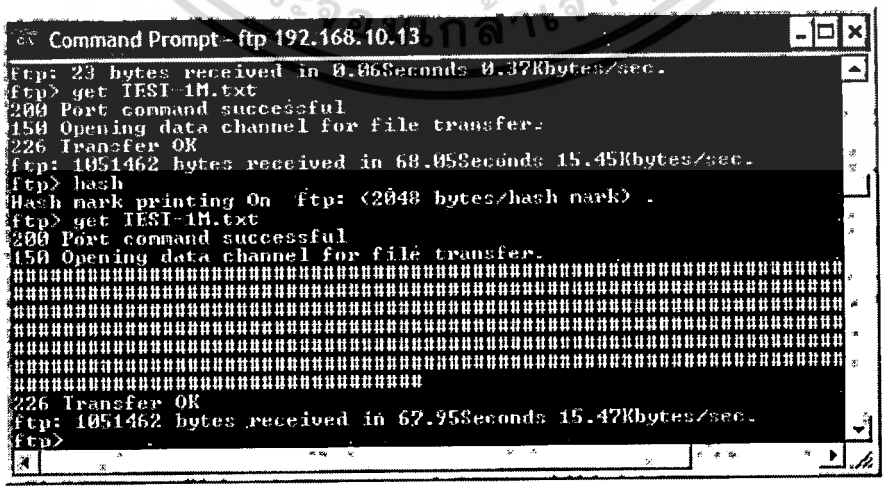
รูปที่ 6.7 การกำหนดค่า Max Bandwidth ของกลุ่มผู้ใช้ที่ผู้ใช้เป็นสมาชิกในการทดสอบรอบที่ 2



รูปที่ 6.8 ยืนยันการล็อกอินของผู้ใช้ในการทดสอบรอบที่ 2



รูปที่ 6.9 ผลการดาวน์โหลดโดยใช้ http ในการทดสอบรอบที่ 2

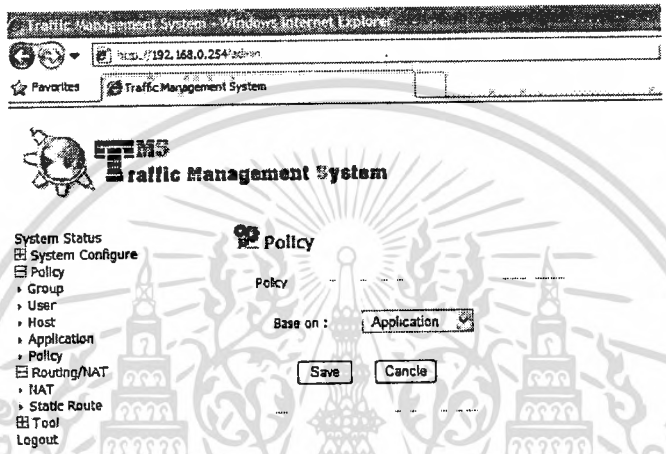


รูปที่ 6.10 ผลการดาวน์โหลดโดยใช้ ftp ในการทดสอบรอบที่ 2

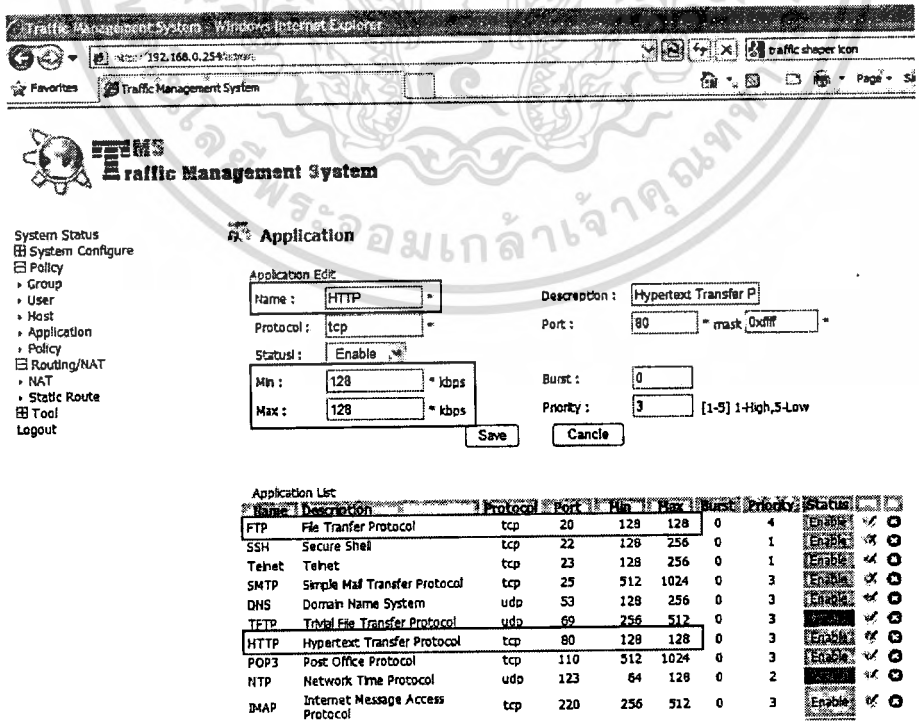
6.1.3 การทดสอบเปิดการจัดการ Policy แบบ Application

ในการทดสอบโดยการเปิด Policy แบบ Application จะเป็นการทดสอบว่าเมื่อมีการควบคุมแบนด์วิดท์ของแต่ละแอพพลิเคชันแล้ว ระบบสามารถที่จะควบคุมได้จริงตามที่ได้กำหนดไว้ ซึ่งการกำหนดค่าคอนฟิคระบบเป็นดังนี้

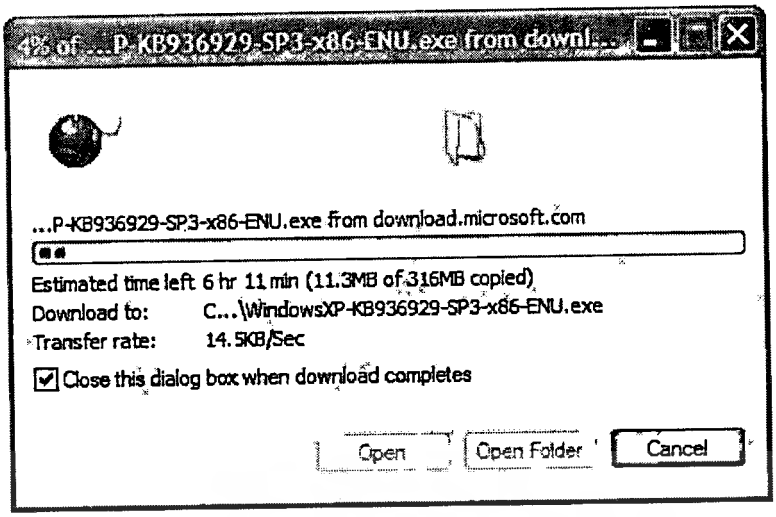
- Policy : Application
- Http Max Bandwidth : 128kbps หรือ 16KB/s
- Ftp Max Bandwidth : 128kbps หรือ 16KB/s



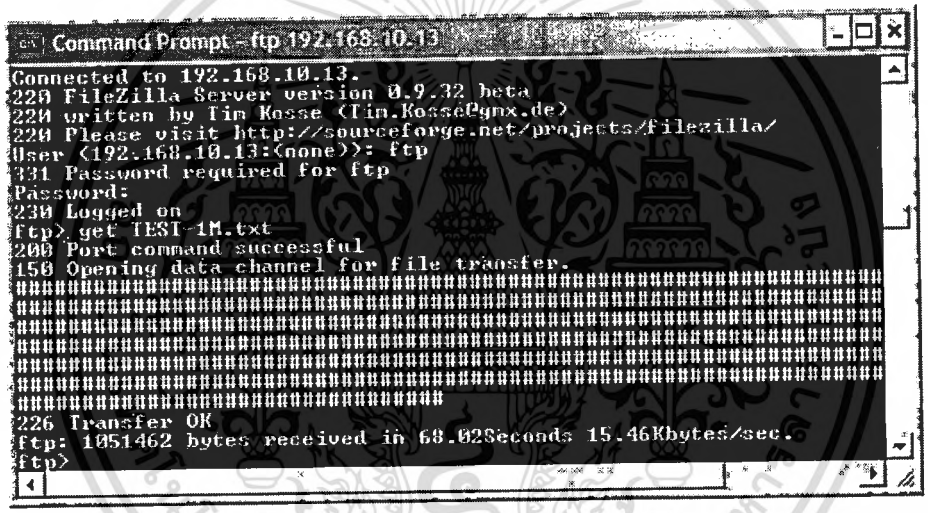
รูปที่ 6.11 การกำหนดค่า Policy ในการทดสอบรอบที่ 3



รูปที่ 6.12 การกำหนดค่า Max Bandwidth ของ http และ ftp ในการทดสอบรอบที่ 3



รูปที่ 6.13 ผลการดาวน์โหลดโดยใช้ http ในการทดสอบรอบที่ 3



รูปที่ 6.14 ผลการดาวน์โหลดโดยใช้ ftp ในการทดสอบรอบที่ 3

ตารางที่ 6.1 รายละเอียดผลการทดสอบระบบ

การกำหนดค่าการทดสอบ			http			ftp		
			KB/s	kbps	รูปที่อ้างอิง	KB/s	kbps	รูปที่อ้างอิง
รอบการทดสอบที่ 1	Policy	Disable	73.5	588	6.2	606.38	4851.04	6.3
	User maximum bandwidth	N/A						
	http maximum bandwidth	N/A						
	ftp maximum bandwidth	N/A						
รอบการทดสอบที่ 2	Policy	User & Host	15.1	120.8	6.8	15.47	123.76	6.9
	User maximum bandwidth	128kbps						
	http maximum bandwidth	N/A						
	ftp maximum bandwidth	N/A						
รอบการทดสอบที่ 3	Policy	Application	14.5	116	6.12	15.46	123.68	6.13
	User maximum bandwidth	N/A						
	http maximum bandwidth	128kbps						
	ftp maximum bandwidth	128kbps						

6.2 สรุปผลการทดสอบระบบ

จากข้อมูลการทดสอบที่ผ่านมาก่อนหน้าได้ทำการรวบรวมผลของการทดสอบไว้ในตารางที่ 6.1 จะเห็นได้ว่าเมื่อทำการกำหนด Policy ให้ระบบทำการจำกัดการใช้งานแบนด์วิดท์ ไม่ว่าจะเป็นแบบ User & Host หรือแบบ Application จะเห็นได้ว่าระบบสามารถควบคุมการใช้งานแบนด์วิดท์ได้ใกล้เคียงกับค่าที่กำหนดไว้มาก โดยที่ไม่เกินค่าที่กำหนดไว้ ดังนั้นจึงสามารถสรุปได้ว่าระบบจัดการและควบคุมข้อมูลในเครือข่ายนี้สามารถทำงานได้อย่างถูกต้อง

6.3 ข้อจำกัดและข้อเสนอแนะ

จากการศึกษาและพัฒนาอุปกรณ์ในการจัดการและควบคุมข้อมูลในเครือข่าย พบว่ายังคงมีข้อจำกัด แต่สามารถเพิ่มคุณสมบัติบางประการ เพื่อลดข้อจำกัดและพัฒนาให้ระบบมีประสิทธิภาพเพิ่มสูงขึ้นได้ โดยสามารถสรุปประเด็นต่างๆ ได้ดังนี้

1. ด้านประสิทธิภาพของระบบ เนื่องจากว่า iBoard มีฮาร์ดแวร์ที่จำกัดไม่สามารถขยายได้จึงทำให้มีข้อจำกัดของประสิทธิภาพในการประมวลผลที่มีความซับซ้อน เช่นการเปิดบริการ

SSL จะทำให้ระบบทำงานช้าลงอย่างเห็นได้ชัดเนื่องจากต้องประมวลผลในเรื่องของการเข้าและถอดรหัสข้อมูล, เมื่อระบบมีข้อมูลวิ่งผ่านระบบเป็นจำนวนมากก็จะทำให้การทำงานช้าลงบ้างเนื่องจากต้องเอาข้อมูลทุกแพคเกตมาตรวจสอบเพื่อส่งต่อ

2. ความสามารถในการขยายได้ของระบบ เนื่องจาก iBoard มีพื้นที่จัดเก็บข้อมูลที่มีเพียง 8MB จึงไม่สามารถที่จะนำแอพพลิเคชันที่มีขนาดใหญ่ และไม่สามารถจัดการเกี่ยวกับการเก็บ Log ไว้ในตัวเองได้ แนวทางที่สามารถจะพอแก้ไขได้โดยการส่ง Log ไปยังระบบ Log Server เพื่อจัดเก็บแทนเพื่อให้ระบบสามาถ ทำงานได้ตาม พรบ. หรือจะเลือกใช้ iBoard ที่มี 2 เน็ตเวิร์คอินเทอร์เฟซ และมีช่องขยายแบบ USB มาเชื่อมต่อกับสื่อเก็บข้อมูลภายนอกก็จะสามารถแก้ปัญหาได้

3. ความสามารถในการแสดงรายงาย เนื่องจากโครงการพัฒนาระบบในครั้งนี้ไม่ได้เน้นในส่วนของรายงานแต่จะเน้นเรื่องการจัดการและควบคุม จึงให้ส่วนแสดงข้อมูลให้ผู้ดูและระบบมีน้อยและไม่ดีเท่าทีควร ถ้าหากมีการพัฒนาต่อในส่วนของรายงานก็จะทำให้ระบบทำงานได้มีประสิทธิภาพและสมบูรณ์มากยิ่งขึ้น ในการพัฒนาต่อยอดระบบการจัดการและควบคุมข้อมูลในเครือข่าย นี้นั้นนอกจากในเรื่องของ Log และรายงานแล้วสามารถพัฒนาต่อในการควบคุมให้ละเอียดขึ้นเช่น ในส่วนผู้ใช้แต่ละคนสามารถกำหนดต่อได้ว่าให้ใช้แต่ละแอพพลิเคชันได้เท่าไหร่จะทำให้การควบคุมทำได้มีประสิทธิภาพมากยิ่งขึ้น

บรรณานุกรม

บัญชา ปะสีละเตสัง. 2551. **พัฒนาเว็บด้วยเทคนิค Ajax และ PHP.** กรุงเทพฯ : ซีเอ็ดยูเคชั่น.

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

สมศักดิ์ โชคชัยชุติกุล. 2550. **insight PHP ฉบับสมบูรณ์.** กรุงเทพฯ : โปรวิชั่น.

Machtelt Garrels. 2008. **Bash Guide for Beginners.** [Online]. Available : http://www.tldp.org/LDP/Bash-Beginners-Guide/Bash-Beginners-Guide.pdf

Martin A. Brown. 2006. **Traffic Control HOWTO.** [Online]. Available: http://tldp.org/HOWTO/Traffic-Control-HOWTO/index.html

Martin Devera aka devik. 2002. **HTB Linux queuing discipline manual - user guide.** [Online]. Available : http://luxik.cdi.cz/~devik/qos/htb/userg.pdf

Milan P. Stanic. 2001. **tc - traffic control (Linux QoS control tool).** [Online]. Available : http://www-igm.univ-mlv.fr/~badis/IR3/QoS/TP/linux-tc-english.pdf

ภาคผนวก

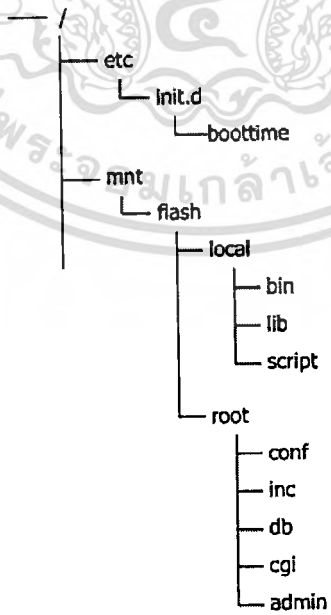
โครงสร้างระบบ และ Shell Script

ในการพัฒนาอุปกรณ์ในการจัดการและควบคุมข้อมูลในเครือข่ายนั้น พัฒนาอยู่บนระบบปฏิบัติการลินุกซ์ จึงเป็นสิ่งที่หลีกเลี่ยงไม่ได้ที่จะต้องใช้ Shell Script เพราะการสั่งงานต่างๆ ของลินุกซ์นั้นจะผ่านทาง Shell เป็นหลัก ในบทนี้จะไม่ได้กล่าวถึงวิธีการเขียน Shell Script เพราะเอกสารที่สอนเกี่ยวกับ Shell Script นั้นมีอยู่มากมายทั้งในรูปแบบหนังสือและบทความบนอินเทอร์เน็ต แต่จะกว่าถึงการออกแบบโครงสร้างของระบบว่ามีโครงสร้างอย่างไร และ Shell Script แต่ละ Script นั้นช่วยจำการอะไรในระบบการจัดการและควบคุมข้อมูลในเครือข่ายนี้ เนื้อหาจะแบ่งเป็นส่วน 3 ส่วนคือ

1. โครงสร้างไดเร็คทอรีที่เก็บข้อมูลของระบบว่าแต่ละส่วนเก็บข้อมูลอะไรบ้าง
2. Shell Script ต่างที่มีอยู่นั้นทำหน้าที่อะไรบ้าง
3. แสดง Source Code ของ Shell Script ที่จำเป็นสำหรับระบบ

1. โครงสร้างการจัดเก็บข้อมูลของระบบ

ในหัวข้อนี้ไม่ได้กว่าถึงโครงสร้างการจัดเก็บข้อมูลทั้งหมดของระบบปฏิบัติการลินุกซ์แต่จะกล่าวถึงเพียงโครงสร้างการจัดเก็บที่ระบบการจัดการและควบคุมข้อมูลในเครือข่ายเท่านั้น



รูปที่ 1 โครงสร้างการจัดเก็บข้อมูลระบบ

ตารางที่ 1.

ไดเร็คทอรี	/etc/init.d/boottime
หน้าที	เป็นไดเร็คทอรีของระบบ เมื่อระบบเปิดการทำงานเมื่อโหลดการทำงานเสร็จสิ้นก็จะมา run ไฟล์ Shell Script ที่มีอยู่ในไดเร็คทอรีนี้ทั้งหมด คล้าย autoexec.bat ของระบบปฏิบัติการวินโดวส์
ไฟล์ที่จัดเก็บ	system.init

ตารางที่ 2.

ไดเร็คทอรี	/mnt/flash
หน้าที	เป็นไดเร็คทอรีเดียวของระบบที่สามารถเขียนข้อมูลลงไปได้เปรียบเสมือนเป็นส่วน Harddisk ของระบบการพัฒนาระบบจะทำอยู่ในไดเร็คทอรีนี่เป็นหลัก
ไฟล์ที่จัดเก็บ	-

ตารางที่ 3.

ไดเร็คทอรี	/mnt/flash/local/bin
หน้าที	เป็นไดเร็คทอรีที่เป็น exec ไฟล์ที่เพิ่มเข้าไปในระบบ
ไฟล์ที่จัดเก็บ	tc, ip

ตารางที่ 4.

ไดเร็คทอรี	/mnt/flash/local/lib
หน้าที	เป็นไดเร็คทอรีที่เป็นไดร์เวอร์ของอุปกรณ์ใหม่ที่เพิ่มเข้าระบบ
ไฟล์ที่จัดเก็บ	dm9601.ko

ตารางที่ 5.

ไดเร็คทอรี	/mnt/flash/local/script
หน้าที	เป็นไดเร็คทอรีที่เก็บ Shell Script ที่สร้างขั้นมาใช้กับการพัฒนาระบบโดย Shell Script ที่นี่จะไม่ถูกเรียกใช้จากเว็บอินเทอร์เฟซ
ไฟล์ที่จัดเก็บ	create_policy_script.php, create_route_script.php, system.conf , create_system_conf.php, init.sh, iptables_init.sh, policy_script.sh, route_script.sh, user_policy.sh

ตารางที่ 6.

ไดเร็คทอรี	/mnt/flash/root/conf
หน้าที	เป็นไดเร็คทอรีที่เก็บค่าคอนฟิคต่างของระบบ
ไฟล์ที่จัดเก็บ	bw.down, bw.up, hostname, nat.enable, net.eth0, net.eth1, policy.enable, system.info, timeserver.conf, udhcpd.conf, udhcpd.enable, usertimeout.conf

ตารางที่ 7.

ไดเร็คทอรี	/mnt/flash/root/inc
หน้าที	เป็นไดเร็คทอรีที่เก็บไฟล์ ไฟล์ฟังก์ชันของ php ที่ระบบต้องเรียกใช้บ่อยๆ จากไฟล์ php อื่นๆ
ไฟล์ที่จัดเก็บ	config.inc.php, redirect.inc.php, util.inc.php

ตารางที่ 8.

ไดเร็คทอรี	/mnt/flash/root/db
หน้าที	เป็นไดเร็คทอรีที่เก็บฐานข้อมูลของระบบ
ไฟล์ที่จัดเก็บ	tms.db

ตารางที่ 9.

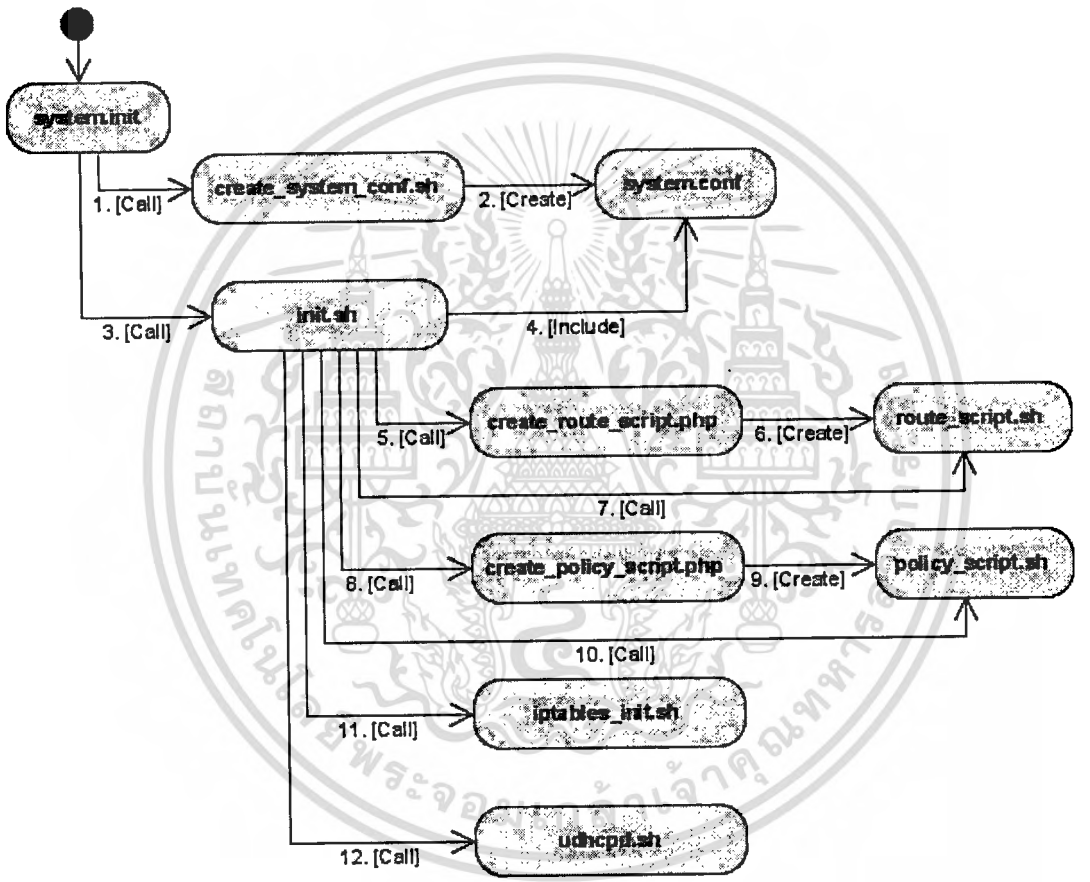
ไดเร็คทอรี	/mnt/flash/root/cgi
หน้าที	เป็นไดเร็คทอรีที่เก็บ Shell Script ที่ต้องถูกไฟล์ php เรียกใช้
ไฟล์ที่จัดเก็บ	date.sh, ifconfig.sh, repassword.sh, routegw.sh, showtim.sh, udhcpd.sh

ตารางที่ 10.

ไดเร็คทอรี	/mnt/flash/root/admin
หน้าที	เป็นไดเร็คทอรีที่หลักที่เก็บเว็บอินเทอร์เฟสสำหรับผู้ดูแลระบบ ซึ่งพัฒนาด้วย php
ไฟล์ที่จัดเก็บ	ฯลฯ

2. การทำงานของ Shell Script ในระบบ

จากที่ทราบไปก่อนหน้านี้แล้วว่าถ้าต้องการให้ Shell Script ทำงานทุกครั้งเมื่อเปิดระบบก็ต้องนำไปเก็บไว้ที่ไดเร็คทอรี /etc/init.d/boottime ในการพัฒนาระบบการจัดการและควบคุมข้อมูลในเครือข่ายก็เช่นกัน มีการสร้าง Shell Script เก็บไว้ที่ /etc/init.d/boottime เพื่อเป็นการเริ่มระบบโดยไฟล์ชื่อ system.init ซึ่งไฟล์ system.init นี้ก็จะเรียก Shell Script อื่นต่อไปตามค่าคอนฟิคที่ระบบกำหนดไว้จะกว่าระบบจะโหลดหารทำงานเสร็จสมบูรณ์ ตามรูปที่ 2.



รูปที่ 2 ลำดับการทำงานของ Shell Script ระบบ

จากรูปสามารถอธิบายขั้นตอนการทำงานได้ดังนี้

1. system.init เรียกให้ create_system_conf.sh ทำงานซึ่งการทำงานของ create_system_conf.sh นั้นจะไปรวบรวมค่าคอนฟิคต่างของระบบที่อยู่ใน /mnt/flash/root/conf นั้นมาแปลงเป็นค่าตัวแปลที่ Shell Script อื่นสามารถนำค่าคอนฟิคนั้นไปใช้ได้ง่าย โดยการ run (. [space] /path/file) ไฟล์คอนฟิคนั้นในสิ่งแวดล้อม

ของ Shell (Shell Environment) Shell Script ก็จะสามารถอ้างถึงค่าตัวแปลเหล่านั้นได้

2. create_system_conf.sh สร้างไฟล์คอนฟิคของระบบออกมาเป็นไฟล์ system.conf
3. system.init เรียกให้ init.sh ทำงาน
4. init.sh เรียกค่าคอนฟิคต่างๆ ของระบบจากไฟล์ system.conf ที่ถูกสร้างไว้มาอยู่ใน Shell Environment ของตัวเอง
5. init.sh เรียกให้ create_route_script.php ทำงานโดย create_route_script.php นี้จะทำการดึงข้อมูลจากฐานข้อมูล (tms.db) ในส่วนของตาราง routing มาสร้างเป็น Shell Script เพื่อที่จะทำการเพิ่มเราท์ติงเข้าระบบ
6. create_route_script.php สร้างไฟล์ route_script.sh ขึ้นมา
7. init.sh เรียกให้ route_script.sh ทำงานเพื่อเพิ่มเราท์ติงเข้าระบบ
8. init.sh เรียกให้ create_policy_script.php ทำงานโดย create_policy_script.php จะทำการตรวจสอบว่า ณ ตอนนี้ระบบถูกกำหนด policy เป็นแบบใดอยู่ [Diable/User&Host/Application] แล้วจะทำการดึงข้อมูลจากฐานข้อมูล (tms.db) ที่ตรงกับ policy นั้นๆ มาสร้างเป็น Shell Script
9. create_policy_script.php สร้างไฟล์ policy_script.sh ขึ้นมา
10. init.sh เรียกให้ iptables_init.sh ทำงานโดยการทำงานของ iptables_init.sh นั้นก็จะทำการตรวจสอบก่อนว่าระบบระบบถูกกำหนด policy เป็นแบบใดอยู่ แล้วจะทำการ run Shell Script ให้เหมาะสมตามที่ได้กำหนดไว้
11. init.sh เรียกให้ udhcpd.sh ทำงานโดยการทำงาน udhcpd.sh จะตรวจสอบก่อนว่าระบบถูกกำหนดให้มีการเปิดใช้งาน dhcp service หรือไม่ และจะทำงานตามที่กำหนดไว้

3. รหัสต้นฉบับของ Shell Script ที่สำคัญในระบบ

3.1 system.init

```
#!/bin/sh

/mnt/flash/local/script/create_system_conf.sh
/mnt/flash/local/script/init.sh all
```

3.2 create_system_conf.sh

```
#!/bin/sh

CPATH="/mnt/flash/root/conf/"
OUTPATH="/mnt/flash/local/script/"

echo "#!/bin/sh\n" > ${OUTPATH}system.conf
echo "# system.conf automatic create by script file 'create_system_conf.sh'\n" >>
${OUTPATH}system.conf

echo "INSIDE_INT=\"eth0\"\n" >> ${OUTPATH}system.conf
echo "OUTSIDE_INT=\"eth1\"\n" >> ${OUTPATH}system.conf

TMP=`cat $CPATH/nat.enable`
echo "NAT=\"$TMP\"" >> ${OUTPATH}system.conf

TMP=`cat $CPATH/bw.up`
echo "BWUP=\"$TMP\"" >> ${OUTPATH}system.conf

TMP=`cat $CPATH/bw.down`
echo "BWDOWN=\"$TMP\"" >> ${OUTPATH}system.conf

TMP=`cat $CPATH/policy.enable`
echo "POLICY=\"$TMP\"" >> ${OUTPATH}system.conf

TMP=`cat $CPATH/timeserver.conf`
echo "TSERVER=\"$TMP\"" >> ${OUTPATH}system.conf

TMP=`cat $CPATH/udhcpd.enable`
echo "UDHCPD=\"$TMP\"" >> ${OUTPATH}system.conf

TMP=`cat $CPATH/usertimeout.conf`
echo "USERTIMEOUT=\"$TMP\"" >> ${OUTPATH}system.conf

TMP=$(cat $CPATH/net.eth0 | cut -d ' ' -f1)
echo "IPETH0=\"$TMP\"" >> ${OUTPATH}system.conf

TMP=$(cat $CPATH/net.eth0 | cut -d ' ' -f2)
echo "MASKETH0=\"$TMP\"" >> ${OUTPATH}system.conf

TMP=$(cat $CPATH/net.eth1 | cut -d ' ' -f1)
echo "IPETH1=\"$TMP\"" >> ${OUTPATH}system.conf

TMP=$(cat $CPATH/net.eth1 | cut -d ' ' -f2)
echo "MASKETH1=\"$TMP\"" >> ${OUTPATH}system.conf

TMP=$(cat $CPATH/net.eth1 | cut -d ' ' -f3)
echo "GW=\"$TMP\"" >> ${OUTPATH}system.conf

chmod +x ${OUTPATH}system.conf
```

3.3 system_conf.sh

```
#!/bin/sh

# system.conf automatic create by script file 'create_system_conf.sh'

INSIDE_INT="eth0"

OUTSIDE_INT="eth1"

NAT="1"
BWUP="1024"
BWDOWN="4096"
POLICY="1"
TSERVER="time.nist.gov"
UDHCPD="1"
USERTIMEOUT="5"
IPETH0="192.168.0.254"
MASKETH0="255.255.255.0"
IPETH1="192.168.10.254"
MASKETH1="255.255.255.0"
GW="192.168.10.3"
```

3.4 init.sh

```
#!/bin/sh
. /mnt/flash/local/script/system.conf

route_add() {
  /usr/bin/php /mnt/flash/local/script/create_route_script.php
  sleep 2
  /mnt/flash/local/script/route_script.sh
}

policy_init() {
  /mnt/flash/local/script/create_policy_script.php
  sleep 2
  /mnt/flash/local/script/policy_script.sh
}

iptables_init() {
  /mnt/flash/local/script/iptables_init.sh
}

set_ip_eth0() {
  /sbin/ifconfig eth0 $IPETH0 netmask $MASKETH0
  echo "\nEth0 IP = ${IPETH0}"
  echo "Eth0 Netmask = ${MASKETH0}"
  echo "Eth0 ip config done..."
}
```

3.4 (ต่อ)

```
set_ip_eth1() {
  /sbin/ifconfig eth1 $IPETH1 netmask $MASKETH1
  echo "\nEth1 IP = ${IPETH1}"
  echo "Eth1 Netmask = ${MASKETH1}"
  echo "Eth1 ip config done.."
}

set_gw(){
  /sbin/route add default gw $GW
  echo "\nDefault Gateway = ${GW}"
  echo "Default gateway add done..."
}

syntime() {
  /usr/bin/ntpclient -l -h ${TSERVER} -c 1 -s
  echo "\nNTP service start..."
}

udhcpd() {
  /mnt/flash/root/cgi/udhcpd.sh mode
}

case $1 in
  route)
    route_add    ;;

  eth0)
    set_ip_eth0    ;;

  eth1)
    set_ip_eth1    ;;

  gw)
    set_gw    ;;

  ntp)
    syntime    ;;

  dhcp)
    udhcpd    ;;

  policy)
    policy_init    ;;

  iptables)
    iptables_init    ;;
```

3.4 (ต่อ)

```
  all)
    set_ip_eth0
    sleep 1
    set_ip_eth1
    sleep 1
    set_gw
    sleep 1
    route_add
    sleep 1
    udhcpd
    policy_init
    iptables_init
    syntime
    ;;

  *)
    echo ""
    echo "init.sh [eth0|eth1|gw|route|ntp|dhcp|policy|iptables|all]"
    echo ""
    ;;
esac
exit 0
```

3.5 create_route_script.php

```
#!/usr/bin/php

<?php

$dbpath = "/mnt/flash/root/db/tms.db";
$froute = "/mnt/flash/local/script/route_script.sh";

$fp = @fopen($froute,"w");
fputs($fp,"#!/bin/sh\n\n");
fputs($fp,"# route_script.sh create from create_route_script.php it read routing
from datbase (tms.db)\n");

$db = new SQLiteDatabase($dbpath);
$result = $db->query("select * from routing ORDER BY dest");

while ($result->valid()) {
        $row = $result->current();
        $route = "/sbin/route add -net " . $row["dest"] . " netmask " .
$row["mask"] . " gw " . $row["gw"] . "\n";
        fputs($fp,$route);
```

3.5 (ต่อ)

```
        $result->next();
}
fputs($fp,"\necho 'Routing add done...'\n");
unset($db);
@fclose($fp);

`chmod +x $froute`;
```

3.6 route_script.sh

```
#!/bin/sh

# route_script.sh create from create_route_script.php it read routing from datbase
(tms.db)
/sbin/route add -net 192.16.0.0 netmask 255.255.0.0 gw 192.168.0.10
/sbin/route add -net 192.168.100.0 netmask 255.255.255.0 gw 192.168.0.100

echo 'Routing add done...'
```

3.7 create_policy_script.php

```
#!/usr/bin/php

<?php
include_once ("/mnt/flash/root/inc/util.inc.php");

function initShellScript ($fp, $polConf) {
        fputs($fp,"#!/bin/sh\n\n");
        fputs($fp,"# policy_script.sh\n");
        fputs($fp,"# This file create auto with create_policy_script.php\n\n");
        fputs($fp,"TC=\"/mnt/flash/local/bin/tc\"\n\n");

        fputs($fp,"\${TC} qdisc del dev eth0 root\n");

        if ($polConf != "0") {
                fputs($fp,"\${TC} qdisc add dev eth0 root handle 1: htb\n");

                getBWConfig ($bwup, $bwdown);
                $tmp = "\${TC} class add dev eth0 parent 1: classid 1:0 htb rate
" . $bwdown . "kbit ceil " . $bwdown . "kbit\n";
                fputs($fp,$tmp);
        }
}
```

3.7 (ต่อ)

```
function groupPolicy ($db, $fp) {
        $result = $db->query("SELECT * FROM grp");
        fputs($fp,"\n# Add group \n");
        while ($result->valid()) {
                $row = $result->current();
                $tmp = "\${TC} class add dev eth0 parent 1:0 classid 1:" .
$row['gid'] . " htb rate " . $row['ceil'] . "kbit ceil " . $row['ceil'] . "kbit\n";

                fputs($fp,$tmp);
                $result->next();
        }
}

function hostPolicy ($db, $fp) {
        $result = $db->query("SELECT * FROM host WHERE enable=1");
        fputs($fp,"\n# Add Host \n");
        while ($result->valid()) {
                $row = $result->current();

                if ($row['prio'] != "") { $prio = " prio " . $row['prio']; } else {
$prio = ""; }
                if ($row['burst'] != 0) { $burst = " burst " . $row['burst'] . "kbit";
} else { $burst = ""; }

                $class = "\${TC} class add dev eth0 parent 1:" . $row['gid'] . "
classid 1:" . $row['hid'] . " htb rate " . $row['rate'] . "kbit ceil " . $row['ceil'] . "kbit"
. $prio . $burst . "\n";

                $filter ="\${TC} filter add dev eth0 protocol ip parent 1:0 prio 1
u32 match ip dst " . $row['ip'] . " flowid 1:" . $row['hid'] . "\n";

                fputs($fp,$class);
                fputs($fp,$filter);

                $result->next();
        }
}

function applicationPolicy ($db, $fp) {
        $result = $db->query("SELECT * FROM app WHERE enable=1");
        fputs($fp,"\n# Add Application \n");
        while ($result->valid()) {
                $row = $result->current();

                if ($row['prio'] != "") { $prio = " prio " . $row['prio']; } else {
$prio = ""; }
```

3.7 (ต่อ)

```
                    if ($row['burst'] != 0) { $burst = " burst " . $row['burst'] . "kbit";
} else { $burst = ""; }

                    $class = "\${TC} class add dev eth0 parent 1:0 classid 1:" .
$row['aid'] . " htb rate " . $row['rate'] . "kbit ceil " . $row['ceil'] . "kbit" . $prio .
$burst . "\n";

                    $filter ="\${TC} filter add dev eth0 protocol ip parent 1:0 prio 1
u32 match ip sport " . $row['port'] . " " . $row['mask'] . " flowid 1:" . $row['aid'] .
"\n";

                    fputs($fp,$class);
                    fputs($fp,$filter);

                    $result->next();
          }
}


// Main program

$fPathOut = "/mnt/flash/local/script/policy_script.sh";
$fConfig = "/mnt/flash/root/conf/policy.enable";

// Read config of policy from [/mnt/flash/root/conf/policy.enable]
$fp = fopen ($fConfig,"r");
$polConf = trim(fgets($fp,100));
fclose($fp);

// Default comment for add to shell script file
$fp = @fopen ($fPathOut,"w");

initShellScript($fp, $polConf);

if ($polConf != "0") {
          dbConnect($db);
          if ($polConf == "1") {
                    // Add Group
                    groupPolicy($db,$fp);
                    // Add Host
                    hostPolicy($db,$fp);
          } elseif ($polConf == "2") {
                    // Add Aplication
                    applicationPolicy ($db, $fp);
          }
          unset($db);
```

3.7 (ต่อ)

```
}

@fclose($fp);

`chmod +x $fPathOut`;
?>
```

3.8 policy_script.sh

```
#!/bin/sh

# policy_script.sh
# This file create auto with create_policy_script.php

TC="/mnt/flash/local/bin/tc"

${TC} qdisc del dev eth0 root
${TC} qdisc add dev eth0 root handle 1: htb
${TC} class add dev eth0 parent 1: classid 1:0 htb rate 4096kbit ceil 4096kbit

# Add group
${TC} class add dev eth0 parent 1:0 classid 1:1 htb rate 1024kbit ceil 1024kbit
${TC} class add dev eth0 parent 1:0 classid 1:2 htb rate 1024kbit ceil 1024kbit
${TC} class add dev eth0 parent 1:0 classid 1:3 htb rate 1024kbit ceil 1024kbit
${TC} class add dev eth0 parent 1:0 classid 1:4 htb rate 1024kbit ceil 1024kbit
${TC} class add dev eth0 parent 1:0 classid 1:5 htb rate 500kbit ceil 500kbit
${TC} class add dev eth0 parent 1:0 classid 1:6 htb rate 500kbit ceil 500kbit

# Add Host
${TC} class add dev eth0 parent 1:1 classid 1:101 htb rate 256kbit ceil 512kbit
prio 3
${TC} filter add dev eth0 protocol ip parent 1:0 prio 1 u32 match ip dst
192.168.0.3 flowid 1:101
${TC} class add dev eth0 parent 1:3 classid 1:103 htb rate 256kbit ceil 1024kbit
prio 1
${TC} filter add dev eth0 protocol ip parent 1:0 prio 1 u32 match ip dst
192.168.0.200 flowid 1:103
```

3.9 iptables_init.sh

```
#!/bin/sh

# iptables_init.sh
# This file check if policy user User&Host [1] create firewall policy for filter
packet

. /mnt/flash/local/script/system.conf

nat_service() {
        if [ "$NAT" = "1" ] ; then
                echo "1" > /proc/sys/net/ipv4/ip_forward
                /sbin/iptables -t nat -A POSTROUTING --out-interface
${OUTSIDE_INT} -j MASQUERADE
        else
                echo "0" > /proc/sys/net/ipv4/ip_forward
        fi
}

# Start Shell Script.

/sbin/iptables -F           # clear forwarding table rule
/sbin/iptables -F -t nat    # clear nat table rule

if [ "$POLICY" = "1" ] ; then
        # redirect all web traffic to authentication page for cheke authorize first
        /sbin/iptables -t nat -A PREROUTING -i $INSIDE_INT -p udp --dport
53 -j ACCEPT
        /sbin/iptables -t nat -A PREROUTING -i $INSIDE_INT -d $IPETH0 -j
ACCEPT
        /sbin/iptables -t nat -A PREROUTING -i $INSIDE_INT -p tcp --dport 80
-j REDIRECT --to-port 80
        /sbin/iptables -t nat -A PREROUTING -i $INSIDE_INT -j DROP
fi

nat_service
```

3.10 udhcpd.sh

```
#!/bin/sh
# This file use by dhcp.php for start and stop dhcp service by web interface

# kill udhcpd service
stop() {
        PID=`cat /var/run/udhcpd.pid`
        kill -9 ${PID}
}

start() {
        /usr/sbin/udhcpd /mnt/flash/root/conf/udhcpd.conf
}

check_mode() {
        MODE=`cat /mnt/flash/root/conf/udhcpd.enable`
        if [ "$MODE" = "1" ]; then
                stop
                start
        else
                stop
        fi
}

case "$1" in
 stop)
   stop    ;;

 start)
   start   ;;

 restart)
   stop
   start   ;;

 mode)
   check_mode
   ;;

 *)
   echo ""
   echo "Usage: udhcpd.sh [start|stop|restart|mode]"
   echo ""
   ;;
esac
exit 0
```

3.11 user_policy.sh

```
#!/bin/sh
# user_policy.sh
# This script use by login.php for add authorize user
# This script use for init, add, delet authen user from web

. /mnt/flash/local/script/system.conf

TC="/mnt/flash/local/bin/tc"

ip=${2}
uid=${3}
gid=${4}
rate=${5}
ceil=${6}
prio=${7}
burst=${8}

case "$1" in
  add)
    # Add ip to firewall for permit traffic
    /sbin/iptables -t nat -I PREROUTING -s ${ip}/32 -i $INSIDE_INT -j ACCEPT

    # Add policy to htb for controll traffic
    if [ "$prio" != "" ] ; then
      prio=" prio ${prio} "
    else
      prio=" "
    fi
    if [ "$burst" = "0" ] ; then
      burst=" "
    else
      burst=" burst ${burst}kbit "
    fi

    ${TC} class add dev eth0 parent 1:${gid} classid 1:${uid} htb rate ${rate}kbit
ceil ${ceil}kbit ${prio} ${burst}
    ${TC} filter add dev eth0 protocol ip parent 1:0 prio 1 u32 match ip dst ${ip}
flowid 1:${uid}
    ;;

  del)
    /sbin/iptables -t nat -D PREROUTING -s ${ip}/32 -i $INSIDE_INT -j
ACCEPT
    HD=`${TC} filter show dev eth0 | grep ${uid} | cut -d " " -f 10`
    ${TC} filter del dev eth0 parent 1:0 prio 1 handle ${HD} u32
    ${TC} class del dev eth0 parent 1:0 classid 1:${uid}
    ;;
```

3.11 (ต่อ)

```
 *)
   echo ""
   echo "${0} v0.1 (2009.09.09-06:33) by Nucha IS23.2"
   echo ""
   echo "Usage: ${0} [OPTION] IP_ADDRESS"
   echo "Options:"
   echo "       add     Add ip address to nat table"
   echo "       del   Delete ip address to nat table"
   echo ""
   exit 1
esac
exit 0
```

ประวัติผู้เขียน

ชื่อผู้จัดทำโครงงาน นายนุชา เสาสีอ่อน

วันเดือนปีเกิด 18 พฤษภาคม 2520

สถานที่เกิด ปัตตานี

ประวัติการศึกษา

มัธยมศึกษา โรงเรียนเดชะปัตตนยานุกูล ปัตตานี

อุดมศึกษา วท.บ. (วิทยาการคอมพิวเตอร์)
มหาวิทยาลัยแม่โจ้

ประวัติการทำงาน

พ.ศ. 2549 – ปัจจุบัน ตำแหน่ง Senior Engineer
บริษัททรูคอร์ปอเรชั่นจำกัด (มหาชน)