ระบบเฝ้าระวังและแจ้งเตือนสถานะของเครื่องแม่ข่ายโดยใช้เอสเอ็นเอ็มพี โพรโตคอลผ่านบริการข้อความด่วน

AN INTERACTIVE LINUX/UNIX SERVER MONITORING USING SNMP PROTOCOL VIA INSTANT MESSAGING SERVICE

กฤษา ไชยเจริญ

ธนชิต วิเชียรฉาย

อาจารย์ที่ปรึกษา

รศ.ดร. โชติพัชร์ ภรณวลัย

ปริญญานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรวิทยาศาสตรบัณฑิต

สาขาวิชาเทคโนโลยีสารสนเทศ

คณะเทคโนโลยีสารสนเทศ

สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

ภาคเรียนที่ 2 ปีการศึกษา 2553

ระบบเฝ้าระวังและแจ้งเตือนสถานะของเครื่องแม่ข่ายโดยใช้เอสเอ็นเอ็มพี โพรโตคอลผ่านบริการข้อความด่วน

AN INTERACTIVE LINUX/UNIX SERVER MONITORING USING SNMP PROTOCOL VIA INSTANT MESSAGING SERVICE

โดย

กฤษยา ไชยเจริญ

ธนชิต วิเชียรฉาย

อาจารย์ที่ปรึกษา

รศ.ดร. โชติพัชร์ ภรณวลัย

ปริญญานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรวิทยาศาสตรบัณฑิต

สาขาวิชาเทคโนโลยีสารสนเทศ

คณะเทคโนโลยีสารสนเทศ

สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

ภาคเรียนที่ 2 ปีการศึกษา 2553

AN INTERACTIVE LINUX/UNIX SERVER MONITORING USING SNMP PROTOCOL VIA INSTANT MESSAGING SERVICE

KRISA CHAIJAROEN

THANACHIT WICHIANCHAI

A PROJECT SUBMITTED IN PARTIAL FULFILLMENT

OF THE REQUIREMENT FOR THE DEGREE OF

BACHELOR OF SCIENCE PROGRAM IN INFORMATION TECHNOLOGY

FACULTY OF INFORMATION TECNOLOGY

KING MONGKUT'S INSTITUTE OF TECHNOLOGY LADKRABANG

2/2010

ใบรับรองปริญญานิพนธ์ ประจำปีการศึกษา 2553

คณะเทคโนโลยีสารสนเทศ

สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

เรื่อง ระบบเฝ้าระวังและแจ้งเตือนสถานะของเครื่องแม่ข่ายโดยใช้เอสเอ็นเอ็มพี โพรโตคอลผ่านบริการข้อความด่วน

AN INTERACTIVE LINUX/UNIX SERVER MONITORING USING SNMP PROTOCOL VIA INSTANT MESSAGING SERVICE

ผู้จัดทำ

1. นายกฤษา ไชยเจริญ รหัสนักศึกษา 50070023
2. นายธนชิต วิเชียรฉาย รหัสนักศึกษา 50070068

.....อาจารย์ที่ปรึกษา

(รศ. ดร. โชติพัชร์ ภรณวลัย)

หัวข้อโครงงาน	ระบบเฝ้าระวังและแจ้งเตือนสถานะของเครื่องแม่ข่าย โดยใช้เอสเอ็นเอ็มพีโพรโตคอลผ่านบริการข้อความด่วน
นักศึกษา	นายกฤษา ไชยเจริญ รหัสนักศึกษา 50070023 นายธนชิต วิเชียรฉาย รหัสนักศึกษา 50070068
ปริญญา	วิทยาศาสตรบัณฑิต
สาขาวิชา	เทคโนโลยีสารสนเทศ
ปีการศึกษา	2553
อาจารย์ที่ปรึกษา	รศ.ดร.โชติพัชร์ ภรณวลัย

บทคัดย่อ

โครงงานนี้ ศึกษาการพัฒนาระบบตรวจสอบเฝ้าระวัง และแจ้งเตือนสถานะของเครื่องแม่ข่าย โดยใช้ โปรโตคอล SNMP ผ่านข้อความด่วนทันใจ โดยออกแบบและพัฒนาบนพื้นฐานของประยุกต์ใช้ การตรวจสอบสถานะของเครื่องแม่ข่ายและอุปกรณ์เครือข่ายผ่านทาง Simple Network Management Protocol, การจัดการ system log ด้วย Syslog-NG และการติดต่อสื่อสารผ่านทาง instant messaging โดยนำข้อมูลเกี่ยวกับสถานะของเครื่องแม่ข่าย บริการบนเครื่องแม่ข่าย เหตุการณ์ต่างๆ ที่ได้จาก SNMP Trap และ Syslog-NG แจ้งเตือนไปยังดูแลระบบ ผ่านทางบริการข้อความด่วน (Instant messaging) เพื่อให้ผู้ดูแลระบบได้รับทราบถึง เหตุการณ์ผิดปกติและข้อผิดพลาด นอกจากนี้การพัฒนาระบบในลักษณะที่เป็น module สามารถช่วยเพิ่มความยืดหยุ่น ในระบบ ทำให้สามารถขยายขอบเขตและความสามารถเพิ่มเติมได้ในอนาคต

โดยในโครงงานนี้ คณะผู้จัดทำได้ทำการออกแบบและพัฒนาระบบทั้งสิ้น 5 โมดูล คือ การทำ Polling, การรับ SNMP Trap, การ query สถานะของเครื่องแม่ข่ายผ่าน SNMP Protocol, การแจ้งเตือน syslog-ng และส่วนของ Web Application สำหรับการตั้งค่าการทำงานของระบบ

Title	An interactive Linux/UNIX Server monitoring using SNMP Protocol via instant messaging service
Student	Mr. Krisa Chaijaroen Student ID 50070023
	Mr. Thanachit Wichianchai Student ID 50070068
Level of Study	Bachelor of Science in Information Technology
Major	Information Technology
Year	2010
Advisor	Assoc. Prof. Dr. Chotipat Pornavalai

ABSTRACT

A project, An interactive Linux/UNIX Server monitoring using SNMP Protocol via instant messaging service, was analyzed, designed and developed on the basis of a tedious task, monitoring networking devices. In addition, the project provides an interactive powerful tool for system and network administrators via, instant messaging, one of the most popular communication channels. The project uses standards protocol to gather network devices information. SNMP protocol is used to provide remote system's information and Syslog-NG is used for centralized system logging. The developed system will send an instant messaging to system and network administrators, when an important message was received. Moreover, the project was developed in modules based architecture to ensure the extensibility with new features.

The current project consists of five modules: SNMP Polling, SNMP Trap, SNMP Get, Syslog-NG and web based application for the system configuration.

กิตติกรรมประกาศ

โครงงานระบบเฝ้าระวังและแจ้งเตือนสถานะของเครื่องแม่ข่ายโดยใช้เอสเอ็นเอ็มพีโพรโตคอลผ่านบริการข้อความด่วน ดำเนินการสำเร็จลุล่วงไปได้ด้วยความช่วยเหลือและการสนับสนุนอย่างดียิ่ง ของ รองศาสตราจารย์ ดร. โชติพัชร์ ภรณวลัย อาจารย์ที่ปรึกษาโครงงาน และผู้ช่วยศาสตราจารย์ ธนิศา นุ่มนนท์ อาจารย์ที่ปรึกษาโครงงานเมื่อครั้งนำไปแข่งขันในโครงการแข่งขันพัฒนาโปรแกรมคอมพิวเตอร์แห่งประเทศไทยครั้งที่ 12 ที่ได้กรุณาให้ความรู้คำแนะนำ และข้อคิดเห็นต่างๆ ที่เป็นประโยชน์ต่อการพัฒนาระบบมาโดยตลอด

คณะผู้บริหารและเจ้าหน้าที่ งานควบคุมเครื่องและเครือข่าย สำนักบริการคอมพิวเตอร์ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง ทุกๆ ท่าน ที่ให้ความอนุเคราะห์ในการใช้อุปกรณ์สำหรับการพัฒนาระบบ ให้คำแนะนำสำหรับการพัฒนาระบบ และให้ความรู้เกี่ยวกับการดูแลระบบคอมพิวเตอร์แม่ข่ายและอุปกรณ์เครือข่ายเป็นอย่างดี

บิดา มารดา และครอบครัว ที่ให้การสนับสนุนในทุกๆ ด้าน

ผู้พัฒนาจึงขอกราบขอบพระคุณทุกท่านเป็นอย่างสูง มา ณ โอกาสนี้

กฤษฎา ไชยเจริญ

ธนชิต วิเชียรฉาย

สารบัญ

สารบัญ (ต่อ)

สารบัญตาราง

สารบัญรูป

สารบัญรูป (ต่อ)

สารบัญรูป (ต่อ)

สารบัญรูป (ต่อ)

บทที่ 1
บทนำ

ปัจจุบันเทคโนโลยีสารสนเทศได้เข้ามามีบทบาทกับองค์กรธุรกิจเป็นอย่างมากในทุกๆ ด้าน โดยถูกนำมาใช้เพื่อพัฒนาหรือเพิ่มประสิทธิภาพและความคล่องตัวของการปฏิบัติงาน การให้บริการ และการบริหารจัดการขององค์กร รวมทั้งเพิ่มขีดความสามารถในการแข่งขันให้กับองค์กรได้ด้วย นอกจากนี้ ระบบสารสนเทศยังช่วยให้การติดต่อสื่อสารมีความสะดวกและรวดเร็วมากขึ้น เช่น ระบบจดหมายอิเล็กทรอนิกส์ (Email) , ระบบ www, การสนทนาผ่านระบบเครือข่ายอินเทอร์เน็ต เช่น MSN, Skype , Google Talk. เป็นต้น การให้บริการระบบสารสนเทศต่างๆ เหล่านี้จะต้องใช้เครื่องคอมพิวเตอร์ที่ทำหน้าที่ในการให้บริการเรียกว่า เครื่องแม่ข่าย หรือเครื่องให้บริการ (Server) เช่น Email Server ,Proxy Server , Name Server ,Web Server, Application server , Database Server , Authentication Server เป็นต้น เครื่องคอมพิวเตอร์และซอฟท์แวร์ที่ทำหน้าที่ให้บริการเป็นเครื่องแม่ข่ายจะต้องมีประสิทธิภาพสูง มีเสถียรภาพ มีความสามารถให้บริการแก่ผู้ใช้ได้เป็นจำนวนมาก และสามารถให้บริการได้อยู่ตลอดเวลา ดังนั้นการดูแลรักษาระบบเครื่องแม่ข่ายให้สามารถให้มีประสิทธิภาพ เสถียรภาพ และความสามารถในการให้บริการได้ตลอดเวลานั้นจึงเป็นสิ่งที่จำเป็นและมีความสำคัญ โดยจะต้องอาศัยความรู้ ความสามารถและความรับผิดชอบของผู้ดูแลระบบ ในการออกแบบระบบ ควบคุมดูแล และบริหารจัดการระบบ ดังนั้นหากมี ซอฟท์แวร์หรือระบบที่ช่วยสนับสนุนการทำงานของผู้ดูแลระบบในการเฝ้าระวังและแจ้งเตือนสถานะของเครื่องแม่ข่ายและระบบสารสนเทศต่างๆ ก็จะสามารถทำให้การปฏิบัติงานของผู้ดูแลระบบ เป็นไปด้วยความสะดวก รวดเร็ว มีประสิทธิภาพมากขึ้นจากการที่บริการต่างๆ ถูกตรวจสอบ เฝ้าระวังและแจ้งเตือนไปยังผู้ดูแลระบบแบบทันเวลา ซึ่งจะทำให้ผู้ดูแลระบบสามารถทำการแก้ไขปัญหาที่เกิดขึ้นได้รวดเร็วมากยิ่งขึ้น ผลที่ได้ก็คือ จะสามารถเพิ่มความสามารถในการให้บริการเครื่องแม่ข่ายขององค์กร โดยสามารถลดระยะเวลาที่ระบบไม่สามารถให้บริการได้อันเนื่องมาจากข้อผิดพลาดของอุปกรณ์และซอฟท์แวร์

ในปริญญานิพนธ์ฉบับนี้ได้นำเสนอการพัฒนา ระบบตรวจสอบเฝ้าระวังและแจ้งเตือนสถานะของเครื่องแม่ข่ายโดยใช้โปรโตคอล SNMP ผ่านข้อความด่วนทันใจ โดยออกแบบและพัฒนาบนพื้นฐานของ ประยุกต์ใช้การตรวจสอบสถานะของเครื่องแม่ข่ายและอุปกรณ์เครือข่ายผ่านทาง Simple Network Management Protocol, การจัดการ system log ด้วย Syslog-NG และการติดต่อสื่อสารผ่านทาง instant messaging โดยนำข้อมูลเกี่ยวกับสถานะของเครื่องแม่ข่าย บริการบนเครื่องแม่ข่าย เหตุการณ์ต่างๆ ที่ได้จาก SNMP Trap และ Syslog-NG แจ้งเตือนไปยังดูแลระบบผ่านทางบริการข้อความด่วนทันใจ (Instant messaging) เพื่อให้ผู้ดูแลระบบได้รับทราบถึงเหตุการณ์

ผิดปกติ และข้อผิดพลาด และสามารถแก้ไขปัญหาต่างๆ ได้ทันท่วงที และสามารถเรียกดูข้อมูลสถานะการให้บริการของเครื่องแม่ข่ายและอุปกรณ์เครือข่ายเบื้องต้นแบบรวมศูนย์ (consolidation) ได้อีกด้วย

1.1 แนวคิดและที่มา

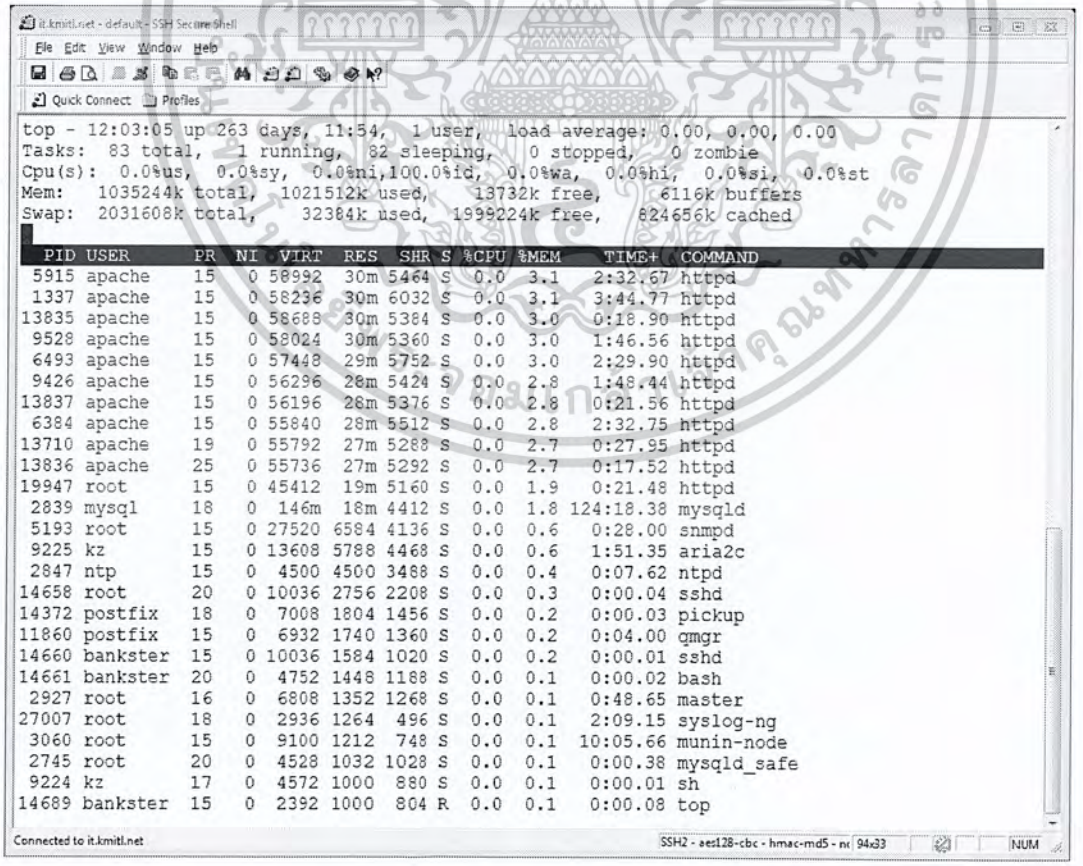
การตรวจสอบสถานะการให้บริการของเครื่องแม่ข่ายและอุปกรณ์เครือข่ายเป็นสิ่งที่มีความสำคัญต่อการให้บริการระบบสารสนเทศ เพราะจะให้ผู้ดูแลระบบทราบถึงสถานะของการให้บริการ เหตุการณ์ต่างๆ ว่าการให้บริการมีข้อผิดพลาดหรือสิ่งผิดปกติเกิดขึ้นหรือไม่ หากการตรวจสอบและเฝ้าระวัง เป็นไปโดยมีประสิทธิภาพและต่อเนื่องก็จะทำให้ผู้ดูแลระบบสามารถรับทราบสถานะโดยรวมของระบบได้ ในกรณีที่มีข้อผิดพลาดเกิดขึ้น ก็จะทำให้สามารถแก้ไขปัญหาได้รวดเร็ว ทันเวลา และถูกต้องมากยิ่งขึ้น ส่งผลให้การให้บริการระบบสารสนเทศเป็นไปอย่างต่อเนื่อง ถูกต้อง และมั่นคงปลอดภัย

ในการตรวจสอบสถานะของการให้บริการของเครื่องแม่ข่ายและอุปกรณ์เครือข่ายนั้น โดยปกติผู้ดูแลระบบจะต้อง login เข้าไปยังเครื่องแม่ข่ายหรืออุปกรณ์เครือข่ายเพื่อทำการตรวจสอบ system log, log file ของ application, สถานะของการใช้งานพื้นที่ disk, จำนวน tcp connection, สถานะของ IO, load average เป็นต้น (ดังรูปที่ 1.1 ที่แสดงการเรียกดูสถานะการใช้งานทรัพยากรของระบบด้วยคำสั่ง top และรูปที่ 1.2 แสดงการใช้คำสั่งเพื่อตรวจสอบสถานะของ disk, file system และ การเรียกดู system log file) หากตรวจพบปัญหาก็ทำการตรวจสอบสาเหตุและทำการแก้ไข วิธีการตรวจสอบนี้สามารถใช้งานได้กับระบบสารสนเทศขนาดเล็กที่มีจำนวนเครื่องแม่ข่ายและอุปกรณ์ไม่มากนัก หากเป็นระบบสารสนเทศขนาดใหญ่ที่มีสถาปัตยกรรมของการบริการที่ซับซ้อน มีความต้องการในด้านประสิทธิและเสถียรภาพของการประมวลผล รวมทั้งความมั่นคงปลอดภัยของระบบ จำนวนเครื่องแม่ข่ายและอุปกรณ์เครือข่ายและซอฟท์แวร์ที่ต้องดูแลรักษาก็จะเพิ่มขึ้นตามไปด้วย การตรวจสอบสถานะการให้บริการต่างๆ นั้น ต้องการทั้งความสม่ำเสมอและความต่อเนื่อง หากจำนวนเครื่องแม่ข่ายและอุปกรณ์เครือข่ายรวมทั้งซอฟท์แวร์มีจำนวนมาก อาจจะให้การดูแลรักษาและตรวจสอบไม่ทั่วถึงและไม่ครบถ้วน หากมีปัญหาเกิดขึ้น จะทำให้ระยะเวลาที่ใช้ในการรับทราบ ตรวจสอบและแก้ไขปัญหาเพิ่มมากขึ้นตามไปด้วย ถึงแม้จะมีระบบการแจ้งเตือนผ่านช่องทางต่างๆ เช่น Email, SMS เป็นต้น แต่ก็ต้องแลกมาด้วยค่าใช้จ่ายที่สูงขึ้นจากค่าลิขสิทธิ์และการบำรุงรักษา

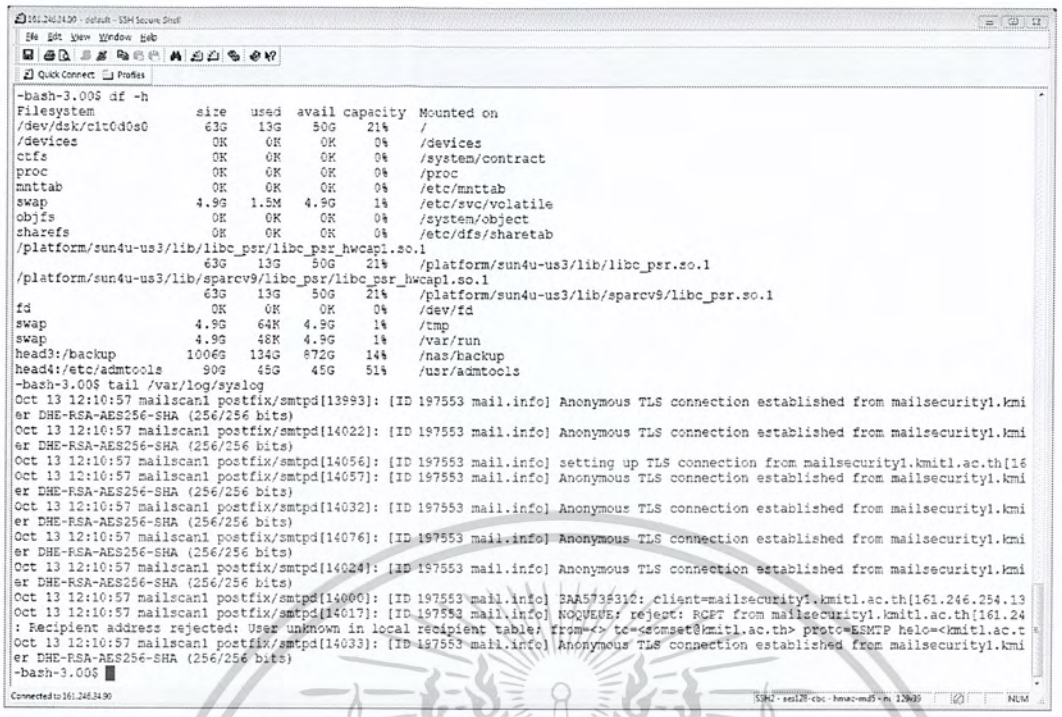
ดังนั้นจึงมีแนวคิดในการพัฒนาระบบตรวจสอบเฝ้าระวังและแจ้งเตือนสถานะของเครื่องแม่ข่ายโดยใช้โปรโตคอล SNMP ผ่านข้อความด่วนทันใจ (Instant Messaging) ดังแสดงในรูปที่ 1.3 โดยจะมีการรวบรวมข้อมูลสถานะการให้บริการของเครื่องแม่ข่ายและอุปกรณ์เครือข่ายด้วย SNMP

Protocol ซึ่งเป็น Protocol มาตรฐานสำหรับการบริหารจัดการอุปกรณ์เครือข่ายและเครื่องแม่ข่าย, ข้อมูลเหตุการณ์ของระบบจาก System Log โดยระบบที่พัฒนาจะนำข้อมูลดังกล่าวมาจัดรูปแบบให้เหมาะสม บันทึกลงในระบบฐานข้อมูล และแจ้งเตือนไปยังผู้ดูแลระบบผ่านทางบริการข้อความด่วนทันใจ (Instant messaging) ซึ่งทำงานบน XMPP Protocol ซึ่งเป็นโปรโตคอลมาตรฐานสำหรับการติดต่อสื่อสารด้วยข้อความแบบมีปฏิสัมพันธ์ที่มีประสิทธิภาพ และได้รับความนิยมอย่างแพร่หลายในผู้ให้บริการขนาดใหญ่ เช่น Facebook Chat, Google Talk เป็นต้น ซึ่งมีข้อดีกว่าการแจ้งเตือนด้วย Email และ SMS ดังนี้

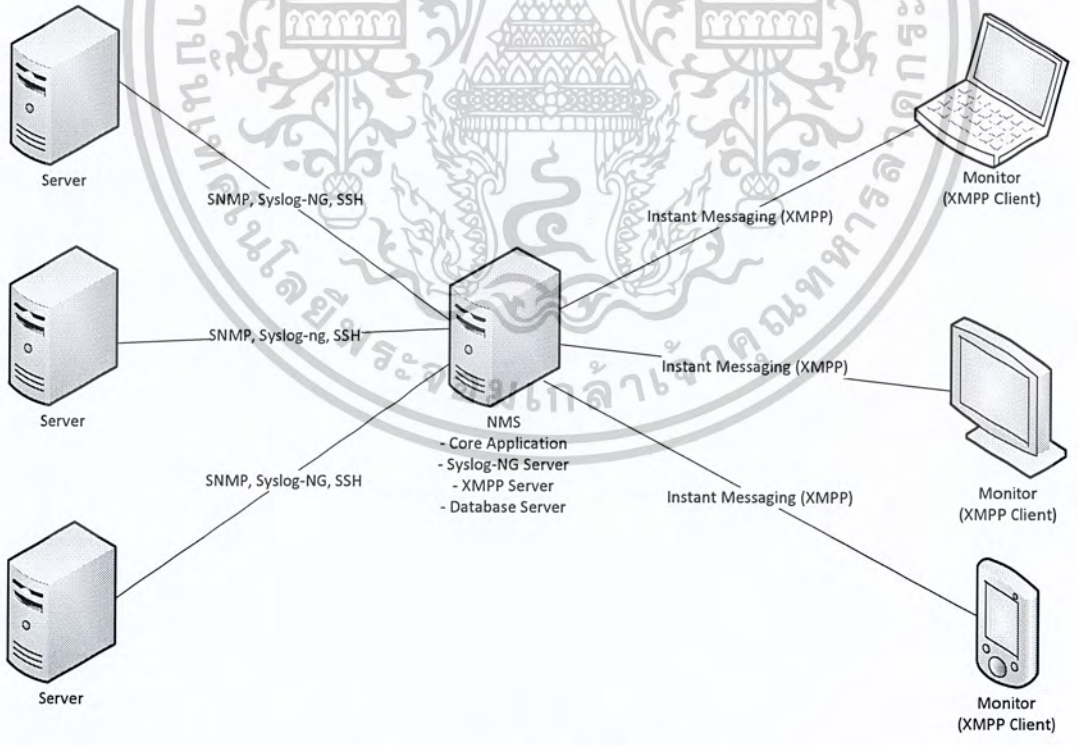
1. การติดต่อระหว่างระบบที่พัฒนาและผู้ใช้งานเป็นแบบมีปฏิสัมพันธ์ จึงทำให้สามารถพัฒนาการระบบการแจ้งเตือนจาก NMS ไปยังผู้ใช้และในขณะเดียวกันผู้ใช้งานเองก็สามารถที่จะทำการร้องขอข้อมูลสถานะของเครื่องแม่ข่าย ณ ขณะนั้นได้ด้วย คล้ายกับการที่ติดต่อกับบุคคลผ่านข้อความด่วน เช่น MSN, Google Talk, Facebook Chat
2. การใช้งานสำหรับโทรศัพท์มือถือสามารถใช้งานแทน SMS ที่มีค่าใช้จ่ายในการใช้บริการ ผู้ใช้งานโทรศัพท์มือถือที่ใช้บริการ internet มักใช้รูปแบบของบริการเป็นแบบ online ตลอด 24 ชั่วโมง ผู้ใช้งานที่ต้องการการแจ้งเตือนผ่านโทรศัพท์มือถือ สามารถทำได้โดยติดตั้งโปรแกรม XMPP Client ลงบนโทรศัพท์มือถือ



รูปที่ 1.1 แสดงผลของคำสั่ง top ใน Linux server



รูปที่ 1.2 แสดงผลของคำสั่ง df -h และ เปิด system log



รูปที่ 1.3 แสดงแนวคิดในการพัฒนาระบบ

1.2 วัตถุประสงค์

ในการพัฒนา ระบบตรวจสอบเฝ้าระวังและแจ้งเตือนสถานะของเครื่องแม่ข่ายโดยใช้โปรโตคอล SNMP ผ่านข้อความด่วนทันใจ มีวัตถุประสงค์ดังนี้

1. เพื่อศึกษาและพัฒนาระบบตรวจสอบและแจ้งเตือนสถานะการให้บริการของเครื่องแม่ข่ายและอุปกรณ์เครือข่าย ที่มีลักษณะการทำงานแบบปฏิสัมพันธ์กับผู้ดูแลระบบผ่านทางบริการข้อความด่วนทันใจ (Instant Messaging)
2. เพื่อเพิ่มประสิทธิภาพของการตรวจสอบสถานะการให้บริการของเครื่องแม่ข่ายและอุปกรณ์เครือข่าย
3. เพื่อศึกษาและพัฒนาซอฟท์แวร์ประยุกต์ทางเครือข่าย (Network Application) บนระบบปฏิบัติการลินุกซ์ (Linux Operating System)

1.3 ขอบเขตของโครงการ

ระบบตรวจสอบเฝ้าระวังและแจ้งเตือนสถานะของเครื่องแม่ข่ายโดยใช้โปรโตคอล SNMP ผ่านข้อความด่วนทันใจ มีขอบเขตการทำงานดังนี้

1. ระบบสามารถตรวจสอบสถานะของ Disk Usage
2. ระบบสามารถตรวจสอบสถานะของ CPU Utilization
3. ระบบสามารถตรวจสอบสถานะของ Load average
4. ระบบสามารถตรวจสอบสถานะของ Uptime
5. ระบบสามารถตรวจสอบสถานะของ Network Interfaces
6. ระบบสามารถตรวจสอบสถานะของ Process
7. ระบบสามารถตรวจสอบ system log หรือ เหตุการณ์สำคัญที่เกิดขึ้นกับเครื่องคอมพิวเตอร์แม่ข่าย
8. ระบบสามารถรับและประมวลผลข้อความจาก SNMP Trap จากเครื่องแม่ข่ายและอุปกรณ์เครือข่าย
9. ระบบสามารถแจ้งเตือนเหตุการณ์ที่สำคัญที่มีที่มาจาก SNMP Trap, system log ไปยังผู้ดูแลระบบผ่านบริการข้อความด่วน (Instant Messaging)
10. ระบบสามารถรับคำสั่งจากผู้ดูแลระบบเพื่อส่งข้อมูลเกี่ยวกับสถานะของเครื่องแม่ข่ายและอุปกรณ์เครือข่ายที่ต้องการ ไปยังผู้ดูแลระบบ ผ่านบริการข้อความด่วนทันใจ (Instant Messaging)

11. ระบบสามารถทำการแก้ไขและบันทึก configuration ของระบบได้ เช่น User Account, Path ของ Syslog, XMPP Server, XMPP Port
12. ระบบสามารถเพิ่มเครื่องแม่ข่ายและอุปกรณ์เครือข่ายที่จะทำการตรวจสอบสถานะได้

1.4 ผลที่คาดว่าจะได้รับ

ผลที่คาดว่าจะได้รับจากการพัฒนาระบบตรวจสอบเฝ้าระวังและแจ้งเตือนสถานะของเครื่องแม่ข่าย มีดังนี้

1. ช่วยลดเวลาในการเฝ้าสังเกตระบบ โดยผู้ดูแลระบบไม่จำเป็นต้องตรวจสอบและเฝ้าสังเกตระบบตลอดเวลา เนื่องจากมีการแจ้งเตือนเมื่อมีเหตุการณ์ผิดปกติผ่านบริการข้อความด่วนทันใจ (Instant Messaging)
2. ช่วยให้ผู้ดูแลระบบสามารถตอบคำถามเรื่องประสิทธิภาพ และสถานะการทำงานแก่ผู้บริหารได้
3. ลดเวลาที่ระบบไม่สามารถทำงานได้ เนื่องจากรับทราบปัญหาที่เกิดขึ้นได้เร็วขึ้น
4. ผู้พัฒนามีความรู้ความเข้าใจเกี่ยวกับ SNMP protocol , xmpp protocol , syslog-ng , และการพัฒนา Java Application บนระบบปฏิบัติการลินุกซ์

บทที่ 2
ทฤษฎีที่เกี่ยวข้องกับการบริหารจัดการเครื่องแม่ข่าย และอุปกรณ์เครือข่าย

2.1 SNMP (Simple Network Management Protocol)

SNMP เป็น Network Management Protocol ตัวหนึ่งซึ่งทำงานในระดับ Application Layer ใช้สำหรับการบริหารจัดการเครือข่าย โปรโตคอลนี้เป็นส่วนหนึ่งในชุดโปรโตคอล TCP/IP ซึ่งช่วยให้ผู้ดูแลระบบสามารถจัดการประสิทธิภาพ, วิเคราะห์ปัญหา และให้ข้อมูลเพื่อใช้สำหรับวางแผนเครือข่ายในอนาคต

SNMP ใช้แนวคิดของผู้จัดการ (Manager) และตัวแทน (Agent) ซึ่ง Manager นั้นส่วนใหญ่จะเป็น Host (PC) ซึ่งควบคุมและติดตามกลุ่มของ Agent มักจะเป็น Router

SNMP เป็นโปรโตคอลที่ทำงานในระดับ Application Layer ซึ่งอาจจะมี 1 หรือ 2-3 สถานีควบคุมของ manager ที่ควบคุมกลุ่มของ agent โปรโตคอลนี้ออกแบบมาทำงานชั้น Application ดังนั้นมันจึงสามาถติดตามควบคุมอุปกรณ์ที่ผลิตมาต่างกัน และการติดตั้งทางกายภาพที่ต่างกัน SNMP มีความเป็นอิสระในการจัดการงานจากทั้งคุณลักษณะทางกายภาพของอุปกรณ์ที่ถูกจัดการและภายใต้เน็ตเวิร์กเทคโนโลยี มันสามารถใช้ในระบบเน็ตเวิร์กที่ไม่เหมือนกันของการเชื่อมต่อ LANs และ WANs โดย Router ซึ่งมีการผลิตที่ต่างกัน

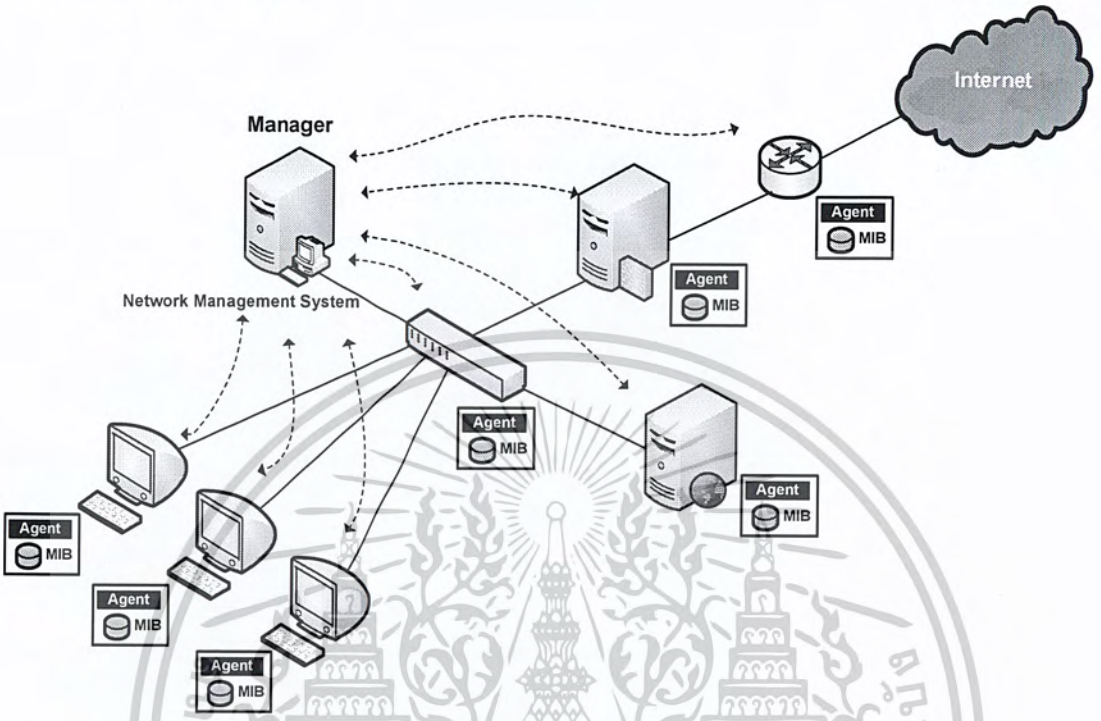
2.1.1 Manager and Agent

สถานีบริหารจัดการเรียก "Manager" ซึ่งเป็น host เครื่อง PC ที่ run "SNMP Client Program" ส่วนสถานีที่ถูกบริหารจัดการเรียก "Agent" เป็นพวก router หรือ host หรืออุปกรณ์เครือข่ายต่างๆ ซึ่งรัน "SNMP Sever Program" โดยการบริหารจัดการจะทำโดยการถ่ายทอดข้อมูลระหว่าง Manager และ Agent

Agent ดำเนินการเก็บข้อมูลของอุปกรณ์ไว้ในฐานข้อมูล Manager สามารถที่จะเข้าถึงค่าของข้อมูลในฐานข้อมูล ตัวอย่างเช่น Router เก็บตัวแปรที่ซึ่งเก็บค่าจำนวนของ packet ที่รับมา และตัวแปรที่เก็บจำนวน packet ที่ส่งต่อ Router ก็สามารถนำค่าของตัวแปรทั้งสอง มาเปรียบเทียบกันว่ามีการแออัด (Congest) ของการส่งต่อ packet หรือไม่

Manager สามารถที่จะกำหนดให้Agent สามารถดำเนินการกระทำบางอย่างได้ เช่น router จะสามารถดำเนินการกระทำบางอย่างได้ เช่น router จะทำการตรวจสอบค่าของ Reboot Counter ในเวลาที่มันควรรีบูตตัวเอง โดยส่ง packet ไปบังคับถ้าค่าของตัวนับเวลาเป็นศูนย์

Agent ก็สามารถช่วยเหลือกระบวนการจัดการได้เหมือนกัน Server Program ที่ทำงานอยู่บน Agent สามารถที่จะเช็คสิ่งแวดล้อมหรือค่าสถานะต่าง ๆ ถ้ามีสิ่งใดผิดปรกติสามารถส่งข้อความ “Trap” ไปเตือน manager ได้



รูปที่ 2.1 โครงสร้างของระบบจัดการเครือข่ายด้วย SNMP

ซึ่งสรุปการบริหารจัดการเครือข่ายโดย SNMP จะกระทำบน 3 แนวคิดพื้นฐานดังนี้

1. Manager จะร้องขอข้อมูลกับ Agent และทำการตรวจสอบพฤติกรรมของ Agent จากข้อมูลที่ส่งกลับมา
2. Manager สั่งให้ Agent ดำเนินการทำงาน โดยทำการเปลี่ยนค่าใหม่ ใน Database ของ Agent
3. Agent สามารถที่จะช่วยเหลือกระบวนการบริหารจัดการ โดยส่งข้อความไปเตือน Manager ถ้ามีสถานะการที่ไม่ปกติเกิดขึ้น

2.1.2 SNMP Version

ในปัจจุบัน SNMP มีแล้ว 3 version คือ

2.1.2.1 SNMP v1 (ซึ่ง SNMP v1ประกาศใน RFC 1155 และใน MIB-1(RFC1156) และ MIB-2 (RFC-1213))

2.1.2.2 SNMP v2 (RFC 1902) ยกระดับความสามารถและประสิทธิภาพการทำงานจากเดิม โดยเพิ่มคำสั่งพื้นฐานสำหรับใช้ในการจัดการเครือข่าย และเพิ่มกลุ่ม Object ในฐานข้อมูล

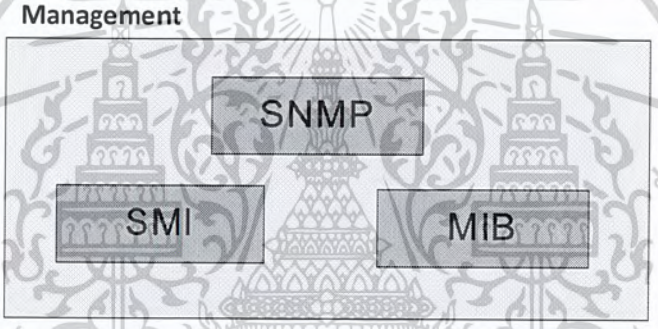
2.1.2.3 SNMP v3 แก้ปัญหาเรื่องความไม่ปลอดภัยของ SNMP (RFC 3411 – RFC 3418) ซึ่งมี Feature ที่สำคัญๆ ดังนี้

- Message Integrity เพื่อให้แน่ใจว่า packet ที่ส่งนั้นจะไม่ถูกเปลี่ยนแปลงทำลาย
- Authentication เป็นการตรวจสอบว่าข้อความนั้นมาจากแหล่งที่ถูกต้อง
- Encryption ทำการเข้ารหัสของ packet เพื่อป้องกันการถูกสอดแนมโดยแหล่งที่ไม่ได้รับอนุญาต

2.1.3 Management Component

ในการทำงานการจัดการเครือข่ายนั้น SNMP จะใช้โปรโตคอลอื่นอีก 2 ตัว คือ Structure of Management Information (SMI) และ Management Information Base (MIB)

การบริการจัดการเครือข่ายบนอินเตอร์เน็ต จะกระทำร่วมมือกัน ของทั้ง 3 โปรโตคอล คือ SNMP, SMI และ MIB ดังรูป 2.2



รูปที่ 2.2 แสดงส่วนประกอบของการจัดการเครือข่ายบนอินเตอร์เน็ต

2.1.3.1 หน้าที่ของ SNMP (Role of SNMP)

SNMP มีหน้าที่เฉพาะมาก ๆ สำหรับการจัดการเน็ตเวิร์ก มันกำหนดรูปแบบของ packet ที่ส่งมาจาก Manager ถึง Agent มันจะทำการแปลผลลัพธ์และจัดทำสถิติ ซึ่งpacket ที่แลกเปลี่ยนกันกับ Agent นั้นจะบรรจุชื่อของ Object และสถานะหรือค่าของ object นั้น SNMP จะอ่านและเปลี่ยนแปลงค่าใน object ใน SNMP packet นี้

2.1.3.2 หน้าที่ของ SMI (Role of SMI)

ในการใช้ SNMP ต้องมีกฎในการตั้งชื่อ object ตรงนี้เป็นส่วนที่สำคัญ เพราะ object จะอยู่แบบโครงสร้างลำดับชั้น (object อาจจะมี object พ่อแม่ หรืออาจมี object ลูก) ส่วน (Part) ของชื่อสามารถอ้างเป็นลำดับจากโหนดพ่อแม่ เราต้องการกฎเพื่อใช้ในการกำหนดประเภทของ object ประเภทของ object อะไรที่สามารถจัดการได้โดยSNMP SNMP สามารถที่จะจัดการประเภทพื้นฐาน หรือโครงสร้างประเภทได้หรือไม่ จำนวนประเภทพื้นฐานที่ใช้งานได้, ขนาดของประเภท, ขอบเขตของประเภท, ในส่วนที่เพิ่มขึ้นมาแต่ระประเภทจะทำการเข้ารหัสอย่างไร

ต้องมีกฎที่ครอบจักวาล เพราะไม่ทราบสถาปัตยกรรมของคอมพิเตอร์ที่ส่ง, รับ หรือเก็บค่าของข้อมูล ผู้ส่งอาจจะเป็นคอมพิวเตอร์ที่มีสมรรถนะสูง ซึ่งเก็บเลขจำนวนเต็ม (Integer) 8-Byte ส่วนผู้รับอาจจะเป็นคอมพิวเตอร์ที่มีสมรรถนะต่ำกว่า เก็บข้อมูลเลขจำนวนเต็มแค่ 4-Byte

SMI เป็นโปรโตคอลที่กำหนดกฎนี้ อย่างไรก็ตามต้องเข้าใจว่า SMI นั้นกำหนดเพียงแค่กฎ แต่มันไม่ได้กำหนดจำนวนของ object ที่ถูกจัดการในส่วนนั้น หรือ object ไหนใช้ Type อะไร SMI จะเก็บกลุ่มของกฎพื้นฐานในการอ้างถึงชื่อของ object และจำแนกประเภทของมัน สำหรับการเลือก object กับประเภทของ object นั้นไม่ได้กระทำโดย SMI

- SMI กำหนดกฎทั่วไปในการอ้างอิงชื่อของ object กำหนดประเภทของ object)ซึ่งรวมลำดับความยาว (และแสดงว่าจะทำการเข้ารหัสของ object และค่าของมันได้อย่างไร
- SMI ไม่ได้กำหนดจำนวนของ object ที่ควรจัดการ หรือชื่อของ objectที่จัดการ หรือกำหนดความสัมพันธ์ระหว่าง object และค่าของมัน

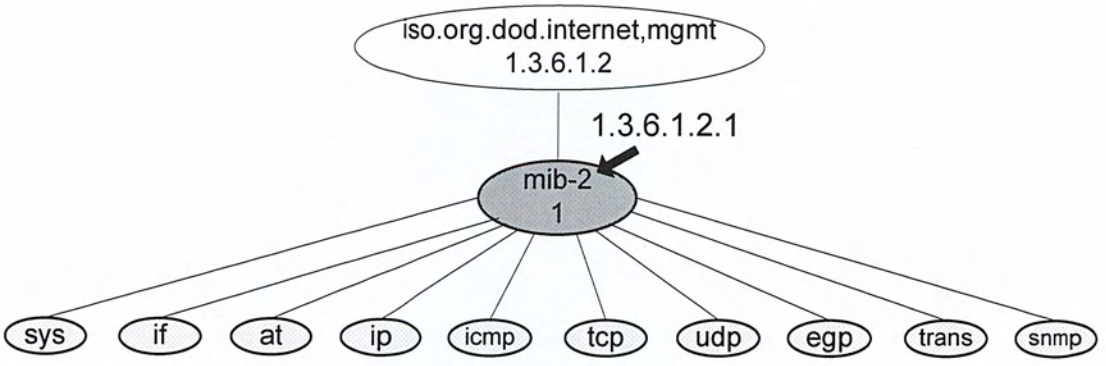
2.1.3.3 หน้าที่ของ MIB (Role of MIB)

ใน Object แต่ละตัวที่ถูกจัดการ โปรโตคอลนี้ต้องกำหนดหมายเลขของ object ชื่อของมันที่สอดคล้องกับกฎที่กำหนดโดย SMI และความสัมพันธ์ของประเภทกับชื่อของ object แต่ละตัว โปรโตคอลนี้คือ MIB, MIB จะทำการกำหนดกลุ่มของ object สำหรับแต่ละ entity เหมือนใน Database

- MIB จะทำการเก็บรวบรวมชื่อของ object ประเภทของมัน และความสัมพันธ์กับ entity อื่น ๆ ที่ถูกจัดการ

2.1.6 Management Information Base (MIB)

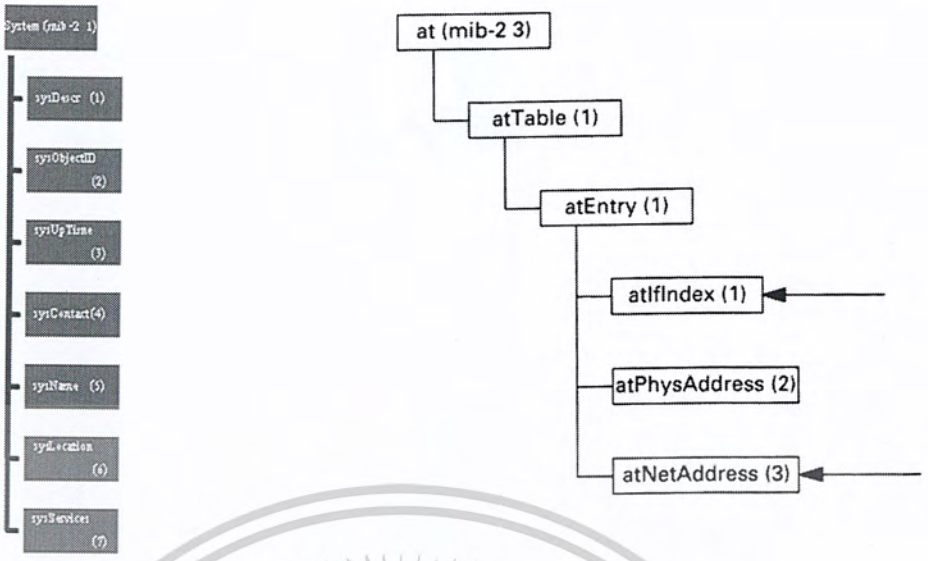
MIB-2 (version 2) เป็นองค์ประกอบที่ 2 ในการจัดการเครือข่าย ในAgent แต่ละตัวจะมี MIB-2 ซึ่งจะเก็บ Object ทุกตัวที่ Manager สามารถจัดการได้ จะมี Object แบ่งเป็นกลุ่มอยู่ 10 ประเภทอยู่ภายใต้ MIB-2 คือ system, interface, address, translation, ip, icmp, tcp, udp, egp, transmission และ snmp กลุมพวกนี้อยู่ภายใต้ object mib-2 ซึ่งมี Object Identifier Tree ดังรูปที่ 2.3



รูปที่ 2.3 Object mib-2

ข้างล่างต่อไปนี้จะเป็นคำอธิบายสรุปของบาง Object ภายใต้ mib-2

- sys (System) object กำหนดรายละเอียดทั่วไปของอุปกรณ์นั้น เช่น Name , ที่ตั้ง (location), ชนิดของ Hardware, ระบบปฏิบัติการ และเวลาชีวิต (Time Life)
- if (Interface) เป็นกลุ่มข้อมูลเกี่ยวกับ Physical Address ของอุปกรณ์ เกี่ยวกับการติดตั้งและข้อมูลที่แสดงถึงเหตุกาณ์ต่าง ๆ ที่เกิดขึ้นกับแต่ละ interface ข้อมูลเหล่านี้ได้แก่ จำนวน interface , ชนิดของ interface, ความเร็ว, Physical Address, ปริมาณข้อมูลที่ไหลเข้าออกในแต่ละ interface เป็นต้น
- at (Address Translation) กำหนดข้อมูลเกี่ยวกับ ARP Table เป็นกลุ่มที่ทำเกี่ยวกับ Address Translation โดยจะประกอบด้วย 1 ตาราง ซึ่งในแต่ละแถวจะประกอบด้วย Network Address จะเป็น IP Address และ Physical Address นั้นขึ้นอยู่กับประเภทของเครือข่าย เช่น ถ้าเป็น Ethernet ก็จะใช้ Ethernet Address เป็น Physical Address เป็นต้น



รูปที่ 2.4 (a) MIB-2 System Group (b) MIB-2 Address-Translation Group

- ip (Internet Protocol)

  ประกอบด้วยข้อมูลเกี่ยวกับ IP ของอุปกรณ์ซึ่งประกอบด้วยตาราง 3 ตาราง คือ

  - ipAddrTable เก็บ IP Address ซึ่งแต่ละ IP Address จะถูกกำหนดให้กับแต่ละ Interface ของอุปกรณ์
  - ipRouteTable เก็บข้อมูลสำหรับการทำการเลือกเส้นทางในเครือข่าย internet (internet routing) ซึ่งข้อมูลเหล่านี้จะขึ้นอยู่กับ Protocolที่ใช้ในการทำการเลือกเส้นทาง
  - ipNetToMediaTable เป็นตารางที่จะใช้ในการแปลง IP Address ให้เป็น Physical Addressโดยข้อมูล IP Address และ Physical Address ในตารางนี้จะเหมือนกับในตาราง atTable

- icmp (Internet Control Message Protocol) object นี้เก็บข้อมูลที่เกี่ยวข้องกับโปรโตคอล ICMP เช่นจำนวนของ Packet ที่ส่งและรับ และจำนวน error ที่เกิดขึ้น
- tcp (Transmission Control Protocol) object นี้เก็บข้อมูลทั่วไปเกี่ยวกับ tcp เช่น ตาราง Connection Table , time-out value, หมายเลขของ port และจำนวนของ packet ที่ส่งและรับ
- udp (User Datagram Protocol) เก็บข้อมูลเกี่ยวกับการทำงานของ UDP ในกลุ่มนี้มีตารางอยู่ 1 ตาราง คือ udpTable ซึ่งจะเก็บข้อมูลของ IP Address และ UDP Port

ซึ่งถูกใช้โดยโปรแกรมที่ทำงานบนอุปกรณ์และกำลัง UDP Datagarm โปรแกรมนี้ถูกเรียกว่า listener

- egp (Exterior Gateway Protocol) เก็บข้อมูลเกี่ยวกับการทำ EGP ของอุปกรณ์ โดยในกลุ่มนี้มีตารางอยู่ 1 ตารางคือ egpNeighTable ข้อมูลในตารางนี้เป็นข้อมูลที่จำเป็นสำหรับการสื่อสารกับอุปกรณ์อื่นที่จะทำ EGP ด้วย
- snmp (Simple Network Management Protocol) เก็บข้อมูลที่เกี่ยวกับการทำงานของ SNMP เอง

2.1.7 Access MIB Variable

ในการแสดงให้เห็นว่าในการที่จะเข้าถึงตัวแปรที่ต่างกัน เราใช้กลุ่มของ UDPในการยกตัวอย่าง ให้ดู ใน UDP group จะมีตัวแปรชนิด Simple 4 ตัว คือ udpInDaragrams, udpNoPorts, udpInErrors, udpOutDatagrams และตัวแปรแบบ sequence of (table of) 1 record คือ udpTable เราจะแสดงว่าเราจะเข้าถึงแต่ละ entity ได้อย่างไร

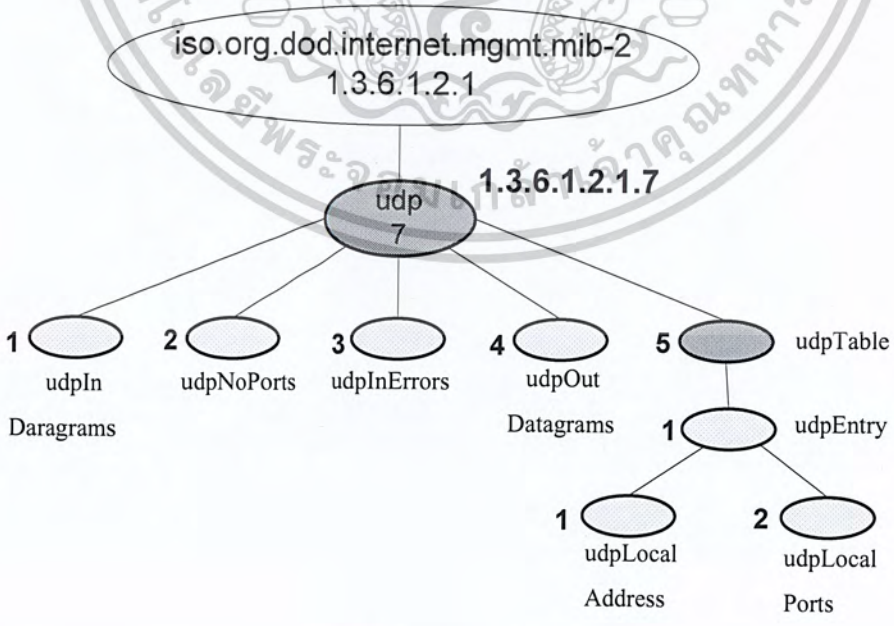
Simple Variable ในการเข้าตัวบางตัวแปรแบบ Simple เราจะใช้ ID ของ group คือ 1.3.6.1.2.1.7 ตามด้วย ID ของตัวแปรดังต่อไปนี้แสดงให้เห็นว่าจะเข้าถึงแต่ละตัวแปรได้อย่างไร

udpInDaragrams -----> 1.3.6.1.2.1.7.**1**

udpNoPorts -----> 1.3.6.1.2.1.7.**2**

udpInErrors -----> 1.3.6.1.2.1.7.**3**

udpOutDatagrams -----> 1.3.6.1.2.1.7.**4**



รูปที่ 2.5 UDP Group

แม้ว่า Object Identifier กำหนดตัวแปรได้ แต่ไม่ใช่ instance ในการแสดง instance ของแต่ละตัวแปรเราจะต้องเพิ่ม instance suffix เช่นเติม 0 ต่อท้ายดังตัวอย่าง

udpInDaragrams.0 -----> 1.3.6.1.2.1.7.1.**0**

udpNoPorts.0 -----> 1.3.6.1.2.1.7.2.**0**

udpInErrors.0 -----> 1.3.6.1.2.1.7.3.**0**

udpOutDatagrams.0 -----> 1.3.6.1.2.1.7.4.**0**

Table ในการจำแนกตาราง ในตอนแรกเราต้องใส่ Table ID ในกลุ่มของ udp มีเพียงแค่ 1 ตาราง ในการเข้าถึงตารางนี้เราใช้ดังนี้

udpTable -----> 1.3.6.1.2.1.7.5

อย่างไรก็ตาม Table ไม่ใช่ปลายทางของโครงสร้างต้นไม้ เราจึงไม่สามารถเข้าถึงตารางได้เราต้องกำหนด entry หรือลำดับ (sequence) ในตาราง (ใ ห้ id =1) ได้เป็น

udpEntry -----> 1.3.6.1.2.1.7.5.**1**

แต่การเข้าถึงนี้ยังไม่ใช่ node ปลาย จึงไม่สามารถเข้าถึงมันได้ เราต้องกำหนด entity ในการเข้าถึง เช่น

udpLocalAddress -----> 1.3.6.1.2.1.7.5.**1.1**

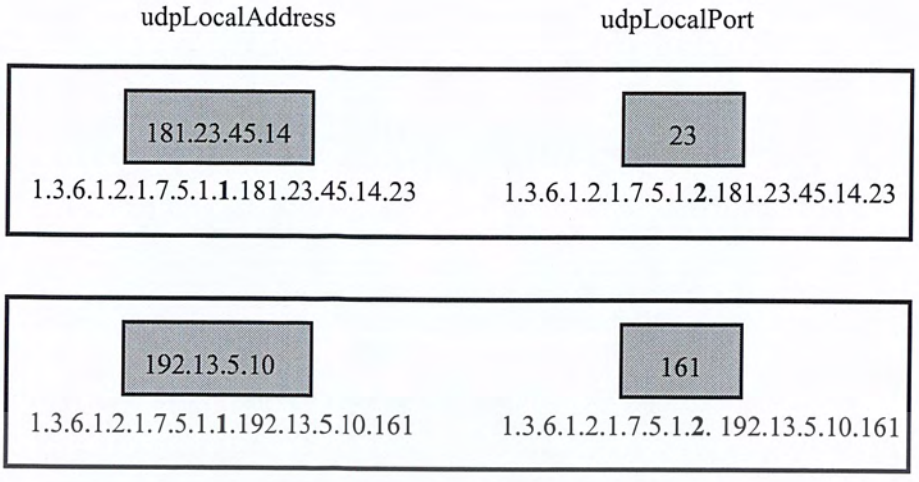
udpLocalPort -----> 1.3.6.1.2.1.7.5.**1.2**

ทั้ง 2 โหนดนี้เป็นโหนดปลายของโครงสร้างต้นไม้ แม้ว่าเราจะสามารถเข้าถึง instance นั้นได้ เราต้องการกำหนด instance อันใหนในบางขณะ ในตารางก็จะมีค่า LocalAddress/LocalPort ได้หลายค่า ในการเข้าถึง instance (row) ของตารางเราต้องใส่ index ของ Array เพิ่มขึ้น ใน MIB index ของ Array ไม่ใช่เลขจำนวนเต็มเหมือนโปรแกรมคอมพิวเตอร์ทั่วไป แต่ index จะใช้ค่าของ 1 field หรือมากกว่าในการเข้าถึง ในตัวอย่างของเรา udpTable มี index เป็นค่าของ LocalAddress และ LocalPort ดังรูปที่ 2.5 ในตารางมี 4 แถว และค่าของแต่ละช่อง index ของแต่ละแถวก็คือการรวมกันของทั้ง 2 ค่า

ในการเข้าถึง instance ของ Local Address สำหรับแถวแรก เราจะใช้การเพิ่มค่า ID กับ index ของ instance นี้

udpLocalAddress.181.23.45.14.23 -----> 1.3.6.1.2.1.7.5.1.1.181.23.45.14.23

หมายเหตุ. ไม่ใช่ทุกตารางที่ใช้ index แบบนี้ บางตารางใช้ค่า field เดียว บางตารางใช้ 2 field

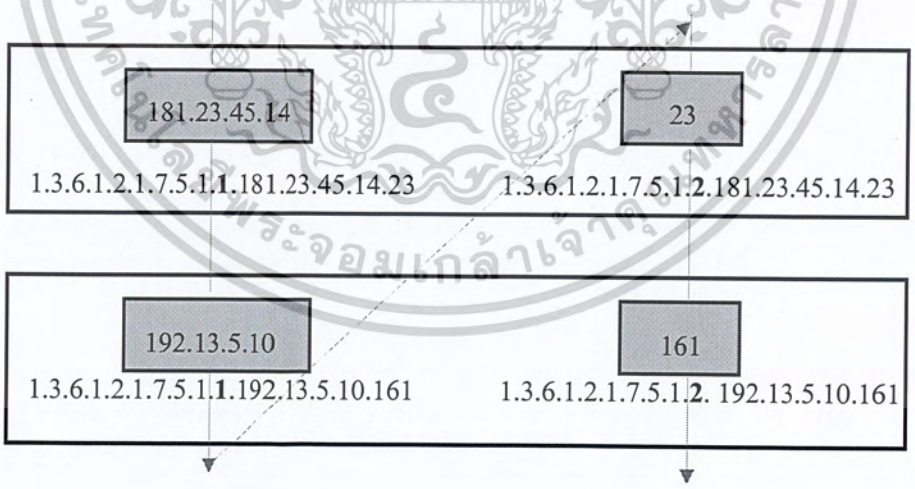


รูปที่ 2.6 Index for udpTable

2.1.8 Lexicographic Ordering

ในจุดที่น่าสนใจเกี่ยวกับตัวแปรในMIB คือ Object Identifier (รวม instance identifier) ตามใน lexicographic order จะไล่จากคอลัมหนึ่งไปยังคอลัมหนึ่ง และในแต่ละคอลัมจะไล่จากบนลงล่าง ดังรูป 2.7

ในการเรียงลำดับ lexicographic ทำให้ manager สามารถเข้าถึงเซทของตัวแปรหนึ่งจากหลาย ๆ ตัวโดยกำหนดตัวแปรตัวแรก เราจะเห็นการใช้ในคำสั่ง GetNextRequest



รูปที่ 2.7 Lexicographic ordering

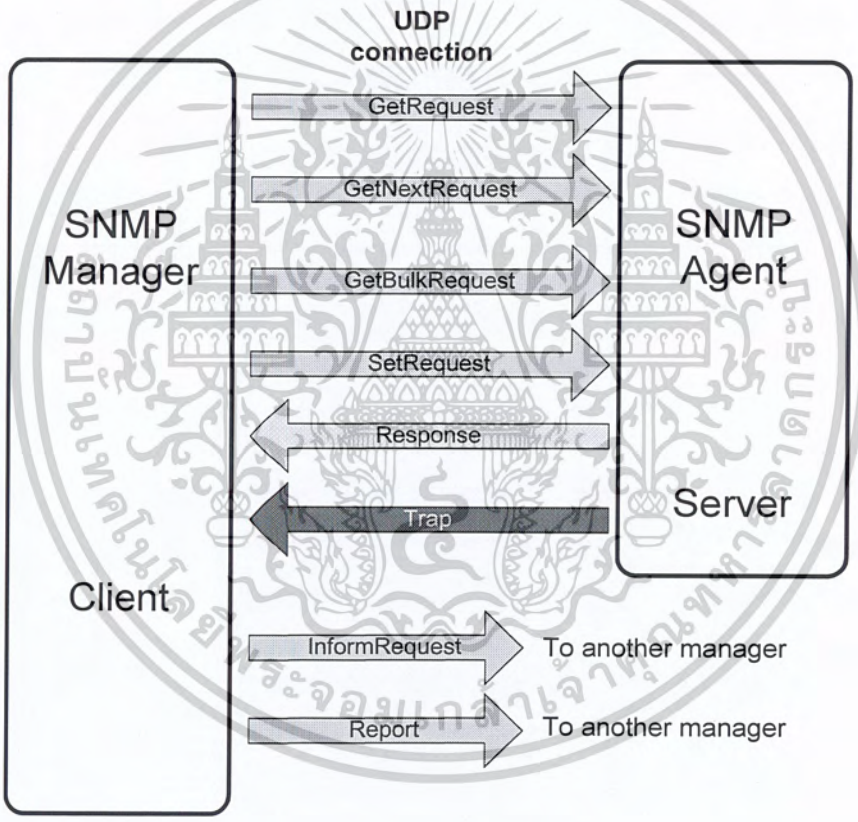
2.1.9 SNMP

ใช้ทั้ง SMI และ MIB ในการจัดการเครือข่าย internet ซึ่งเป็น Application Program ที่ทำดังต่อไปนี้

1. Manager จะเอาค่าของ Object ที่กำหนดจาก Agent
2. Manager จะเก็บค่าของ Object ที่กำหนดใน Agent
3. Agent จะส่งข้อความเตือนเกี่ยวกับเหตุการณ์ที่ไม่ปกติให้กับ Manager

2.1.9.1 PDUs (Protocol Data Unit)

ใน SNMPv3 ได้กำหนดประเภทของ Packet (หรือ PDU) ไว้ 8 ประเภทด้วยกัน คือ GetRequest, GetNextRequest, GetBulkRequest, SetRequest, Response, Trap and Report



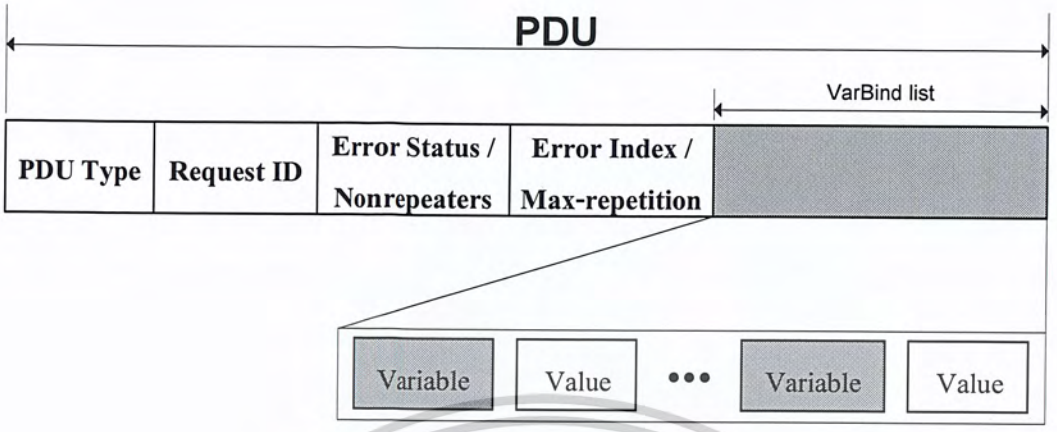
รูปที่ 2.8 SNMP UDPs

- GetRequest ใน GetRequest PDU จะส่งจาก Manager ไปยัง Agent เพื่อบอกว่า Manager ต้องการทราบข้อมูลอะไรจาก Agent ซึ่งกำหนดโดย Object Identifier ที่ส่งไปพร้อมกับ Message เช่น Manager ระบุ Object Identifier เป็น 1.3.6.1.2.1.1.1.0 ซึ่งเป็นการระบุว่า

ต้องการทราบข้อมูล sysDescr หรือส่วนของรายละเอียดของอุปกรณ์ที่ตัว Agent ทำงานอยู่ ซึ่งทาง Agent ก็จะตอบข้อมูลรายละเอียดของอุปกรณ์ตัวที่มันทำงานอยู่กลับมา

- GetNextRequest ใน GetNextRequest PDU จะส่งจาก Manager ถึง Agent ในการที่จะเอาค่าตัวแปร โดยค่าที่ได้จะเป็นค่าของ Object ตัวต่อจาก Object ID ตัวที่กำหนดใน PDU ส่วนใหญ่มักจะใช้ในการเอาค่าของการเข้าไปตาราง ถ้า Manager ไม่รู้ index ของการเข้ามันจะไม่สามารถดึงค่าออกมาได้ แต่เราใช้ GetNextRequest และกำหนด Object ID ของตารางเพราะ entry ตัวแรกจะต่อจาก Object ID ของตารางทันที ก็จะได้ค่าของ First entry ตัวแรกกลับมา Manager สามารถใช้ Object ID นี้ในการเอาค่าของตัวถัดไปเรื่อย ๆ ตัวอย่างเช่น Manager ส่ง GetNextRequest ที่ให้ Object Identifier เป็น 1.3.6.1.2.1.1 ซึ่งเป็นการเข้าถึงกลุ่ม System ใน MIB โดยที่ไม่ได้ระบุว่า ต้องการ ทราบข้อมูลอะไรในกลุ่ม System ดังนั้นเมื่อเวลาที่ Agent ส่ง GetResponse กลับมาให้มันก็จะส่งค่าของ Object Identifier เป็น 1.3.6.1.2.1.1.1.0 ซึ่งก็คือค่าของ SysDescr ที่อยู่ในกลุ่ม System ซึ่งเป็นค่าของ Object Identifier ตัวถัดไปใน Tree นั่นเอง
- GetbulkRequest ใน GetbulkRequest PDU นั้นจะส่งจาก Manager ไปที่ Agent ในการที่จะเอาค่าของข้อมูลจำนวนมาก มันสามารถใช้ GetRequest หรือ GetNextRequest หลายๆ ครั้งแทนได้
- SetRequest ใน SetRequest PDU ส่งจาก Manager ไปที่ Agent ในการ Set (Store) ค่าลงในตัวแปร หรือเปลี่ยนแปลงค่า Configuration ต่าง ๆของข้อมูลใน MIB ของอุปกรณ์นั้นๆ
- Response ในResponse PDU ส่งจาก Agent มาที่ Manager ในการตอบสนอง GetRequest หรือ GetNextRequest ซึ่งจะบรรจุค่าของตัวแปรที่ร้องขอโดย Manager
- Trap ใน Trap PDU (เรียก SNMPv2 Trap เพื่อให้ต่างจาก SNMPv1 Trap) ส่งจาก Agent ไปสู่ Manager เพื่อรายงานเหตุกาณ์ตัวอย่าง เช่น ถ้า Agent ทำการรีบูต (Reboot) ก็จะแจ้ง Manager และรายงานเวลาที่ทำการรีบูต
- InFormRequest ใน InFormRequest PDU จะส่งจาก Manager ตัวหนึ่งไปยัง Manager ตัวอื่น ๆ ที่อยู่ไกลออกไป เพื่อรับค่าของตัวแปรบางตัวจาก Agent ภายใต้การควบคุมของ Manager ที่อยู่ไกลออกไปนั้น ซึ่ง Manager ที่อยู่ไกลออกไปนั้นจะตอบสนองมาด้วย Response PDU
- Report ใน Report PDU ออกแบบมาเพื่อรายงานข้อผิดพลาดบางประเภทระหว่าง Manger ด้วยกัน

2.9.1.2 SNMP PDU Form



รูปที่ 2.9 SNMP PDU Format

รูปแบบของ 8 SNMP PDUs แสดงในรูปที่ 2.9 ในส่วนของ GetBulkRequest PDU จะต่างจาก PDU ตัวอื่นอยู่ 2 ช่อง

- PDU Type ช่องนี้จะกำหนดประเภทของ PDU (ดูตาราง 2.1)
- Request ID ช่องนี้เป็นหมายเลขการร้องขอที่ใช้โดย Manager ใน Request PDU และใช้ซ้ำโดยAgent ในการตอบสนอง ซึ่งเป็นการใช้คู่กันระหว่างการร้องของและการตอบสนอง
- Error Status ช่องนี้เป็นเลขจำนวนเต็มที่ใช้เพียงแค่ใน Response PDU เท่านั้นในการแสดงประเภทของข้อผิดพลาดที่รายงานโดย Agent ซึ่งค่าในช่องนี้เป็น 0 ใน Request PDU ดังตารางที่ 2.3 เป็นการจำแนกประเภทของข้อผิดพลาดที่เกิดขึ้นได้

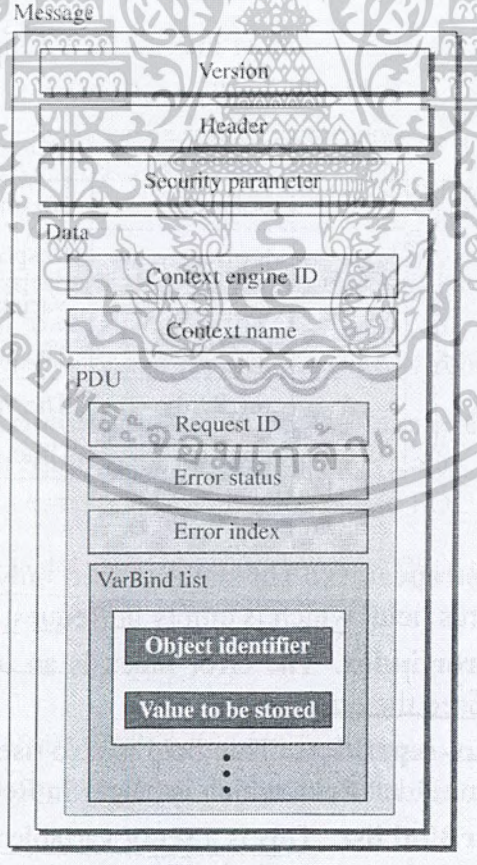
ตารางที่ 2.1 ประเภทของข้อผิดพลาด

Status	Name	Meaning
0	noError	ไม่มีข้อผิดพลาด
1	tooBig	Response ใหญ่เกินที่จะเก็บลงใน 1 ข้อความได้
2	noSuchName	Variable นั้นไม่มีอยู่
3	badValue	ค่าที่ถูกจัดเก็บนั้นไม่ถูกต้อง
4	ReadOnly	ไม่สามารถเปลี่ยนแปลงค่าได้
5	genErr	ข้อผิดพลาดประเภทอื่นๆ

- Nonrepeaters ช่องนี้ใช้ใน GetBulkRequest ซึ่งแทนที่ Error Status ซึ่งช่องนี้จะว่างถ้าเป็น Request PDU
- Error Index ใน Error Index นี้ ใช้บอกกับ manager ว่าตัวแปรไหนเป็นสาเหตุของข้อผิดพลาด
- Max-repetition ช่องนี้ใช้ใน GetBulkRequest ซึ่งแทนที่ช่อง Error Index ซึ่งช่องนี้จะว่างถ้าเป็น Request PDU
- VarBind list นี่เป็นเซทของตัวแปรกับค่าของมัน ที่Manager ต้องการเอามาหรือต้องการกำหนด ซึ่งค่าจัเป็น null ใน GetRequest และ GetNextRequest ในTrap PDU มันจะแสดงตัวแปรและค่าที่เกี่ยวกับการกำหนด PDU

2.9.1.3 Message

SNMP ไม่ได้ส่งแค่ PDU แต่มันฝัง PDU ลงใน Message ซึ่งข้อความใน SNMPv3 สร้างขึ้นจาก 4 องค์ประกอบ คือ Version, Header, Security parameter, และ Data (ซึ่งจะรวมเข้ากับรหัสของ PDU) แสดงดังรูปที่ 2.10



รูปที่ 2.10 SNMP Message

เพราะว่าความยาวของส่วนประกอบที่ต่างกันจากข้อความถึงข้อความ SNMP จึงใช้ BER ทำการเข้ารหัสแต่ละส่วน จำได้ว่า BER ใช้ Tag และ Length ในการกำหนด Value

- ในส่วนของ Version กำหนดให้ใช้รุ่นปัจจุบันคือ version3
- ส่วน Header จะบรรจุค่า Message Identifier , maximum message size (ขนาดใหญ่ สุดของการตอบกลับ) , message flag (1 octetของข้อมูลแบบ OCTET STRING ซึ่งแต่ละบิตกำหนด Security type เช่น privacy หรือ authentication หรือข้อมูลอื่นๆ) และ Message security model ส่วน Message Security Parameter ใช้ในการสร้าง message digest
- ในส่วนของ Data จะบรรจุ PDU ถ้าข้อมูลถูกเขารหัสตรงนี้ก็จะเป็นข้อมูลเกี่ยวกับ encrypting engine และ encrypting context (ประเภทของการเข้ารหัส) ตามมาด้วยการเข้ารหัสของ PDU ถ้าข้อมูลไม่ได้ทำการเข้ารหัส ส่วนของ Data จะมีแค่ PDU ในการกำหนดประเภทของ PDU นั้น SNMP ใช้ Tag ส่วน Class จะเป็น context-sensitive(10), รูปแบบเป็น Structure(1), และNumber จะเป็น 0, 1, 2, 3, 5, 6, 7, และ 8 ดังตารางที่ 2.2

หมายเหตุ ใน SNMP v1 จะกำหนด A4 สำหรับ Trap

ตารางที่ 2.2 รหัสสำหรับ SNMP Message

Data	Class	Format	Number	Whole Tage (Binary)	Whole Tage (Hex)
GetRequest	10	1	00000	**10100000**	**A0**
GetNextRequest	10	1	00001	**10100001**	**A1**
Response	10	1	00010	**10100010**	**A2**
SetRequest	10	1	00011	**10100011**	**A3**
GetBulkRequest	10	1	00101	**10100101**	**A5**
InformRequest	10	1	00110	**10100110**	**A6**
Trap (SNMPv2)	10	1	00111	**10100111**	**A7**
Report	10	1	01000	**10101000**	**A8**
Trap (SNMPv1)	10	1	00100	**10100100**	**A4**

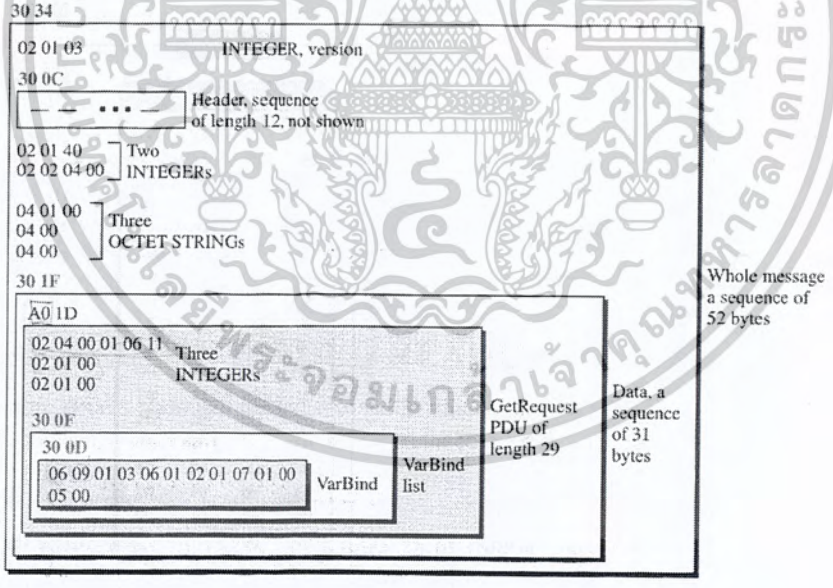
ตัวอย่าง

ในตัวอย่างนี้ Manager station (SNMP Client) ใช้ GetRequest message ในการเอาค่าหมายเลข UDP Datagram ที่ Router เก็บไว้

จะมีแค่ 1 VarBind entity ตัวแปรใน MIB ที่ตรงกับข้อมูลใน udpInDatagram คือ Object Identifier 1.3.6.1.2.1.7.1.0 ในส่วนของ Manager ต้องการเอาค่า (ไม่ได้กำหนดค่า) ในส่วนของช่อง Value จึงใส่เป็น null entity ดังรูปที่ 15 แสดงแนวคิดของ packet และเป็นลำดับชั้น เราใช้กล่องสีขาวและสีและสีเทาสำหรับ PDU

ในส่วนของ VarBind list มีเพียงแค่ 1 VarBind ประเภทของตัวแปร คือ 06 และยาว 09 ค่า value คือ 05 และยาว 00 ในส่วนของ VarBind เป็นลำดับยาว 0D (13) ส่วน Varbind list เป็นลำดับยาว 0F (15) ในส่วน GetRequest PDU ยาว 1D (29)

ตอนนี้เรามี 3 OCTET STRING เกี่ยวกับ Security Parameter , Security Model และ Flags ดังนั้นเรามี 2 จำนวนเต็มที่ที่กำหนด Maximum Size (1024) และ Message ID (64) ส่วน Header มีลำดับความยาว 12 ซึ่งเราวางที่ว่างทางซ้ายสำหรับความง่าย ตรงนั้นคือจำนวนเต็มนึง, version 3, message มีลำดับความยาว 52 Byte ดังรูปที่ 2.11



รูปที่ 2.11 ตัวอย่าง Message

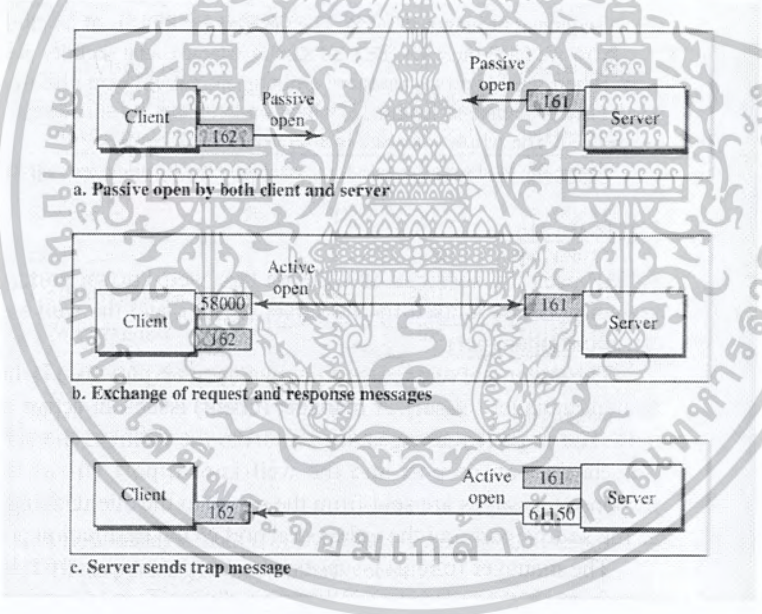
2.9.1.4 UDP Port

SNMP ใช้บริการส่งข้อมูลของ UDP บน 2 Well-know port คือ 161 และ 162

- Port 161 ใช้โดย Server (Agent)
- Port 162 ใช้โดย Client (Manager)

Agent (Server) ปล่อย passive open บน port161 และรอการเชื่อมต่อจาก Manager (Client) , Manager ปล่อย active open โดยใช้ port ชั่วคราว ข้อความร้องขอ (Request Message) ส่งจาก client ไปสู่ server และใช้พอร์ตชั่วคราวเป็น source port และ พอร์ต161 เป็น destination port ส่วน response message ส่งจาก server ถึง client ใช้พอร์ต 161 เป็น source port และพอร์ตชั่วคราวเป็น destination port

Manager ปล่อย passive open บนพอร์ต162 และรอการเชื่อมต่อจาก Agent เวลาที่ Trap message, Agent จะปล่อย Active open ใช้พอร์ตชั่วคราวนี้เป็นการเชื่อมต่อทางฝ่ายเดียวจาก Server ถึง Client



รูปที่ 2.12 หมายเลขพอร์ตสำหรับ SNMP

2.9.1.5 Security

ข้อใหญ่ที่ต่างกันระหว่าง SNMPv2 กับ SNMP v3 คือเรื่องการยกระดับการรักษาความปลอดภัย SNMPv3 ได้ให้การรักษาความปลอดภัย 2 ประเภท คือ general และ specific SNMP v3 ได้เตรียมข้อความ authentication, privacy และ manager authorization ซึ่งยอมให้ manager สามารถรักษาความปลอดภัยในระยะไกลได้ ซึ่งหมายถึง manager ไม่ต้องปรากฏอยู่ที่ manager station ได้

2.2 Syslogd

Syslogd เป็นกลไกที่ใช้ในการเก็บข้อมูลล็อกของ kernel และ application บนระบบยูนิกซ์และลินุกซ์ เป็น daemon ที่ถูกติดตั้งมาให้พร้อมกับระบบปฏิบัติการในเกือบทุกระบบ โดยผู้ดูแลระบบสามารถปรับแต่งไฟล์ configuration เพื่อควบคุมการทำงานของ syslogd ได้ เช่น ให้ syslogd เก็บข้อมูลไปไว้ที่ไฟล์ใด หรือให้ส่งข้อมูลล็อกนี้ไปเก็บไว้ยังเครื่องอื่นในเครือข่าย

ข้อมูลล็อกที่ควบคุมโดย syslogd นั้น จะถูกกำหนดให้มีค่า facility และ priority โดยส่วนของ facility นั้น เป็นข้อมูลที่อธิบายถึงแหล่งกำเนิดของข้อมูลล็อกนั้นๆ เช่น ข้อมูลล็อกที่ส่งมาจากระบบเมล์ก็จะมี facility เป็น mail ส่วน priority นั้น จะแสดงถึงระดับความสำคัญของเหตุการณ์ที่เกิดสำหรับแต่ละ facility ทั้งนี้ข้อมูลล็อกทุกอันจำเป็นต้องมี facility และ priority เสมอ

ตาราง 2.3 แสดง facility

Facility	คำอธิบาย
auth	เกี่ยวข้องกับการทำ authentication
authpriv	การทำ private authentication เท่านั้น
cron	cron daemon
daemon	system daemons
kern	ส่วนของ kernel
lpr	line printer spooling system
mail	sendmail และซอฟแวร์อื่นที่เกี่ยวกับเมล์
mark	ให้บันทึกเวลาขณะเกิดเหตุการณ์ด้วย
news	usenet news system
security	เหมือนกับ auth
syslog	ข้อมูลล็อกภายในของ syslogd
user	ส่วนของโปรเซสของ user
uucp	สำรองไว้สำหรับ UUCP
local0 - local7	local messages

ตาราง 2.4 แสดง priority

Priority	คำอธิบาย
emerg	ภาวะฉุกเฉิน
alert	แจ้งเตือนเร่งด่วน
crit	ล่อแหลม
err	มีข้อผิดพลาด
warning	คำเตือน
notice	ข้อสังเกตุ
info	ข้อมูลทั่วไป
debug	สำหรับใช้ดีบักเท่านั้น

/etc/syslog.conf

การทำงานของ syslogd นั้น จะขึ้นอยู่กับไฟล์ /etc/syslog.conf เป็นหลัก การแก้ไขใดๆ ที่เกิดขึ้นกับไฟล์นี้นั้น จะยังไม่มีผลต่อการทำงานของ syslogd ในทันที จะต้องทำการ restart syslogd service ใหม่เสียก่อน รูปแบบคำสั่งในไฟล์ /etc/syslog.conf นั้นมีรูปแบบดังนี้

facility.level **action**

facility1, facility2.level **action**

facility1.level1; facility2.level2 **action**

***.level** **action**

***.level;badfacility.none** **action**

หมายความว่า เมื่อมีข้อมูลล็อกที่มี facility และ level ที่ตรงหรือมากกว่ากับที่ตั้งไว้ ก็ จะกระทำ action ตามที่กำหนดไว้ ทั้งนี้เพราะ level ที่ตั้งไว้นั้น เป็นค่า minimum ซึ่งหมายความว่าถ้าเราตั้ง level เป็น debug ก็จะคลอบคุลมทุก level ของ facility นั้นๆ เลย ทั้งนี้เราสามารถใช้เครื่องหมาย *แทนทุกๆ ค่าใน facility หรือ priority level นั้นๆ ได้ เช่น mail.* /var/log/mail หมายความว่าให้ syslogd เก็บข้อมูลล็อกของ mail ทุก level ไปไว้ยังไฟล์ /var/log/mail ในขณะที่ level ที่เป็น none นั้น หมายความว่าไม่ให้สนใจ facility ที่ประกาศค่า level เป็น none เช่น*.emerg;mail.none /var/log/emer.log คือให้เก็บข้อมูลล็อกที่มี level เป็น emerg สำหรับทุก facility ยกเว้น mail facility สำหรับ action นั้นสามารถเลือกได้ดังนี้คือ

- filename : เก็บข้อมูลล็อกนั้นลงในไฟล์ที่กำหนด
- @hostname : ส่งต่อข้อมูลล็อกไปยัง syslogd บน host ที่กำหนด
- @ipaddress : ส่งต่อข้อมูลล็อกไปยัง host ที่มี ip address ตามที่กำหนด

- user1, user2 : ส่งข้อมูลล็อกไปยังหน้าจอของ user ที่กำหนด ถ้า user เหล่านั้นยังล็อกอินอยู่ในระบบ
- * : ส่งข้อมูลล็อกไปยังทุกๆ user ที่ยังล็อกอินอยู่ในระบบ
- /dev/console เพื่อส่งข้อมูลล็อกไปยัง console device หรือ device อื่นๆ ตามที่ต้องการ

สำหรับ Red Hat นั้นได้ขยายความสามารถของ syslogd เพิ่มเติม โดยอนุญาตให้ข้อมูลล็อกสามารถถูกส่งแบบ pipe ไปยังไฟล์ได้ โดยแก้ไขใน syslog.conf และยังสามารถใช้เครื่องหมาย = และ ! ใน syslog.conf ได้

- โดยเครื่องหมาย = หมายถึง priority ที่กำหนดเท่านั้น
- เครื่องหมาย ! หมายถึง priority อื่นที่ไม่ใช่ priority นี้และสูงกว่า

ตัวอย่าง เช่น

mail.info ความหมายคือ ข้อมูลล็อกที่เกี่ยวข้องกับเมล์และมี priority เป็น info และสูงกว่า

mail.=info ความหมายคือ ข้อมูลล็อกที่เกี่ยวข้องกับเมล์และมี priority เป็น info เท่านั้น

mail.info;mail.!err ความหมายคือ ข้อมูลล็อกที่เกี่ยวข้องกับเมล์และมี priority เป็น info , notice และ warning mail.debug;mail.!=warning ความหมายคือ ข้อมูลล็อกที่เกี่ยวข้องกับเมล์และมี priority ทุกระดับที่ไม่ใช่ warning

Red Hat นั้น โดยปกติจะเก็บข้อมูลล็อกไว้ในไฟล์ซึ่ง อยู่ภายใต้โฟลเดอร์ /var/log และถูกติดตั้งมาพร้อมกับ logrotate ซึ่งเป็นเครื่องมือที่ช่วยจัดการล็อกไฟล์ได้อย่างมีประสิทธิภาพ ปกติแล้วจะ rotate ล็อกไฟล์อาทิตย์ละครั้ง และจะเก็บล็อกไว้ 4 รอบ หรือ 1 เดือน ผู้ดูแลระบบสามารถปรับเปลี่ยนค่าเหล่านี้ได้ที่ /etc/logrotate.conf

ตัวอย่างของไฟล์ configuration สำหรับ stand-alone machine

```
#ในกรณีฉุกเฉินให้แจ้งเตือน user ทุกคนที่ล็อกอินอยู่
*.emerg                                 *
#เก็บข้อมูลล็อกที่สำคัญไว้ในไฟล์
*.warning;daemon,auth.info,user.none    /var/log/messages
#printer errors
lpr.debug                               /var/log/lpd-errs
```

ตัวอย่างของไฟล์ configuration สำหรับ network client

```
#ในกรณีฉุกเฉินให้แจ้งเตือน user ทุกคนที่ล็อกอินอยู่
*.emerg;user.none                *
#ส่งข้อมูลล็อกที่สำคัญไปยังเครื่องที่ทำหน้าที่เก็บล็อก
*.warning;lpr,local1.none        @netloghost
daemon,auth.info                 @netloghost
#ส่งข้อมูลล็อกของ local ไปยังเครื่องที่ทำหน้าที่เก็บล็อก
local2.info;local0,local7.debug  @netloghost
#printer errors
lpr.debug                        /var/log/lpd-errs
#ข้อมูลของ sudo ที่ local2 ให้เก็บไว้ในไฟล์
local2.info                      /var/log/sudo-logs
#ข้อมูลของ kernel ให้เก็บไว้ในไฟล์
kern.info                        /var/log/kern.log
```

ในกรณีนี้ข้อมูลส่วนใหญ่จะถูกส่งไปยังเครื่องที่ทำหน้าที่เก็บล็อก ซึ่งหมายความว่าถ้าเครื่องนั้นไม่สามารถให้บริการได้ข้อมูลล็อกก็จะสูญไป ดังนั้นจึงควรเก็บข้อมูลล็อกบางส่วนไว้ที่เครื่องของตัวเองด้วย

Syslog-ng เป็นโปรแกรมที่มีความหยืดหยุ่นในการทำงาน เหมาะสำหรับการนำมาใช้งานเป็น centralized log server เพราะสามารถเก็บข้อมูลล็อกแยกตามเครื่องที่ส่งล็อกมาได้ นอกจากนี้ยังสามารถทำงานร่วมกับโปรแกรม sqlsyslogd เพื่อนำข้อมูลล็อกทั้งหมดบันทึกลงในฐานข้อมูลได้

แนะนำ Syslog-ng (Syslog new generation)

syslog-ng สามารถแก้ไขข้อบกพร่องส่วนใหญ่ของ syslog ได้ โดย

- syslog-ng สามารถทำงานได้ทั้งบน TCP และ UDP
- syslog-ng สามารถทำการกรอง)filter) ข้อมูลได้ด้วย regular expression
- syslog-ng สามารถทำงานในรูปแบบที่อ้างอิง priority/facility ได้ ดังนั้น มันจึงสามารถทำงานแทนที่ syslog ได้
- syslog-ng สนับสนุน log forwarding ซึ่งทำให้สามารถทราบได้ว่า ต้นทางของล็อกถูกส่งมาจากเครื่องใด และผ่านเครื่องใดมาบ้า

นอกจากนี้ syslog-ng ยังมีรูปแบบของไฟล์ configuration ที่ง่าย แต่มีความยืดหยุ่นสูง สามารถนำไปประยุกต์ใช้ให้ตรงความต้องการได้โดยง่าย

Configuration ของ syslog-ng มีความยุ่งยากมากกว่าของ syslog แต่ก็ให้ประโยชน์ในแง่ของความยืดหยุ่นที่ได้และความสามารถที่มีมากกว่า หลังจากที่ทำความเข้าใจ configuration แล้ว ผู้ดูแลระบบสามารถสร้างไฟล์ configuration ง่ายๆ ขึ้นมาได้ด้วยตัวเอง และสามารถปรับปรุงให้เหมาะสมกับระบบของตนต่อไป

โดยปกติแล้ว syslog-ng จะอ่านข้อมูล configuration จากไฟล์ /etc/syslog-ng/syslog-ng.conf

ตัวอย่างที่ 1 แสดง configuration อย่างง่ายของ syslog-ng

```
options {
use_fqdn(no);
sync(0);
};
source s_sys { unix-stream("/dev/log"); internal(); };
source s_net { udp(); };
destination d_security { file("/var/log/security"); };
destination d_meages { file("/var/log/meages"); };
destination d_console { usertty("root"); };
filter f_authpriv { facility(auth, authpriv); };
filter f_meages { level(info .. emerg) and not facility(auth, authpriv); };
```

```
filter f_emergency { level(emerg); };
log { source(s_sys); filter(f_authpriv); destination(d_security); };
log { source(s_sys); filter(f_meages); destination(d_meages); };
log { source(s_sys); filter(f_emergency); destination(d_console); };
```

จากตัวอย่างจะเห็นได้ว่า ส่วนประกอบหลักของ configuration ประกอบไปด้วย 5 statement หลักคือ options{}, source{}, destination{}, filter{}, log{} ซึ่งแต่ละ statement จะคั่นด้วยเครื่องหมาย semicolon(;)

จะเห็นได้ว่ารูปแบบ configuration ของ syslog-ng.conf จะคล้ายคลึงกับรูปแบบของภาษาซี (C) ซึ่งทุกๆ statement จะต้องลงท้ายด้วยเครื่องหมาย semicolon ส่วน whitespace หรือช่องว่างนั้นไม่มีผลใดๆ ใน configuration จะใช้งานเพียงเพื่อให้สามารถอ่านได้ง่ายเท่านั้น

Global options

เป็นออปชันที่ถูกประกาศใช้งานภายใน options {} statement ซึ่งบางออปชันนั้นนอกจากสามารถใช้งานได้ใน option {} เองแล้วยังสามารถใช้งานใน statement อื่น เช่น source {}, destination {}, filter {}, log {} ได้อีกด้วย

ตารางที่ 2.5 options{}

Option	Description
chain_hostnames(yes | no)	หลังจากแสดง hostname ของเครื่องที่ส่งล็อกมายังเครื่องนี้ผ่านทาง tcp/udp แล้ว ให้แสดง hostname ของทุกเครื่องที่ข้อมูลล็อกถูก handle (โดย syslog-ng) มาตลอดทาง ซึ่งเหตุการณ์นี้จะเกิดขึ้นเมื่อล็อกถูกส่งต่อจาก syslog-ng server ไปยัง syslog-ng server อื่นๆ เป็นทอดๆ)default = yes)
keep_hostname(yes | no)	ให้เชื่อใจ)trust) ค่า hostname ที่อยู่ใน tcp/udp message (default = no)
use_fqdn(yes | no)	บันทึก full name ของเครื่องที่ส่ง tcp/udp message (default = no)
use_dns(yes | no)	ให้ resolve ค่า IP address ในข้อมูลล็อก เป็น hostname (default = yes)
use_time_recvd(yes | no)	ตั้งค่า message timestamp เป็นเวลาที่ล็อกเดินทางมาถึง ซึ่งโดยปกติแล้วจะใช้เวลาที่ระบุในล็อก (default = no)

ตารางที่ 2.5 (ต่อ) options{}

time_reopen(NUMBER)	เมื่อมีแพ็กเกต tcp ที่สูญหายระหว่างทางหรือเหตุที่ทำให้ไม่สามารถสื่อสารได้ตามปกติ syslog-ng จะพยายามสร้างการสื่อสารใหม่ขึ้นมา โดยจะรอเวลาตามที่ระบุ)NUMBER) หน่วยเป็นวินาที)default = 60)
time_reap(NUMBER)	เมื่อ syslog-ng เปิดไฟล์ที่เป็น inactive file (ไม่มีการเขียนข้อมูลลงไฟล์(syslog-ng จะพยายามปิดไฟล์ดังกล่าว โดยจะรอเวลาตามที่ระบุ)NUMBER) หน่วยเป็นวินาที (default = 60)
log_fifo_size(NUMBER)[a]	ขนาดของ message ที่จะถูกนำไปเข้าคิวในหน่วยความจำก่อนที่จะถูกประมวลผล ถ้าคิวเต็มและ syslog-ng ไม่สามารถทำงานได้ตามปกติ)busy) ข้อความล็อกที่ส่งเข้ามาจะถูกละทิ้ง แต่หากระบุขนาด FIFO จำนวนมากเกินไปก็จะทำให้สิ้นเปลืองหน่วยความจำ)default = 100)
sync(NUMBER)[a]	จำนวนบรรทัดของ message ที่จะเขียนลงไฟล์ก่อนที่ไฟล์จะถูก synchronize (default = 0)
owner(string)[a]	ตั้งค่าชื่อ user สำหรับไฟล์ล็อกที่ syslog-ng สร้างขึ้นมาใหม่)default = root)
group(string)[a]	ตั้งค่าชื่อ group สำหรับไฟล์ล็อกที่ syslog-ng สร้างขึ้นมาใหม่)default = root)
perm(NUMBER)[a]	ตั้งค่า file permission สำหรับไฟล์ล็อก)default = 0600)
create_dirs(NUMBER)[a]	เป็นตัวบอกว่าจะให้ syslog-ng สร้างไดเรกทอรีใหม่ได้หรือไม่ ในกรณีที่ path ที่ระบุไม่มีอยู่จริงในระบบ)default = no)
dir_owner(string)[a]	ตั้งค่าชื่อ user สำหรับไดเรกทอรีที่ syslog-ng สร้างขึ้นมาใหม่)default = root)
dir_group(string)[a]	ตั้งค่าชื่อ group สำหรับไดเรกทอรีที่ syslog-ng สร้างขึ้นมาใหม่)default = root)
dir_perm(NUMBER)[a]	ตั้งค่า directory permission เมื่อ syslog-ng สร้างไดเรกทอรีใหม่)default = 700)

[a] : ออปชันที่สามารถนำไปใช้กับ file() ใน destination{} ได้

สำหรับออปชันที่เกี่ยวข้องกับ hostname ได้แก่ chain_hostnames(), keep_hostname(), use_fqdn() และ use_dns() นั้น สนใจเฉพาะค่า hostname ของเครื่องที่ส่งล็อกมาเท่านั้น ไม่เกี่ยวข้องกับ hostname ที่ระบุใน message body แต่อย่างใด

use_dns()

เช่น หากใน syslog-ng.conf มี statement ดังต่อไปนี้

options { use_dns(yes); };

และเครื่อง joe-chong ซึ่งมีไอพีเป็น 10.0.0.7 ส่งล็อกดังต่อไปนี้มาที่ log server

Oct 13 19:56:56 s_sys@**10.0.0.7** sshd[1222]: Accepted publickey for ROOT from 10.0.0.222 port 1355 ssh2

เครื่อง log server จะทำการบันทึกล็อกดังนี้

Oct 13 19:56:56 s_sys@**joe-chong** sshd[1222]: Accepted publickey for ROOT from 10.0.0.222 port 1355 ssh2

จากตัวอย่างจะเห็นว่าไอพี 10.0.0.7 นั้นถูก resolve ให้เป็น joe-chong แต่ข้อมูลไอพีอื่นที่อยู่ใน message body คือ 10.0.0.222 นั้น ไม่ได้ถูก resolve ไปด้วย ดังนั้นจึงสรุปได้ว่าออปชัน use_dns(yes) นั้นจะทำการ resolve เฉพาะ hostname ที่อยู่ในส่วนต้นบรรทัดของ message เท่านั้น

นอกจากนี้ออปชันบางตัวที่เกี่ยวข้องกับไฟล์และไดเรกทอรี ยังสามารถใช้งานได้ทั้งใน global options() และ destination() ซึ่งก็คือ modifier ของออปชัน file() เช่น owner(), group() เป็นต้น ทั้งนี้หากมีการระบุค่าออปชันบางตัวที่ซ้ำกันใน options() section และ section อื่นๆ ค่าที่ระบุใน section อื่นๆ จะถูกนำไปใช้แทนที่ค่าใน options() section

keep_hostname() เป็นออปชันที่ใช้งานค่อนข้างมาก ซึ่งจะตั้งค่าดีฟอลต์เป็น no ซึ่งหมายถึง syslog-ng จะไม่ใช้ค่า hostname ที่ส่งมา มันจะทำการ resolve หา hostname จาก source IP address ของแพ็กเกตที่ส่งล็อกเข้ามา เพื่อป้องกันการปลอม hostname จากเครื่องที่ส่งล็อกเข้ามา ซึ่งจะแตกต่างจาก syslog ซึ่งใช้ค่า hostname ตามที่ได้รับมาจาก log message

chain_hostnames() โดยดีฟอลต์มีค่าเป็น yes ซึ่งหมายถึง syslog-ng จะแสดงรายชื่อ host ทุก host ที่ message ถูกส่งต่อมา)relayed by syslog-ng) โดย host ดังกล่าวต้องเป็น host ที่ติดตั้ง syslog-ng และทำหน้าที่ redirect ข้อมูลล็อกมายัง log server (ไม่ใช่ host ที่เป็น network host ตามปกติ เช่น router, firewall)

ตัวอย่างที่ 2 แสดงผลของการใช้งาน keep_hostname() และ chain_hostnames() ซึ่งทั้งสองค่าถูกตั้งค่าดีฟอลต์ให้เป็น yes โดยในตัวอย่างข้อมูลล็อกจะถูกสร้างขึ้นโดยเครื่องปัจจุบัน)locally) จากนั้นจะถูกส่งต่อไปยัง host1 ซึ่งมี hostname จริงๆ เป็น "linux" ซึ่งจะส่งข้อมูลล็อกต่อไปยัง host2 โดย host2 จะทำหน้าที่ตรวจสอบ hostname ผ่านทาง DNS จากนั้นล็อกจึงจะถูกส่งต่อไปยัง host3 ต่อไป

ตัวอย่างที่ 2 แสดงตัวอย่างล็อกที่ถูกส่งต่อผ่านโฮสต์

```
Original log entry on host1:
Oct 9 23:57:16 s_loc@linux syslog-ng[1656]: syslog-ng version 1.4.13 starting
Entry as sent to and recorded by host2:
Oct 9 23:57:16 s_loc@linux/host1 syslog-ng[1656]: syslog-ng version 1.4.13 starting
Same log entry as relayed from host2 to host3:
Oct 9 23:57:16 s_loc@linux/host1/host2 syslog-ng[1656]: syslog-ng version 1.4.13 starting
```

สิ่งที่น่าสนใจจากตัวอย่างที่ 2 คือ

- เมื่อ host2 บันทึกข้อมูลล็อก ตัว syslog-ng ได้ตรวจสอบข้อมูลจาก DNS แล้วพบว่า จริงๆ แล้ว host1 นั้นมี DNS name เป็น linux แต่ syslog-ng เองก็ยังไม่มั่นใจ จึงเพิ่ม hostname "linux" ต่อท้าย hostname "host1" (host1 อาจจะเป็นชื่อที่ปลอมมา
- timestamp ที่ระบุในล็อกทั้งสามชุดมีเวลาที่ตรงกัน ซึ่งหมายถึง เวลาที่เห็นนั้นถูกสร้างขึ้นจากเครื่องที่ให้กำเนิดล็อกแล้วจึงส่งล็อกต่อไป เรื่อยๆ ผ่านโฮสต์ต่างๆ ซึ่งโฮสต์เหล่านั้นไม่ได้ตั้งค่า use_time_recd() ให้เป็น yes โฮสต์ต่างๆ จึงไม่ได้แก้ไขข้อมูล timestamp จึงมีผลให้เวลาทั้งสามจุดตรงกันหมด
- จากข้อมูลล็อกที่ host1 จะพบคำว่า s_loc อยู่ ซึ่งค่าดังกล่าวเป็นค่า source{} ของ syslog-ng ที่อยู่บน host1

ตัวอย่างที่ 3 แสดง configuration ของ syslog-ng บนเครื่อง host1

```
options{};
source s_loc {unix-stream("/dev/log"); internal(); };
destination d_host2 { udp("host2" port(514)); };
destination d_local { file("/var/log/messages"); };
log { source(s_loc); source(s_net); destination(d_host2); destination(d_local); };
```

Sources

จากตัวอย่างที่ 3 มีการประกาศค่า source{} หนึ่งครั้ง โดยข้อมูลภายใน source{} ซึ่งก็คือ source driver ทำหน้าที่ระบุถึงแหล่งที่มาของข้อมูลล็อก ทั้งนี้ใน syslog-ng.conf หนึ่งๆ สามารถประกาศ source{} ได้ไม่จำกัดครั้ง ซึ่งภายใน source{} แต่ละตัวนั้นสามารถบรรจุ driver ได้ไม่จำกัดเช่นกัน

รูปแบบการประกาศ source{}

source sourcelabel1 { drivers([options]); drivers([options]); etc. };

โดย sourcelabel หมายถึง string ที่ใช้เพื่ออ้างอิงกลุ่มของ source driver เพื่อให้สามารถนำไปใช้งานต่อได้อย่างสะดวก เช่น

source s_loc { unix-stream("/dev/log"); internal(); };

จากบรรทัดด้านบน s_loc เป็นชื่อที่ถูกใช้เพื่ออ้างอิงถึงข้อมูลล็อกที่ถูกดึงมาจาก /dev/log และข้อมูลล็อกจาก syslog-ng เอง

syslog-ng มีความหยืดหยุ่นอย่างมากในการใช้งาน source driver ซึ่งสามารถรับข้อมูลล็อกได้จาก Unix socket เช่น /dev/log หรือล็อกจาก syslog-ng เอง รวมทั้งล็อกที่ส่งมาจากเครื่องอื่นผ่านทาง TCP, UDP protocol และยังสามารถรับล็อกจากไฟล์พิเศษเช่น ไฟล์ใน /proc ได้อีกด้วย ตารางที่ 2.6

ตารางที่ 2.6 Source drivers

Source	Description
internal()	ล็อกที่รับมาจาก syslog-ng daemon เอง
file("filename" [options])	ล็อกที่อ่านมาจากไฟล์ที่ระบุไว้ เช่น /proc/kmsg
pipe("filename")	ล็อกที่รับมาจาก name pipe
unix-stream("filename" [options])	ล็อกที่รับมาจาก Unix socket ที่อยู่ใน โหมด connection-oriented stream เช่น /dev/log (maximum concurrent connections default = 100)
unix-dgram("filename" [options])	ล็อกที่รับมาจาก Unix socket ที่อยู่ใน โหมด connectionless datagram เช่น ล็อกของ klogd จาก /dev/log
tcp([ip(address)] [port(#)] [max-connections(#)])	ล็อกที่รับมาจากเครื่องอื่นที่ส่งข้อมูลผ่านทาง TCP ตามหมายเลขพอร์ตที่ระบุ (default = 514) โดยรับข้อมูลจาก local network interface (default = all) และสามารถระบุจำนวน concurrent connections ได้)default = 10)
udp([ip(address)] [port(#)])	ล็อกที่รับมาจากเครื่องอื่นที่ส่งข้อมูลผ่านทาง UDP ตามหมายเลขพอร์ตที่ระบุ (default = 514) โดยรับข้อมูลจาก local network interface (default = all)

internal()

syslog-ng เองจะส่งข้อมูลล็อก เช่น startup message, errors หรือล็อกอื่นๆ ไปยัง internal() ดังนั้น หากต้องการรับล็อกของตัวโปรแกรม syslog-ng จะต้องระบุ internal() ไว้ใน source{} ด้วย

file()

file() ใช้เพื่อระบุชื่อไฟล์ที่ต้องการให้ syslog-ng ไปดึงข้อมูลล็อกมา เช่น ไฟล์ /proc/kmsg ซึ่งเป็นไฟล์ข้อมูลล็อกของเคอร์เนลหากต้องการให้ syslog-ng ดึงข้อมูลล็อกจาก text file ปกติ เช่น ล็อกของ httpd นั้น จะต้องสร้างสคริปต์ขึ้นมาเพิ่มเติมเพื่อทำหน้าที่ pipe ผลลัพธ์ของคำสั่ง tail -f [filename] ไปยัง logger (ดูรายละเอียดเพิ่มเติมเกี่ยวกับการใช้งาน logger ได้จากคำสั่ง # man logger)

unix-stream(), unix-dgram()

เป็น source driver ที่สำคัญ โดยจะรับข้อมูลจากการเชื่อมต่อแบบ connection-oriented และ connectionless Unix socket สำหรับลีนุกซ์ที่ใช้เคอร์เนลเวอร์ชัน 2.4.1 หรือสูงกว่านั้น จะใช้งาน Unix datagram socket ดังนั้นหากต้องการเก็บข้อมูลล็อกของ /dev/log จะต้องใช้ unix-dgram("/dev/log") เท่านั้น จึงจะสามารถได้รับล็อกตามปกติ เช่น

source s_loc { unix-dgram("/dev/log"); internal(); };

หากใช้ลีนุกซ์ที่มีเวอร์ชันของเคอร์เนลเป็น 2.4.0 หรือต่ำกว่า จะต้องใช้ unix-stream() ในการเก็บข้อมูลล็อกจาก /dev/log

tcp(), udp()

ทั้ง tcp() และ udp() จะรับข้อมูลล็อกจาก remote host ผ่านทาง TCP protocol (connection-oriented) และ UDP protocol (connectionless) โดยทั้งคู่สามารถตั้งให้รอรับข้อมูลล็อกผ่านทาง IP address และ port ที่ระบุได้ โดยดีฟอลต์แล้ว syslog-ng จะรอรับการเชื่อมต่อที่ 0.0.0.0:514 ซึ่งหมายถึง "รอรับการเชื่อมต่อที่ทุก network interface, port 514"

การระบุ IP address มีประโยชน์สำหรับโฮสต์ที่มี network interface มากกว่าหนึ่ง และต้องการเปิดพอร์ตรอรับล็อกจากบาง interface เท่านั้น ดังตัวอย่างที่ 4

ตัวอย่างที่ 4 ตัวอย่างการระบุ ip, port ใน source{}

```
source s_tcpmessages { tcp( ip(192.168.1.19) port(10514) ); };
source s_udpmessages { ucp(); };
```

จากตัวอย่างที่ 4 ซึ่งกำหนดให้ s_tcpmessages รับข้อมูลล็อกทุกอันที่ส่งมายัง network interface ที่มีไอพีเป็น 192.178.190.190 TCP port 10514 ส่วน s_udpmessages นั้นรอรับข้อมูลล็อกทุกอันผ่านทาง UDP port 514 ในทุกๆ local network interface

ip(), port(), max_connections()

นอกเหนือจาก ip() และ port() แล้ว ยังมี max_connections() ซึ่งใช้ร่วมกับ tcp() เพื่อจำกัดจำนวนการเชื่อมต่อพร้อมกันสูงสุด ซึ่งการใช้งานออปชันนี้ต้องใช้ค่าที่เหมาะสมกับระบบ เพราะหากกำหนดค่าที่มากไปอาจจะมีผลให้ล็อกบางส่วนถูกทิ้ง)drop) ไปเมื่อเซิร์ฟเวอร์ทำงานเกินพิกัด หากกำหนดน้อยเกินไปและมีการเชื่อมต่อเพื่อส่งล็อกถึงขีดที่กำหนดไว้ จะมีผลให้ข้อมูลล็อกถูก drop ไป จนกระทั่งจะมีช่องว่างเพียงพอที่จะสร้างการเชื่อมต่อ

ตัวอย่างที่ 5 ตัวอย่างการใช้งาน max-connections()

```
source s_tcpmessages { tcp( ip(192.168.1.19) port(10514) max-connections(100)); };
```

ค่าดีฟอลต์ของ max-connections() สำหรับ unix-stream() มีค่าเป็น 100 และสำหรับ tcp() มีค่าเป็น 10

Destinations

syslog-ng สามารถเก็บข้อมูลล็อกในรูปแบบเดียวกันกับที่ syslog เก็บได้ ไม่ว่าจะเป็น ASCII file, name pipe, remote host (ผ่านทาง UDP) และแสดงผลออกทาง TTY นอกจากนี้ syslog-ng ยังสามารถส่งข้อมูลล็อกไปยัง Unix socket, remote host (ผ่าน TCP) และส่งต่อไปยัง standard input ของโปรแกรมอื่น

ตารางที่ 2.7 Destination drivers

Driver	Description
file("filename [$MACROS]")	เก็บข้อมูลล็อกลง Ascii file ตามปกติ หาก syslog-ng ไม่พบไฟล์ตามที่ระบุ มันจะสร้างให้โดยอัตโนมัติ ส่วน MACRO นั้น ใช้เพื่อกำหนดชื่อไฟล์แบบ dynamic เช่น ตั้งชื่อไฟล์ตาม facility ของข้อมูลล็อก (โปรดอ่านรายละเอียดเพิ่มเติม ที่เอกสารเผยแพร่เรื่อง "ทำความรู้จักกับ syslogd")
tcp("address" [port(#);])	ส่งข้อมูลล็อกไปยัง IP address หรือ hostname ที่ระบุผ่านทาง TCP port ที่ระบุ)default port = 514)
udp("address" [port(#);])	ส่งข้อมูลล็อกไปยัง IP address หรือ hostname ที่ระบุผ่านทาง UDP port ที่ระบุ)default port = 514)
pipe("pipename")	ส่งข้อมูลล็อกไปยัง name pipe เช่น /dev/xconsole

ตารางที่ 2.7 (ต่อ) Destination drivers

unix-stream("filename" [options])	ส่งข้อมูลล็อกไปยัง Unix socket แบบ connection-oriented เช่น /dev/log
unix-dgram("filename" [options])	ส่งข้อมูลล็อกไปยัง Unix socket แบบ connectionless เช่น /dev/log
usertty(username)	ส่งข้อมูลล็อกไปยัง console ของ user ที่ระบุ
program("/path/to/program")	ส่งข้อมูลล็อกเพื่อนำไปเป็น standard input ของโปรแกรมที่ระบุ

syslog-ng สามารถเก็บข้อมูลลงไฟล์ได้และมีความสามารถมากกว่า syslog ตรงที่มีการใช้งานมาโคร มาโครช่วยให้สามารถตั้งชื่อไฟล์ที่ใช้เก็บข้อมูลล็อกได้อย่างน่าดี เช่น ตั้งชื่อไฟล์ตามปี เดือนวัน หรือตั้งชื่อไฟล์ตาม facility, priority

ตัวอย่างที่ 6 ตัวอย่างการใช้งานมาโคร

```
destination d_dailylog { file("/var/log/messages.$WEEKDAY"); };
```

จากตัวอย่าง configuration ด้านบน เมื่อ syslog-ng ต้องการเขียนข้อมูลล็อกลงไฟล์ มันจะสร้างไฟล์ชื่อ /var/log/messages.Tues, /var/log/messages.Wed ซึ่งขึ้นกับวันที่เก็บข้อมูลล็อกดังกล่าว

ตารางที่ 2.8 Macros supported in file() destinations

Macro	Expands to
Program	ชื่อของโปรแกรมที่ส่งล็อกเข้ามา
HOST	ชื่อโฮสต์ที่เป็นจุดกำเนิดล็อก
FACILITY	facility ของล็อกที่ถูกส่งเข้ามา
PRIORITY or LEVEL	priority ของล็อกที่ถูกส่งเข้ามา
YEAR	ปีปัจจุบัน[a]
MONTH	เดือนปัจจุบัน[a]
DAY	วันที่ปัจจุบัน[a]
WEEKDAY	วันปัจจุบัน[a] เช่น Monday
HOUR	ชั่วโมงปัจจุบัน[a]
MIN	นาทีปัจจุบัน[a]
SEC	วินาทีปัจจุบัน[a]

หากออปชัน use_time_recvd() ถูกตั้งค่า yes แล้ว ข้อมูลเวลาจะอ้างอิงจาก local system ขณะที่ล็อกเดินทางมาถึง แต่หาก use_time_recvd() มีค่าเป็น no ก็จะอ้างอิงเวลาจากเวลาที่ปรากฏในข้อมูลล็อก syslog-ng จะสร้างไฟล์ขึ้นมาใหม่ หากไฟล์ที่ระบุใน file() ไม่มีอยู่จริง นอกจากนี้ syslog-ng ยังสามารถกำหนดออปชันบางตัวในระดับทั่วไป)general rule) คือให้มีผลกับ configuration ทั้งไฟล์ได้ ขณะเดียวกันก็สามารถกำหนดออปชันในระดับ per-log-file ได้ ซึ่งการกำหนดออปชันชนิดหลังนี้จะเป็นการ overridden ออปชันในระดับ general rule

ตัวอย่างที่ 7 การควบคุม file()

```
destination d_mylog { file("/var/log/ngfiles/mylog" create_dirs(yes)\
dir_owner(root) dir_group(root) dir_perm(700)); };
```

จากตัวอย่างที่ 7 เป็นการระบุออปชัน dir_owner(), dir_group(), dir_perm() ใน destination{} ซึ่งค่าที่ระบุนี้จะมีผลแทนที่ค่าที่ระบุใน options{} โดยอัตโนมัติ นอกจากนี้ยังสามารถระบุออปชัน owner(), group(), perm() ได้เช่นเดียวกันกับออปชันด้านบน

โดยปกติ syslog-ng จะสร้างไฟล์ล็อกที่ไม่มีอยู่ในระบบโดยอัตโนมัติ เว้นเสียแต่ว่าไฟล์ที่ระบุดังกล่าวจะอยู่ใน path ที่ไม่มีอยู่จริงและออปชัน create_dirs() ถูกตั้งค่าเป็น no

sync() ถูกใช้เพื่อจำกัดความถี่ในการ synchronize ไฟล์ล็อก หากมีค่าสูงๆ จะทำให้ข้อมูลล็อกถูกนำไปเก็บไว้ที่แคช)cache) เป็นจำนวนมากก่อนที่จะถูก synchronize หรือบันทึกลงไฟล์ล็อกต่อไป หาก sync() มีค่าต่ำ ก็เป็นการลดความเสี่ยงในการสูญเสียข้อมูล เพราะข้อมูลที่ถูกประมวลผลแล้วจะถูกบันทึกลงไฟล์ล็อกทันที

โดยดีฟอลต์แล้ว ค่าล็อกถูกตั้งค่าเป็นศูนย์ ซึ่งหมายถึงให้บันทึกข้อมูลล็อกทุกอันในทันที โดยปกติค่า sync() ต่ำๆ จะเหมาะสำหรับระบบที่ข้อมูลล็อกไม่เยอะมาก ส่วนระบบที่มีข้อมูลล็อกจำนวนมากควรใช้ค่า sync() สูง ซึ่งค่าระหว่าง 100 ถึง 1000 นั้นถือว่ามีค่าสูงพอสมควร ซึ่งผู้ดูแลระบบจะต้องทดสอบเพื่อหาค่าที่เหมาะสมกับระบบของตนต่อไป

อย่างไรก็ตามหากระบบที่ติดตั้ง syslog-ng ได้ติดตั้งโปรแกรมจำพวก log monitoring tool เช่น Swatch แล้วไม่ควรตั้งค่า sync() ไว้สูงมากนัก เพราะอาจจะทำให้ไม่สามารถแจ้งเตือนผู้ดูแลระบบได้ในกรณีที่ไฟล์ล็อกโดนลบ

Filters

filter หรือการกรองข้อมูลเป็นส่วนที่มีความสำคัญส่วนหนึ่ง นอกเหนือจากการกรองข้อมูลโดยใช้ facility, priority แล้ว syslog-ng ยังสามารถตรวจสอบชื่อโปรแกรมที่ส่งข้อมูลล็อกมา ชื่อเครื่องที่ทำหน้าที่ส่งต่อล็อกมา และยังสามารถกรองข้อมูลล็อกตาม regular expression ที่ตั้งไว้อีกด้วย

filter{} statement ประกอบไปด้วย label (ชื่อเรียกของ filter{} ชุดนั้นๆ (และคำสั่งในการกรองข้อมูลอย่างน้อย 1 คำสั่ง โดยสามารถใช้ and, or, not ในการเชื่อมคำสั่งในการกรองข้อมูลได้

ตารางที่ 2.9 filter{} funtions

Function (criteria)	Description
facility(facility-name)	facility ที่ต้องการ
priority(priority-name) **priority**(priority-name1, priority-name2, etc,) **priority**(priority-name1 .. priority-name2)	ระดับของ priority ที่ต้องการ - สามารถใช้เครื่องหมาย comma (,) คั่น หากต้องการมากกว่าหนึ่งระดับได้ - สามารถใช้เครื่องหมาย ..แทน priority ที่ต้องการระหว่าง priority ที่กำหนดได้ เช่น info .. warn
level(priority-name)	เช่นเดียวกันกับ priority
program(program-name)	ชื่อโปรแกรมที่สร้างล็อกขึ้นมา
host(hostname)	ชื่อ host ที่ล็อกนี้ถูกสร้าง
match(regular-expression)	regular expression ที่จะถูกนำไปเปรียบกับกับส่วน body ของล็อก
filter(filter-name)	ชื่อ filter อื่นที่ต้องการนำมากรองอีกครั้ง

จากตัวอย่างที่ 8 แสดง syslog-ng.conf ในระบบปฏิบัติการลินุกซ์เดเบียน 2.2 (Debian 2.2)

ตัวอย่างที่ 8 ตัวอย่างการใช้งาน filter{}

```
filter f_mail { facility(mail); };
filter f_debug { not facility(auth, authpriv, news, mail); };
filter f_messages { level(info .. warn) and not facility(auth, authpriv, cron, daemon, mail, news);
};
filter f_cother { level(debug, info, notice, warn) or facility(daemon, mail); };
```

บรรทัดแรกในตัวอย่างที่ 8 filter f_mail กรองได้ข้อมูลล็อกทุกอันที่อยู่ใน facility mail

บรรทัดที่สอง filter f_debug กรองได้ข้อมูลล็อกทุกอันยกเว้น facility auth, authpriv, news, และ mail

บรรทัดที่สาม filter f_messages กรองได้ข้อมูลล็อกทุกอันที่มี priority ตั้งแต่ info จนถึง warn ยกเว้นข้อมูลล็อกที่มี facility เป็น auth, authpriv, cron, daemon, mail, news

บรรทัดสุดท้าย filter f_cother กรองข้อมูลล็อกที่มี priority เป็น debug, info, notice และ warn **หรือ** ข้อมูลล็อกที่มี facility เป็น daemin และ mail

Log statements

หลังจากที่ทำความเข้าใจส่วนประกอบต่างๆ คือ sources, filters และ destinations แล้ว ก็จะนำส่วนประกอบทั้งหมดมารวมไว้ใน log{}

ตัวอย่างที่ 9 ตัวอย่าง syslog-ng.conf

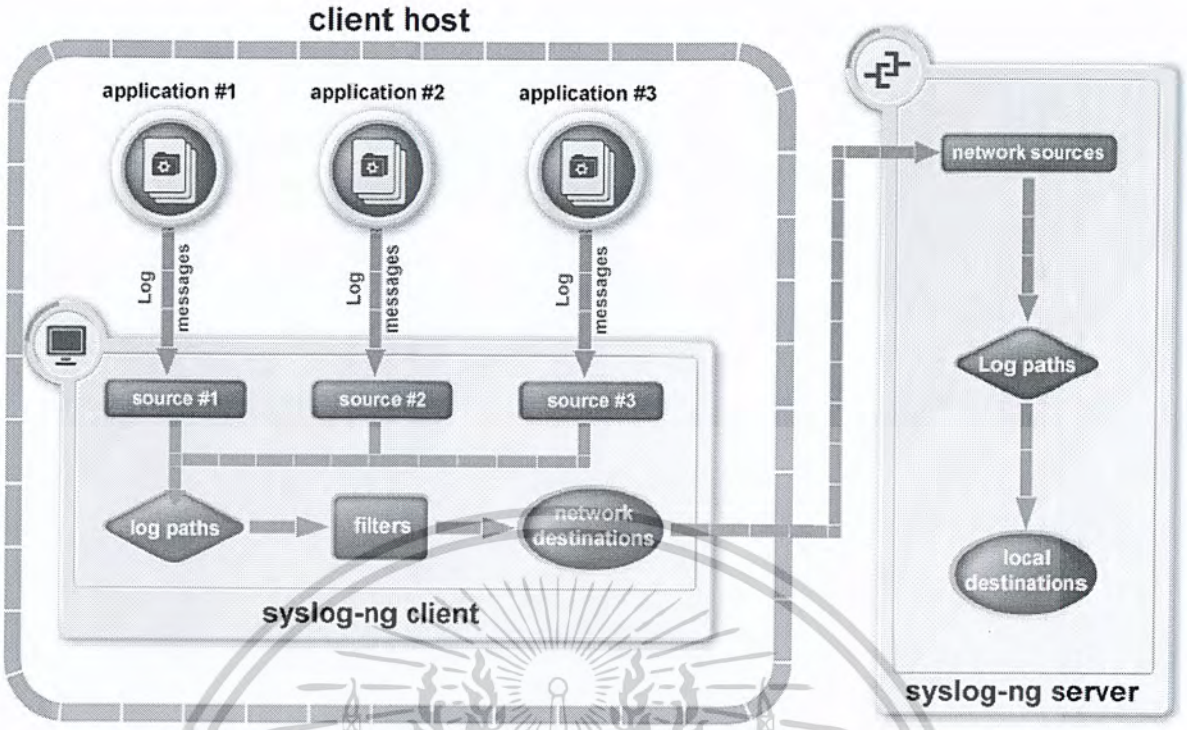
```
source s_loc { unix-stream("/dev/log"); internal(); };
source s_tcpmessages { tcp( ip(192.168.1.19); port(10514);); };
destination d_dailylog { file("/var/log/messages.$WEEKDAY"); };
destination d_untlog { file("/var/log/untlog" owner(unt)) perm(0600)); };
filter f_mail { facility(mail); };
filter f_messages { level(info .. warn) and not facility(auth, authpriv, cron, daemon, mail, news);
};
log { source(s_tcpmessages); destination(d_untlog); };
log { source(s_loc); filter(f_mail); destination(d_untlog); };
log { source(s_loc); filter(f_messages); destination(d_dailylog); };
```

จาก log statement บรรทัดแรกนั้น จะทำให้ข้อมูลล็อกทุกอันที่มาจากเครื่อง 192.168.1.19 จะถูกบันทึกลงในไฟล์ /var/log/untlog บรรทัดที่สองจะทำให้ข้อมูลล็อกของเมล์ (facility mail) ของ localhost ถูกบันทึกลงในไฟล์ /var/log/untlog บรรทัดที่สามจะทำให้ข้อมูลล็อกของ localhost ที่ผ่านการกรองของ filter f_messages ถูกบันทึกลงในไฟล์ /var/log/messages.**$WEEKDAY** เช่น /var/log/Mon, /var/log/Sun

จากตัวอย่างที่ 9 อาจจะเกิดข้อสังสัยว่า ล็อกบางส่วนที่ไม่ได้ถูกจัดเก็บโดย log{} statement ทั้งสามตัวนั้นจะถูกจัดเก็บไว้ที่ใด syslog-ng มีค่า filter(DEFAULT) ซึ่งสามารถใช้ระบุในตอนท้ายเพื่อสั่งให้ syslog-ng บันทึกข้อมูลล็อกที่ไม่ได้ถูกจัดเก็บโดย log{} ก่อนหน้านี้ได้ ดังตัวอย่างที่ 10

ตัวอย่างที่ 10 ตัวอย่าง syslog-ng.conf

```
log { source(s_tcpmessages); destination(d_untlog); };
log { source(s_loc); filter(f_mail); destination(d_untlog); };
log { source(s_loc); filter(f_messages); destination(d_dailylog); };
log { source(s_loc); filter(DEFAULT); destination(d_dailylog); };
```



รูปที่ 2.13 แสดงการทำงานของ centralized log server

2.3 Extensible Messaging and Presence Protocol (XMPP)

Extensible Messaging and Presence Protocol (XMPP) เป็นโปรโตคอลสำหรับการสื่อสารที่ใช้การส่งข้อความเป็นหลัก โดยมีอีกชื่อหนึ่งว่า Jabber ซึ่งถูกใช้กันอย่างแพร่หลายในโปรแกรมประเภท ส่งข้อความด่วนเช่น Facebook chat หรือ Google Talk โดยหลักการทำงานของโปรโตคอล XMPP จะใช้ภาษา XML ในการสื่อสารเป็นหลัก ซึ่งมีลักษณะคล้ายกับระบบจดหมายอิเล็คทรอนิคส์ หรือ อีเมลล์ โดยใครก็สามารถเปิดให้บริการ XMPP ได้โดยไม่จำเป็นต้องมีเครื่องแม่ข่ายกลาง และยังสามารถส่งข้อความถึงกันได้ แม้ว่าจะไม่ได้อยู่ในระบบเครือข่าย XMPP เดียวกันก็ตาม

หลักการทำงาน

การกระจายตัวและการอ้างอิง

เครือข่าย XMPP นั้นมีสถาปัตยกรรมแบบ ลูกข่าย-แม่ข่าย พอร์ท TCP ที่ใช้ในการสื่อสารคือ 5222 และ 5223 เมื่อเปิดใช่งาน TLS โดยเครื่องลูกข่ายไม่สามารถเชื่อมต่อกันได้โดยตรง จะต้องผ่านเครื่องแม่ข่ายกลางก่อนเสมอ แต่อย่างไรก็ตาม โปรโตคอล XMPP สามารถทำงานได้โดยไม่ต้องมี เครื่องแม่ข่ายกลาง เหมือนระบบการส่งข้อความด่วนแบบอื่นๆ เช่น Windows Live Messenger AOL Instant Messenger

โปรโตคอลนี้ใช้ บัญชีชื่อผู้ใชงาน)Jabber ID หรือ JID) ในการติดต่อสื่อสารระหว่างกัน โดยมีลักษณะคล้ายระบบอีเมลล์คือ มีชื่อผู้ใช้ตามด้วยเครื่องหมาย @ และลงท้ายด้วยชื่อโดเมน หรื

อาจจะมีส่วนของ resource ติดท้ายมาด้วย เช่น username@example.com/home ส่วนของ resource จะเป็นตัวบอกถึงคุณลักษณะพิเศษของลูกข่ายแต่ละตัว เช่นในกรณีที่ผู้ใช้งานมีเครื่องที่ต้องการใช้งานหลายๆเครื่อง ผู้ใช้งานสามารถเติมส่วนของ resource ลงไปเองได้เพื่อบ่งบอกที่อยู่ของเครื่องลูกข่าย

การส่งข้อความ

ในกรณีที่ผู้ส่งและผู้รับมีบัญชีผู้ใช้งานอยู่คนละโดเมน

ตัวอย่าง a@aaa.com และ b@bbb.com ต้องการส่งข้อความถึงการ

1. a ทำการส่งข้อความไปหา b ผ่านเครื่องแม่ข่าย aaa.com ถ้า bbb.com นั้นถูกบล๊อคจาก aaa.com การส่งข้อความจะถูกปฏิเสธ
2. เครื่องแม่ข่าย aaa.com เชื่อมต่อไปยัง เครื่องแม่ข่าย bbb.com ถ้า aaa.com นั้นถูกบล๊อคจาก bbb.com การส่งข้อความจถูกปฏิเสธ
3. bbb.com ตรวจสอบว่า ผู้ใชงาน b ทำการเชื่อมต่ออยู่หรือไม่ ถ้าไม่ ข้อความจะถูกเก็บไว้เผื่อส่งให้ภายหลัง
4. ถ้า b ทำการเชื่อมต่ออยู่ bbb.com จะทำการส่งข้อความให้ทันที

การเชื่อมต่อกับโปรโตคอลสื่อสารอื่นๆ

โปรโตคอล XMPP สามารถทำการเชื่อมต่อกับโปรโตคอลสื่อสารอื่นๆ ไม่ว่าจะเป็น ICQ SMS Windows Live messenger หรือ อีเมลล์ก็ตาม โดยทำการเชื่อมต่อผ่านเกตเวย์ของโปรโตคอลนั้นๆ ซึ่งอาจจะต้องการข้อมูลที่เกี่ยวข้องกับการเชื่อมต่อ ของโปรโตคอลนั้นเพิ่มเติม ซึ่งเครื่องลูกข่ายนั้น ไม่จำเป็นต้องมีความรู้เกี่ยวกับโปรโตคอลอื่น เพราะเกตเวย์จะทำหน้าในการสื่อสารแทนทั้งหมด

การนำไปประยุกต์ใช้งาน

โปรโตคอล XMPP ถูกนำไปประยุกต์ใช้เป็นทั้ง โปรแกรมสำหรับลูกข่าย โปรแกรมสำหรับเตรื่องแม่ข่าย และไลบรารี่

ลูกข่าย

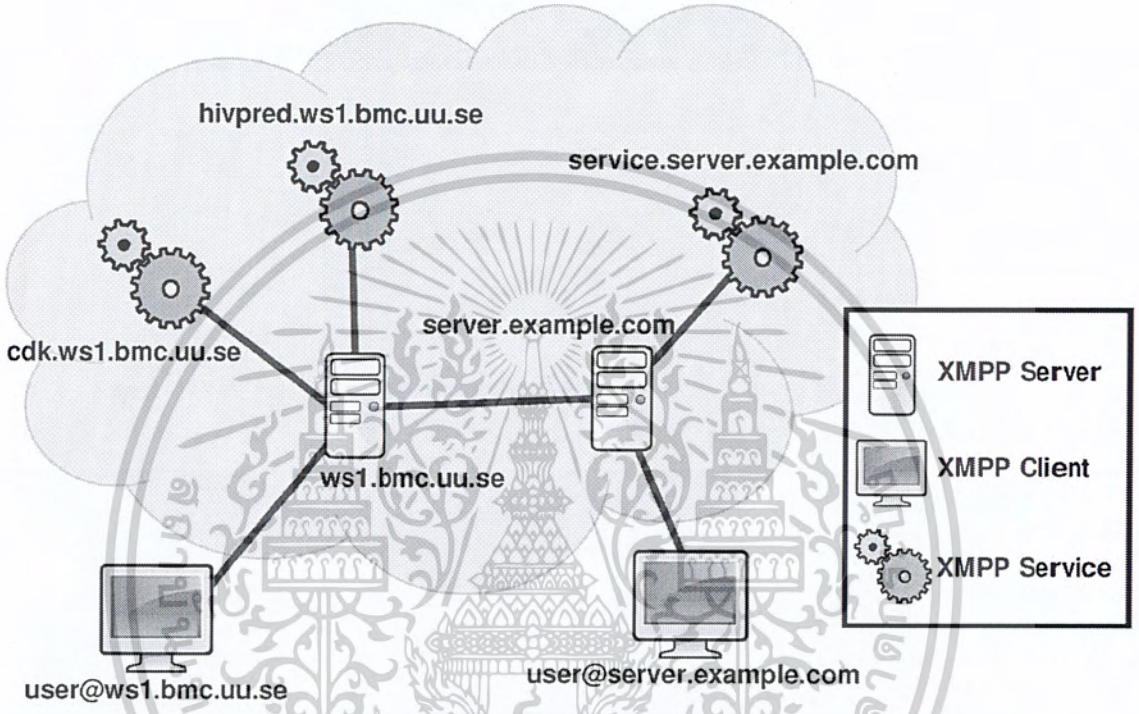
- Adium
- Pandion
- Empathy

แม่ข่าย

- Openfire
- jabberd
- Open IM
- Sun Java System IM

ไลบรารี่

- Smack (Java)
- Twisted Words (Python)
- dojox.xmpp (JavaScipt)



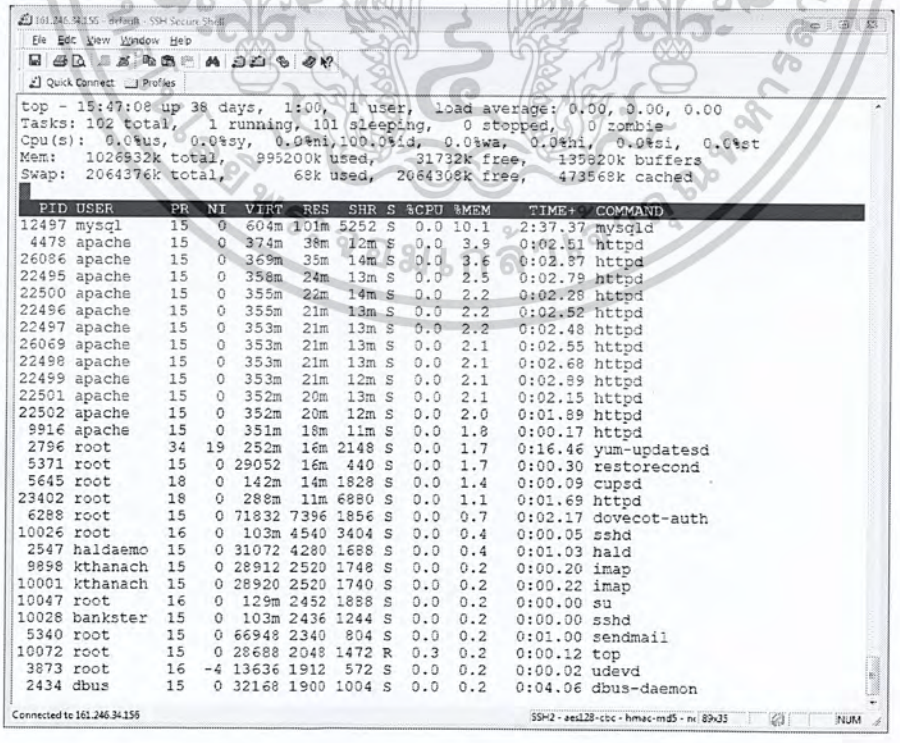
รูปที่ 2.14 แสดงการทำงานของ XMPP

บทที่ 3
การวิเคราะห์และออกแบบระบบงาน

3.1 การตรวจสอบสถานะการทำงานของลินุกซ์/ ยูนิกซ์ เซิร์ฟเวอร์

3.1.1 การตรวจสอบประสิทธิภาพในการทำงาน

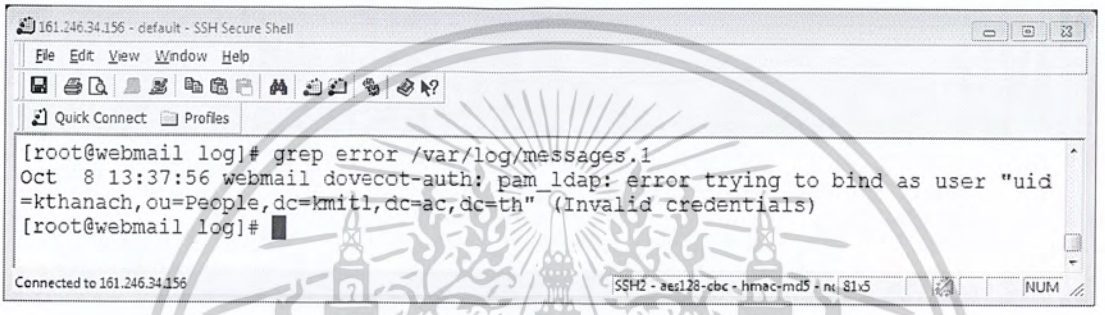
ในการตรวจสอบสถานะการทำงานและการให้บริการของเครื่องแม่ข่ายที่ใช้ระบบปฏิบัติการ Linux หรือ UNIX server ผู้ดูแลระบบจะทำการ login เข้าสู่เครื่องแม่ข่ายผ่าน SSH โดยโช้ Secure Shell client ทั่วไปเช่น putty เป็นต้น และจะคอยเฝ้าสังเกตสถานะการทำงานของเครื่องแม่ข่ายผ่านคำสั่ง top บน Linux หรือ prstat บน Sun Solaris ผลที่ได้คือการใช้ CPU (CPU Utilization) , Load Avarage, IO Wait, การใช้หน่วยความจำ (Memory Usage) ส่วนที่เหลือจะเป็นรายการของ process ที่ทำงานอยู่ในระบบ ซึ่งจะแจ้งให้ทราบถึงรายละเอียด ว่าใครเป็นเจ้าของ Process นั้นๆ และ process นั้นๆ มีการมีใช้ CPU, Memory เป็นจำนวนมากน้อยเพียงใด และมีระยะเวลาที่ Process นั้นทำงานอยู่ในระบบด้วย โดยผู้ดูแลระบบจะใช้ top เปนหลักในการตรวจสอบสถานะและประสิทธิภาพ คำสั่ง top นั้นจะทำการ refresh ตัวเองทุกๆ 5 วินาที ดังเช่นในตัวอย่าง รูปที่ 3.1 จะเห็นได้ว่าการสังเกตถึงความผิดปกติของการใช้ทรัพยากรต้องมีการเฝ้าตรวจสอบอยู่ตลอดเวลา ซึ่งในความเป็นจริงแล้วเป็นเรื่องที่ทำได้ยาก



รูปที่ 3.1 แสดงผลของคำสั่ง top ใน Linux Server

3.1.2 การตรวจสอบเหตุการณ์ผิดปกติ

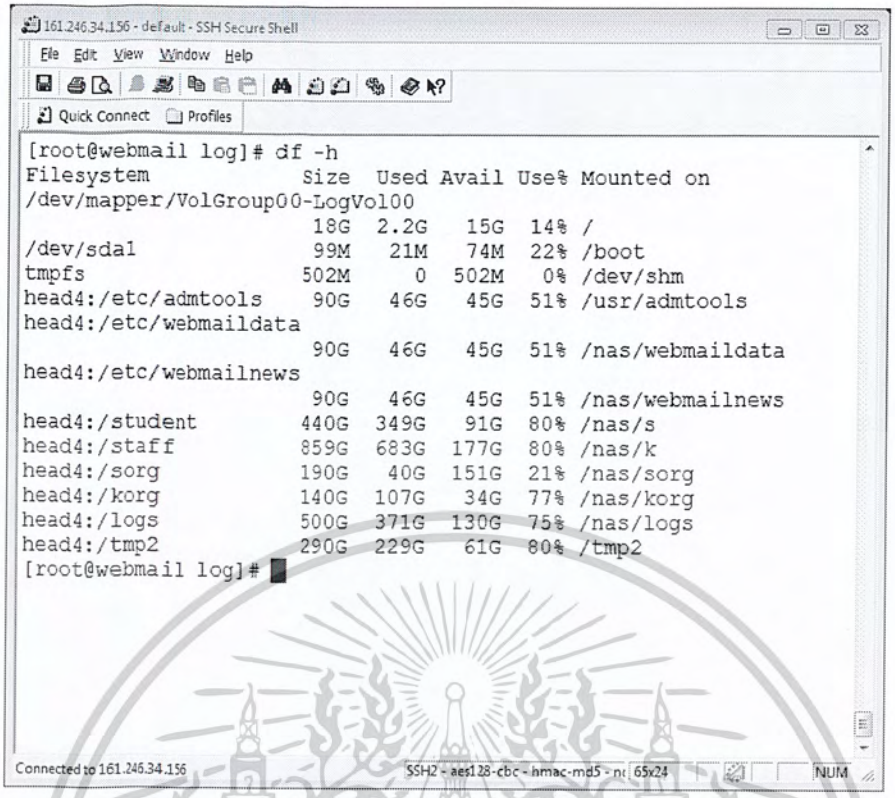
ระบบปฏิบัติการ Linux และ UNIX จะมีการเก็บข้อมูลเหตุการณ์ต่างๆ ของระบบปฏิบัติการและซอฟท์แวร์ ไว้ในรูปของ syslog โดยเก็บข้อมูลลงใน file แยกตามประเภทของ Log เช่น system log อาจจะเก็บไว้ที่ /var/log/syslog หรือที่ /var/adm/messages Email log อาจจะเก็บไว้ที่ /var/log/maillog เป็นต้น การตรวจสอบเหตุการณ์ผิดปกติที่เกิดขึ้นกับเครื่องแม่ข่ายนั้นสามารถทำได้โดยการเปิด log file ขึ้นมา และทำการ filter ข้อมูลที่ต้องการด้วยการใช้คำสั่ง grep, sed, awk, cat เป็นต้น จากตัวอย่างในรูป 3.2 เป็นการแสดงผล ส่วนของไฟล์ messages.1 ที่มีข้อความ “error”



รูปที่ 3.2 แสดงผลของคำสั่ง grep error /var/log/messages.1

3.1.3 การตรวจสอบการใช้งานพื้นที่ Disk และ สถานะของ file system

การตรวจสอบ disk usage สามารถใช้คำสั่ง df –h โดยคำสั่งนี้จะแสดง Disk Usage ของแต่ละ file system ในระบบปฏิบัติการ จากตัวอย่างในรูป 3.3 เป็นการเรียกใช้งานคำสั่ง df –h เพื่อแสดง file system โดยจะแสดงในลักษณะที่มนุษย์สามารถเข้าใจได้ทันที

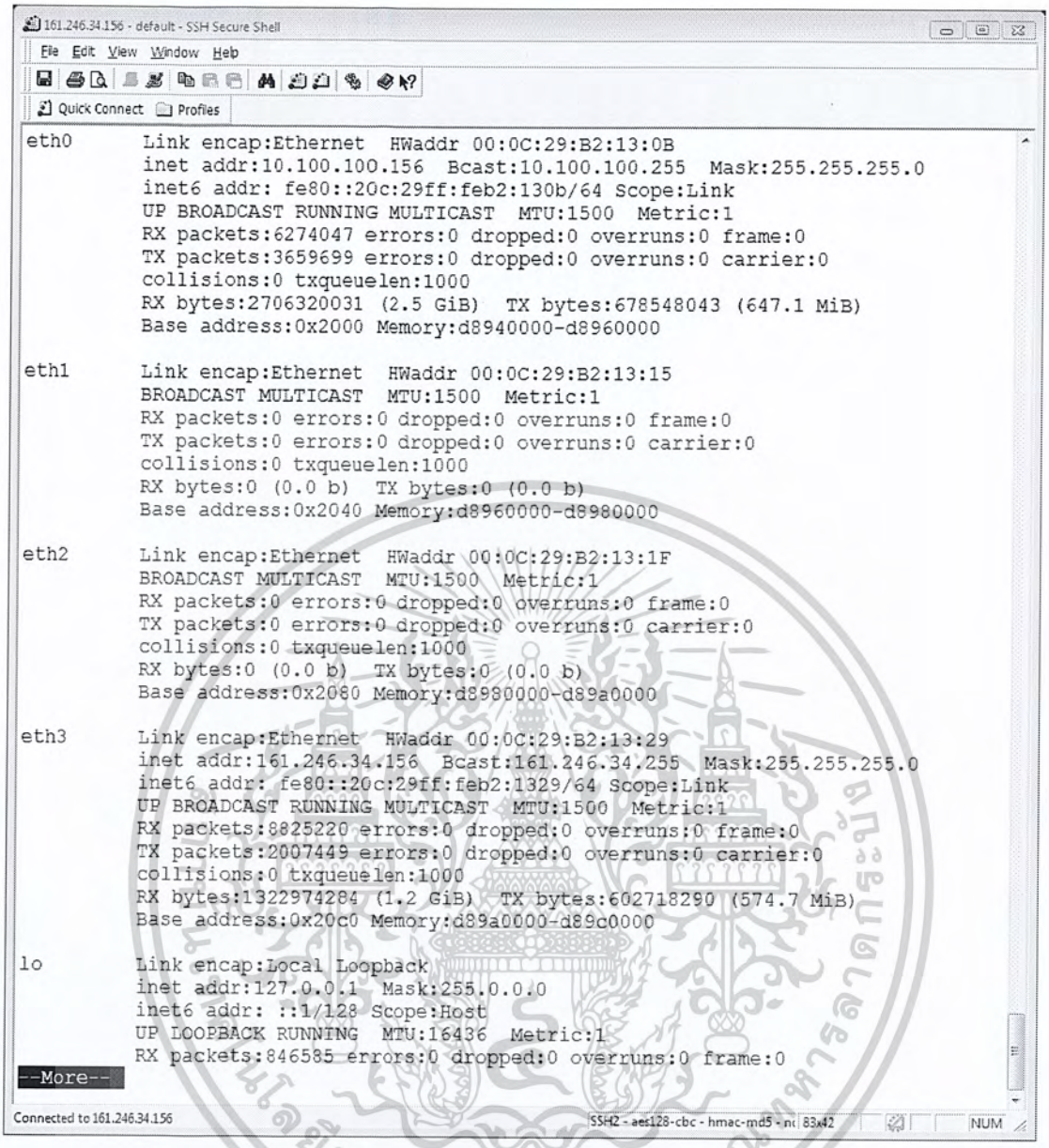


รูปที่ 3.3 แสดงผลของคำสั่ง df –h ใช้ในการตรวจสอบพื้นที่ disk

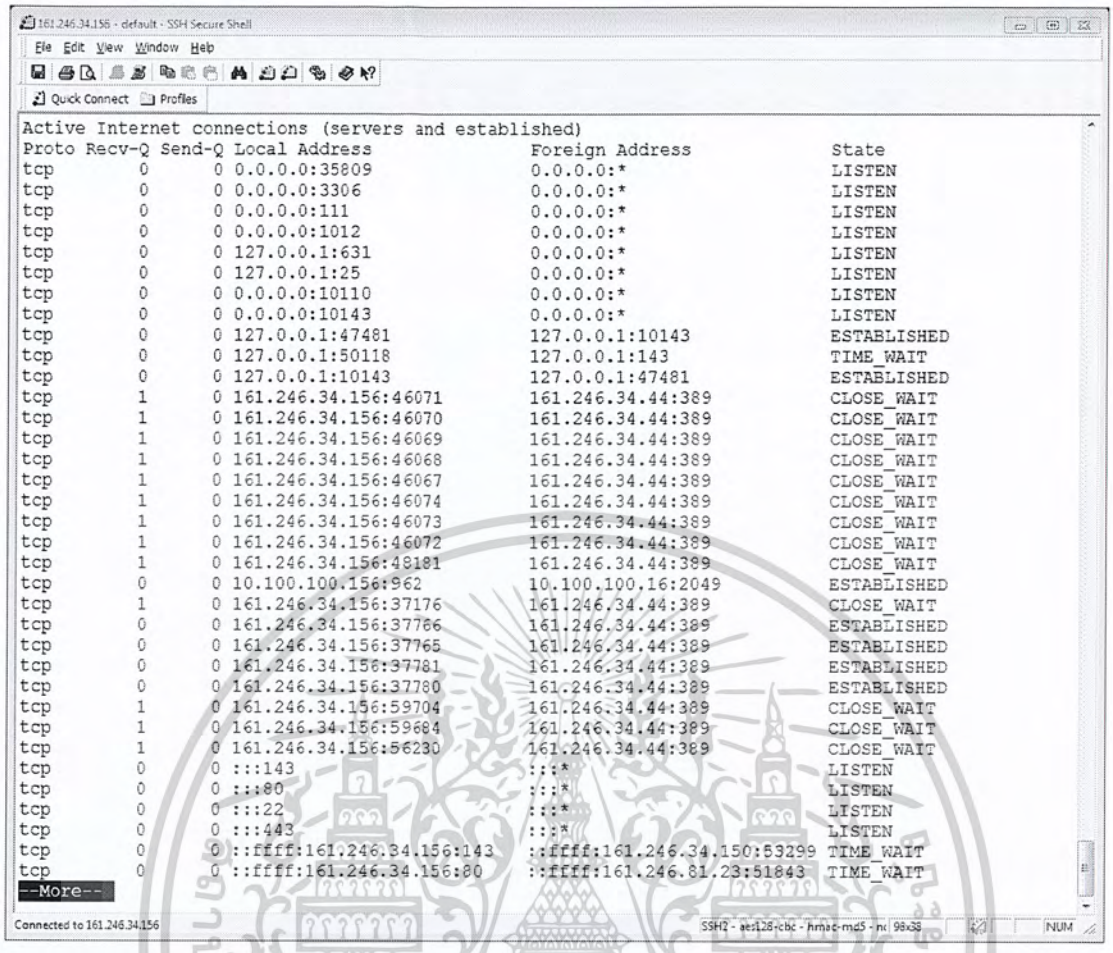
3.1.4 การตรวจสอบ TCP Connection และ network interface card

การตรวจสอบสถานะของ TCP Connection สามารถตรวจสอบได้โดยใช้คำสั่ง netstat –na การตรวจสอบสถานะของ Network Interface Card จะใช้คำสั่ง ifconfig –a แล้วดูสถานะว่า up หรือ down เป็นต้น จากตัวอย่างในรูปที่ 3.4 เป็นการเรียกใช้งานคำสั่ง ifconfig –a โดยผลลัพธ์ของคำสั่งนั้นจะแสดงรายละเอียดของ Network Interface Card ทั้งหมดในเครื่องแม่ข่าย

จากรูปที่ 3.5 เป็นการเรียกใช้งานคำสั่ง netstat –na โดยผลลัพธ์ของคำสั่งนั้นจะแสดงการเชื่อมต่อทั้งหมดที่เกิดขึ้นกับเครื่องแม่ข่าย



รูปที่ 3.4 แสดงผลของคำสั่ง ifconfig -a



รูปที่ 3.5 แสดงผลของคำสั่ง netstat -na

สรุป

การตรวจสอบสถานะการทำงานของเครื่องแม่ข่ายและอุปกรณ์เครือข่าย ด้วยวิธีการที่มีอยู่เดิมนั้น ผู้ดูแลระบบต้องทำหน้าที่ในการตรวจสอบเอง โดย Login เข้าสู่เครื่องแม่ข่ายและอุปกรณ์แต่ละตัวและทำการเรียกใช้คำสั่งเพื่อตรวจสอบสถานะ ซึ่งมีข้อเสียคือ ขาดประสิทธิภาพเนื่องจากอาจจะเกิดปัญหาความต่อเนื่องของการตรวจสอบ ไม่มีการเฝ้าระวังและระบบแจ้งเตือนปัญหาที่เกิดขึ้นโดยอัตโนมัติ

3.2 วิเคราะห์ความต้องการ

ภาพรวม

ระบบจัดการบริหารเครือข่ายหรือ NMS นั้น มีความสำคัญอย่างมากกับองค์กรที่มีเครือข่ายคอมพิวเตอร์ขนาดใหญ่ แต่ซอฟต์แวร์ที่มีอยู่ในปัจจุบันนั้น ผู้ใช้งานหรือผู้ดูแลระบบยังไม่สามารถตอบสนองความต้องการของผู้ดูแลระบบได้ทั้งหมด ดังนั้นจึงทำให้มีการพัฒนาระบบแจ้งเตือนและบริหารเครือข่ายผ่านทางข้อความด่วนทันใจขึ้นมา

ประโยชน์ที่จะได้รับจากการใช้ระบบใหม่นี้ จะช่วยให้ข้อความหรือข้อผิดพลาดที่เกิดขึ้นกับระบบส่งไปถึงมือผู้รับได้รวดเร็ว และผู้ใช้งานยังสามารถส่งคำสั่งกลับไปจัดการระบบในเบื้องต้นได้ ทำให้เวลาที่สูญเสียไปกับการแก้ปัญหานั้นลดน้อยลง อีกทั้งยังช่วยเพิ่มศักยภาพทางเทคโนโลยีสารสนเทศขององค์กรได้อีกด้วย

ความต้องการของระบบ

Functional Requirements

ก. ระบบทำงานแบบรวมศูนย์ (Centralized)
   i. เครื่องแม่ข่ายสำหรับโปรโตคอล XMPP
   ii. เครื่องแม่ข่ายสำหรับการเก็บ Log (Centralized Log Server)
   iii. เครื่องแม่ข่ายสำหรับระบบ NMS

ข. NMS 1 ระบบมีบัญชีผู้ใช้งานของระบบเพียง 1 บัญชี

ค. ระบบสามารถตรวจสอบอุปกรณ์และเฝ้าระวังด้วยโปรโตคอล SNMP

ง. ระบบติดต่อกับผู้ใช้ผ่านโปรโตคอล XMPP

จ. ระบบรับข้อความ Log จาก server ต่างๆในระบบด้วย Syslog-ng

ฉ. ระบบอ่าน Log จาก pipe ของเครื่องแม่ข่าย

ช. ผู้ใช้งานทำการติดต่อหรือรับข้อความจากอุปกรณ์ที่ถูกเฝ้าระวังผ่านบัญชีผู้ใช้งานของ NMS

ซ. ผู้ใช้งานสามารถสื่อสารกับอุปกรณ์ได้ก็ต่อเมื่อมีระดับการเข้าถึง มากกว่าหรือเท่ากับอุปกรณ์นั้น

ฌ. ระบบจัดเก็บ log ในฐานข้อมูล

ญ. ระบบจัดเก็บค่าปรับแต่งทั้งหมดอยู่ในฐานข้อมูล

ฎ. ระบบมีหน้าเว็บอินเตอร์เฟสให้ผู้ใช้ เพิ่มเติม ลบ แก้ไข ข้อมูลได้

ฏ. ผู้ใช้สั่งงานระบบผ่านการป้อนคำสั่ง

ฐ. คำสั่งของระบบมีลักษณะคล้ายกับ IOS ของ CISCO

Non-Functional requirements

ก. เวลาการตอบสนอง (Response Time) ระบบควรตอบสนองต่อข้อความที่ส่งมาจากอุปกรณ์ได้ทันที

ข. ความง่ายของการใช้งาน ในส่วนของเว็บอินเตอร์เฟสและการส่งคำสั่ง ต้องสามารถเข้าใจได้ง่าย และกระชับมากที่สุด

ค. การให้บริการ ระบบต้องสามารถทำงานได้อย่างต่อเนื่อง โดยมี Availability มากกว่า 99%

3.3 การออกแบบระบบ

ในปริญญานิพนธ์ฉบับนี้ได้นำเสนอการพัฒนา ระบบตรวจสอบเฝ้าระวังและแจ้งเตือนสถานะของเครื่องแม่ข่ายโดยใช้โปรโตคอล SNMP ผ่านข้อความด่วนทันใจ (Instant Messaging) โดยออกแบบและพัฒนาบนพื้นฐานของ ประยุกต์ใช้การตรวจสอบสถานะของเครื่องแม่ข่ายและอุปกรณ์เครือข่ายผ่านทาง Simple Network Management Protocol, การจัดการ system log ด้วย syslog-ng และการติดต่อสื่อสารผ่านทางบริการข้อความด่วนทันใจ (Instant Messaging) โดยนำข้อมูลเกี่ยวกับสถานะของเครื่องแม่ข่าย บริการบนเครื่องแม่ข่าย เหตุการณ์ต่างๆ ที่ได้จาก SNMP และ syslog-ng แจ้งเตือนไปยังดูแลระบบ ผ่านทางบริการ instant messaging เพื่อให้ผู้ดูแลระบบได้รับทราบถึงเหตุการณ์ผิดปกติ และข้อผิดพลาด ต่างๆ โดยระบบจะมีการนำข้อมูลจากแหล่งต่างๆ มาประมวลผล ดังนี้

a. SNMP

เป็นการเรียกดูสถานะของเครื่องแม่ข่ายและอุปกรณ์เครือข่ายแต่ละตัวที่ติดตั้ง SNMP Agent ผ่านทาง SNMP Protocolโดยจะอ่านข้อมูลจากเอ็มไอบี (MIB: Management Information Base

b. SNMP Trap

ระบบจะรับข้อความแจ้งเตือน จากเครื่องแม่ข่าย ผ่านทาง SNMP trap และนำข้อความมาประมวลผล และส่งต่อให้ผู้ดูแลระบบผ่านทางบริการ Instant Messaging

c. System log

ระบบจะอ่านข้อมูลเหตุการณ์ และข้อผิดพลาดจากของเครื่องแม่ข่ายจาก syslog และแจ้งเตือนผู้ดูแลระบบผ่านทางบริการข้อความด่วนทันใจ (Instant Messaging) ตามเงื่อนไขที่ได้กำหนดไว้

3.3.1 การออกแบบระบบส่วนของโปรแกรม

สามารถแบ่งระบบโดยภาพรวมได้เป็น 2 ส่วน ได้ ดังนี้

3.3.1.1 ส่วนของ NMS

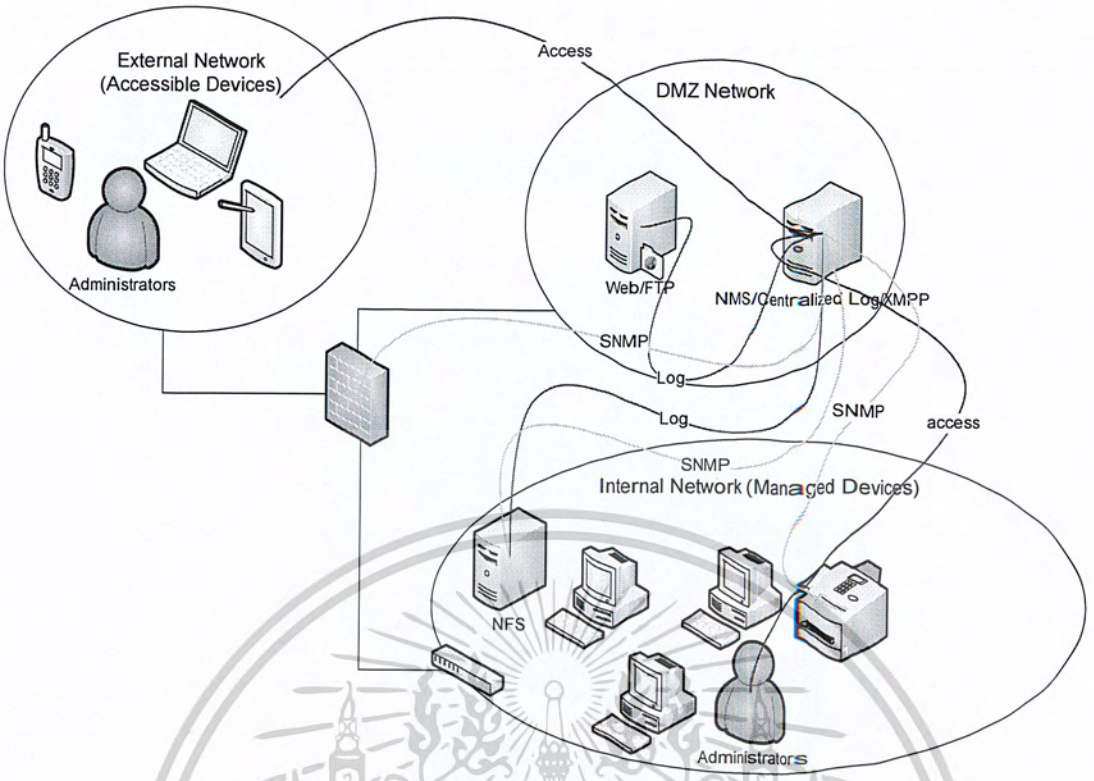
พัฒนาขึ้นโดยใช้ภาษา JAVA ร่วมกับ Library SNMP4j สำหรับการติดต่อผ่านโปรโตคอล SNMP Library Smack สำหรับการติดต่อผ่านโปรโตคอล XMPP และโปรแกรม OpenFire สำหรับทำหน้าที่เป็น XMPP server โดยมีฐานข้อมูลเป็น MySQL ส่วนระบบของเครื่องแม่ข่ายที่ใช้นั้นจะเป็น UNIX/Linux

การออกแบบนั้นจะใช้เทคนิคการแยกส่วนประกอบต่างๆในระบบ เช่น ส่วนของการเฝ้าระวังข้อความ Trap การติดต่อผ่านโปโตคอล XMPP การเรียกคำสั่ง SNMP เป็นต้น โดยส่วนประกอบต่างๆเหล่านี้จะถูกเรียกใช้ผ่านตัวกลางที่ทำหน้าที่รันโปรแกรมทั้งหมด ทำให้การขยายหรือลดฟังก์ชั่นการทำงานทำได้ง่ายขึ้น

ข้อมูลต่างๆในระบบถูกจัดเก็บอยู่ในฐานข้อมูลแทบทั้งหมด ดังนั้นส่วนที่สำคัญอีกส่วนหนึ่งก็คือส่วนของ การเข้าถึงข้อมูล โดยได้ประยุกต์ใช้งานเทคนิค Data Access Object (DAO) ซึ่งจะทำหน้าที่เป็นอินเตอร์เฟสในการเชื่อมต่อกับฐานข้อมูล โดยที่การใช้งานนั้นไม่จำเป็นต้องรู้รายละเอียดของฐานข้อมูล ทำให้การเข้าถึงข้อมูลสามารถทำได้ง่ายและรวดเร็ว ไม่ต้องทำการสร้างคำสั่ง SQL ใหม่ทุกๆครั้งที่ทำงาน

3.3.1.1 ส่วนของเครือข่าย

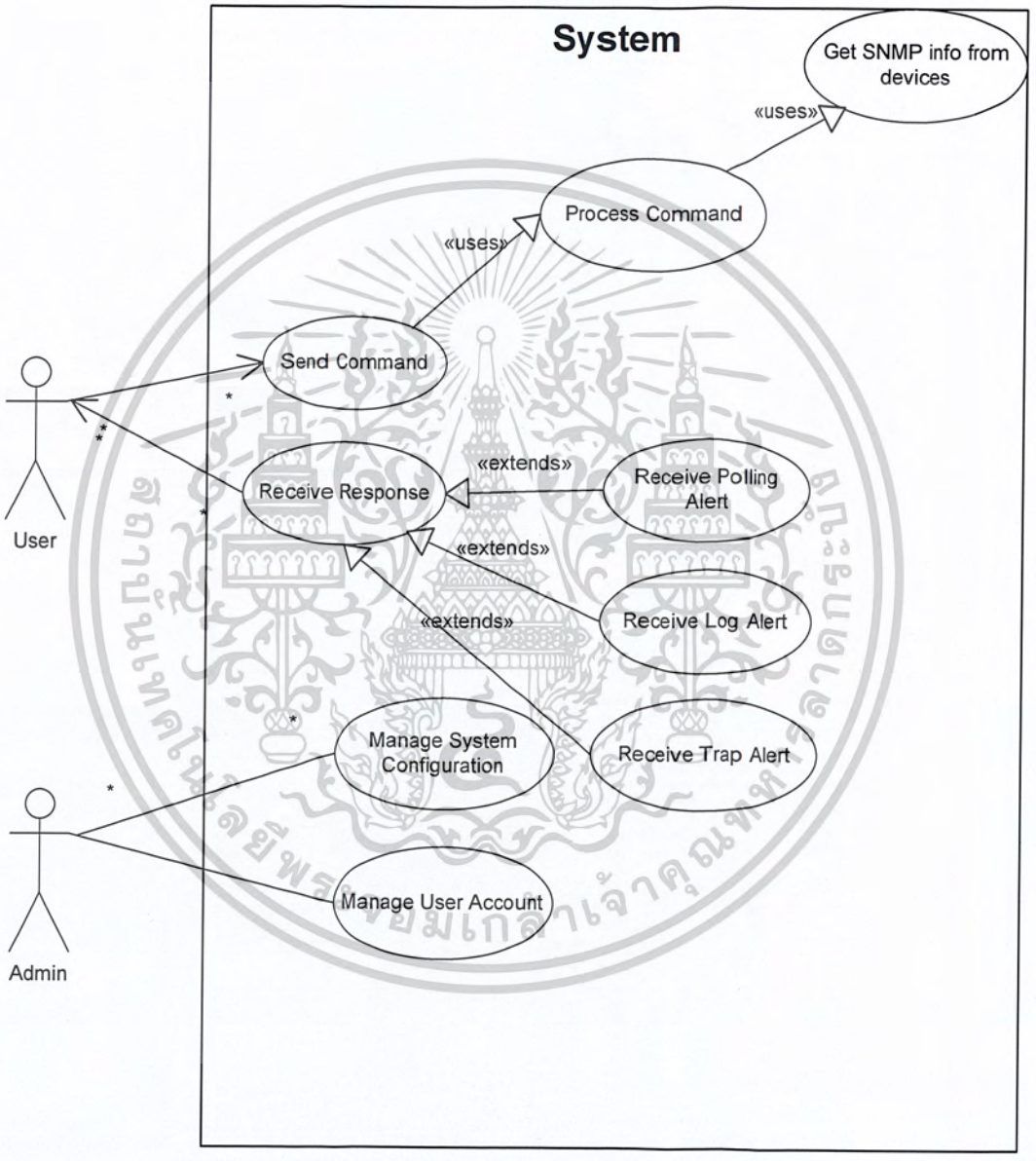
ระบบ NMS นี้สามารถนำไปใช้ได้กับเครือข่ายหลายลักษณะ โดยในตัวอย่างนี้จากรูปที่ 3.6 ระบบจะทำงานอยู่ในส่วนของ DMZ ที่มีการอนุญาตให้ผู้ใช้งานสามารถติดต่อเข้ามาได้จากทั้งด้านนอกและด้านในของเครือข่ายองค์กร ซึ่งช่วยให้การเข้าถึงระบบสามารถทำจากที่ได้จากทุกๆที่ ผ่านอุปกรณ์หลากหลายประเภท ไม่ว่าจะเป็น โทรศัพท์มือถือ หรือคอมพิวเตอร์พกพา ก็ตาม โดยอุปกรณ์ที่ระบบสามารถตรวจสอบเฝ้าระวังได้นั้นคือ อุปกรณ์เครือข่ายทุกประเภทที่สนับสนุนโปรโตคอล SNMP



รูปที่ 3.6 แสดงแผนภาพเครือข่ายของระบบที่พัฒนา

3.4 แผนภาพยูสเคส

ยูสเคสในระบบ มีทั้งหมด 5 ยูสเคส โดยจากรูปที่ 3.7 ระบบจะประกอบไปด้วยตัวละคร 2 ตัว คือส่วนของผู้ใช้งาน (User) และ ผู้ดูแลระบบ (Administrator) โดยตัวละครแต่ละตัวทำหน้าที่ติดต่อกับระบบดังต่อไปนี้



รูปที่ 3.7 แสดงแผนภาพยูสเคส

3.4.1 Use Case (UC1): ส่งคำสั่ง (Send Command)

Actor: ผู้ใช้

Preconditions: ผู้ใช้อยู่ในระบบ

Postconditions: ผู้ใช้ได้รับข้อความตอบกลับจากระบบ

Main scenario:

1. ผู้ใช้พิมพ์คำสั่งส่งมายังระบบ
2. ระบบตรวจสอบคำสั่ง
   2.1. คำสั่งถูกต้องหรือไม่
   2.2. ผู้ใช้งานมีสิทธิเรียกคำสั่งนี้หรือไม่
3. ระบบทำงานตามคำสั่งที่ได้รับ
4. ระบบส่งข้อความผลลัพธ์กลับไปยังผู้ใช้

Alternatives:

2.1 คำสั่งผิด ระบบส่งข้อความแจ้งกลับไปยังผู้ใช้

2.2 ระบบส่งข้อความแจ้งกลับไปยังผู้ใช้ ผู้ใช้ไม่มีสิทธิใช้งาน

3.4.2 Use Case (UC2): รับข้อความจากระบบ (Receive Response)

Actor: ผู้ใช้

Preconditions: ผู้ใช้อยู่ในระบบ

Postconditions: ผู้ใช้ได้รับข้อความจากระบบ

Main scenario:

1. ระบบได้รับข้อความแจ้งเตือนจากเครื่องแม่ข่ายและอุปกรณ์เครือข่าย ผ่านทาง Pipe SNMP Trap หรือการ Poll
2. ระบบตรวจสอบข้อมูลที่เข้ามาว่าตรงตามค่าที่ผู้ใช้ต้องการหรือไม่
3. ระบบตรวจสอบหาผู้ใช้ที่มีสิทธิรับข้อความจากระบบภายนอก
4. ระบบส่งข้อความออกไปยังผู้ใช้

Alternatives:

2. ข้อมูลที่เข้ามาไม่ตรงตามค่าที่ต้องการ ระบบจะไม่สนใจข้อมูลนั้น

3.4.3 Use Case (UC3): ประมวลผลคำสั่ง (Process Command)

Actor: ระบบ

Preconditions: มีการส่งคำสั่งมาจากผู้ใช้

Postconditions: ระบบทำงานแล้วคืนค่ากลับไปยังผู้ใช้

Main scenario:

1. ผู้ใช้พิมพ์คำสั่งส่งมายังระบบ
2. ระบบตรวจสอบคำสั่ง
   2.1. คำสั่งถูกต้องหรือไม่
   2.2. ผู้ใช้งานมีสิทธิเรียกคำสั่งนี้หรือไม่
3. ระบบทำงานตามคำสั่ง
4. ระบบส่งข้อความผลลัพธ์กลับไปยังผู้ใช้

Alternatives:

2.1 คำสั่งผิด ระบบส่งข้อความแจ้งกลับไปยังผู้ใช้

2.2 ผู้ใช้ไม่มีสิทธิใช้งาน ระบบส่งข้อความแจ้งกลับไปยังผู้ใช้

3.4.4 Use Case (UC4): อ่านค่าด้วย SNMP GET (GET SNMP Info from device)

Actor: ระบบ

Preconditions: มีการเรียกใช้งานจากส่วนประมวลผล

Postconditions: คืนค่ากลับไปยังส่วนประมวลผล

Main scenario:

1. ส่วนประมวลผลส่งการทำงานมายังระบบ
2. ระบบเชื่อมต่อไปยังอุปกรณ์ที่ผู้ใช้ได้เลือกไว้
3. ระบบส่งคำสั่ง SNMP GET ไปยังอุปกรณ์
4. ระบบคืนค่าผลลัพธ์

Alternatives

2. ระบบเชื่อมต่อเข้ากับอุปกรณ์ไม่ได้ ระบบส่งข้อความแจ้งกลับไปยังผู้ใช้

3.4.5 Use Case (UC5) ปรับแต่งค่าในระบบ (Manage Systems Configuration)

Actor: ผู้ใช้

Preconditions: ผู้ใช้อยู่ในระบบ

Postconditions: ระบบแก้ไขค่าตามที่ต้องการ

Main scenario:

5. ผู้ใช้เข้ามาในส่วนการปรับแต่งค่าผ่านทางเว็บอินเตอร์เฟส
6. ผู้ใช้ปรับแต่งค่าต่างๆ แล้วกดบันทึก
7. ระบบยืนยันการเปลี่ยนแปลง

3.4.6 Use Case (UC6): จัดการบัญชีผู้ใช้ในระบบ (Manage User Account)

Actor: ผู้ใช้

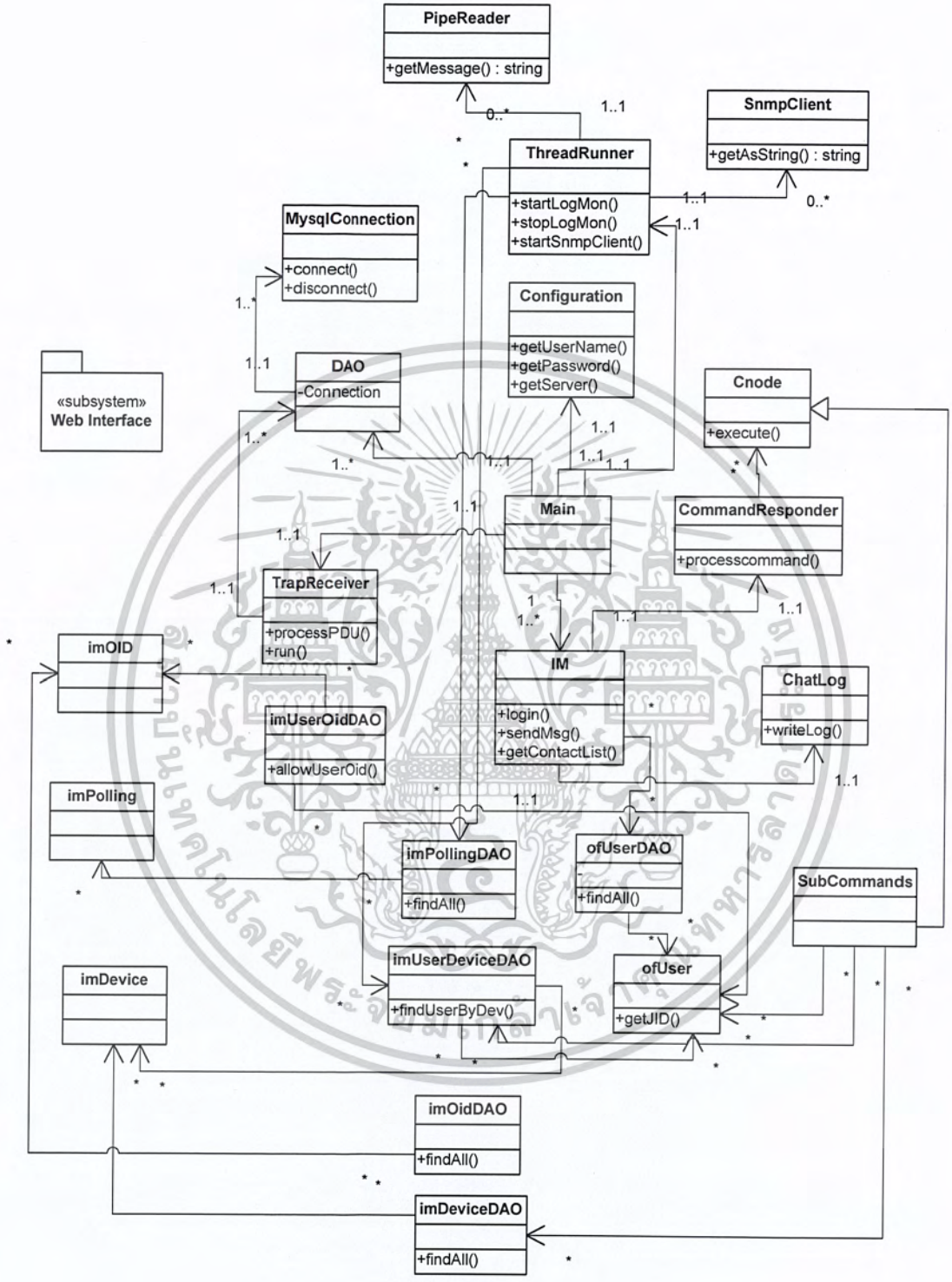
Preconditions: ผู้ใช้อยู่ในระบบ

Postconditions: ระบบแก้ไขค่าตามที่ต้องการ

Main scenario:

1. ผู้ใช้เข้ามาในส่วนจัดการผู้ใช้ผ่านเว็บอินเตอร์เฟส
2. ผู้ใช้ทำการเลือก เพิ่ม แก้ไข หรือ ลบ
3. ระบบยืนยันการเปลี่ยนแปลง

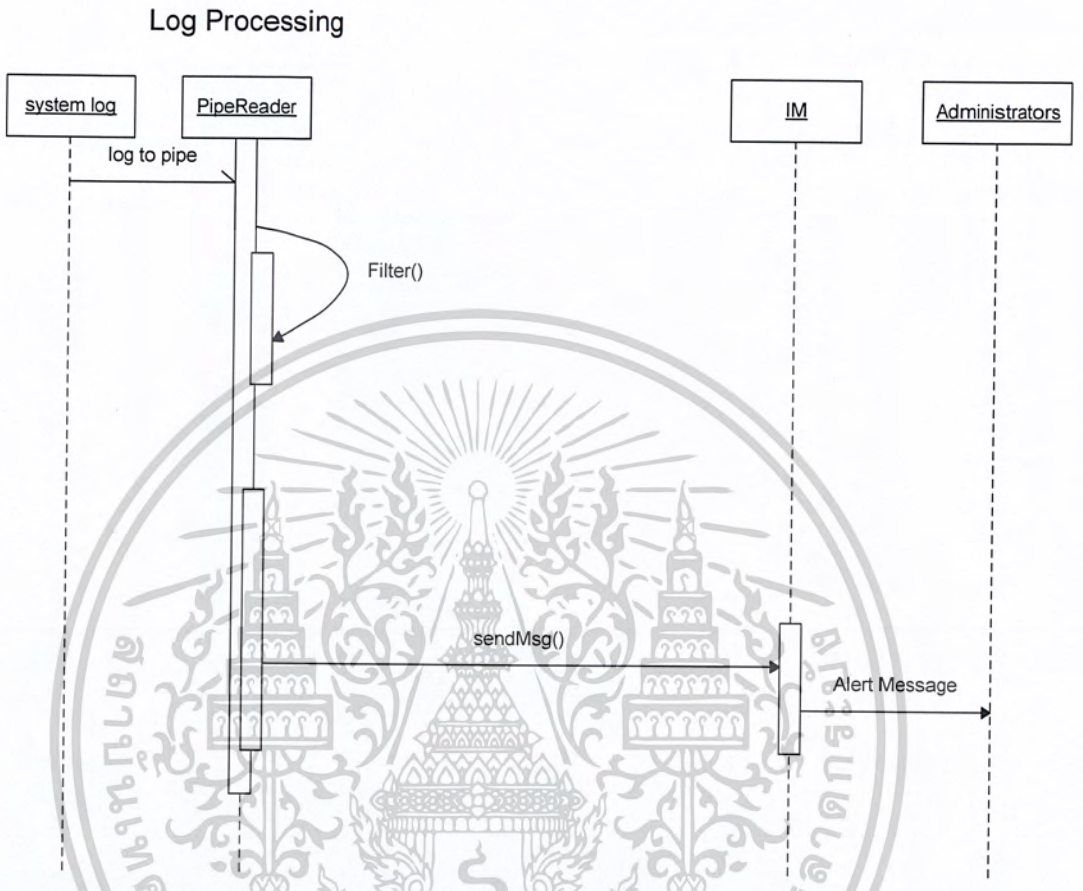
3.5 แผนภาพคลาส



รูปที่ 3.8 แสดงแผนภาพคลาสของระบบที่พัฒนา

3.6 แผนภาพซีเควนซ์

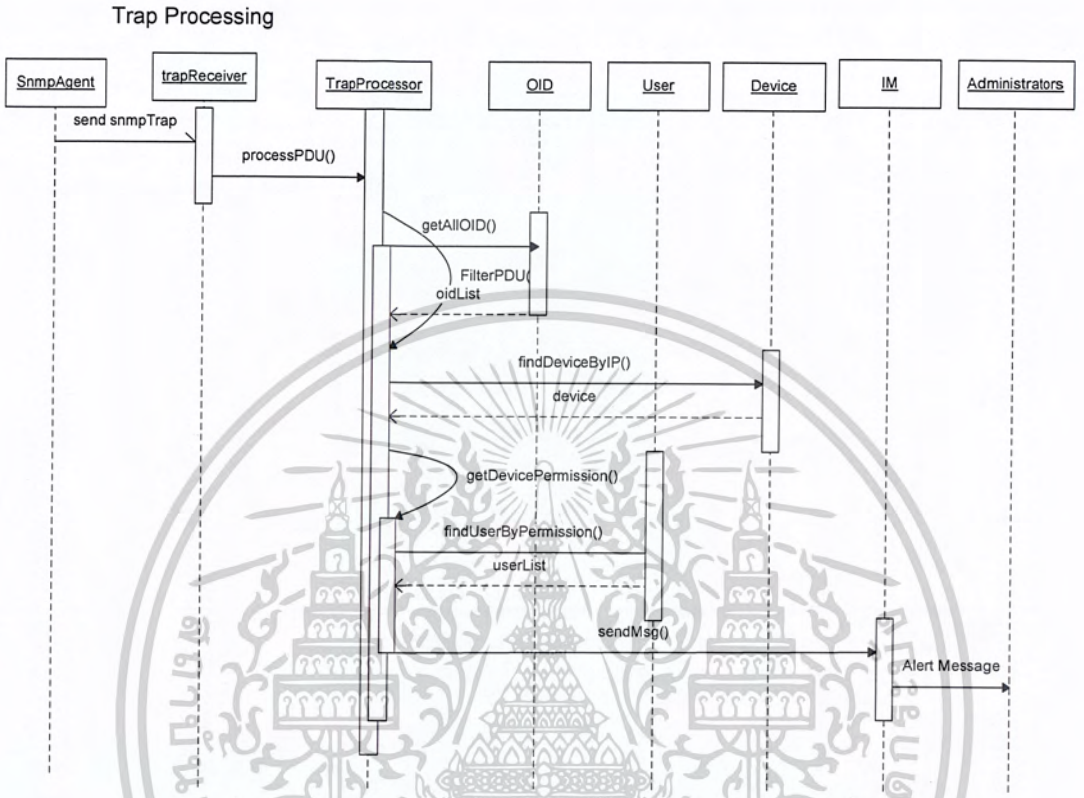
3.6.1 การประมวลผล Log (Log Processing)



รูปที่ 3.9 แสดงแผนภาพซีเควนซ์ Log Processing

จากรูปที่ 3.9 เป็นการส่งข้อความ System Log จากระบบ ไปยังผู้ดูแลระบบโดยเมื่อ ส่วนของการตรวจสอบ Named Pipe ตรวจพบ system log ที่เขียนเพิ่มขึ้นมาใหม่ ระบบจะทำการตรวจสอบ log ก่อนว่าซ้ำกับครั้งก่อนหรือไม่ หลังจากนั้นระบบจะส่งข้อความไปแจ้งเตือนยังผู้ใช้งานผ่านทางบริการข้อความด่วนทันใจ (Instant Messaging)

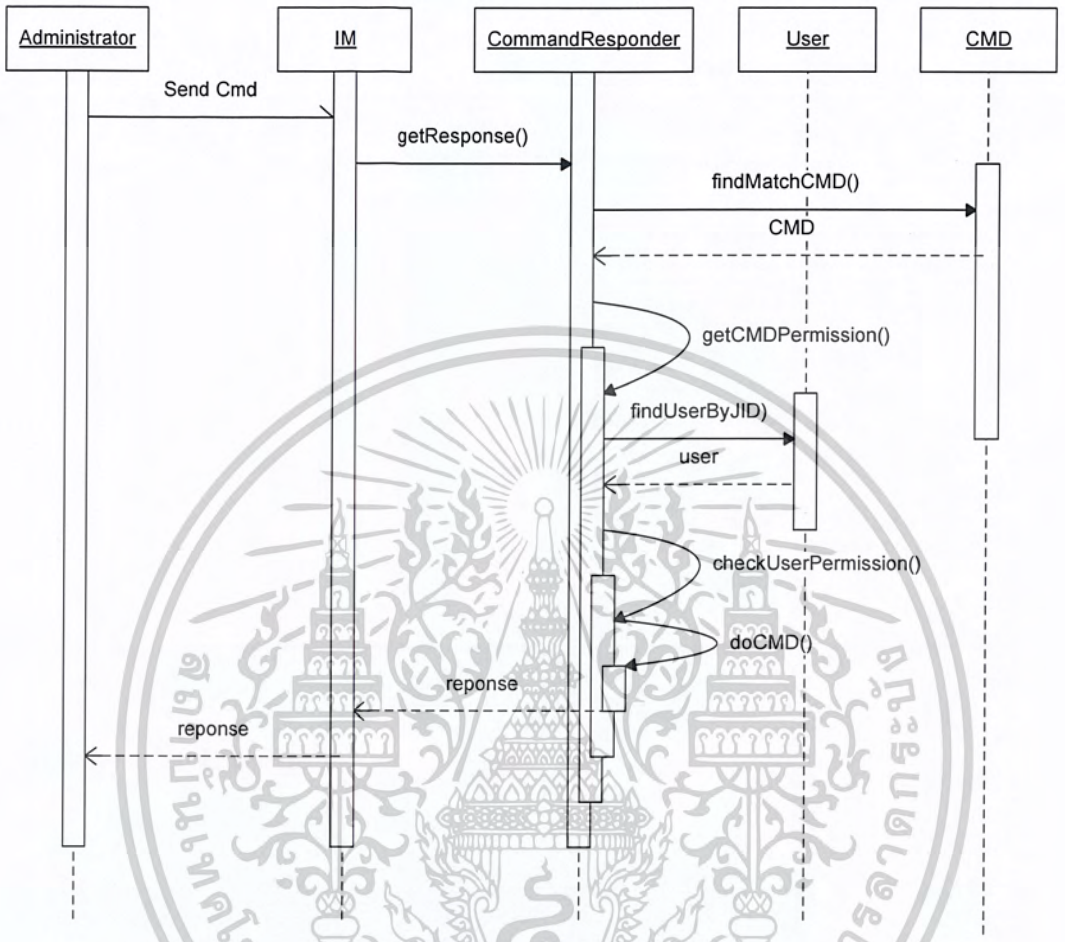
3.6.2 การประมวลผล Trap (Trap Processing)



รูปที่ 3.10 แสดงแผนภาพซีเควนซ์ Trap Processing

จากรูปที่ 3.10 เป็นการส่งข้อความ SNMP Trap จากระบบไปยังผู้ดูแลระบบโดยเมื่อ SNMP Agent ของอุปกรณ์เครือข่ายใดๆก็ตาม ทำการส่ง SNMP Trap มา ระบบที่ดูแลส่วนของการรับ SNMP Trap จะทำการตรวจสอบ OID ของ SNMP Trap ที่เข้ามาว่าตรงกับ OID ที่มีอยู่ในฐานข้อมูลหรือไม่ แล้วทำการค้นหาผู้ใช้งานจากสิทธิของอุปกรณ์ที่ทำการส่ง SNMP Trap มา หลังจากนั้นระบบจะส่งข้อความไปแจ้งเตือนยังผู้ใช้งานผ่านทางบริการข้อความด่วนทันใจ (Instant Messaging)

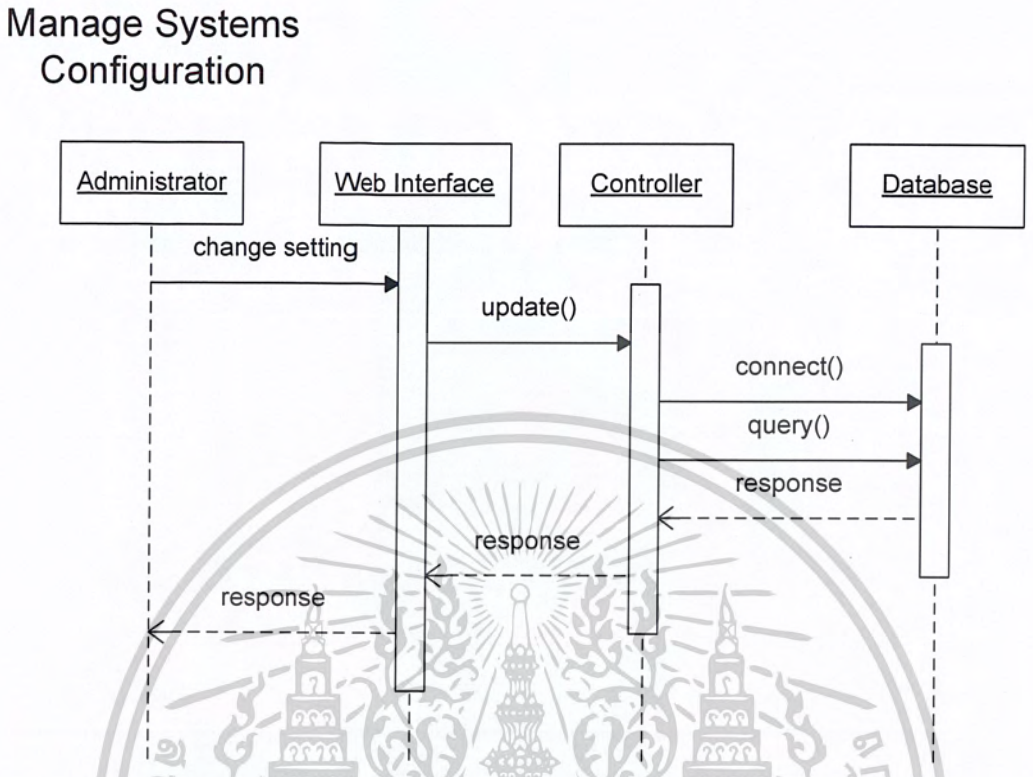
3.6.3 การประมวลผลคำสั่ง (Process Command)



รูปที่ 3.11 แสดงแผนภาพซีเควนซ์ View System Status

จากรูปที่ 3.11 เป็นการส่งคำสั่งจากผู้ดูแลระบบ มายังระบบ เพื่อเรียกดูสถานะของระบบที่ทำการตรวจสอบและเฝ้าระวัง โดยหลังจากที่ได้รับข้อความมาแล้ว ระบบจะทำการประมวลผลคำสั่งว่าถูกต้อง หรือไม่ถูกต้องอย่างไร ถ้าถูกต้อง ก็จะทำการหาผู้ใช้จาก Jabber ID แล้วทำการตรวจสอบสิทธิของผู้ใช้งาน ว่าสามารถเรียกคำสั่งดังกล่าวได้หรือไม่ ถ้าพบว่ามีสิทธิ์ถูกต้อง ระบบก็ทำคำสั่งนั้น แล้วระบบจะทำการส่งข้อความผลลัพธ์กลับไปทางบริการข้อความด่วนทันใจ (Instant Messaging)

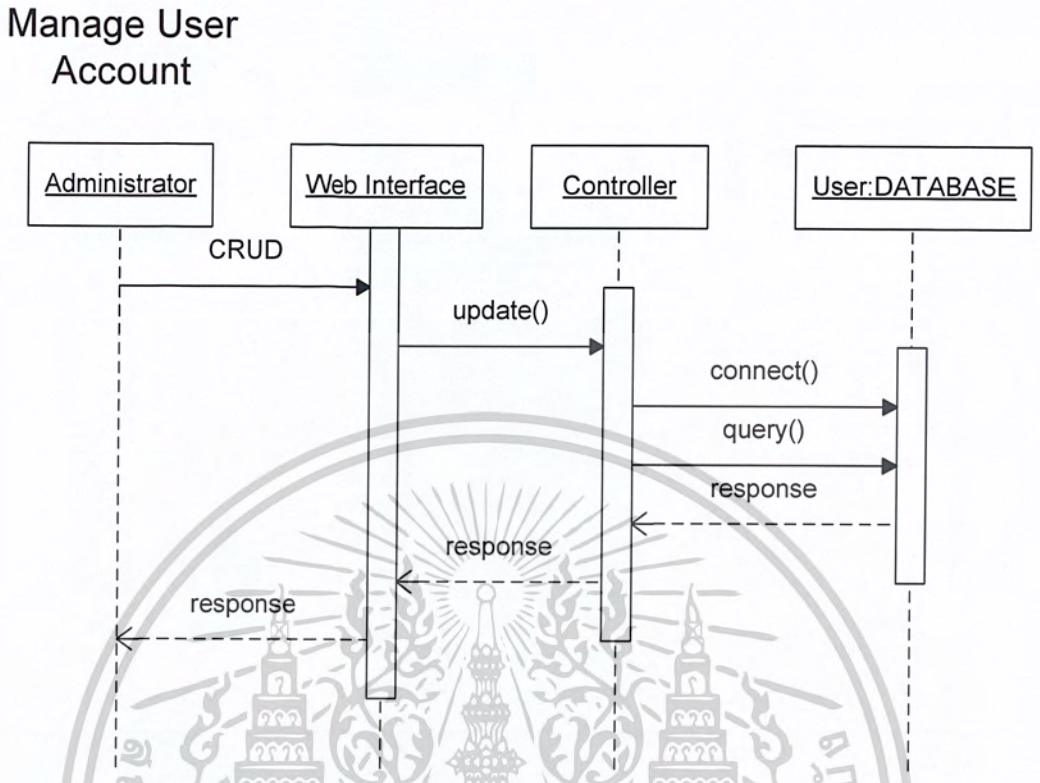
3.6.4 การปรับแต่งค่าในระบบ (Manage Systems Configuration)



รูปที่ 3.12 แสดงแผนภาพซีเควนซ์ Manage Systems Configuration

จากรูปที่ 3.12 ผู้ใช้จะเข้าไปปรับเปลี่ยนการตั้งค่าต่างๆ ผ่านทางเว็บอินเตอร์เฟส หลังจากนั้นข้อมูลจะถูกส่งไปยังส่วนควบคุมของระบบเพื่อทำการเชื่อมต่อและแก้ไขข้อมูล เมื่อระบบทำงานเสร็จแล้วจะมีข้อความตอบกลับว่าทำรายการสำเร็จหรือไม่

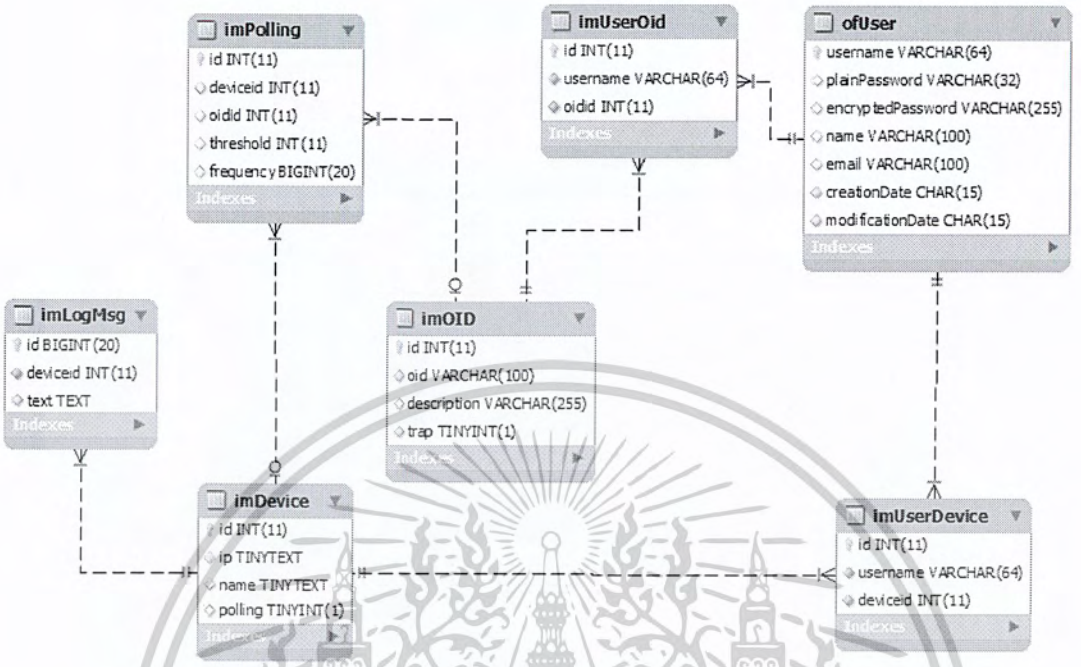
3.6.5 การจัดการบัญชีผู้ใช้งาน (Manage User Account)



รูปที่ 3.13 แสดงแผนภาพซีเควนซ์ Manage User Account

จากรูปที่ 3.13 เป็นการส่งคำสั่ง CRUD (Create, Update, Delete) ผ่านทางเว็บอินเตอร์เฟส โดยข้อมูลที่ได้รับจะถูกส่งไปยังส่วนควบคุมของระบบเพื่อทำการเชื่อมต่อและแก้ไขข้อมูล เมื่อระบบทำงานเสร็จจะมีข้อความตอบกลับว่าทำรายการสำเร็จหรือไม่

3.7 การออกแบบฐานข้อมูล



รูปที่ 3.14 แผนภาพแสดงโครงสร้างฐานข้อมูแบบ Extended Entity-Relationship

การออกแบบฐานข้อมูล จะแบ่งเป็นสองส่วน โดยส่วนแรกจะเป็นข้อมูลที่เกี่ยวข้องกับ โอไอดี อุปกรณ์ การโพล และการเก็บข้อความล้อก เช่น โอไอดี หมายเลขอะไร มีรายละเอียดอะไรบ้าง หรือ อุปกรณ์ มีหมายเลขไอพีอะไร มีชื่อว่าอะไร เป็นต้น

ส่วนที่สองคือข้อมูลสิทธิการใช้งานของผู้ใช้งาน โดยจะแบ่งแยกเป็นส่วนที่เกี่ยวข้องกับอุปกรณ์ และ ส่วนที่เกี่ยวข้องกับโอไอดี โดยในตารางของส่วนนี้จะนำข้อมูลจากตารางผู้ใช้งาน (ofUser) และ โอไอดี (imOID) หรือ อุปกรณ์ (imDevice) มาอ้างอิงเป็น FK

พจนานุกรมโครงสร้างข้อมูล

ตาราง imDevice

Field	Type	Null	Key	Default
Id	Int(11)	NO	Primary	Auto_increment
Ip	Tinytext	NO		
Name	Tinytext	YES		
Polling	Tinyint(1)	YES		

ตาราง imOID

Field	Type	Null	Key	Default
Id	Int(11)	NO	Primary	Auto_increment
Oid	Varchar(100)	NO		
Description	Varchar(255)	YES		
Trap	Tinyint(1)			

ตาราง imUserDevice

Field	Type	Null	Key	Default
Id	Int(11)	NO	Primary	Auto_increment
Username	Varchar(64)	NO	FK (ofUser.username)	
Deviceid	Int(11)	NO	FK (imDevice.id)	

ตาราง imUserOid

Field	Type	Null	Key	Default
Id	Int(11)	NO	Primary	Auto_increment
Username	Varchar(64)	NO	FK (ofUser.username)	
Oidid	Int(11)	NO	FK (imOID.id)	

ตาราง ofUser

Field	Type	Null	Key	Default
Username	Varchar(64)	NO	Primary	
plainPassword	Varchar(32)	YES		
encryptedPassword	Varchar(255)	YES		
Name	Varchar(100)	YES		
Email	Varchar(100)	YES		
creationDate	Char(15)	NO		
modificationDate	Char(15)	NO		

3.8 เครื่องมือที่ใช้ในการพัฒนา

3.8.1 ระบบปฏิบัติการ

3.8.1.1 Linux Ubuntu 10.04 LTS

3.8.2 โปรแกรมที่ใช้ในการพัฒนา

3.8.2.1 Netbeans 6.9.1

3.8.2.2 Syslog-NG 2.0.9

3.8.2.3 Net-SNMP 5.4.2.1

3.8.2.4 MySQL 5.1.41

3.8.2.5 SSHD

3.8.2.6 Openfire (XMPP Server)

3.8.3 API/library

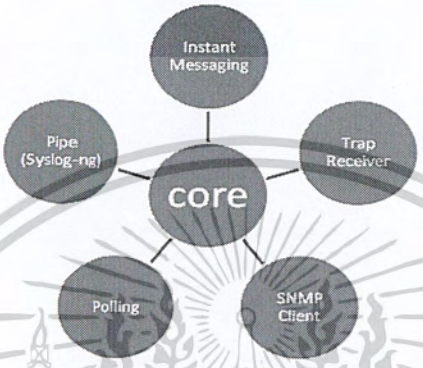
3.8.3.1 Smack 3.0

3.8.3.2 Snmp4j 1.11.1

3.8.3.3 SnakeYaml 1.7

บทที่ 4
การพัฒนาระบบ

4.1 โครงสร้างหลักของระบบ (Core Systems)



รูปที่ 4.1 แสดงโครงสร้างหลักของระบบ

การพัฒาโครงสร้างหลัก ใช้การเรียก โมดูลแต่ละโมดูล ซึ่งเป็นคลาสต่างๆ ใน ฟังก์ชั่นเมน โดยจะอ่านรายละเอียดที่จำเป็นสำหรับการเริ่มต้นทำงาน จากส่วนของไฟล์คอนฟิกที่อยู่ใน /etc/imguardian/config.yaml ส่วนข้อมูลอื่นๆจะถูกเก็บ ไว้ในฐานข้อมูลทั้งหมด

โดยข้อมูลที่เก็บอยู่ในไฟล์คอนฟิกจะเกี่ยวข้องกับการเชื่อมต่อฐานข้อมูลและการเชื่อมต่อเข้ากับเครื่องแม่ข่าย XMPP

4.1.1 ขั้นตอนการเริ่มต้นระบบ

4.1.1.1 ระบบอ่านค่าเริ่มต้นที่จำเป็นจากไฟล์ /etc/imguardian/config.yaml

4.1.1.2 ระบบทำการเชื่อมต่อกับฐานข้อมูล openfire ใน localhostเชื่อมต่อเข้าสู่เครื่องแม่ข่าย XMPP

4.1.1.3 เริ่มต้นการรับ Trap

4.1.1.4 เริ่มต้น SNMP Client ทั้งหมดในระบบ

4.1.1.5 ทำการอ่าน pipe เพื่อรับ syslog

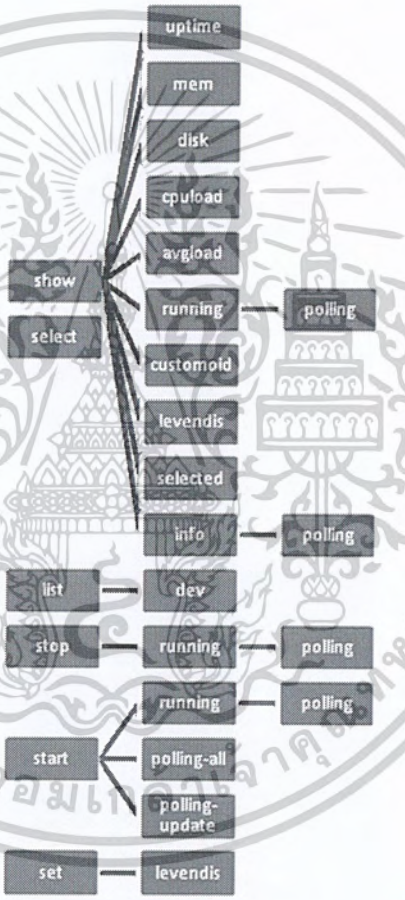
4.1.1.6 เริ่มต้นการ Polling

4.2 ระบบการส่งข้อความด่วน (Instant Messaging)

เป็นส่วนที่ทำหน้าที่ส่งและรับข้อความ รวมถึงสถานะการออนไลน์ต่างๆ จากเครื่องแม่ข่าย XMPP (openfire) โดยเมื่อผู้ใช้งานส่งข้อความมายัง nms ระบบจะทำการส่งข้อความไปประมวลผลที่คลาส CommandResponder

ในการพัฒนานั้นจะใช้ไลบารี่ Smack 3.0 ช่วยในการติดต่อผ่านโปรโตคอล XMPP

4.3 โครงสร้างของคำสั่ง



รูปที่ 4.2 แสดงโครงสร้างคำสั่งทั้งหมดในระบบ

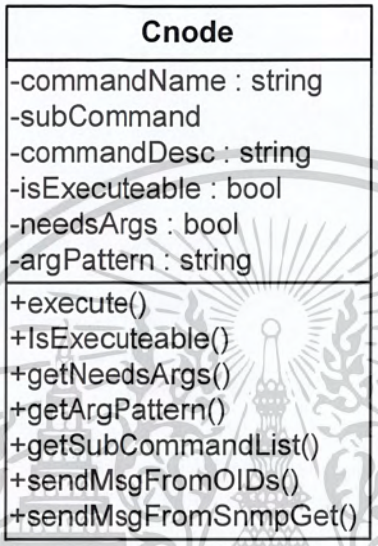
โครงสร้างของคำสั่งที่ใช้ติดต่อระหว่างเครื่อง NMS กับผู้ใช้งานมีลักษณะเป็นคำสั่งแบบลำดับชั้นแบบต้นไม้ ดังนั้นการออกแบบโปรแกรมเพื่อให้รองรับคำสั่งในลักษณะนี้ จะทำการสร้างคลาสสำหรับแต่ละคำสั่ง โดยในแต่ละชุดคำสั่ง (ต้นไม้) จะมีโหนดพ่อแม่ (parent node) หนึ่งโหนด

หรือหนึ่งคลาสเท่านั้น แต่จะมี โหนดลูก (child node) ซึ่งจะเป็นคลาสย่อย (nested class) ของของโหนดพ่อแม่นั้นกี่โหนดก็ได้

คำสั่ง	ผลลัพธ์	อาร์กิวเมนต์
Show uptime	ข้อมูล uptime ของอุปกรณ์	
Show mem	ข้อมูลหน่วยความจำชั่วคราวของอุปกรณ์	
Show disk	ข้อมูลหน่วยความจำถาวรของอุปกรณ์	
Show cpuload	ข้อมูลโหลดของอุปกรณ์	
Show avgload	ข้อมูลโหลดของอุปกรณ์โดยเฉลี่ย	
Show running polling	ข้อมูลของ poll ที่ทำงานอยู่	
Show customoid	SNMP GET ของค่าอาร์กิวเมนต์	หมายเลข OID
Show levendis	แสดงค่า Levenshtein Distance ที่ใช้อยู่	
Show info polling	แสดงข้อมูล poll ทั้งหมด	
Show selected	แสดงรายการอุปกรณ์ที่ได้เลือกไว้	
select	เลือกอุปกรณ์	Ip address หรือ hostname ของอุปกรณ์
Start running polling	เริ่มการทำงานของ poll	หมายเลข id ของ poll
Start polling-all	เริ่มการทำงานของ poll ทั้งหมด	
Start polling-update	อัพเดทค่า threshold และ frequency จากฐานข้อมูล	
Stop running polling	หยุดการทำงานของ poll	หมายเลข id ของ poll
Set levendis	ตั้งค่า Levenshtein Distance	ตัวเลขมากกว่า 0
List dev	แสดงอุปกรณ์ทั้งหมดที่ใช้งานอยู่	

4.3.1 การออกแบบคลาสคำสั่ง (คลาสโหนด) (Cnode)

คลาสโหนดเป็นคลาสนามธรรม (abstract class) ที่จะทำหน้าที่เก็บข้อมูลและการทำงานของคำสั่งเช่น ชื่อคำสั่ง ชุดคำสั่งลูก การเรียกใช้งาน การส่งข้อความจากการเรียน SNMP GET เป็นต้น โดยทุกๆคำสั่งในโปรแกรมจะต้องทำการ extend ต่อจากคลาสนี้เพื่อเรียกใช้งานคำสั่งที่จำเป็น โดยมีโครงสร้างดังต่อไปนี้



รูปที่ 4.3 แสดงโครงสร้างของคลาส Cnode

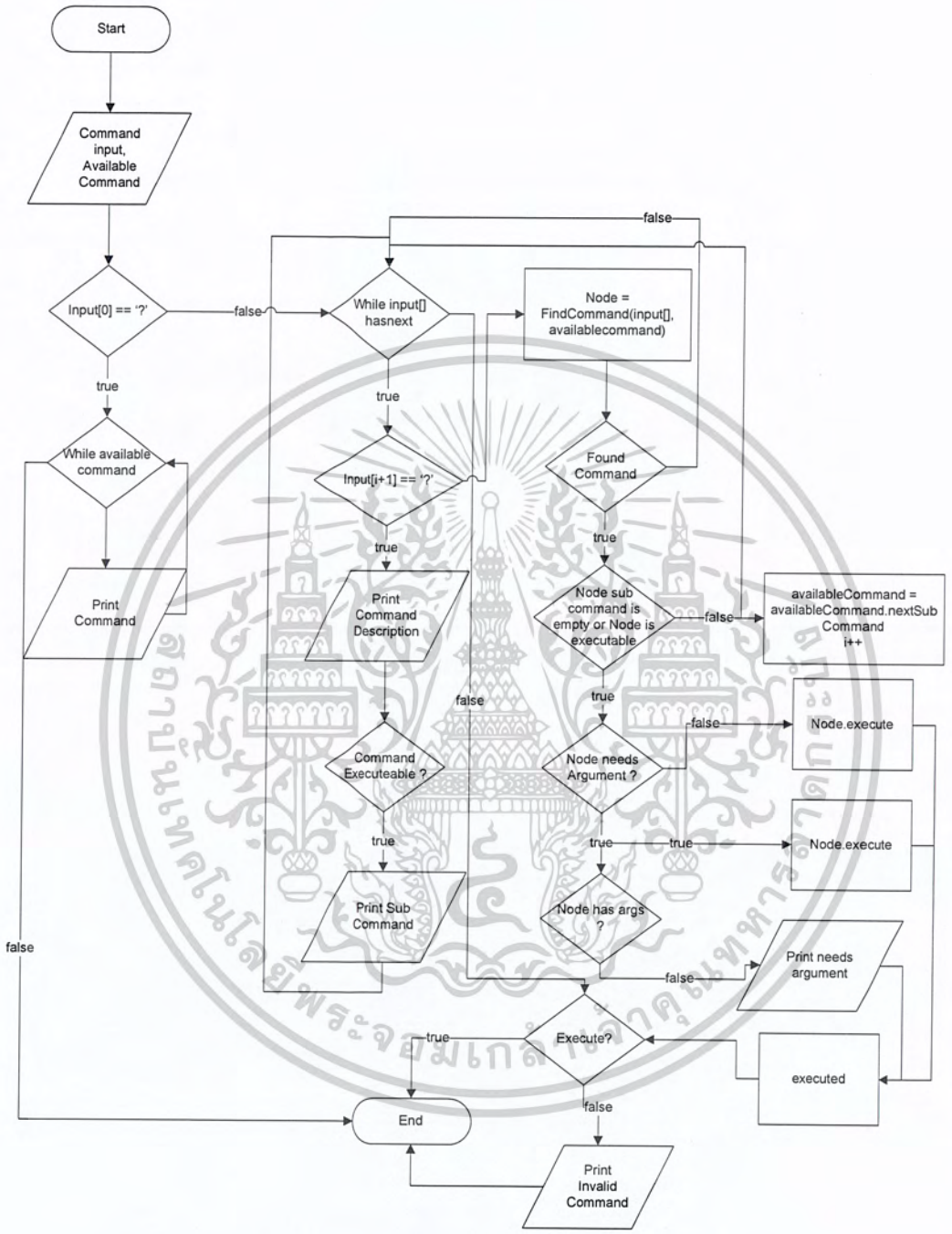
ลักษณะการเรียกใช้งานจะเป็นในลักษณะนี้ โดยตัวอย่างข้างล่างจะเป็นส่วนของคำสั่ง “set” ที่มีคำสั่ง levendis เป็นคำสั่งลูก จะสังเกตเห็นได้ว่า ทั้งคำสั่ง set และ levendis ต่างก็ extend คลาส Cnode ทำให้คำสั่งทั้งสองนั้นมีคุณสมบัติเหมือนกัน แต่ต่างกันที่ว่าคำสั่ง levendis ถูกนำมาใส่ในตัวแปร subcommand ของคำสั่ง set โดยเมื่อผู้ใช้งานป้อนคำสั่ง “set levendis” คำสั่งนี้จะถูกส่งไปยัง CommandResponder เพื่อหาคลาสคำสั่ง ในที่นี้คือ levendis หลักจากนั้นจึงสั่ง execute

```
public class set extends Cnode{
        public set(){
        setArgPattern("positive number");
         setCommand("set");
    setIsExecuteable(false);
        getSubCommand().add(new levendis());
    }
    @Override
    void execute(ofUser user) {
```

```
    }
    @Override
    void execute(String args, ofUser user) {
    }
    class levendis extends Cnode{
          public levendis(){
           setCommandDesc("LevenshteinDistance");
           setCommand("levendis");
    setNeedsArgs(true);
    setIsExecuteable(true);
    }
      @Override
      void execute(ofUser user) { }
      @Override
      void execute(String args, ofUser user) {}
    }
  } }
```

การออกแบบโครงสร้างในลักษณะนี้ทำให้สามารถจัดระเบียบโครงสร้างของคำสั่งได้ง่าย ผู้พัฒนาสามารถพัฒนาระบบได้ง่ายขึ้น แต่ข้อเสียคือความยืดหยุ่นของระบบต่ำ เนื่องจากคลาสอยู่ภายใต้กันในลักษณะลำดับชั้น ทำให้การนำคลาสลูกมาใช้ใหม่นั้นทำได้ยากกว่าปกติ

4.4 ระบบประมวลผลคำสั่ง (Command Responder)



รูปที่ 4.4 Flow Chart แสดงการทำงานของ Command Responder

คลาส Command Responder ทำหน้าที่ประมวลผลคำสั่งจากข้อความที่ผู้ใช้ป้อนเข้ามา โดยเมื่อได้รับข้อความจากผู้ใช้แล้ว ระบบจะทำการแบ่งคำสั่งออกจากกันด้วยการแบ่งช่องว่าง จึงทำให้ได้อะเรย์ผลลัพธ์ ซึ่งระบบจะทำการค้นหาชุดคำสั่งที่ตรงกับอะเรย์ชุดนี้

ชุดคำสั่งทั้งหมดจะถูกเก็บไว้ตัวแปรแบบเวกเตอร์ โดยระบบจะทำการอ่านชื่อคำสั่ง ในคลาสคำสั่ง และทำการเปรียบเทียบกับข้อความในอะเรย์ เมื่อระบบตรวจสอบคำสั่งทั้งหมดว่าถูกต้องแล้ว ระบบจะทำการอ่านชุดคำสั่ง execute ในคลาสคำสั่งนั้น เพื่อทำการประมวลผลต่อไป ในกรณีที่มีการป้อนอากิวเม้นต่อท้ายเข้ามา ระบบจะทำการอ่านว่าคำสั่งนั้น รองรับค่าอาร์กิวเมนต์หรือไม่

4.5 ระบบแจ้งเตือน

4.5.1 Syslog-NG

ข้อมูลที่รับนั้นผ่านเข้ามาทาง pipe ที่ผู้ใช้งานได้สร้างไว้ในเครื่อง NMS (ต้องกำหนดผ่านไฟล์คอนฟิก) เมื่อระบบรับข้อมูลเข้ามาแล้วจะทำการคัดกรองข้อมูลที่อาจจะซ้ำกันในข้อความ แต่แตกต่างกันในเรื่องของเวลา ในเบื้องต้นด้วยการใช้ อัลกอริทึม Levenshtein Distance โดยถ้ามีความแตกต่างน้อยกว่า 5 (สามารถเปลี่ยนแปลงได้ด้วยคำสั่ง “set levendis “positive number”) ให้ถือว่าเป็นข้อมูลเดิม หลังจากนั้นระบบจะอานค่า ip address จากข้อความ log เพื่อนำไปหาผู้ใช้งาน ที่มีสิทธิในการรับข้อความ log ต่อไป

4.5.2 SNMP Trap

ระบบจะทำการรับข้อมูลแจ้งเตือนด้วยพอร์ท UDP 162 โดยเมื่อระบบได้รับข้อความ Trap มาจากอุปกรณ์ ระบบจะนำค่า OID ในข้อความ Trap มาตรวจสอบว่า ตรงตามกับที่ผู้ใช้งานได้เลือกไว้หรือไม่ ถ้าตรงระบบจะส่งข้อความ trap ไปยังผู้ใช้งานที่มีสิทธิในการรับ Trap จากอุปกรณ์ชิ้นนี้ โดยก่อนที่ส่งนั้นระบบจะใช้อัลกอริทึม Levenshtein Distance ตรวจสอบเช่นเดียวกับ Syslog-NG แต่แตกต่างกันที่ ในส่วนของ SNMP จะมีการเปรียบเทียบกับรายการข้อความที่ส่งไปไม่เกิน 1 นาทีที่ผ่านมา

4.6 ระบบเฝ้าระวังด้วยการโพล (Polling)

ระบบอ่านข้อมูลที่จำเป็นต่อการ Poll ทั้งหมดจากฐานข้อมูล ได้แก่ หมายเลขไอพี ค่า OID ความถี่ในการโพล(วินาที) ค่าสูงสุดที่รับได้ (threshold) โดยระบบจะทำการสร้างเทรดใหม่สำหรับแต่ละโพล โดยเมื่อได้รับข้อมูลที่เข้ามาจากการใช้ SNMP GET(OID) แล้วระบบจะทำการตรวจสอบว่าเกินค่าสูงสุดที่กำหนดไว้หรือไม่ ถ้าเกินระบบจะทำการส่งข้อความแจ้งเตือนไปยังผู้ใช้ และหากถ้าค่าสูงสุดยังเกินกว่าที่กำหนดไว้อีก (ในการโพลรอบถัดมา) ระบบจะทำการตรวจสอบว่าในช่วงเวลาที่ผ่านมา ได้มีการส่งข้อความไปรึยัง ด้วยสูตร (15(s) x times) โดยที่ times คือจำนวนครั้งของการส่งข้อความ ถ้า times มีค่าเป็น 1 ระบบจะต้องทำการรอไปอีก 15 วิถึงจะส่งข้อความเดิม

ไปยังผู้ใช้งาน นอกจากนี้ผู้ใช้ยังสามารถปิดการทำงานของ Polling ชั่วคราว หรือทำการอัพเดทข้อมูล โพล หรือ ค่าสูงสุดผ่านทางคำสั่ง ได้

4.7 ความปลอดภัยและสิทธิของผู้ใช้

ระบบทำงานภายใต้โมเดลความปลอดภัยแบบ Mandatory Access Control (MAC) โดยจะมีเพียงแอดมินคนเดียวเท่านั้นที่สามารถแก้ไขสิทธิการเข้าใช้งานของผู้ใช้ในอุปกรณ์หรือ OIDต่างๆ ได้ โดยที่ผู้ใช้งานทั่วไปไม่สามารถเข้ามาแก้ไขได้ด้วยตัวเอง ลักษณะการทำงานนั้นจะตรวจสอบว่าผู้ใช้มีสิทธิในการเข้าถึง วัตถุหรือไม่ โดยวัตถุก็คือ OID และอุปกรณ์

ในการส่งข้อความออกไปในแต่ละครั้ง ระบบจะอาศัยตัวแปรอย่างต่ำ 1 ตัวคือ อุปกรณ์ที่เกี่ยวข้อง โดยระบบจะทำการหาผู้ใช้ที่มีสิทธิใช้งานอุปกรณ์นี้ทั้งหมด และหากว่าการส่งข้อความนั้นมีโอไอดีเข้ามาเกี่ยวข้อง ระบบจะทำการตรวจสอบว่าผู้ใช้แต่ละคนที่ได้ผลมานั้น มีสิทธิในการอ่านค่าโอไดดีเหล่านั้นหรือไม่ หลังจากนั้นจึงส่งข้อความไปยังผู้ใช้

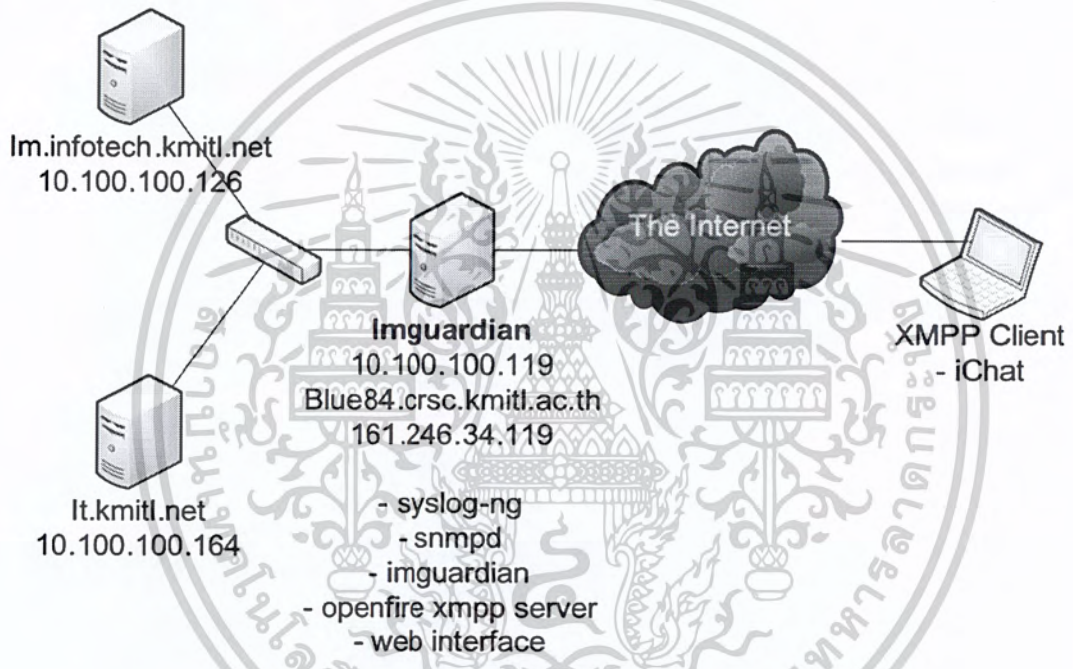
4.8 ระบบจัดการผ่านเว็บ (Web Interface)

เป็นส่วนที่ผู้ดูแลระบบคนที่มีสิทธิมากที่สุด สามารถเข้ามาดูและจัดการ แก้ไขปรับปรุงข้อมูลต่างๆของระบบได้ โดยในส่วนของเว็บไซท์นั้นจัดทำด้วย Java Enterprise Edition ซึ่งต้องอาศัย Application Server ในการรันระบบ เช่น GlassFish Enterprise Server

บทที่ 5
การทดสอบการทำงานของระบบ

ในบทนี้จะแสดงให้เห็นถึงสถาปัตยกรรม การตั้งค่าต่างๆ สำหรับการทดสอบ และการทดสอบการทำงานต่างๆ ของระบบตามที่ได้ออกแบบและพัฒนาขึ้นมา โดยจะแสดงในรูปของการทดสอบใช้งานระบบในรูปแบบต่างๆ ดังต่อไปนี้

5.1 สถาปัตยกรรมของระบบ



รูปที่ 5.1 แสดงสถาปัตยกรรมของระบบที่ทำการทดสอบ

ส่วนประกอบของระบบที่ใช้ในการทดสอบสามารถอธิบายได้ดังนี้

Imguardian เป็นเครื่องแม่ข่ายหลักของระบบมีหน้าดังนี้

1. รับการติดต่อจากผู้ใช้งานผ่าน Protocol XMPP ในการเรียกดูสถานะของเครื่องแม่ข่ายที่ทำการเฝ้าระวัง รวมทั้งส่งการแจ้งเตือนไปยังผู้ใช้งานเมื่อมีเหตุการณ์ที่ผิดปกติ
2. เป็น centralized log server สำหรับรับและจัดเก็บ syslog จากเครื่องแม่ข่ายในระบบเครือข่าย
3. ทำหน้าที่เป็นเครื่องแม่ข่ายสำหรับให้บริการ Instant Messaging
4. ทำหน้าที่เป็น Application Server สำหรับให้บริการ web interface สำหรับตั้งค่าการทำงานของระบบ

Im.infotech.kmitl.net และ it.kmitl.net

เป็นเครื่องแม่ข่ายที่ถูกเฝ้าระวังโดยระบบที่พัฒนาขึ้น โดยติดตั้ง SNMP Agent และตั้งค่าให้ส่ง syslog ไปยังเครื่อง imguardian เพื่อจัดเก็บและนำไปประมวลผล โดยโปรแกรมที่พัฒนาขึ้นโดยเครื่อง im.infotech.kmitl.net ให้บริการ http และ ssh

Xmpp client เป็นผู้ใช้งานที่เรียกใช้บริการของระบบผ่านทางระบบ Internet โดยใช้โปรแกรม xmpp client โดยในการทดสอบใช้โปรแกรม ichat บนระบบปฏิบัติการ Mac OS X

5.2 โปรแกรมที่ใช้การทดสอบ

5.2.1 Imguardian

1. Ubuntu 10.04.1 LTS X86_64 Server Edition
2. Imguardian version 2.6
3. NET-SNMP version: 5.4.2.1
4. Sun GlassFish Enterprise Server (GlassFish server) v3
5. Syslog-ng 2.0.9
6. openfire xmpp server 3.6.4
7. MySQL Server Version 5.1.41-3ubuntu12.8

5.2.2 it.kmitl.net

1. CentOS release 5.5 (Final) X86_64
2. NET-SNMP version: 5.3.2.2
3. syslog-ng 2.1.4

5.2.3 im.infotech.kmitl.net

1. Ubuntu 10.04 LTS X86_64 Server Edition
2. NET-SNMP version: 5.4.2.1

5.2.4 client

1. Apple iChat version 5.0.3 (745)

5.3 การตั้งค่าต่างๆ สำหรับการทดสอบ

5.3.1 การตั้งค่าการทำงานของโปรแกรมต่างๆ บนเครื่อง Imguardian

การตั้งค่าการทำงานของโปรแกรมต่างๆ บนเครื่อง Imguardian สำหรับการทดสอบได้มีการตั้งค่าดังนี้

5.3.1.1 การตั้งค่าการทำงานของโปรแกรม imguardian

การตั้งค่าการทำงานของโปรแกรม imguardian สามารถตั้งค่าได้ที่ file /etc/imguardian/conf.yaml ดังมีรายละเอียดดังนี้

OpenfireIP : localhost

OpenfireUser : imguardian

OpenfirePassword : 1234

OpenfirePort : 5222

MysqlIP : localhost

MysqlUser : root

MysqlPassword : secret

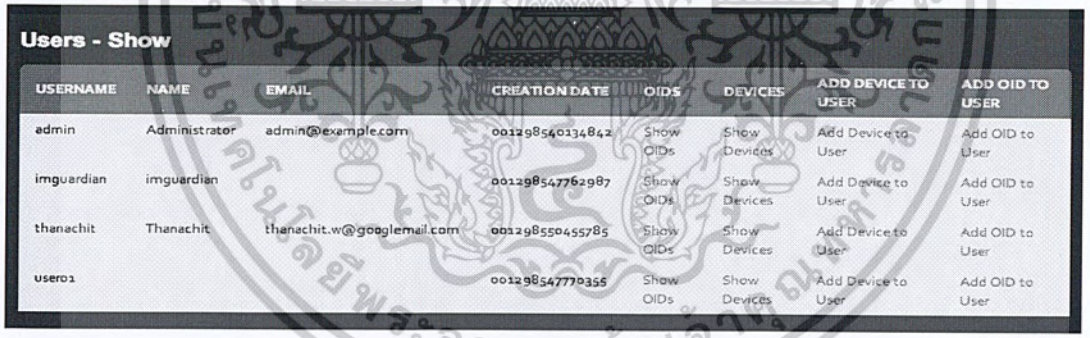
MysqlPort : 3306

MysqlDB : openfire

PipePath : /var/log/pipe

ข้อมูลและการตั้งค่าการทำงานของโปรแกรมในฐานข้อมูล

ข้อมูลรายการ User

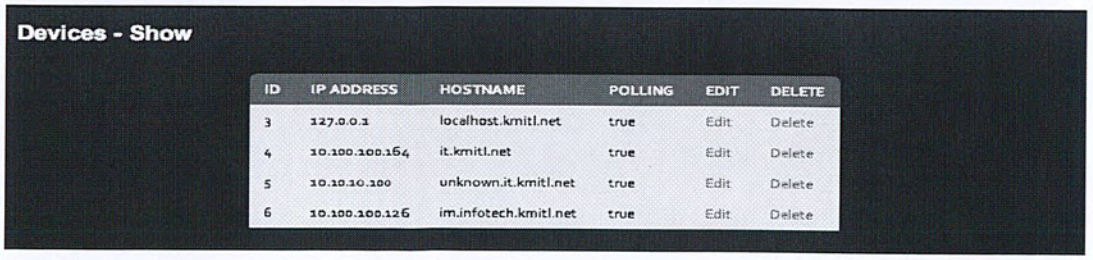


Users - Show

USERNAME	NAME	EMAIL	CREATION DATE	OIDS	DEVICES	ADD DEVICE TO USER	ADD OID TO USER
admin	Administrator	admin@example.com	001298540134842	Show OIDs	Show Devices	Add Device to User	Add OID to User
imguardian	imguardian		001298547762987	Show OIDs	Show Devices	Add Device to User	Add OID to User
thanachit	Thanachit	thanachit.w@googlemail.com	001298550455785	Show OIDs	Show Devices	Add Device to User	Add OID to User
user01			001298547770355	Show OIDs	Show Devices	Add Device to User	Add OID to User

รูปที่ 5.2 แสดงข้อมูลรายการ User

ข้อมูลรายการ Devices

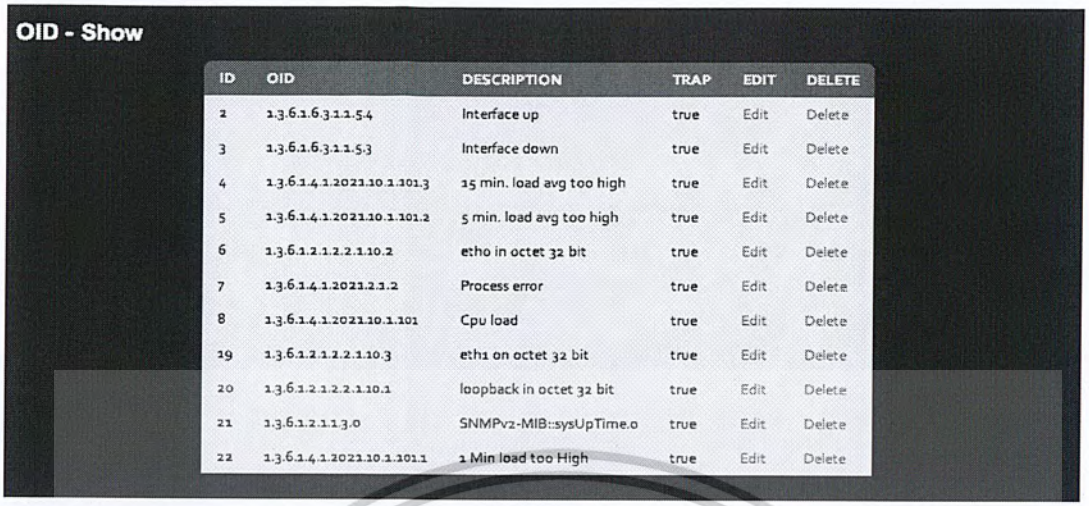


Devices - Show

ID	IP ADDRESS	HOSTNAME	POLLING	EDIT	DELETE
3	127.0.0.1	localhost.kmitl.net	true	Edit	Delete
4	10.100.100.164	it.kmitl.net	true	Edit	Delete
5	10.10.10.100	unknown.it.kmitl.net	true	Edit	Delete
6	10.100.100.126	im.infotech.kmitl.net	true	Edit	Delete

รูปที่ 5.3 แสดงข้อมูลรายการ Devices

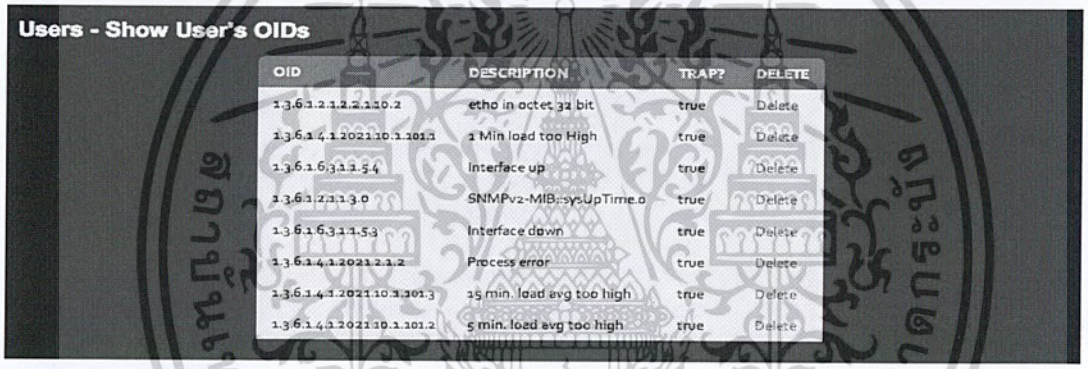
ข้อมูลรายการ OID



OID - Show

ID	OID	DESCRIPTION	TRAP	EDIT	DELETE
2	1.3.6.1.6.3.1.1.5.4	Interface up	true	Edit	Delete
3	1.3.6.1.6.3.1.1.5.3	Interface down	true	Edit	Delete
4	1.3.6.1.4.1.2021.10.1.101.3	15 min. load avg too high	true	Edit	Delete
5	1.3.6.1.4.1.2021.10.1.101.2	5 min. load avg too high	true	Edit	Delete
6	1.3.6.1.2.1.2.2.1.10.2	etho in octet 32 bit	true	Edit	Delete
7	1.3.6.1.4.1.2021.2.1.2	Process error	true	Edit	Delete
8	1.3.6.1.4.1.2021.10.1.101	Cpu load	true	Edit	Delete
19	1.3.6.1.2.1.2.2.1.10.3	eth1 on octet 32 bit	true	Edit	Delete
20	1.3.6.1.2.1.2.2.1.10.1	loopback in octet 32 bit	true	Edit	Delete
21	1.3.6.1.2.1.1.3.0	SNMPv2-MIB::sysUpTime.0	true	Edit	Delete
22	1.3.6.1.4.1.2021.10.1.101.1	1 Min load too High	true	Edit	Delete

รูปที่ 5.4 แสดงข้อมูลรายการ OID

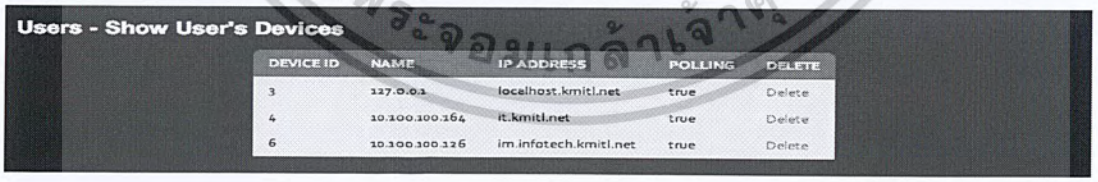


Users - Show User's OIDs

OID	DESCRIPTION	TRAP?	DELETE
1.3.6.1.2.1.2.2.1.10.2	etho in octet 32 bit	true	Delete
1.3.6.1.4.1.2021.10.1.101.1	1 Min load too High	true	Delete
1.3.6.1.6.3.1.1.5.4	Interface up	true	Delete
1.3.6.1.2.1.1.3.0	SNMPv2-MIB::sysUpTime.0	true	Delete
1.3.6.1.6.3.1.1.5.3	Interface down	true	Delete
1.3.6.1.4.1.2021.2.1.2	Process error	true	Delete
1.3.6.1.4.1.2021.10.1.101.3	15 min. load avg too high	true	Delete
1.3.6.1.4.1.2021.10.1.101.2	5 min. load avg too high	true	Delete

รูปที่ 5.5 แสดงข้อมูล User's OIDs ของ username: thanachit

ข้อมูล User's Devices ของ username: thanachit



Users - Show User's Devices

DEVICE ID	NAME	IP ADDRESS	POLLING	DELETE
3	127.0.0.1	localhost.kmitl.net	true	Delete
4	10.100.100.164	it.kmitl.net	true	Delete
6	10.100.100.126	im.infotech.kmitl.net	true	Delete

รูปที่ 5.6 แสดงข้อมูล User's Devices ของ username: thanachit

5.3.1.2 รายละเอียดการตั้งค่าการทำงานของ syslog-ng

การตั้งค่าการทำงานของ syslog-ng จะต้องมีการกำหนดค่าที่ไฟล์ /etc/syslog-ng/syslog-ng.conf

ในส่วนแรกเป็นส่วนของการกำหนด option ซึ่งในการทดลองนี้ได้ กำหนด option สำหรับ syslog-ng ดังนี้

```
options {
        time_reopen(10);
        time_reap(360);
        log_fifo_size(2048);
        create_dirs(yes);
        group(adm);
        perm(0640);
        dir_perm(0755);
        use_dns(no);
        stats(0);
        bad_hostname("^gconfd$");
        chain_hostnames(yes);
        keep_hostname (yes);
};
```

ส่วนที่ 2 เป็นการกำหนด source คือ แหล่งที่มาของ log ซึ่งในที่นี้คือ s_all โดยระบุว่า log จะมาจากทุกที่ที่ log ถูกสร้างขึ้นในเครื่องแม่ข่าย ดังนี้

```
# all known message sources
source s_all {
        internal();
        unix-stream("/dev/log");
        file("/proc/kmsg" log_prefix("kernel: "));
};
```

ส่วนที่ 3 เป็นการกำหนดค่า source ที่มาจาก client โดย syslog-ng จะรอรับการเชื่อมต่อ syslog-ng client ที่ ip 0.0.0.0 และ tcp port 514 พร้อมทั้งกำหนดจำนวน connection สูงสุดไว้ที่ 300 connections

```
source client {
        tcp(ip(0.0.0.0) port(514) max-connections(300));
};
```

ส่วนที่ 4 เป็นการกำหนดปลายทางของ log โดยตั้งชื่อเป็น hosts และระบุว่าให้จัดเก็บ log ลง file ที่ /var/log/HOSTS/$HOST/$YEAR/$MONTH/$DAY/ $FACILITY โดยมีการกำหนด option ดังนี้

```
owner(root) group(root) perm(0600) dir_perm(0700) create_dirs(yes));
destination hosts {
  file("/var/log/HOSTS/$HOST/$YEAR/$MONTH/$DAY/$FACILITY"
  owner(root) group(root) perm(0600) dir_perm(0700) create_dirs(yes));
 };
```

ส่วนที่ 5 เป็นการกำหนดปลายทางของ log โดยตั้งชื่อเป็น pipe และระบุว่าให้จัดเก็บ log ลง named pipe ที่ /var/log/pipe และมีการกำหนด template ของ log ที่จะถูกเขียนลง named pipe เป็น template("$YEAR-$MONTH-$DAY $HOUR:$MIN:$SEC $HOST $SOURCEIP [$FACILITY.$PRIORITY] $PROGRAM : $MSGONLY \n") เพื่อให้ระบบที่พัฒนาสามารถอ่าน named pipe ขึ้นไปแสดงผลได้ในรูปแบบที่ต้องการ

```
destination d_pipe {
pipe("/var/log/pipe" \
template("$YEAR-$MONTH-$DAY $HOUR:$MIN:$SEC $HOST $SOURCEIP
[$FACILITY.$PRIORITY] $
PROGRAM : $MSGONLY \n") template-escape(yes));
};
```

ส่วนที่ 6 เป็นการกำหนด filter คือการกรอง โดยการทดสอบนี้ได้ตั้งค่าให้กรอง syslog จาก priority ของ syslog โดยจะกรองเฉพาะ log ที่มี priority ตั้งค่า warning ขึ้นไปจนถึง emerg โดยตั้งชื่อ filter นี้ว่า f_at_least_warn

```
filter f_at_least_warn { level(warn..emerg); };
```

ส่วนที่ 7 เป็นส่วนสำคัญที่สุด เพราะส่วนที่รวบรวมเอา configuration ในการส่วนต่างๆ ที่ได้ระบุไว้มาประกอบกันเพื่อใช้งาน โดย directive log จะเป็นตัวบอกให้ syslog-ng ทำงาน ในการทดลองนี้ได้ตั้งค่าในส่วนของ log ไว้ดังนี้

```
log {
    source(s_all);
    filter(f_at_least_warn);
    destination(hosts);
```

```
    destination(d_pipe);
};

log {
    source(client);
    filter(f_at_least_warn);
    destination(hosts);
    destination(d_pipe);
};
```

5.3.1.3 การตั้งค่าการทำงานของโปรแกรม net-snmpd

การตั้งค่าการทำงานของโปรแกรม netsnmpd นั้นสามารถทำได้ที่ file /etc/snmpd/snmpd.conf โดยในการทดลองนี้ มีการตั้งค่าการทำงานของ netsnmpd เพื่อทำงานหน้าที่เป็น snmp agent ดังนี้

ส่วนที่ 1 เป็นการกำหนดค่าเกี่ยวกับการส่ง snmp trap โดย trap2sink คือ การกำหนดค่า destination ของ snmp trap version 2 , informsink เป็นการกำหนดค่า destination ของ snmp informsink version 2 และได้กำหนด trap community เป็น public (สอดคล้องกับ module trap receiver ในโปรแกรม imguardian)

```
trap2sink  localhost public 162
informsink  localhost public 162
trapcommunity  public
authtrapenable  1
```

ส่วนที่ 2 เป็นการกำหนดค่าการเข้าถึงข้อมูล MIB ของ snmp client โดยจะมีการกำหนดค่า rocommunity คือ ค่า community สำหรับการอ่านอย่างเดียว และ read write community สำหรับการอ่านและการเขียน

```
rocommunity  public
rwcommunity  public
```

ส่วนที่ 3 เป็นการกำหนดค่าการเฝ้าระวังต่างๆ ของเครื่องแม่ข่าย โดยในการทดลองนี้ ได้มีการเฝ้าระวัง process ที่ควรจะต้องทำงานอยู่ในเครื่องแม่ข่าย คือ sshd และ apache โดยระบุค่า Max และ Min สำหรับจำนวน process ในสภาวะปกติ หากมีจำนวน process มากกว่าหรือน้อยกว่าที่กำหนด snmpd จะทำการส่ง snmp trap ออกไปยัง destination ที่ระบุไว้

```
proc  sshd 1 0
proc  apache 10 1
```

รวมทั้งได้กำหนดให้เฝ้าระวัง load average ของเครื่องแม่ข่ายด้วย โดยการตั้งค่าจะอยู่ใน format ของ 5 mins 10 mins 15 mins ดังนี้

```
load  2 3.1 5.1
```

ส่วนที่ 4 เป็นการกำหนดสิทธิ์ในการเรียกดูข้อมูลต่างๆ ซึ่งจะอนุญาตการเข้าถึงจาก localhost และ เครือข่าย 10.100.100.100/24 ดังนี้

```
com2sec local   localhost public
com2sec localNet 10.100.100.0 /24 public
group MyROSystem v1 local
group MyROSystem v2c local
group MyROSystem usm local
group MyROGroup v1 localnet
group MyROGroup v2c localnet
group MyROGroup usm localnet
group MyRWGroup v1 local
group MyRWGroup v2c local
group MyRWGroup usm local
```

ส่วนที่ 5 เป็นการตั้งค่า credentials เพื่อใช้ในการรับค่าที่ทำการเฝ้าระวัง ดังนี้

```
createUser    _internal MD5 "the first sign of madness"
iquerySecName _internal
rouser        _internal
```

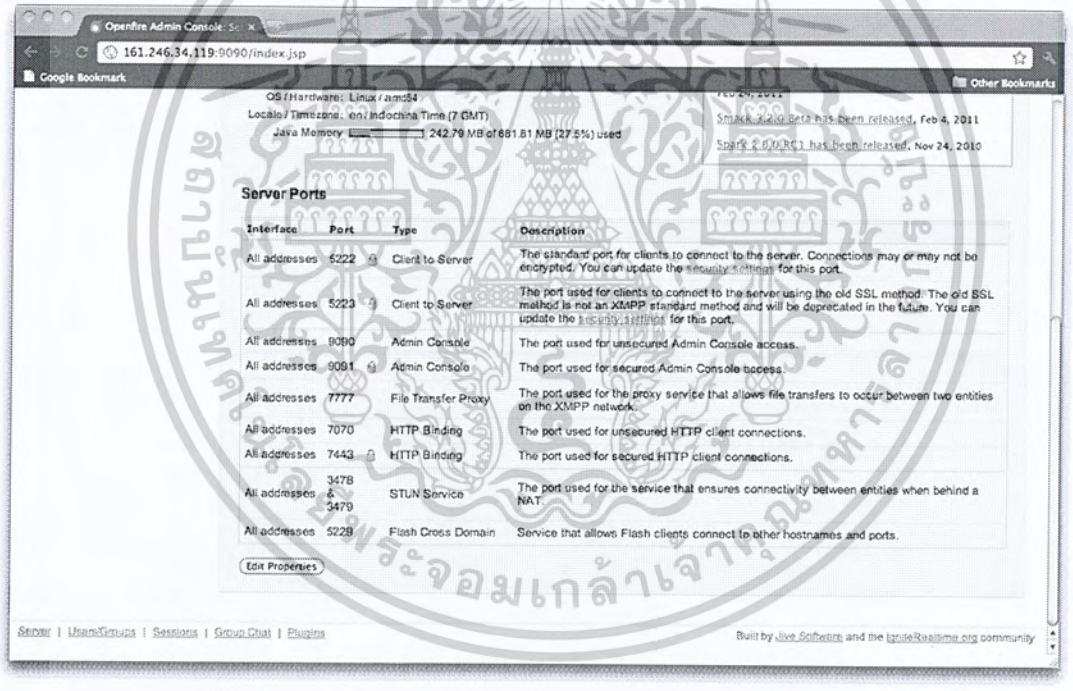
ส่วนที่ 6 เป็นการตั้งค่าเพื่อเปิดการใช้งานของ standard monitoring เช่น linkup link down เป็นต้น

```
# Active the standard monitoring entries
defaultMonitors       yes
linkUpDownNotifications yes
```

5.3.1.4 การตั้งค่าการทำงานของ openfire xmpp server

การตั้งค่าการทำงานของ openfire xmpp server นั้นสามารถได้โดยตั้งค่าตอน install และแก้ไขปรับเปลี่ยนการตั้งค่าจาก web interface

โดยการมีการตั้งค่าที่สำคัญคือ XMPP port ที่รอรับ connection จาก XMPP client โดย openfire จะรอรับ xmpp connection ที่ tcp port 5222 และ tcp port 5223 (SSL) ดังแสดงในรูปที่ 5.7



รูปที่ 5.7 แสดง XMPP Port ที่ Openfire เปิดเพื่อรอรับการเชื่อมต่อ

และรายละเอียดของ xmpp user โดยมีการสร้าง user : imguardian, admin , thanachit , user01 ดังแสดงในรูปที่ 5.8

รูปที่ 5.8 แสดงรายละเอียดบัญชีผู้ใช้บน XMPP Server

5.3.1.5 การตั้งค่าการทำงานของ glassfish enterprise server

การตั้งค่าการทำงานของ glassfish enterprise server ที่ใช้ในการทดสอบที่สำคัญมีดังนี้

domain : domain1

domain directory: /home/glassfish/glassfish/domains/domain1/

application

location="${com.sun.aas.instanceRootURI}/applications/imguardian/"

Web Server Address : 0.0.0.0 tcp port 8080

5.3.2 การตั้งค่าการทำงานของโปรแกรมต่างๆ บนเครื่อง it.kmitl.net

สำหรับการทดสอบได้มีการตั้งค่าดังนี้

5.3.2.1 รายละเอียดการตั้งค่าการทำงานของ syslog-ng

การตั้งค่าการทำงานของ syslog-ng เพื่อทำหน้าที่เป็น syslog-ng client จะต้องมีการกำหนดค่าที่ไฟล์ /etc/syslog-ng/syslog-ng.conf

ในส่วนแรกเป็นส่วนของการกำหนด option ซึ่งในการทดลองนี้ได้ กำหนด option สำหรับ syslog-ng ดังนี้

```
options {
    sync (0);
    time_reopen (10);
    log_fifo_size (1000);
    long_hostnames (off);
    use_dns (no);
    use_fqdn (no);
    create_dirs (no);
```

```
    keep_hostname (yes);
    stats(0);
};
```

ส่วนที่ 2 เป็นการกำหนด source คือ แหล่งที่มาของ log ซึ่งในที่นี้คือ s_sys โดยระบุว่า log จะมาจากทุกที่ที่ log ถูกสร้างขึ้นในเครื่องแม่ข่าย ดังนี้

```
source s_sys {
    file ("/proc/kmsg" log_prefix("kernel: "));
    unix-stream ("/dev/log");
    internal();
};
```

ส่วนที่ 3 เป็นการกำหนด destination คือ ปลายทางของ log ซึ่งในที่นี้คือ logserver โดย ระบุว่า log จะมีปลายทางไปที่ syslog-server address 10.100.100.119 และ tcp port 514

```
destination logserver{
    tcp("10.100.100.119" port (514));
};
```

ส่วนที่ 4 เป็นการกำหนด filter คือการกรอง โดยการทดสอบนี้ได้ตั้งค่าให้กรอง syslog จาก priority ของ syslog โดยจะกรองเฉพาะ log ที่มี priority ตั้งค่า warning ขึ้นไปจนถึง emerg โดยตั้งชื่อ filter นี้ว่า f_at_least_warn

```
filter f_at_least_warn { level(warn..emerg); };
```

ส่วนที่ 5 เป็นการกำหนดให้ syslog-ng ทำงานตามที่ได้ตั้งค่าไว้ โดย log ที่มีต้นทางมาจาก s_sys จะถูก filter ด้วย filter f_at_least_warn และส่งไปยังปลายทางคือ log server ตามที่ได้กำหนด ไว้ก่อนหน้านี้แล้ว ดังนี้

```
log{
    source(s_sys);
    filter(f_at_least_warn);
    destination(logserver);
};
```

5.3.3 การตั้งค่าการทำงานของโปรแกรมต่างๆ บนเครื่อง im.infotech.kmitl.net

สำหรับการทดสอบได้มีการตั้งค่าดังนี้

5.3.3.1 การตั้งค่าการทำงานของโปรแกรม netsnmpd

สามารถทำได้ที่ file /etc/snmpd/snmpd.conf โดยในการทดลองนี้ มีการตั้งค่าการทำงานของ netsnmpd เพื่อทำงานหน้าที่เป็น snmp agent ดังนี้

ส่วนที่ 1 เป็นการกำหนดค่าเกี่ยวกับการส่ง snmp trap โดย trap2sink คือการกำหนดค่า destination ของ snmp trap version 2 , informsink เป็นการกำหนดค่า destination ของ snmp informsink version 2 และได้กำหนด trap community เป็น public (สอดคล้องกับ module trap receiver ในโปรแกรม imguardian)

```
trap2sink  10.100.100.119 public 162
informsink  10.100.100.119 public 162
trapcommunity  public
authtrapenable  1
```

ส่วนที่ 2 เป็นการกำหนดค่าการเข้าถึงข้อมูล MIB ของ snmp client โดยจะมีการกำหนดค่า rocommunity คือ ค่า community สำหรับการอ่านอย่างเดียว และ read write community สำหรับการอ่านและการเขียน

```
rocommuniry public
rwcommunity public
```

ส่วนที่ 3 เป็นการกำหนดค่าการเฝ้าระวังต่างๆ ของเครื่องแม่ข่าย โดยในการทดลองนี้ ได้มีการเฝ้าระวัง process ที่ควรจะต้องทำงานอยู่ในเครื่องแม่ข่าย คือ sshd และ apache โดยระบุค่า Max และ Min สำหรับจำนวน process ในสภาวะปกติ หากมีจำนวน process มากกว่าหรือน้อยกว่าที่กำหนด snmpd จะทำการส่ง snmp trap ออกไปยัง destination ที่ระบุไว้

```
proc  sshd 1 0
proc  apache2 10 1
```

รวมทั้งได้กำหนดให้เฝ้าระวัง load average ของเครื่องแม่ข่ายด้วย โดยการตั้งค่าจะอยู่ใน format ของ 5 mins 10 mins 15 mins ดังนี้

```
load  0.2 0.1 0.1
```

ส่วนที่ 4 เป็นการกำหนดสิทธิ์ในการเรียกดูข้อมูลต่างๆ ซึ่งจะอนุญาตการเข้าถึงจาก localhost และ เครือข่าย 10.100.100.100/24 ดังนี้

```
com2sec local   localhost public
com2sec localNet 10.100.100.0 /24 public
group MyROSystem v1 local
group MyROSystem v2c local
group MyROSystem usm local
group MyROGroup v1 localnet
group MyROGroup v2c localnet
group MyROGroup usm localnet
group MyRWGroup v1 local
group MyRWGroup v2c local
group MyRWGroup usm local
```

ส่วนที่ 5 เป็นการตั้งค่า credentials เพื่อใช้ในการรับค่าที่ทำการเฝ้าระวัง ดังนี้

```
createUser   _internal MD5 "the first sign of madness"
iquerySecName _internal
rouser       _internal
```

ส่วนที่ 6 เป็นการตั้งค่าเพื่อเปิดการใช้งานของ standard monitoring เช่น linkup link down เป็นต้น

```
# Active the standard monitoring entries
defaultMonitors        yes
linkUpDownNotifications yes
```

5.3.4 การตั้งค่าการทำงานของโปรแกรมต่างๆ บนเครื่อง client

สำหรับการทดสอบได้มีการตั้งค่าดังนี้

5.3.4.1 โปรแกรม iChat

ในการทดสอบนี้เลือกใช้โปรแกรม xmpp client ชื่อ iChat เนื่องจากเป็นโปรแกรมที่ใช้งานง่ายและมีมาพร้อมกับระบบปฏิบัติการ Mac OS X โดยการตั้งค่าที่สำคัญจะอยู่ที่การ สร้างบัญชีผู้ใช้ และการกำหนดชื่อเครื่องแม่ข่าย XMPP ที่จะทำการ login เข้าสู่ระบบ

โดยมีการกำหนดค่าดังนี้

Account : thanachit@blue84.crsc.kmitl.ac.th

Account Type: Jabber

Description: XMPP Account for Senior Project

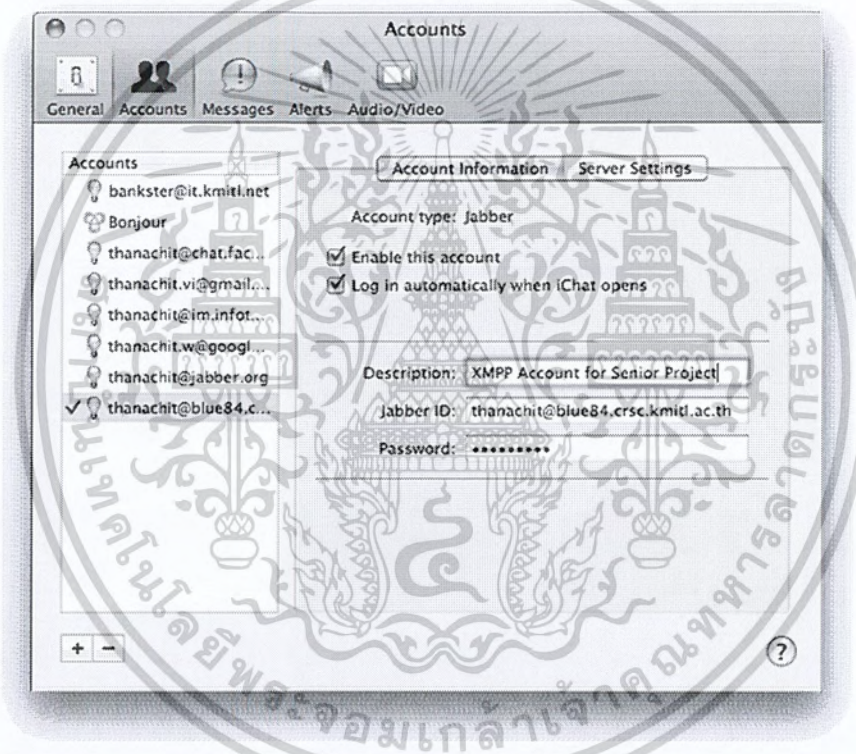
Jabber ID: thanachit@blue84.crsc.kmitl.ac.th

Password : xxxxxxxxxx

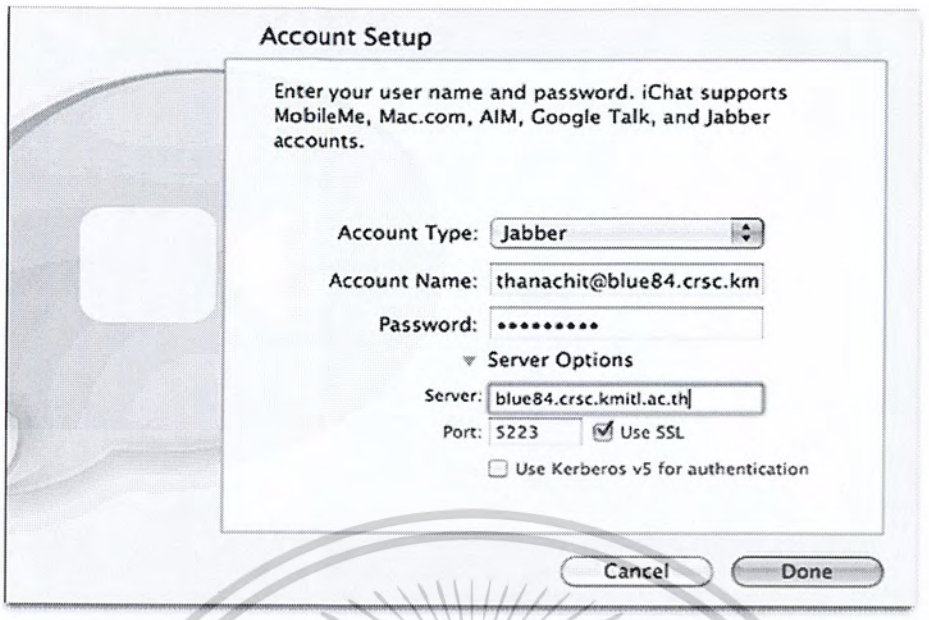
Server: blue84.crsc.kmitl.ac.th (161.246.34.119)

Port : 5223 (SSL)

ซึ่งสามารถแสดงได้ดังรูปที่ 5.9 และ 5.10



รูปที่ 5.9 แสดงการตั้งค่า account information เพื่อเพิ่มบัญชีผู้ใช้ในโปรแกรม iChat

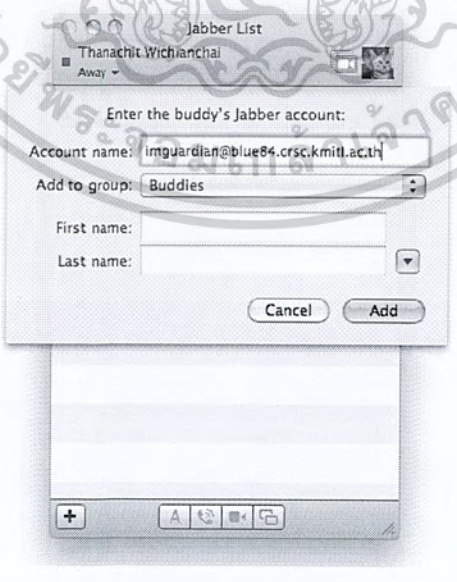


รูปที่ 5.10 แสดงการตั้งค่า server settings เพื่อเพิ่มบัญชีผู้ใช้ในโปรแกรม iChat

5.4 การทดสอบการทำงานของระบบผ่าน Instant Messaging Client

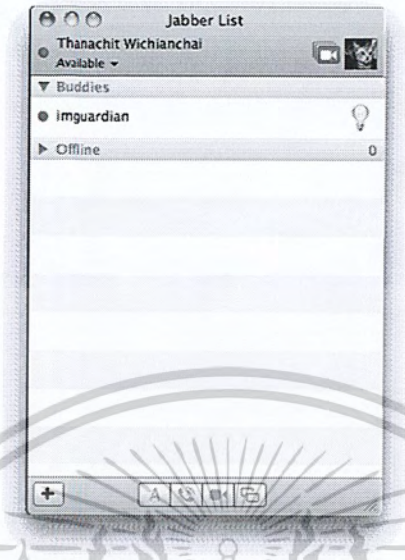
5.4.1 ทดสอบการเข้าสู่ระบบ XMPP Server จาก IM Client ด้วยบัญชีผู้ใช้ที่สร้างขึ้น

หลังจากที่เปิดการทำงานของ account ที่ได้กำหนดไว้แล้ว ichat จะทำการ login เข้าสู่ xmpp server ให้โดยอัตโนมัติ จากนั้นจะทำการเพิ่มบัญชีผู้ใช้ของโปรแกรม imguardian คือ imguardian@blue84.crsc.kmitl.ac.th เข้ามาใน contact list ดังรูปที่ 5.11



รูปที่ 5.11 แสดงการเพิ่มบัญชีผู้ใช้ของโปรแกรม imguardian

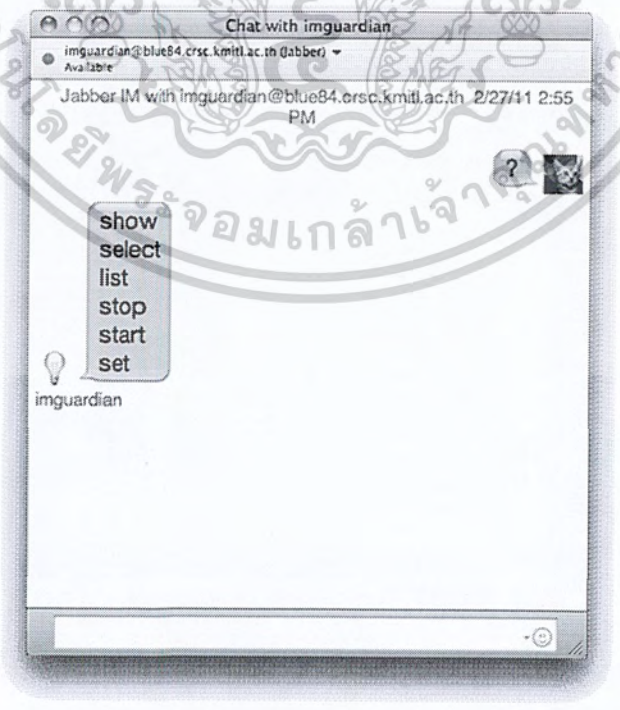
เมื่อเพิ่ม contact เรียบร้อยแล้ว imguardian จะแสดงสถานะ online อยู่ใน contact list ของผู้ใช้งาน ซึ่งสามารถเริ่มต้นการติดต่อสื่อสารกับโปรแกรมได้ทันที ดังรูปที่ 5.12



รูปที่ 5.12 แสดง contact list หลังจากการเข้าสู่ระบบและเพิ่ม contact

5.4.2 ทดสอบการส่งคำสั่งเพื่อเรียกใช้ snmpget

ทดลองส่งคำสั่ง ? เพื่อให้โปรแกรมแสดงคำสั่งทั้งหมดที่สามารถใช้งานได้ออกมา โดยโปรแกรมได้ตอบกลับมายังผู้ใช้งาน ดังรูปที่ 5.13



รูปที่ 5.13 แสดงการเรียกใช้คำสั่ง ? และผลลัพธ์

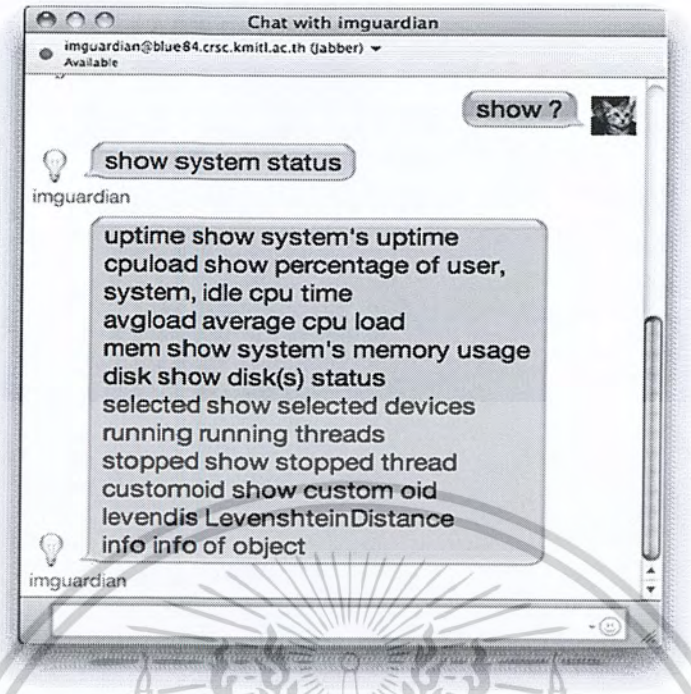
การทดสอบเลือกทำการติดต่อกับเครื่องแม่ข่าย

เพื่อให้สามารถใช้คำสั่งที่โปรแกรมได้จัดเตรียมไว้สำหรับการ query ข้อมูลต่างๆ ผ่าน snmp client ไปยัง snmp agent ของเครื่องแม่ข่ายที่เลือก ในที่นี้ จะเลือกทำการติดต่อกับเครื่อง 10.100.100.164 คือเครื่อง it.kmitl.net โดยสามารถเรียกใช้ได้ด้วยคำสั่ง select "10.100.100.164"

โปรแกรมจะตอบกลับมาว่า Successfully, started snmp client on it.kmitl.net แสดงว่าโปรแกรมได้สร้าง connection ไปยัง snmp agent ของเครื่อง it.kmitl.net ให้สำเร็จแล้ว ดังรูปที่ 5.14

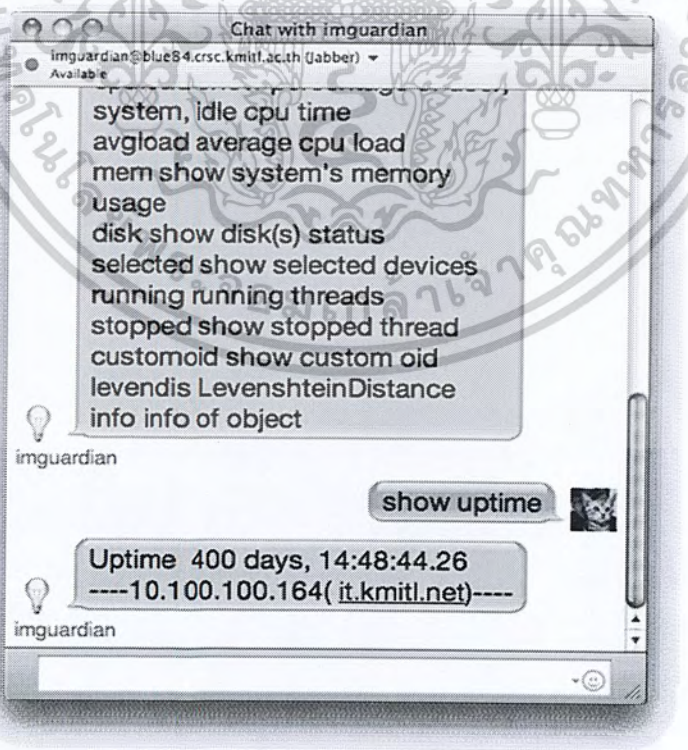
รูปที่ 5.14 แสดงการเรียกใช้คำสั่ง select "10.100.100.164" และผลลัพธ์

การทดสอบการเรียกใช้คำสั่ง show ? เพื่อแสดงรายละเอียดว่า โปรแกรมสามารถติดต่อไปยัง snmp agent ของเครื่องแม่ข่าย เพื่อ query ข้อมูลอะไรได้บ้าง โดยใช้คำสั่ง show?โปรแกรมจะตอบกลับเป็นรายการ ของคำสั่งที่สามารถใช้งานได้ และคำอธิบาย ดังรูปที่ 5.15



รูปที่ 5.15 แสดงการเรียกใช้คำสั่ง show ? และผลลัพธ์

ทดสอบเรียกใช้คำสั่ง show uptime โดยโปรแกรมจะตอบกลับค่า uptime ที่ได้จากการ query จาก snmp agent ของเครื่อง it.kmitl.net เป็น 400 Days ดังรูปที่ 5.16

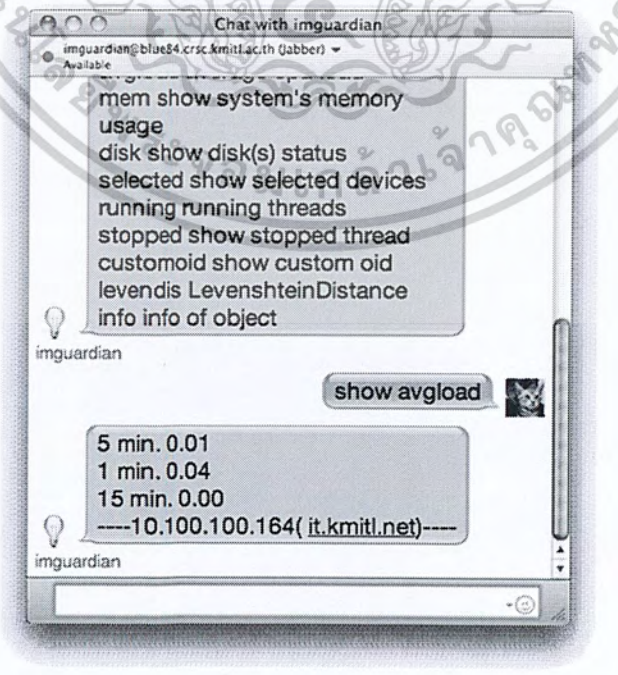


รูปที่ 5.16 แสดงการเรียกใช้คำสั่ง show uptime และผลลัพธ์

ทดสอบเรียกใช้คำสั่ง show cpuload โดยโปรแกรมจะตอบกลับค่า cpu utilization ที่ได้จากการ query จาก snmp agent ของเครื่อง it.kmitl.net เป็นค่า cpu idle , cpu system , cpu user ดังรูปที่ 5.17

รูปที่ 5.17 แสดงการเรียกใช้คำสั่ง show cpuload และผลลัพธ์

ทดสอบเรียกใช้คำสั่ง show avgload โดยโปรแกรมจะตอบกลับค่า load average ที่ได้จากการ query จาก snmp agent ของเครื่อง it.kmitl.net เป็นค่า load average in 5 mins , 10 mins และ 15 mins ดังรูปที่ 5.18

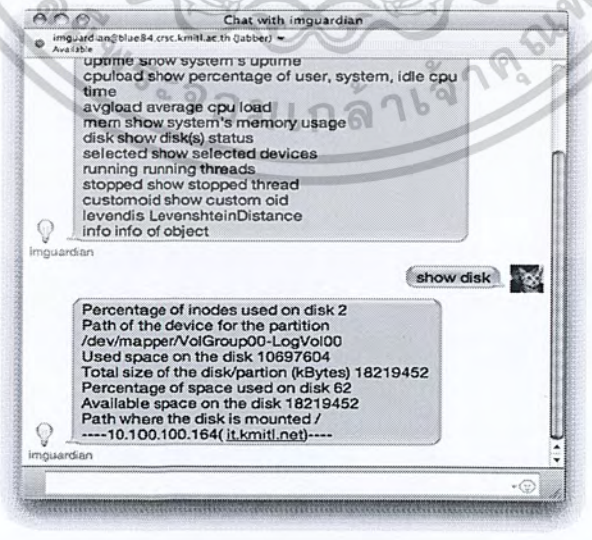


รูปที่ 5.18 แสดงการเรียกใช้คำสั่ง show avgload และผลลัพธ์

ทดสอบเรียกใช้คำสั่ง show mem โดยโปรแกรมจะตอบกลับค่า memory utilization ที่ได้จากการ query จาก snmp agent ของเครื่อง it.kmitl.net เป็นค่า Total Swap Size, Total Ram Buffered, Total Ram Shared, Total Ram in Machine, Total Cache Memory, Total Ram Free, Total Ram Used และ Available Swap Space ดังรูปที่ 5.19

รูปที่ 5.19 แสดงการเรียกใช้คำสั่ง show mem และผลลัพธ์

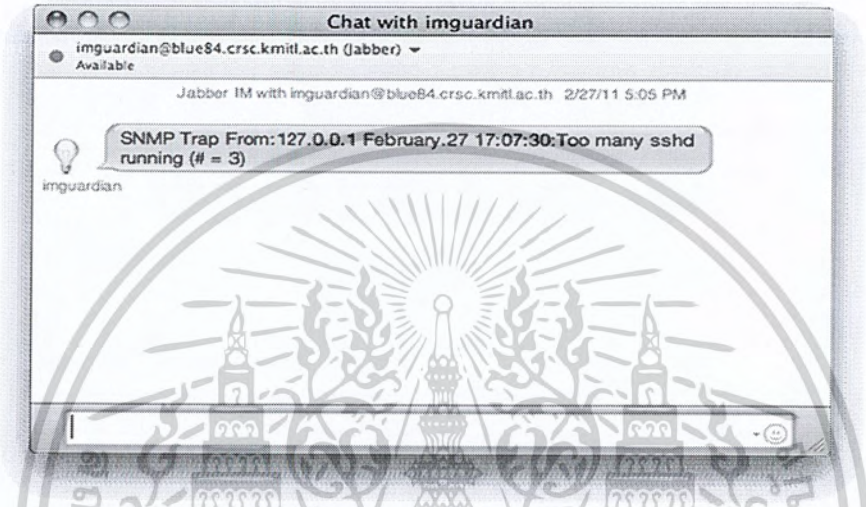
ทดสอบเรียกใช้คำสั่ง show disk โดยโปรแกรมจะตอบกลับค่า disk usage ที่ได้จากการ query จาก snmp agent ของเครื่อง it.kmitl.net เป็นค่า Percentage of inodes used , Path of the devices or partitions , Used Space on disk, Total size of the disk/partition , Percentage of Space used on disk, Available Space on the disk, และ Path where the disk is mounted ดังรูปที่ 5.20



รูปที่ 5.20 แสดงการเรียกใช้คำสั่ง show disk และผลลัพธ์

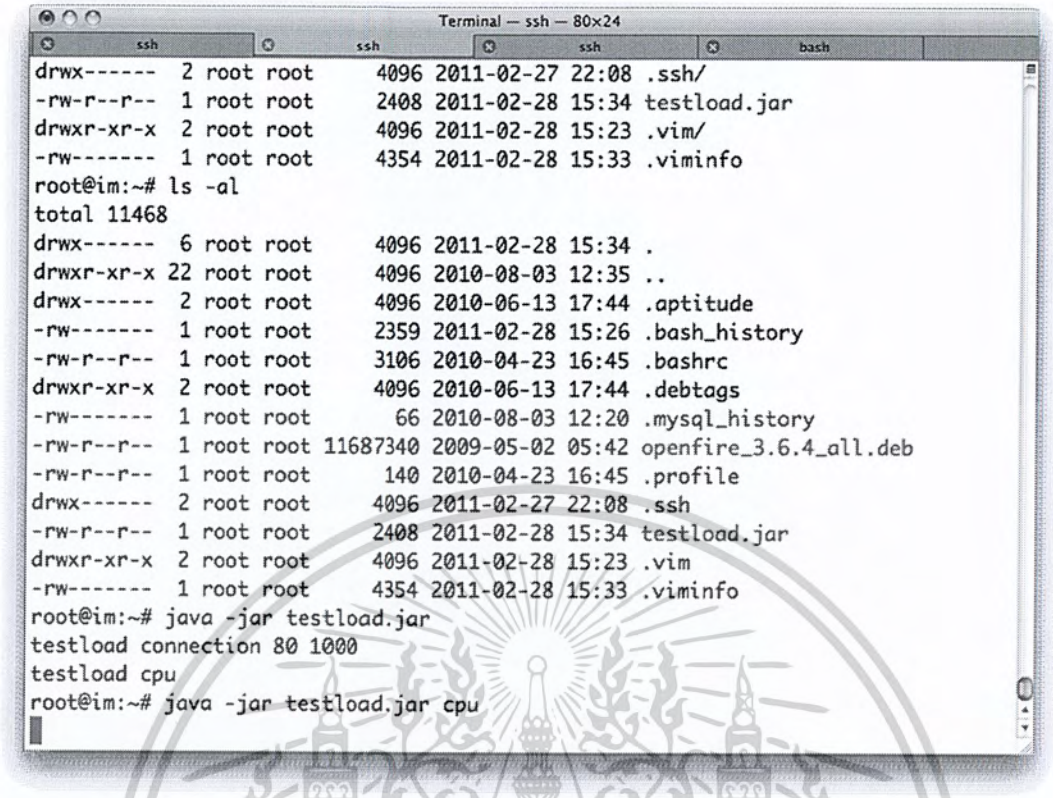
5.4.2 ทดสอบ SNMP Trap

การทดสอบ SNMP Trap ให้ snmp agent ส่ง snmp trap มาในกรณีที่จำนวน process มากกว่าหรือน้อยกว่าที่กำหนดไว้ โดยได้เลือกทำการทดสอบกับ process sshd ซึ่งได้กำหนด ค่าไว้ว่าต้องมีจำนวน process ไม่เกิน 1 process จึงได้ทดสอบ secure shell เข้าสู่เครื่อง imguardian เพื่อให้ sshd process เพิ่มขึ้น และ snmp agent จะได้ส่งข้อความ snmp trap มายังระบบที่พัฒนาเพื่อแจ้งเตือนให้ user ทราบ ดังรูปที่ 5.21



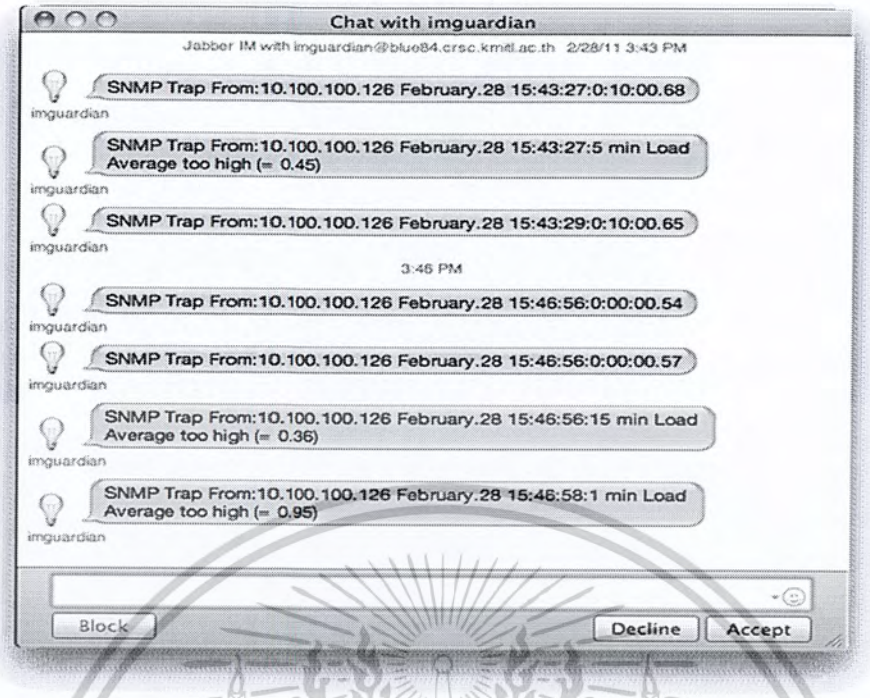
รูปที่ 5.21 แสดงข้อความ SNMP Trap แจ้งเตือน Process มีปัญหา

การทดสอบการแจ้งเตือนเมื่อ load average ของเครื่องแม่ข่ายสูงกว่าที่กำหนดไว้ สามารถทำการทดสอบได้โดย run program testload.java เพื่อใช้งานการประมวลผลของ cpu อย่างหนักจนทำให้ load average ของระบบเพิ่มขึ้น ดังรูปที่ 5.22 ซึ่งเป็นการเรียกใช้โปรแกรม testload.java บนเครื่อง im.infotech.kmitl.net



รูปที่ 5.22 แสดงการเรียกใช้โปรแกรม testload.java บนเครื่อง im.infotech.kmitl.net

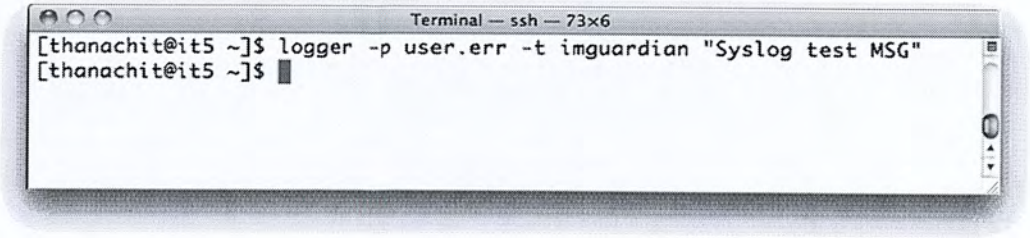
เมื่อ load average เพิ่มขึ้นจนถึงค่าที่กำหนดไว้ใน configuration file ของ snmpd บนเครื่อง im.infotech.kmitl.net คือ 0.2 0.1 0.1 snmp agent บนเครื่อง im.infotech.kmitl.net จะส่งข้อความ snmp trap มายัง ระบบที่พัฒนาเพื่อนำไปแจ้งเตือนแต่ user ที่มีสิทธิ์ ผ่านทาง instant messaging ดังรูปที่ 5.23 โดยจะแสดงข้อความแจ้งเตือน load average ขึ้นสูงมากกว่าที่กำหนดไว้



รูปที่ 5.23 แสดงข้อความแจ้งเตือน load average ขึ้นสูงมากกว่าที่กำหนดบนเครื่อง im.infotech.kmitl.net

5.4.3 ทดสอบการรับ System Log และแจ้งเตือนไปยังผู้ใช้งาน

ในการทดสอบการรับ System Log และแจ้งเตือนไปยังผู้ใช้งานนั้น จะทดสอบโดยใช้ โปรแกรม logger ใน linux ในการสร้าง log message ขึ้นมา โดยสามารถใช้งานโปรแกรม logger โดยระบุ priority ในรูปแบบของ facility.level กำหนด tag ของ log และต่อท้ายด้วย log message ใน double quote โดย system log ที่ถูกสร้างขึ้นจากโปรแกรม logger จะถูกอ่านโดยโปรแกรม syslog-ng ในเครื่อง it.kmitl.net และส่งไปยัง เครื่อง centralized log server เพื่อเขียนลง file และ named pipe ต่อไป การใช้งาน logger สำหรับการทดลอง สามารถทำได้ดังรูปที่ 5.24

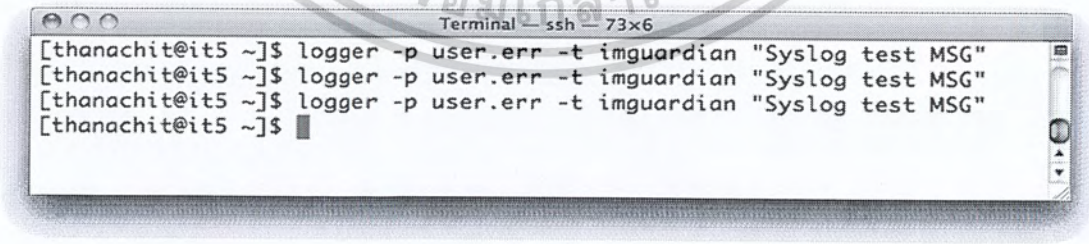


รูปที่ 5.24 แสดงการเรียกใช้งานโปรแกรม logger เพื่อสร้าง log message

เมื่อ syslog ถูกส่งมาถึงเครื่อง imguardian โปรแกรม syslog-ng จะเขียน syslog ลงใน file และ named pipe ตามรูปแบบ (template) ที่กำหนด ในขณะเดียวกันโปรแกรม imguardian (module pipeReader) จะคอยทำการอ่าน syslog จาก named pipe เมื่อมี log เข้ามาใหม่ imguardian จะทำการส่งข้อความ log ไปแจ้งเตือนแก่ผู้ใช้งานผ่าน instant messaging ดังรูปที่ 5.25 โดยจะแสดงข้อความ syslog log ที่ได้รับจากการแจ้งเตือน

รูปที่ 5.25 แสดงข้อความ syslog log ที่ได้รับจากการแจ้งเตือน

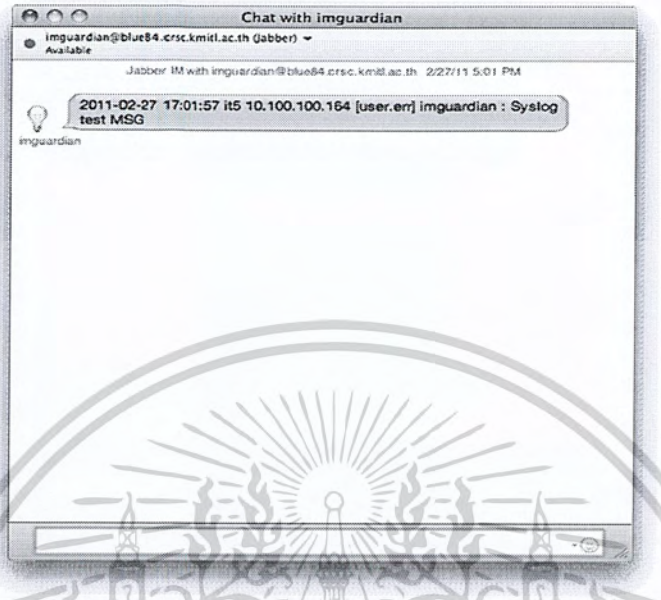
ทดสอบสร้าง log เนื้อหาเหมือนเดิมหลายๆ ครั้ง ดังรูปที่ 5.26



รูปที่ 5.26 แสดงการทดสอบการสร้าง Syslog ด้วยโปรแกรม Logger

จะพบว่าระบบจะไม่ส่งข้อความ log ที่ซ้ำ เนื่องจากได้มีการใช้ algorithm levendis ในการตรวจสอบการซ้ำกันของข้อความ syslog เพื่อไม่ให้เป็นการส่ง system log ซ้ำๆ และไม่เกิดประโยชน์

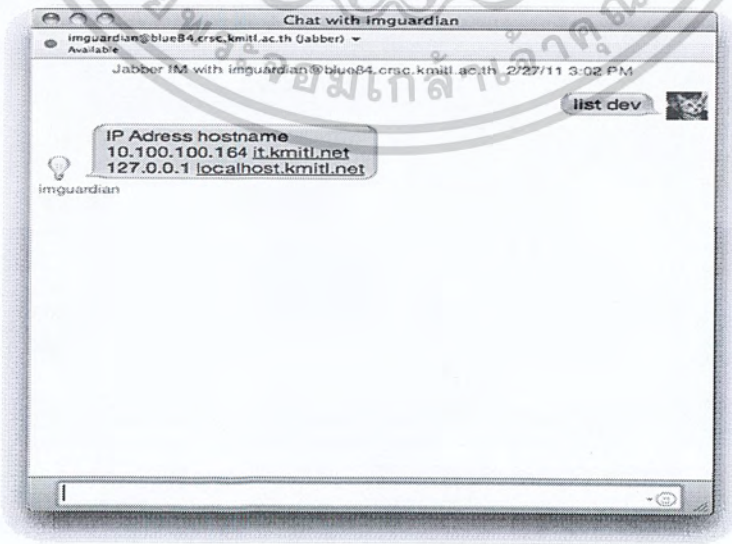
ดังแสดงในรูปที่ 5.27 โดยแสดงให้เห็นว่า ถึงแม้จะมีการส่ง log ซ้ำกันหลายๆ ครั้ง แต่โปรแกรมก็แจ้งเตือนข้อความนั้นครั้งเดียว



รูปที่ 5.27 แสดงการไม่ส่งข้อความ log ซ้ำของระบบ

5.4.5 การทดสอบเรียกใช้คำสั่งอื่นๆ

ทดสอบเรียกใช้คำสั่ง list dev โดยโปรแกรมจะตอบกลับค่า devices ที่ user thanachit มีสิทธิ์ในการได้รับข้อความแจ้งเตือนและ query ข้อมูลต่างๆ ผ่าน snmp protocol เป็นค่า ip address และ hostname ดังรูปที่ 5.28

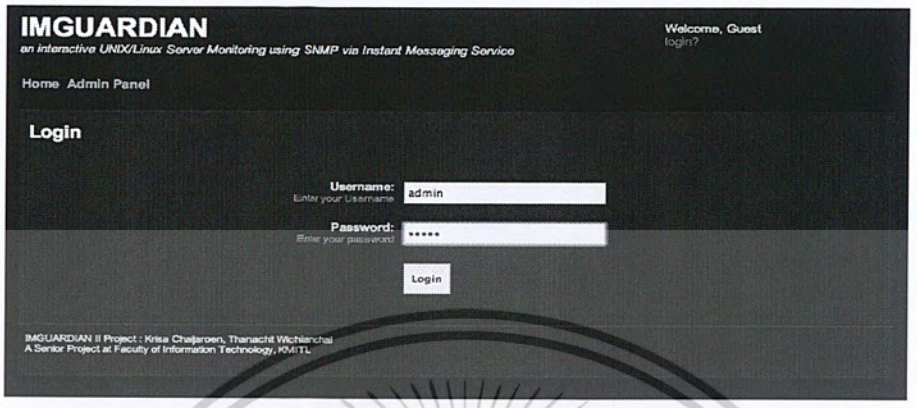


รูปที่ 5.28 แสดงการเรียกใช้คำสั่ง list dev และผลลัพธ์

5.4.6 การทดสอบการทำงานของระบบในส่วนของ Web Interface

ทดสอบเข้าสู่ระบบ

ทำการทดสอบเข้าสู่ระบบด้วย username: admin, password: admin ดังรูปที่ 5.29



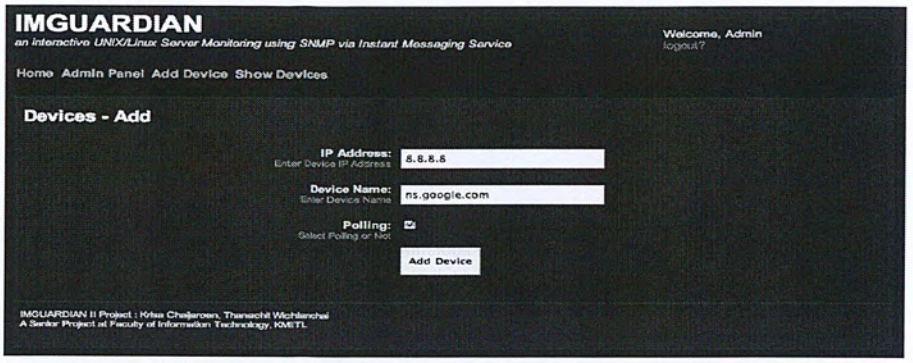
รูปที่ 5.29 แสดงการเข้าสู่ระบบ

เมื่อเข้าสู่ระบบเสร็จสิ้น ระบบจะทำการสร้าง session และแสดงสดงหน้า admin panel ดังรูปที่ 5.30

รูปที่ 5.30 แสดงหน้า admin panel

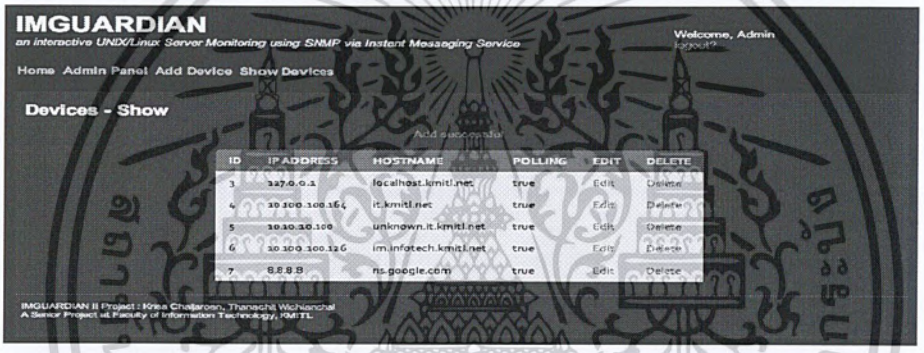
การจัดการ Device

ทดสอบเพิ่ม Device โดยทดสอบกรอกข้อมูล IP Address : 8.8.8.8 และ Device Name หรือ hostname เป็น ns.google.com ดังรูปที่ 5.31



รูปที่ 5.31 แสดงทดสอบแสดงรายการ Device

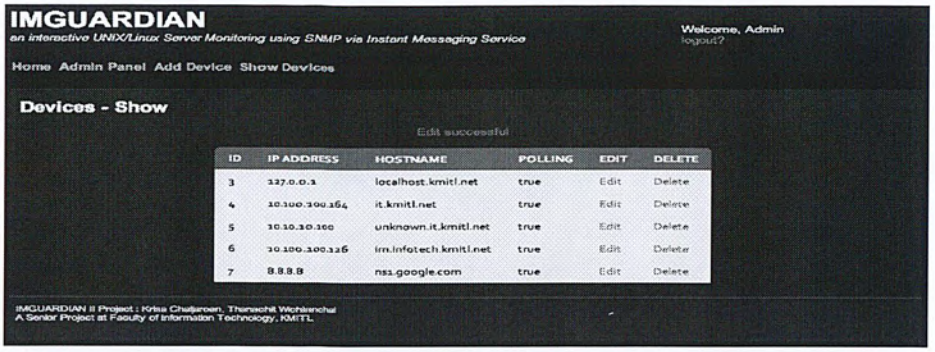
เมื่อทำการเพิ่ม device แล้ว ระบบจะ redirect มายังหน้าแสดงรายการ device จะพบว่า record ที่เพิ่มเข้าไปถูกแสดงอยู่บรรทัดล่างสุด ดังรูปที่ 5.32



รูปที่ 5.32 แสดงรายการ Device หลังทำการเพิ่ม Device

ทดสอบแก้ไข Device โดยทดสอบแก้ไขชื่อของ device ns.google.com เป็น ns1.google.com ดังรูปที่ 5.33 และรูปที่ 5.34

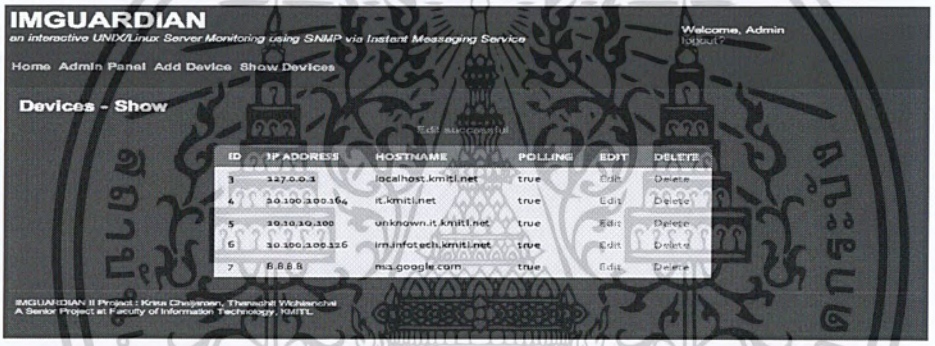
รูปที่ 5.33 แสดงการแก้ไขชื่อ Device



รูปที่ 5.34 แสดงรายการ device หลังจากการแก้ไขชื่อ

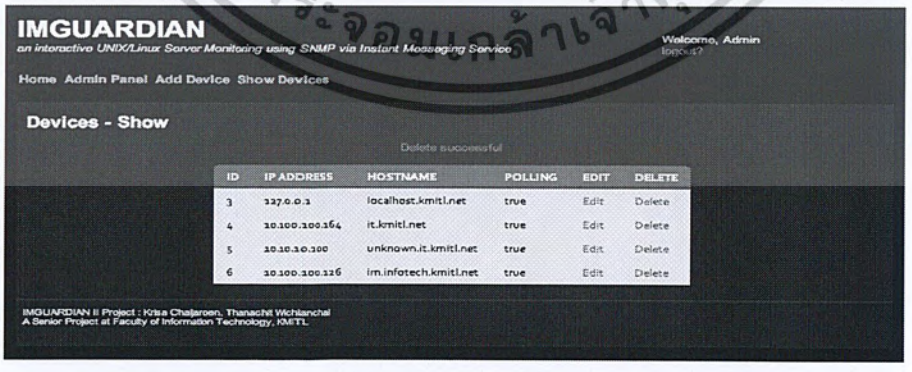
ทดสอบลบ Device

ทดสอบลบ device ns1.google.com ออกจากฐานข้อมูล ทำได้โดยกด link delete ที่ด้านท้าย record ของ ns1.google.com ดังรูปที่ 5.35 และรูปที่ 5.36



รูปที่ 5.35 แสดงรายการ Device และปุ่ม delete

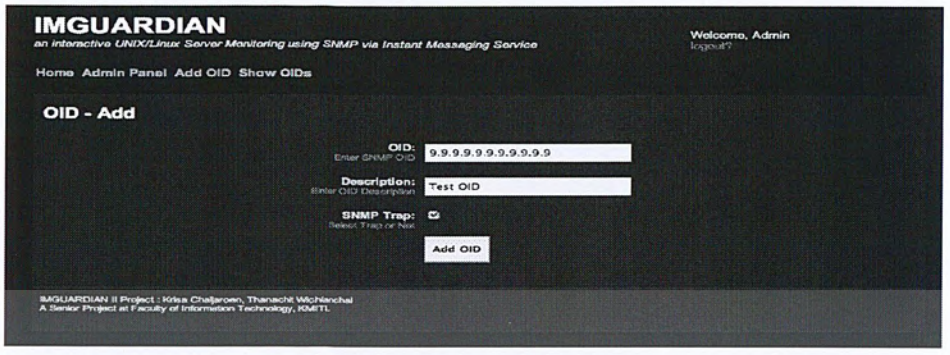
device ดังกล่าวจะถูกลบออกจากระบบ ดังรูปที่ 5.36



รูปที่ 5.36 แสดงรายการ device หลังจากการลบ Device

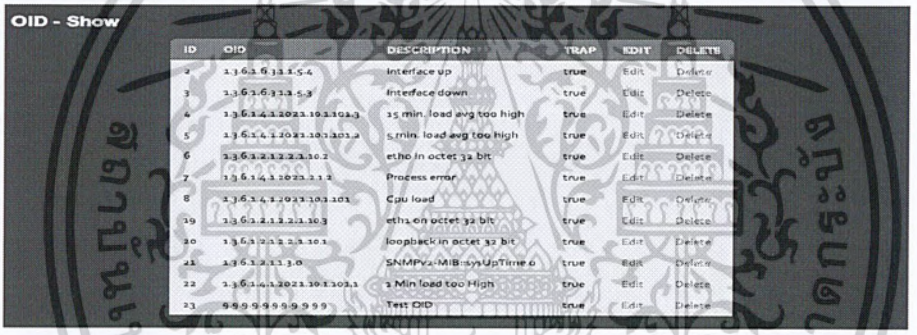
การจัดการ OID

ทดสอบเพิ่ม OID โดยเข้าไปยังระบบ ดังรูปที่ 5.37



รูปที่ 5.37 แสดงการเพิ่ม OID

ทดสอบแสดงรายการ OID



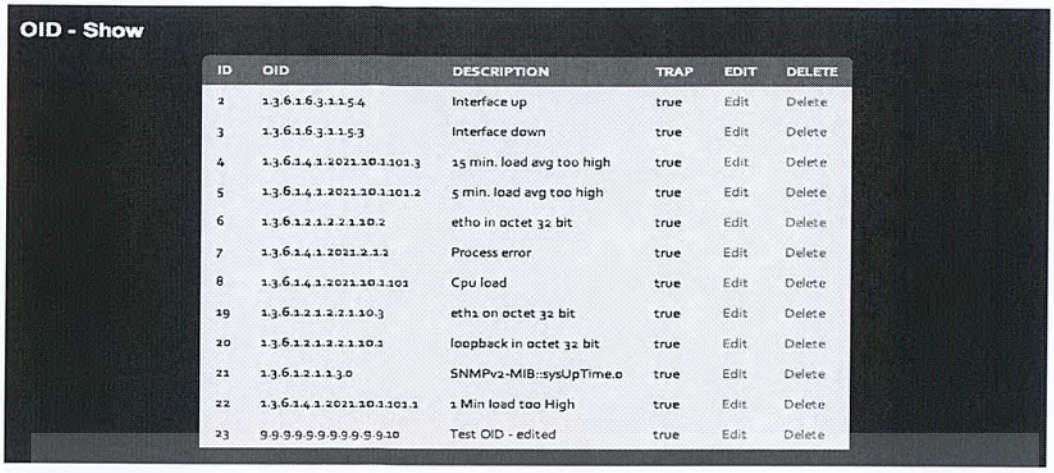
รูปที่ 5.38 แสดงรายการ OID

ทดสอบแก้ไข OID

ทดสอบแก้ไข OID และ description ดังรูปที่ 5.39

รูปที่ 5.39 แสดงการแก้ไข description ของ OID

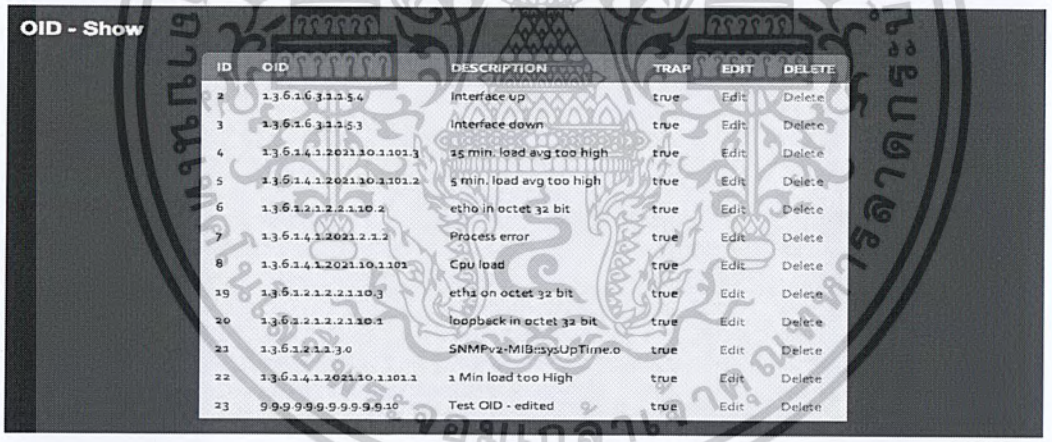
ผลลัพธ์ ดังแสดงในรูปที่ 5.40



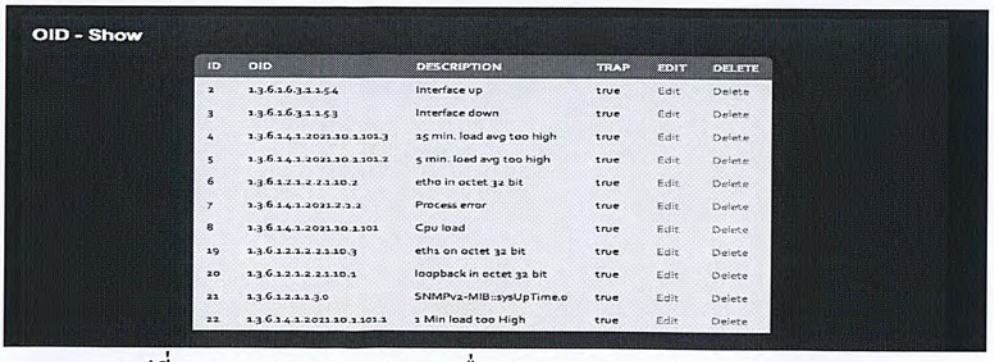
รูปที่ 5.40 แสดงรายการ OID หลังการแก้ไข

ทดสอบลบ OID

ทดสอบลบ OID 9.9.9.9.9.9.9.9.9.9.10 ออกจากฐานข้อมูล ทำได้โดยกด link delete ที่ด้านท้าย record ของ 9.9.9.9.9.9.9.9.9.9.10 ดังรูปที่ 5.41 ระบบจะทำการลบ OID ออกจากระบบและแสดงรายการ OID ดังรูปที่ 5.42



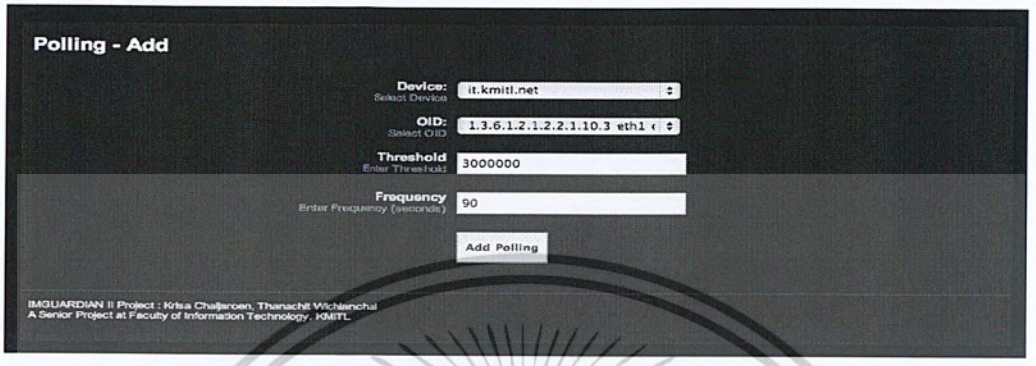
รูปที่ 5.41 แสดงรายการ OID และปุ่ม delete



รูปที่ 5.42 แสดงรายการ OID เมื่อทำการ delete OID ออกจากระบบ

การจัดการ Polling

การเพิ่ม record polling โดมเพิ่ม polling ไปยังเครื่อง it.kmitl.net ค่า eth1 in octet 32 bit เพื่อเฝ้าระวังค่า traffic ขาเข้าสู่ NIC eth1 โดยมี threshold เป็น 3000000 และ frequency เป็น 90 seconds ดังรูปที่ 5.43



รูปที่ 5.43 แสดงการเพิ่มรายการ Polling

แสดงรายการ polling

สามารถแสดงได้ดังรูปที่ 5.44

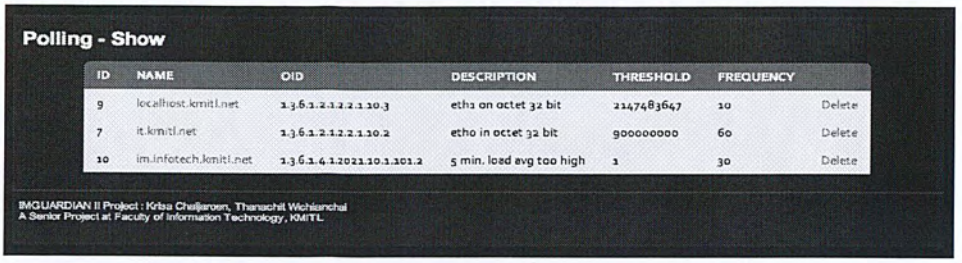
รูปที่ 5.44 แสดงรายการ polling

การลบ polling

สามารถทำการลบได้โดย click link Delete ด้านท้าย record it.kmitl.net ดังรูปที่ 5.45

รูปที่ 5.45 แสดงรายการ Polling และปุ่ม delete

เมื่อลบแล้วแสดงรายการ polling จะพบว่า record ที่ทำการลบถูกลบออกจากระบบแล้ว



รูปที่ 5.46 แสดงรายการ Polling หลังจากทำการ Delete แล้ว

การจัดการ Users และสิทธิ์การใช้งาน

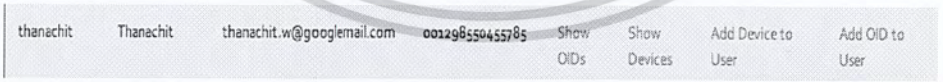
แสดงรายการ User

การแสดงรายการ User ในตาราง ofUser ซึ่งเป็นตารางเดียวกับที่ openfire xmpp server ใช้ สามารถแสดงได้ดังรูปที่ 5.47

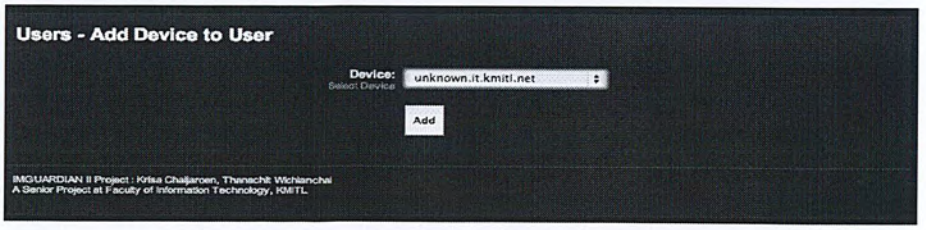
รูปที่ 5.47 แสดงรายการผู้ใช้ที่มีในระบบ

การเพิ่มสิทธิ์ในการเฝ้าระวัง Device ให้กับ User

ทดสอบทำการเพิ่ม device unknown.it.kmitl.net ให้กับ user thanachit โดย click link Add Device to User บน record ของ user thanachit ดังรูปที่ 5.48 และ รูปที่ 5.49

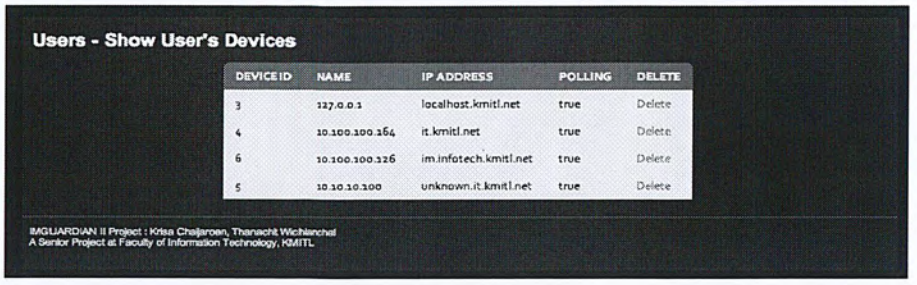


รูปที่ 5.48 แสดงรายละเอียดของ User Thanachit และ Action ต่างๆ



รูปที่ 5.49 แสดงการเพิ่ม Device ให้กับ User: thanachit

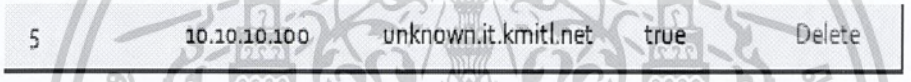
เมื่อเพิ่มเสร็จแล้ว ระบบจะแสดง devices ทั้งหมดที่ user thanachit สามารถเฝ้าระวังและ query ข้อมูลจาก snmp agent ได้ ดังรูปที่ 5.50



รูปที่ 5.50 แสดงรายการ device ที่ user: thanachit มีสิทธิ์เฝ้าระวัง

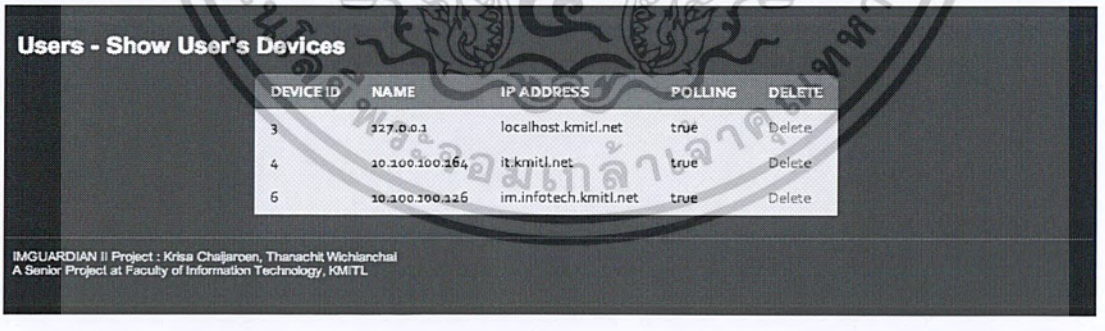
การลบ Device ออกจากรายการ Device ของ User

ทดสอบลบ device unknown.it.kmitl.net ทำได้โดย click link Delete ที่ด้านท้าย record ของ unknown.it.kmitl.net ดังรูปที่ 5.51



รูปที่ 5.51 แสดงรายการ device และปุ่ม delete

ระบบจะแสดงรายการ devices ของ user thanachit โดยแสดงให้เห็นว่า device unknown.it.kmitl.net ถูกลบออกจากระบบเรียบร้อยแล้ว ดังรูปที่ 5.52



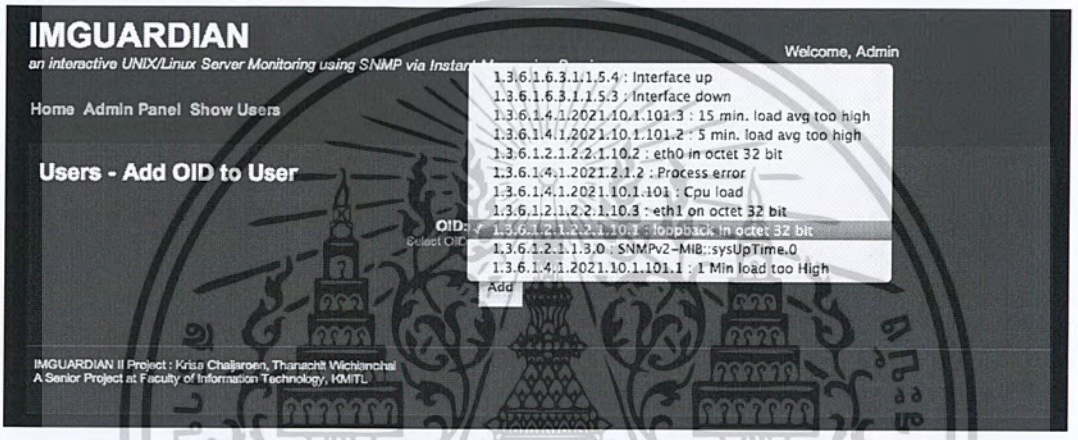
รูปที่ 5.52 แสดงรายการ device หลังจากทำการ delete device แล้ว

การจัดการสิทธิ์ที่มีต่อ OID ของ User

การเพิ่ม OID ให้ User สามารถทำได้โดย click link Add OID to User ที่ด้านของ record ของ user: thanachit ในหน้า show user ดังรูปที่ 5.53

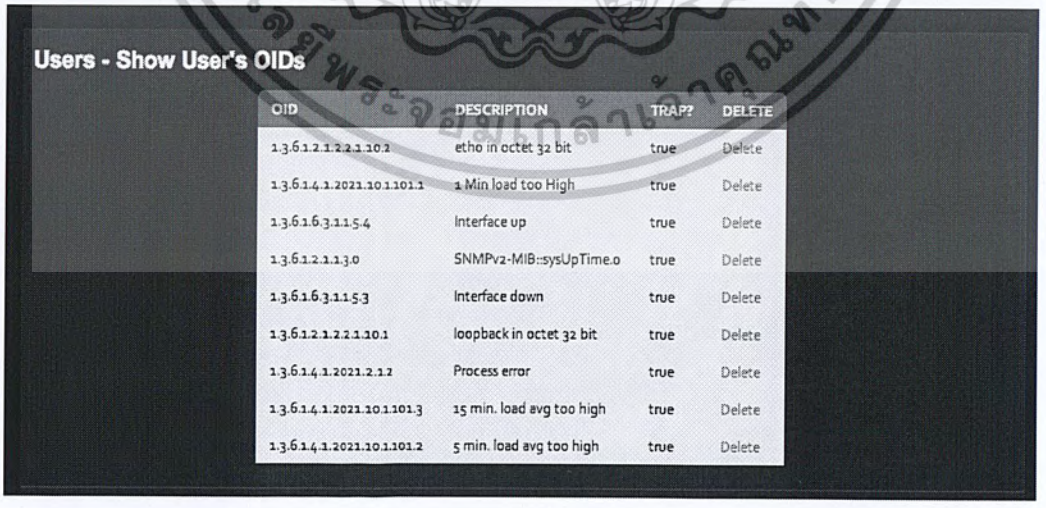
รูปที่ 5.53 แสดงรายละเอียดของ user: thanachit และปุ่ม Action ต่างๆ

ทำการเลือก OID จากรายการ โดยการทดสอบจะเลือก loopback in octet 32 bit ดังรูปที่ 5.54



รูปที่ 5.54 แสดงการเพิ่ม OID ให้กับ User: thanachit

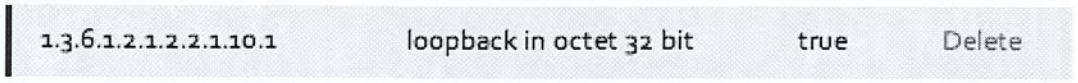
เมื่อเพิ่ม OID แล้วระบบจะแสดงรายการ OID ที่ user มีสิทธิ์ใช้ ดังรูปที่ 5.55



รูปที่ 5.55 แสดงรายการ OID ที่ user: thanachit มีสิทธิ์เฝ้าระวัง

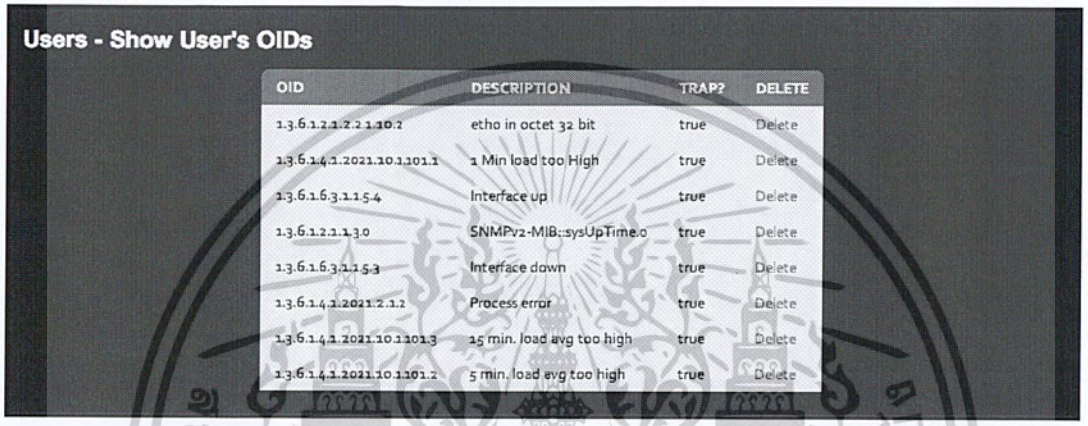
ทดสอบ delete OID

สามารถทำการลบ User OID ออกได้โดย click link Delete ด้านท้าย record ของ OID ดังรูปที่ 5.56



รูปที่ 5.56 แสดงรายละเอียดของ OID และปุ่ม Delete

เมื่อลบเสร็จแล้วระบบจะทำการแสดงรายการ OID ของ user ดังรูปที่ 5.57



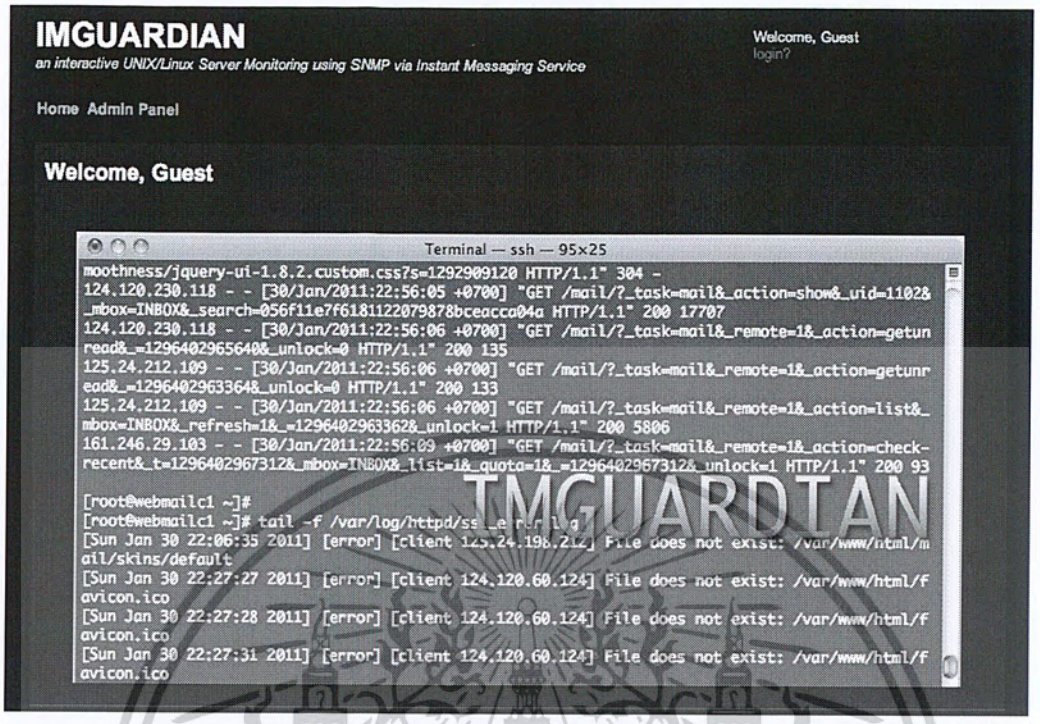
รูปที่ 5.57 แสดงรายการ OID หลังจากทำการลบออกจากระบบ

การออกจากระบบ

สามารถออกจากระบบได้โดยclick link logout? ที่ด้านบนขวาของหน้าจอ ดังรูปที่ 5.58

รูปที่ 5.58 แสดงปุ่ม logout? ที่ใช้สำหรับการออกจากระบบ

เมื่อทำการ clear session แล้ว ระบบจะ redirect user ไปยังหน้าหลักของระบบ ดังรูปที่ 5.59



รูปที่ 5.59 แสดงหน้าหลักของระบบ

บทที่ 6
บทสรุป

6.1 ผลจากการดำเนินการ

จากการพัฒนาระบบตรวจสอบเฝ้าระวังและแจ้งเตือนสถานะของเครื่องแม่ข่าย โดยใช้โปรโตคอล SNMP ผ่านข้อความด่วนทันใจ โดยออกแบบและพัฒนาบนพื้นฐานของประยุกต์ใช้การตรวจสอบสถานะของเครื่องแม่ข่ายและอุปกรณ์เครือข่ายผ่านทาง SNMP การจัดการ system log ด้วย Syslog-NG และการติดต่อสื่อสารผ่านทาง instant messaging โดยนำข้อมูลเกี่ยวกับสถานะของเครื่องแม่ข่าย บริการบนเครื่องแม่ข่าย เหตุการณ์ต่างๆ ที่ได้จาก SNMP Trap และ Syslog-NG แจ้งเตือนไปยังดูแลระบบ ผ่านทางบริการ ข้อความด่วนทันใจ (Instant messaging) เพื่อให้ผู้ดูแลระบบได้รับทราบถึง เหตุการณ์ผิดปกติและ ข้อผิดพลาด นอกจากนี้การพัฒนาระบบในลักษณะที่เป็น module สามารถช่วยเพิ่มความยืดหยุ่น ในระบบ ทำให้สามารถขยายขอบเขตและความสามารถเพิ่มเติมได้ในอนาคต

โดยในโครงงานนี้ คณะผู้จัดทำได้ทำการออกแบบและพัฒนาระบบทั้งสิ้น 5 โมดูล ดังนี้ 1. การทำ SNMP Polling 2. การรับ SNMP Trap 3. การ query สถานะของเครื่องแม่ข่ายผ่าน protocol SNMP 4. การแจ้งเตือน syslog-ng 5. Web Interface สำหรับการตั้งค่าการทำงานของระบบ

6.2 ประโยชน์ที่ได้รับ

ระบบช่วยลดเวลาในการเฝ้าสังเกตระบบโดยผู้ดูแลระบบไม่จำเป็นต้องตรวจสอบและเฝ้าสังเกตระบบตลอดเวลาเนื่องจากมีการแจ้งเตือนเมื่อมีเหตุการณ์ผิดปกติผ่านบริการข้อความด่วนทันใจ (Instant Messaging) ทั้งยังช่วยให้ ผู้ดูแลระบบสามารถตอบคำถามเรื่องประสิทธิภาพ และสถานะการทำงานแก่ผู้บริหารได้ รวมทั้ง ลดเวลาที่ระบบไม่สามารถทำงานได้ เนื่องจากรับทราบปัญหาที่เกิดขึ้นได้เร็วขึ้น

6.3 แนวทางในการดำเนินงานในอนาคต

เพื่อให้ระบบสามารถทำงานได้อย่างมีประสิทธิภาพ ควรมีการพัฒนาเพิ่มเติมในส่วนของการสนับสนุน plugin, tools ต่างๆ เพื่อทำให้ระบบมีสามารถในการทำงานได้หลากหลายมากขึ้น เช่น ช่องทางแจ้งเตือน อาจเพิ่มเป็น email, sms เป็นต้น รวมทั้งอาจประยุกต์ใช้ SNMP Set เพื่อทำการแก้ไขและตั้งค่าให้กับ SNMP Daemon เพื่อนำไปสู่การแก้ไขปัญหาเบื้องต้น

บรรณานุกรม

[1] Wikipedia. **Java (programming language)**. [Online] Available:
http://en.wikipedia.org/wiki/Java_(programming_language)

[2] Wikipedia. **Object-oriented programming.** [Online] Available:
http://en.wikipedia.org/wiki/Object-oriented_programming

[3] ภูวดล ด่านระหาญ. **Syslog-ng (Syslog new generation)**. [Online] Available:
http://www.thaicert.org/paper/unix_linux/syslog-ng.php

[4] Ignite Realtime.**Smack API**. [Online] Available:
http://www.igniterealtime.org/projects/smack/

[5] Mauro, D. and Schmidt, K.J. 2005. **Essential SNMP**. 2nd Edition. O'Reilly Media: USA.

ประวัติผู้เขียน

นายกฤษา ไชยเจริญ

วันเดือนปีเกิด	24 มิถุนายน 2531
อายุ	23 ปี
การศึกษา	
ประวัติการศึกษา	2550 เทคโนโลยีสารสนเทศ สาขาเทคโนโลยีสารสนเทศ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง
ประสบการณ์และผลงาน	วิทยากรบรรยายการใช้งานโปรแกรม Windows Movie Maker ค่ายสมองแก้ว ครั้งที่ 22

นายธนชิต วิเชียรฉาย

วันเดือนปีเกิด	1 เมษายน 2531
อายุ	23 ปี
การศึกษา	
ประวัติการศึกษา	2550 เทคโนโลยีสารสนเทศ สาขาเทคโนโลยีสารสนเทศ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง
ประสบการณ์และผลงาน	
พฤษภาคม 2552	ได้รับรางวัลรองชนะเลิศอันดับหนึ่ง จากผลงาน fireWatch โครงการ Microsoft Imagine Cup 2009
21-24 ตุลาคม 2551	วิทยากรบรรยายเกี่ยวกับ Computer System Administration ค่ายไอทีแคมป์ ครั้งที่ 5 คณะเทคโนโลยีสารสนเทศ สจล.
มกราคม 2551-ปัจจุบัน	Unix System Administrator สำนักบริการคอมพิวเตอร์ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง