ระบบตรวจจับการบุกรุกด้วย Snort Rules ที่เหมาะสม

Optimizing Snort Rules in Intrusion Detection System

โดย

นายพิศาล ศิริบัณฑิตย์

รหัส 45061634

H002191

อาจารย์ที่ปรึกษา

ผศ.ดร. จันทร์บูรณ์ สถิตวิริยวงศ์

รายงานนี้เป็นส่วนหนึ่งของวิชาโครงการพัฒนาระบบงาน

หลักสูตรวิทยาศาสตรมหาบัณฑิต สาขาวิชาเทคโนโลยีสารสนเทศ

ภาคเรียนที่ 1 ปีการศึกษา 2547

คณะเทคโนโลยีสารสนเทศ

สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

ชื่อหัวข้อ	ระบบตรวจจับการบุกรุกด้วย Snort Rules ที่เหมาะสม
นักศึกษา	นายพิศาล ศิริบัณฑิตย์
อาจารย์ที่ปรึกษา	ผศ.ดร. จันทร์บูรณ์ สถิตวิริยวงศ์
ระดับการศึกษา	วิทยาศาสตรมหาบัณฑิต สาขาวิชาเทคโนโลยีสารสนเทศ
แขนงวิชา	วิทยาการสารสนเทศ
ปีการศึกษา	2547

บทคัดย่อ

ระบบตรวจจับการบุกรุก (Intrusion Detection System) เป็นกลยุทธ์หนึ่งที่ช่วยสร้างปราการอันเข็มแข็งสำหรับใช้ป้องกันภัยคุกคามและผู้บุกรุกในรูปแบบต่างๆ ทั้งจากภายในและภายนอกองค์กร ในช่วงหลายปีที่ผ่านมาได้มีบริษัทต่างๆ พยายามพัฒนาซอฟต์แวร์ที่ทำการตรวจจับการบุกรุก แต่เนื่องจากราคาที่ค่อนข้างสูง ดังนั้น Open Source Community จึงได้พัฒนาซอฟต์แวร์ดังกล่าวที่มีราคาถูกกว่าและต้องการการลงทุนด้านฮาร์ดแวร์เพียงเล็กน้อย ทำให้ได้รับความนิยมเป็นอย่างมาก อย่างไรก็ตามผู้บุกรุกในปัจจุบันนี้มีความพยายามที่จะบุกรุกโดยใช้รูปแบบใหม่ๆ อยู่เสมอ ดังนั้นระบบตรวจจับการบุกรุกจึงต้องมีการสร้างและปรับปรุงรูปแบบของการป้องกันการบุกรุกให้ทันสมัยตลอดเวลา Snort เป็นระบบตรวจจับการบุกรุกแบบ Open Source Software หนึ่งที่ได้รับความนิยมโดยหัวใจหลักของระบบตรวจจับการบุกรุกด้วย Snort คือกฎการตรวจจับผู้บุกรุกด้วยเหตุที่ Snort มีกฎเป็นจำนวนมาก Snort จึงเกิดการตรวจจับที่ผิดพลาดได้มากเช่นเดียวกัน ดังนั้นการสร้างระบบตรวจจับการบุกรุกโดยใช้ Snort Rules ให้เหมาะกับแต่ละองค์กรจึงเป็นสิ่งจำเป็น ซึ่งแต่ละองค์กรย่อมมีระบบสารสนเทศที่แตกต่างกันไป ดังนั้นการใช้ Snort Rules ที่เหมาะสมทำให้ระบบตรวจจับการบุกรุกทำงานได้อย่างมีประสิทธิภาพมากขึ้น ช่วยลดการแจ้งเตือนที่ผิดพลาด ใช้หน่วยความจำลดลงและหน่วยประมวลผลทำงานได้มากขึ้น ปัจจุบันกฎของ Snort และคอนฟิกไฟล์นั้นเป็นเพียงเท็กซ์ไฟล์ทำให้การเพิ่มและแก้ไขกฎดังกล่าวทำได้ไม่สะดวก ดังนั้นโครงการพัฒนาระบบงานนี้จึงสร้างเครื่องมือในการจัดการกับกฎของ Snort และคอนฟิกไฟล์ดังกล่าวให้สะดวกขึ้นด้วย Graphic User Interface (GUI)

Title	Optimizing Snort Rules in Intrusion Detection System
Student	Mr. Pisarn Siribandit
Advisor	Asst. Prof. Chanboon Sathitwiriyawong, Ph.D.
Level of Study	Master of Science in Information Technology
Major	Information Science
Academic Year	2004

ABSTRACT

Intrusion detection system is a strategic to help and protect criminals and intruders from both internal and external organization. In many years ago companies tried to develop intrusion detection's software but it s' rather expensive. So, open source community develop intrusion detection's software that price is cheapness and require a little hardware therefore it so much popular. However nowadays intruder always try to attack with modern style. So, Intrusion detection system always must be update. Snort is popular open source intrusion detection system that heart of Snort intrusion detect system is Snort rules. However Snort generates more fault alarms because it has many rules therefore it is important to develop optimizing Snort rules for each organization that has different Information Systems, thus the optimize action of Snort rules can increase the detection rate, decrease the number of fault alarms, use lower memory and CPU. Nowadays Snort rules and configuration files are text files. So, creation and updating them are inconvenience. This system development project create Snort management tool for manage Snort rules and configuration files with Graphic User Interface (GUI).

กิตติกรรมประกาศ

ผู้จัดทำขอขอบพระคุณ ผศ.ดร. จันทร์บูรณ์ สถิตวิริยวงศ์ ซึ่งได้ให้คำปรึกษาทั้งแนวคิดและข้อเสนอแนะต่างๆ และขอขอบใจเพื่อนๆทุกคนที่ให้กำลังใจและความช่วยเหลือมาโดยตลอด

สารบัญ

หน้า

สารบัญ (ต่อ)

หน้า

สารบัญตาราง

หน้า

ตารางที่

สารบัญรูป

หน้า

รูปที่

สารบัญรูป (ต่อ)

หน้า

รูปที่

สารบัญรูป (ต่อ)

หน้า

รูปที่

บทที่ 1
บทนำ

1.1 ความเป็นมาและความสำคัญของโครงการพัฒนาระบบงาน

ปัจจุบันระบบคอมพิวเตอร์และอินเทอร์เน็ตได้เข้ามามีบทบาทอย่างมาก จนเริ่มเป็นสิ่งจำเป็นสิ่งหนึ่งที่ขาดไม่ได้ในชีวิตประจำวันของมนุษย์ ข้อมูลทุกสิ่งทุกอย่างไม่ว่าจะเป็นข้อมูลส่วนตัว ข้อมูลทางการค้า การเงิน ข้อมูลข่าวสาร ข้อมูลที่เป็นความลับและข้อมูลที่สำคัญอื่นๆ จะถูกบันทึกและทำการประมวลผลด้วยคอมพิวเตอร์ คอมพิวเตอร์แต่ละเครื่องสามารถติดต่อรับส่งข้อมูลผ่านอินเทอร์เน็ต ทำให้เกิดความสะดวกในการแลกเปลี่ยนและสืบค้นข้อมูลได้อย่างอิสระ อย่างไรก็ตามสิ่งเหล่านี้เปรียบเสมือนดาบสองคม เพราะทำให้ผู้ไม่ประสงค์ดีที่มีความรู้ด้านคอมพิวเตอร์ และอินเทอร์เน็ตมากพอสามารถรุกล้ำเข้าไปยังคอมพิวเตอร์ต่างๆบนเครือข่ายที่เชื่อมต่อกับอินเทอร์เน็ตได้แม้จะไม่ได้รับอนุญาตก็ตาม เมื่อผู้บุกรุกเหล่านี้เข้ามาในระบบแล้ว ก็สามารถที่จะกระทำการบางอย่างกับเครื่องคอมพิวเตอร์ของเราได้ตามใจชอบ ไม่ว่าจะเป็นการลอบดู ขโมย ทำลาย หรือเปลี่ยนแปลงข้อมูลที่สำคัญ ทั้งแบบที่ตั้งใจและไม่ได้ตั้งใจ จึงเป็นเหตุให้ผู้ดูแลระบบ (System Administrator) ต้องเสียเวลาอย่างมากในการค้นหาหนทางเพื่อป้องกันการบุกรุกของผู้บุกรุกเหล่านี้

ในช่วงหลายปีที่ผ่านมาได้มีบริษัทต่างๆ พยายามพัฒนา ซอฟต์แวร์และฮาร์ดแวร์ ซึ่งใช้ทำการตรวจจับการบุกรุก แต่เนื่องจากราคาที่ค่อนข้างสูงของซอฟต์แวร์และฮาร์ดแวร์ดังนั้น open source community จึงได้พัฒนาซอฟต์แวร์ที่มีราคาถูกกว่า ในปัจจุบันมีการใช้ระบบตรวจจับการบุกรุกที่เป็น open source software อย่างแพร่หลาย ซึ่งซอฟต์แวร์ดังกล่าวสามารถดาวน์โหลดได้จากอินเทอร์เน็ต และต้องการการลงทุนทางด้านฮาร์ดแวร์เพียงเล็กน้อยเท่านั้น ตัวอย่างของระบบตรวจจับการบุกรุกที่เป็น open source software เช่น Snort, Tcpdump, Logsurfer, Shadow เป็นต้น

ปัจจุบัน Snort เป็นซอฟต์แวร์ระบบตรวจจับการบุกรุกที่ได้รับความนิยมอย่างมาก มีกฎมากมายในการตรวจจับผู้บุกรุก ด้วยเหตุที่มีกฎเป็นจำนวนมาก Snort จึงเกิดการตรวจจับที่ผิดพลาดได้มากเช่นเดียวกัน จากกรณีศึกษาเรื่อง "Three Open Source Security Management Tools" พบว่า Snort มีการตรวจจับการบุกรุกได้มากที่สุด 44% แต่ก็มีการตรวจจับที่ไม่จำเป็นมากที่สุดถึง 99% ส่วน Pakemon มีการตรวจจับการบุกรุกได้ 34% แต่ก็มีการตรวจจับที่ไม่จำเป็น 95% (Kayacik and Zincir-Heywood. 2003) ดังนั้นโครงงานพัฒนาระบบนี้จึงสร้าง Snort rules ที่เหมาะกับสารสนเทศที่มีใช้อยู่ในแต่ละองค์กรเพื่อลดการตรวจจับที่ไม่จำเป็นให้น้อยลง

หัวใจหลักของระบบตรวจจับการบุกรุกด้วย Snort คือ กฎการตรวจจับ ซึ่งกฎการตรวจจับการบุกรุกนั้นมีมากมาย บางครั้งกฎบางข้อก็ไม่ได้ใช้งานเพราะไม่มีการใช้งานสารสนเทศนั้น เช่น บางองค์กรไม่มีการใช้งานซอฟต์แวร์ระบบฐานข้อมูล Oracle ก็ไม่มีความจำเป็นที่จะต้องเอาข้อมูลที่ต้องการตรวจจับมาเปรียบเทียบกับกฎที่เกี่ยวกับ Oracle ดังนั้นจึงไม่มีความจำเป็นที่จะต้องโหลดกฎที่เกี่ยวกับ Oracle ใน Snort ทำให้ลดการแจ้งเตือนที่ผิดพลาด การทำงานมีประสิทธิภาพมากขึ้น หน่วยประมวลผลทำงานได้ดีขึ้นและใช้หน่วยความจำน้อยลง ปัจจุบันกฎของ Snort นั้นเป็นเพียงเท็กซ์ไฟล์ทำให้การเพิ่ม แก้ไข ลบ กฎดังกล่าวทำได้ไม่สะดวก โครงการพัฒนาระบบนี้จึงสร้างเครื่องมือในการจัดการกับกฎดังกล่าวให้สะดวกขึ้นด้วย Graphic User Interface (GUI)

1.2 เป้าหมายของโครงการพัฒนาระบบงาน

ระบบตรวจจับการบุกรุกด้วย Snort Rules ที่เหมาะสมใช้ซอฟต์แวร์ Snort ทำงานโดยการวิเคราะห์ content ของข้อมูลซึ่งจะเป็นการตรวจหา Signature ใน payload การทำงานแบบนี้คล้ายกับการทำงานของซอฟต์แวร์ป้องกันไวรัสคอมพิวเตอร์ ซึ่งจำเป็นที่ต้องกำหนด Signature ให้กับระบบตรวจจับการบุกรุก การเขียนกฎนั้นจำเป็นที่ต้องใช้ความละเอียด เพราะเราคงไม่ต้องการการแจ้งเตือนที่ไม่จำเป็นจากระบบตรวจจับการบุกรุก หรือ false alarm และก็ไม่ต้องการให้เกิดการตรวจจับที่ผิดพลาดเช่นเดียวกัน

ระบบตรวจจับการบุกรุกด้วย Snort Rules ที่เหมาะสมมีเป้าหมายในการทำงานดังนี้

- สร้างเครื่องมือสำหรับการจัดการกับ Snort โดยใช้กฎที่เหมาะสมกับระบบสารสนเทศที่มีอยู่ให้สามารถทำงานได้อย่างมีประสิทธิภาพมากขึ้น
- สามารถเพิ่ม แก้ไขรูลไฟล์และคอนฟิกไฟล์ (Rule & Configuration files)
- สามารถสร้างรูลไฟล์และคอนฟิกไฟล์เพื่อนำไปใช้กับระบบตรวจจับการบุกรุกด้วย Snort ได้
- ลดการแจ้งเตือนที่ไม่จำเป็นจากการตรวจจับการบุกรุกโดยใช้รูลไฟล์ที่ได้จากโครงการ

1.3 วัตถุประสงค์ของโครงการพัฒนาระบบงาน

- เพื่อศึกษาหลักการทำงานของระบบตรวจจับการบุกรุก ซอฟต์แวร์ Snort และส่วนที่เกี่ยวข้องกับโครงการพัฒนาระบบงาน
- เพื่อวิเคราะห์และออกแบบกฎที่เหมาะสมที่ใช้ในระบบตรวจจับการบุกรุกด้วยซอฟต์แวร์ Snort
- เพื่อลดการแจ้งเตือนที่ผิดพลาดของซอฟต์แวร์ Snort โดยใช้กฎที่เหมาะสม

1.4 ขอบเขตของการศึกษาและพัฒนาระบบงาน

การพัฒนาระบบตรวจจับการบุกรุกด้วย Snort Rules ที่เหมาะสม ถูกพัฒนาภายใต้ระบบปฏิบัติการ WINDOWS โดยแบ่งการทำงานออกเป็น 5 ส่วนหลักดังนี้

- ส่วนการสร้าง แก้ไข ลบ Snort IDS Sensor
- ส่วนการเพิ่ม แก้ไข กู้กลับ กฎการตรวจจับและคอนฟิกไฟล์ของ Snort
- ส่วนการสร้างรูลไฟล์และคอนฟิกไฟล์ให้เหมาะกับสารสนเทศที่มีอยู่
- ส่วนสั่งการให้ระบบตรวจจับการบุกรุกด้วย Snort Rules ที่เหมาะสมทำงานหรือหยุดทำงาน
- ส่วนการนำ Analysis Console for Intrusion Database (ACID) มาใช้งานร่วมกับระบบตรวจจับการบุกรุกด้วย Snort Rules ที่เหมาะสม

1.5 ประโยชน์ที่คาดว่าจะได้รับ

ผู้ดูแลระบบสามารถจัดการกับรูลไฟล์และคอนฟิกไฟล์ในการตรวจจับการบุกรุกได้อย่างมีประสิทธิภาพ เพราะเป็นการจัดการด้วย GUI ทำงานได้ง่ายกว่าการแก้ไขจากเท็กซ์ไฟล์ รวมทั้งลดการแจ้งเตือนที่ไม่จำเป็นจากระบบตรวจจับการบุกรุกหรือ false alarm และเพิ่มอัตราการตรวจจับการบุกรุกที่มีประสิทธิภาพมากขึ้น

1.6 ขั้นตอนการศึกษาและพัฒนาระบบ

ในการศึกษาและพัฒนาระบบนี้ได้มีการกำหนดขั้นตอนไว้ดังต่อไปนี้

- ศึกษาหลักการระบบตรวจจับการบุกรุกและการใช้งานซอฟต์แวร์ Snort
- ศึกษาปัญหาที่เกิดขึ้นจากการใช้กฎของ Snort
- ศึกษาแนวทางในการแก้ไขปัญหา และทฤษฎีต่างๆ ที่เกี่ยวข้อง
- ทำการวิเคราะห์ และออกแบบระบบตรวจจับการบุกรุกด้วย Snort Rules ที่เหมาะสม
- ศึกษาเครื่องมือที่นำมาใช้ในการพัฒนาระบบงาน
- พัฒนาระบบตรวจจับการบุกรุกด้วย Snort Rules ที่เหมาะสม
- ทดสอบการใช้งานโปรแกรม
- ปรับปรุงแก้ไขโปรแกรมที่พัฒนาแล้ว
- สรุปผลการทดสอบจากการใช้งาน
- จัดทำเอกสารประกอบโครงการ

1.7 รายละเอียดของแต่ละบท

- บทที่ 2 หลักการและความรู้ที่เกี่ยวข้องกับการพัฒนาระบบตรวจจับการบุกรุกด้วย Snort Rules ที่เหมาะสม ประกอบด้วย ความรู้เบื้องต้นเกี่ยวกับระบบตรวจจับการบุกรุก
- บทที่ 3 หลักการและการทำงานของซอฟต์แวร์ Snort ประกอบด้วย โหมดการทำงานของ Snort รูปแบบและวิธีการสร้างกฎของ Snort
- บทที่ 4 ออกแบบและพัฒนาระบบตรวจจับการบุกรุกด้วย Snort Rules ที่เหมาะสม ซึ่งประกอบด้วย ส่วนประกอบของกฎ รูปแบบของรูลไฟล์และคอนฟิกไฟล์ในระบบ ตัวอย่างคอนฟิกไฟล์และรูลไฟล์ในโครงการพัฒนาระบบ ออกแบบระบบงาน ฐานข้อมูลสำหรับรูลไฟล์และคอนฟิกไฟล์ พจนานุกรมข้อมูล ซอฟต์แวร์ที่เกี่ยวข้องในการพัฒนาระบบงาน การออกแบบหน้าจอการทำงานของระบบงาน และฟังก์ชันการทำงานในระบบตรวจจับการบุกรุกด้วย Snort Rules ที่เหมาะสม
- บทที่ 5 ออกแบบการทดสอบ ทดสอบการใช้งานระบบตรวจจับการบุกรุกด้วย Snort Rules ที่เหมาะสม ผลที่ได้จากการทดสอบ และการเปรียบเทียบผลการทดสอบของโครงการพัฒนาระบบงาน
- บทที่ 6 สรุปผลและข้อเสนอแนะโครงการพัฒนาระบบงาน

บทที่ 2
หลักการระบบตรวจจับการบุกรุก

เมื่อประตูสู่โลกการค้าบนอินเทอร์เน็ตได้เปิดขึ้น สิ่งที่ตามมาคือผู้ไม่ประสงค์ดีที่อาจสร้างความเสียหายให้กับระบบและข้อมูลขององค์กร ดังนั้นระบบรักษาความปลอดภัยจึงต้องมีความมั่นคงพร้อมรับมือกับการจู่โจมที่อาจเกิดขึ้น ไฟร์วอลล์เพียงอย่างเดียวมิได้ช่วยให้ระบบปลอดภัยร้อยเปอร์เซ็นต์ แต่ระบบรักษาความปลอดภัยที่ดีต้องป้องกันการโจมตีในทุกระดับ ดังนั้นการทำงานร่วมกับระบบตรวจจับการบุกรุก (IDS: Intrusion Detection System) จะช่วยเพิ่มความปลอดภัยอีกขั้นหนึ่ง ซึ่งระบบตรวจจับการบุกรุกจะทำการเฝ้าดู วิเคราะห์เหตุการณ์ต่างๆ ว่าเป็นการบุกรุกหรือมีความพยายามในการบุกรุกหรือไม่ โดยอาศัยค่าต่างๆ อาทิเช่น Network traffic, CPU Utilization, I/O Utilization หรือ File activities และเตือนภัยต่างๆ ที่เกิดขึ้นบนเครือข่ายคอมพิวเตอร์ (Koziol. 2003)

2.1 ความหมายของระบบตรวจจับการบุกรุก

ระบบตรวจจับการบุกรุก (Intrusion Detection System) หมายถึง ระบบที่ใช้ในการเฝ้าดู, วิเคราะห์ตรวจจับเหตุการณ์และเตือนภัยเมื่อมีผู้บุกรุกหรือ มีสิ่งผิดปกติที่จะเข้ามาในระบบ ซึ่งความพยายามในการใช้งานระบบของผู้บุกรุกนั้นขัดกับข้อบังคับและเจตุจำนงค์การใช้งานส่งผลต่อความปลอดภัยของระบบคอมพิวเตอร์หรือเครือข่ายคอมพิวเตอร์ 3 ประการคือ Integrity, Confidentiality, Availability (Whitman and Mattord. 2003)

2.2 ศาสตร์ของการบุกรุก

การป้องกันการบุกรุกได้อย่างมีประสิทธิภาพนั้นจำเป็นที่จะต้องทราบหลักการของการบุกรุกก่อน รูปที่ 2.1 ศาสตร์ของการบุกรุก (Anatomy of Hack) จากรูปพบว่าขั้นตอนที่สำคัญที่สุดของการบุกรุกคือขั้นตอน Escalating Privilege ซึ่งเป็นเป้าหมายหลักของผู้บุกรุกทั่วไป เพราะถึงแม้ผู้บุกรุกจะสามารถผ่านขั้นตอน Footprinting, Scanning, Enumeration และ Gaining access มาได้แล้วก็ยังไม่สามารถทำอันตรายกับระบบของเราได้ แต่ถ้าผู้บุกรุกสามารถได้อำนาจเทียบเท่ากับผู้ดูแลระบบ ผู้บุกรุกจะสามารถทำทุกสิ่งทุกอย่างกับระบบของเราได้ตามใจชอบ (มีข้อยกเว้นคือการทำ Denial of Service (DoS) ไม่ต้องผ่านขั้นตอน Escalating Privilege โดย DoS มักเป็นทางเลือก

สุดท้ายของผู้บุกรุกที่ใช้ในการทำลายระบบการให้บริการของเครื่องคอมพิวเตอร์เป้าหมาย) นอกจากนี้ ขั้นตอนอีก 2 ขั้นตอนที่ตามมาที่เรียกว่า Covering Tracks และ Creating Back Doors ก็มีความสำคัญมากเช่นกัน เพราะถ้าผู้ดูแลระบบรู้ตัวเร็วเท่าใด ก็สามารถทำการแก้ไขระบบให้กลับเป็นปกติได้ง่ายขึ้นเท่านั้น (Stevens. 1997)

รูปที่ 2.1 ศาสตร์ของการบุกรุก (Anatomy of Hack)

2.3 การแบ่งประเภทของผู้บุกรุก

Hacker และ Cracker เป็นคำที่ใช้แทนผู้บุกรุก ซึ่ง hacker คือบุคคลที่ชอบเจาะเข้าสู่ระบบต่างๆ hacker ที่ดีคือบุคคลที่พยายามที่จะเข้าสู่ระบบคอมพิวเตอร์ของตัวเอง เพื่อที่จะเข้าใจการทำงานของระบบ แต่ hacker ที่เป็นผู้ร้ายหรือ cracker คือบุคคลที่พยายามเจาะเข้าสู่ระบบของผู้อื่นโดยมีเจตนาเพื่อทำร้ายระบบให้เกิดความเสียหายสำหรับบุคคลหรือสิ่งอื่นใดก็ตามที่เข้าสู่ระบบโดย

ไม่ได้รับอนุญาต แล้วกระทำการใดๆที่อาจก่อให้เกิดความเสียหายแก่ระบบ เราจะเรียกว่า "ผู้บุกรุก" ซึ่งผู้บุกรุกจะถูกแบ่งเป็น 2 ประเภทคือ

2.3.1 Outsides หมายถึง ผู้บุกรุกจากภายนอกเครือข่ายและบุคคลที่อาจจะโจมตีมาจากภายนอก เช่น การเปลี่ยนแปลงหน้าตาของ web server หรือการส่งต่อเมล์ผ่านทาง e-mail server ซึ่งการบุกรุกจากภายนอกอาจมาจากอินเทอร์เน็ต, การหมุนโมเด็ม, การบุกรุกเข้าไป, เครือข่ายของคู่ค้าที่เชื่อมต่อมายังเครือข่าย

2.3.2 Insider หมายถึง ผู้บุกรุกที่มีสิทธิ์ในการใช้เครือข่ายภายใน รวมทั้งผู้ใช้ที่ใช้สิทธิ์ในทางที่ผิด หรือการลักลอบใช้สิทธิ์ของผู้ใช้คนอื่นๆ ที่มีสิทธิ์เหนือกว่า (SANS Institute. 2002)

2.4 สาเหตุที่ผู้บุกรุกสามารถเข้าสู่ระบบ

2.4.1 ข้อผิดพลาดของซอฟต์แวร์ ปรากฏอยู่ใน server daemons, โปรแกรมต่างๆ และระบบปฏิบัติการ ตัวอย่างของ Software bugs เช่น Buffer Overflows, Input Invalid เป็นต้น

2.4.2 System Configuration สามารถจำแนกได้ดังต่อไปนี้

- Default Configurations ระบบส่วนใหญ่จะถูกจัดส่งจากผู้ขายด้วย default configuration ซึ่งง่ายในการใช้ แต่นั้นก็หมายถึงง่ายในการเจาะระบบด้วย
- ความไม่เอาใจใส่ของผู้ดูแลระบบ มีระบบอยู่ไม่น้อยที่ถูกติดตั้งมาให้ไม่ต้องใส่รหัสผ่านของ root ซึ่งจุดนี้อาจทำให้ผู้บุกรุกเจาะเข้าสู่ระบบได้

2.4.3 Password Cracking

- การเลือกใช้รหัสผ่านที่ค่อนข้างอ่อนแอ โดยทั่วไปคนส่วนใหญ่จะเลือกใช้รหัสผ่านที่เป็นชื่อของตนเอง, ชื่อลูก, ชื่อสามี/ภรรยา, สัตว์เลี้ยง, รุ่นของรถ หรืออาจไม่ใส่เลย จะเห็นว่าผู้บุกรุกสามารถทำการเดาสุ่มรหัสผ่านเหล่านี้ได้ง่าย
- Dictionary Attacks ผู้บุกรุกอาจจะใช้โปรแกรมที่ทำการ crack รหัสผ่านโดยที่โปรแกรมดังกล่าวจะเลือกคำที่เป็นไปได้ในพจนานุกรมแล้วเปรียบเทียบกับคำในพจนานุกรมที่ถูกเข้ารหัสไว้
- Brute force attacks จะคล้ายกับ dictionary attacks ผู้บุกรุกจะพยายามนำอักขระต่างๆมาผสมกันเป็นรหัสผ่าน เช่นรหัสผ่านที่มีอักษร 3 ตัว และเป็นตัวพิมพ์เล็กเพียงอย่างเดียว อาจจะถูก crack ภายในเวลาแค่ไม่กี่นาที แต่ถ้าเป็นรหัสผ่านความยาว 8 ตัวอักษร ซึ่งมีทั้งตัวพิมพ์เล็กและตัวพิมพ์ใหญ่อาจต้องใช้เวลาหลายเดือนในการ crack

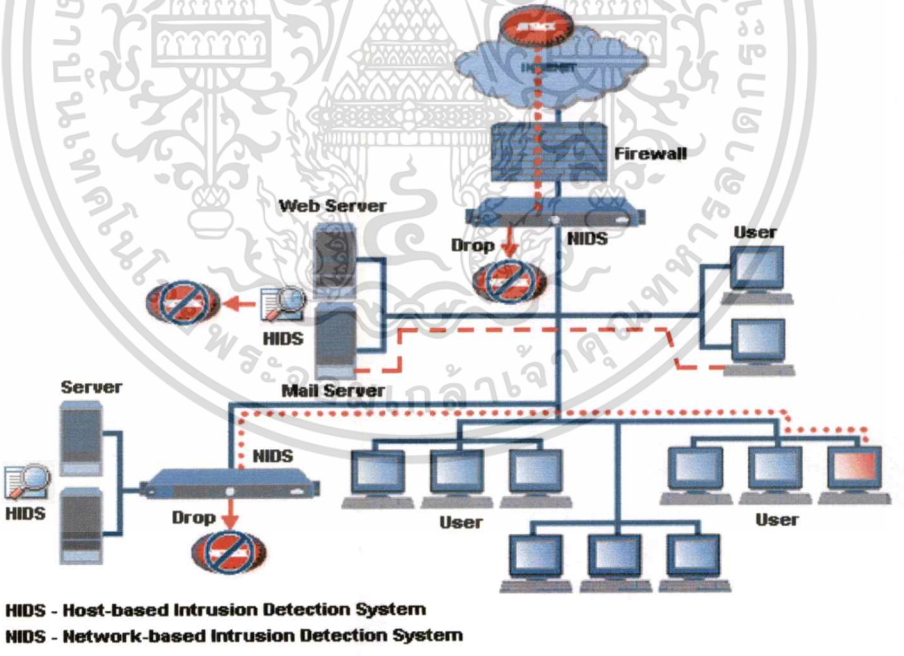
2.4.4 ข้อบกพร่องของการออกแบบ ถึงแม้ว่าการใช้ซอฟต์แวร์จะถูกต้องตามที่ออกแบบไว้ แต่อาจจะมีข้อผิดพลาดได้ซึ่งจะนำไปสู่การเจาะระบบได้ เช่น TCP/IP เนื่องจาก TCP/IP ได้ถูกออกมาก่อนที่วิวัฒนาการทางด้านการบุกรุกระบบจะเหมือนในปัจจุบัน เพราะฉะนั้นเป็นไปได้ที่

อาจจะมีข้อผิดพลาดทางด้านการออกแบบที่จะนำไปสู่ปัญหาทางด้านความปลอดภัยได้ ตัวอย่างเช่น IP spoofing หรือ SYN floods ได้มีการพัฒนา Internet Protocol Security (IPSec) เพื่อที่จะแก้ปัญหาต่างๆ เหล่านี้ (Amoroso. 1999)

2.5 ประเภทของระบบตรวจจับการบุกรุก

2.5.1 Host based Intrusion Detection System (HIDS)

HIDS เป็นระบบตรวจจับการบุกรุกประเภทแรกที่มีการพัฒนาและใช้งาน ซึ่งระบบจะทำการตรวจสอบผู้บุกรุกเฉพาะเครื่องที่ได้ลงระบบ IDS เอาไว้ โดยระบบนี้จะทำการตรวจจับข้อมูลที่เข้าและออกในคอมพิวเตอร์ และยังตรวจสอบความสมบูรณ์ของ system files รวมทั้งเฝ้าดูกิจกรรมที่น่าสงสัย แสดงดังรูปที่ 2.2 ข้อมูลที่ HIDS ใช้ในการตรวจจับการบุกรุกเช่น Windows NT/2000 Security Event Logs, RDBMS audit, UNIX Syslog เป็นต้น ข้อดีคือการทำงานจะรวดเร็วเนื่องจากสนใจแต่ข้อมูลที่เกี่ยวกับเครื่องๆเดียว แต่มีข้อเสียในเรื่องการจัดการเช่นเมื่อต้องการเปลี่ยนแปลง configuration ก็ต้องทำการเปลี่ยนแปลงทุกเครื่องที่ลงระบบนี้ (Proctor. 2000)



รูปที่ 2.2 Host-based IDS และ Network-based IDS

2.5.2 Network based Intrusion Detection System (NIDS)

ระบบการตรวจจับการบุกรุกแบบ Network based นั้นจะทำการเฝ้าดูข้อมูลบนเครือข่าย โดยทำการรับข้อมูลทั้งหมดที่อยู่บนส่วนของเครือข่ายที่รับผิดชอบ หากนอกเหนือจากส่วนของ

เครือข่ายที่รับผิดชอบ และชนิดของการสื่อสารอื่นๆ แล้วระบบดังกล่าวก็ไม่สามารถทำการตรวจจับ packet ต่างๆได้ sensor ของระบบ IDS จะมองเห็นเฉพาะ packet ที่ผ่านส่วนของเครือข่ายที่ sensor นั้นติดอยู่ packet ต่างๆ จะเป็นที่สนใจของ sensor ก็ต่อเมื่อ packet นั้นเข้ากับ signature ที่กำหนดซึ่งปกติแล้ว signature จะมี 3 ประเภทคือ

2.5.2.1 String signatures จะมองหา text string ซึ่งอาจบ่งบอกถึงการโจมตี

2.5.2.2 Port signatures จะเฝ้าดูการพยายามติดต่อเข้ามาทาง port ที่รู้จักกันดี และมักจะถูกโจมตี เช่น telnet จะใช้ TCP port 23, FTP จะใช้ TCP port 21/20 และ IMAP จะใช้ TCP port 143 ซึ่งถ้าระบบไม่ได้เปิด port ดังกล่าว แต่มีการพยายามเชื่อมต่อเข้ามาแสดงว่า packet ดังกล่าวอาจประสงค์ร้ายก็ได้

2.5.2.3 Header condition signatures พยายามมองหา combination ที่อันตรายและผิดกฎของ packet header ตัวอย่างที่เห็นได้ชัดของ header signature คือ TCP packet ซึ่งมีทั้ง SYN และ FIN Flags (Innella and McMillan. 2002)

2.6 กลวิธีในการตรวจจับการบุกรุก

2.6.1 การตรวจจับพฤติกรรมที่ผิดปกติ (Anomaly Detection)

ใช้การพิจารณาจากสถิติต่างๆ เช่น CPU Utilization การใช้ดิสก์ การเข้าใช้ระบบของผู้ใช้ การใช้ไฟล์ โดยค่าทางสถิติเหล่านี้จะมีค่า Threshold เพื่อเป็นตัวเปรียบเทียบว่าเหตุการณ์นั้นมีพฤติกรรมที่ผิดปกติหรือมีการบุกรุกหรือไม่ ประโยชน์ของการใช้วิธีนี้ คือสามารถจับความผิดปกติต่างๆ โดยที่ไม่ต้องรู้สาเหตุของความผิดปกตินั้น และสามารถสร้างรูปแบบการบุกรุกที่เกิดขึ้นใหม่มาใช้กับการตรวจจับรูปแบบของการโจมตี (Bace. 1998)

- การตรวจจับพฤติกรรมที่ผิดปกติมีข้อดีดังนี้
  - สามารถตรวจจับพฤติกรรมที่ผิดปกติที่ไม่เคยเกิดขึ้นมาก่อนได้
  - สามารถสร้างรูปแบบการบุกรุกที่เกิดขึ้นใหม่แล้วนำกลับมาใช้กับการตรวจจับการใช้ที่ผิด
  - สถิติในการจัดเก็บง่ายต่อการเข้าใจ
- การตรวจจับพฤติกรรมที่ผิดปกติมีข้อเสียดังนี้
  - การกำหนดค่า Threshold ที่ถูกต้องทำได้ยาก อาจทำให้ระบบตัดสินใจผิดพลาดได้ เป็นเหตุให้ผู้ใช้ที่มีเจตนาดีถูกมองเป็นผู้บุกรุกได้

2.6.2 การตรวจจับรูปแบบของการโจมตี (Signature Detection)

กลวิธีการตรวจจับรูปแบบของการโจมตี หรือเรียกอีกอย่างหนึ่งว่า การตรวจจับการใช้ที่ผิด (Misuse Detection) ใช้วิธีการตรวจสอบหรือศึกษารูปแบบการโจมตี ที่เป็นที่รู้จักกันดีหมายความว่าเทคนิคต่างๆที่ผู้บุกรุกใช้ก็จะถูกบันทึกเป็นกฎเข้าสู่ระบบเพื่อทำการตรวจจับการบุกรุกต่อไป

ตัวอย่างของ HIDS ที่ใช้เช่น การเข้าสู่ระบบผิดพลาด 3 ครั้ง ส่วนตัวอย่างของ NIDS ที่ใช้เช่นการตรวจดูภายใน packet ว่าประกอบด้วยรูปแบบที่อาจแสดงถึงการพยายามเข้าสู่ระบบ เช่นถ้า packet ประกอบด้วย "cgi-bin/phf?" หมายถึงการพยายามเข้าถึง CGI script ที่อยู่บน web server ประโยชน์ของการใช้วิธีนี้คือ เร็ว เชื่อถือได้และมีประสิทธิภาพมาก แต่มีข้อเสียคือต้องมีการปรับปรุงรูปแบบของการป้องกันการบุกรุกให้ทันสมัยอยู่ตลอด เพื่อป้องกันรูปแบบการบุกรุกใหม่ๆที่เข้าสู่ระบบ (Bace. 1998)

- การตรวจจับพฤติกรรมการใช้ที่ผิดมีข้อดีดังนี้
  - มีประสิทธิภาพมากต่อการตรวจจับโดยไม่มีการเตือนภัยที่ผิดพลาด
  - การทำงานไม่จำเป็นต้องใช้การคำนวณที่ซับซ้อนเพื่อเก็บข้อมูลทางสถิติ
  - เชื่อถือได้
- การตรวจจับพฤติกรรมการใช้ที่ผิดมีข้อเสียดังนี้
  - การทำงานตามกฎนั้น จำเป็นต้องนำเหตุการณ์มาเปรียบเทียบกับกฎทุกกฎ ซึ่งถ้ามีกฎมากทำให้การทำงานช้าได้
  - กฎต่างๆ ส่วนใหญ่ถูกพบเมื่อมีเหตุการณ์บุกรุกเกิดขึ้นแล้ว ถ้ามีการบุกรุกด้วยวิธีใหม่ๆ ระบบจะไม่สามารถตรวจพบได้

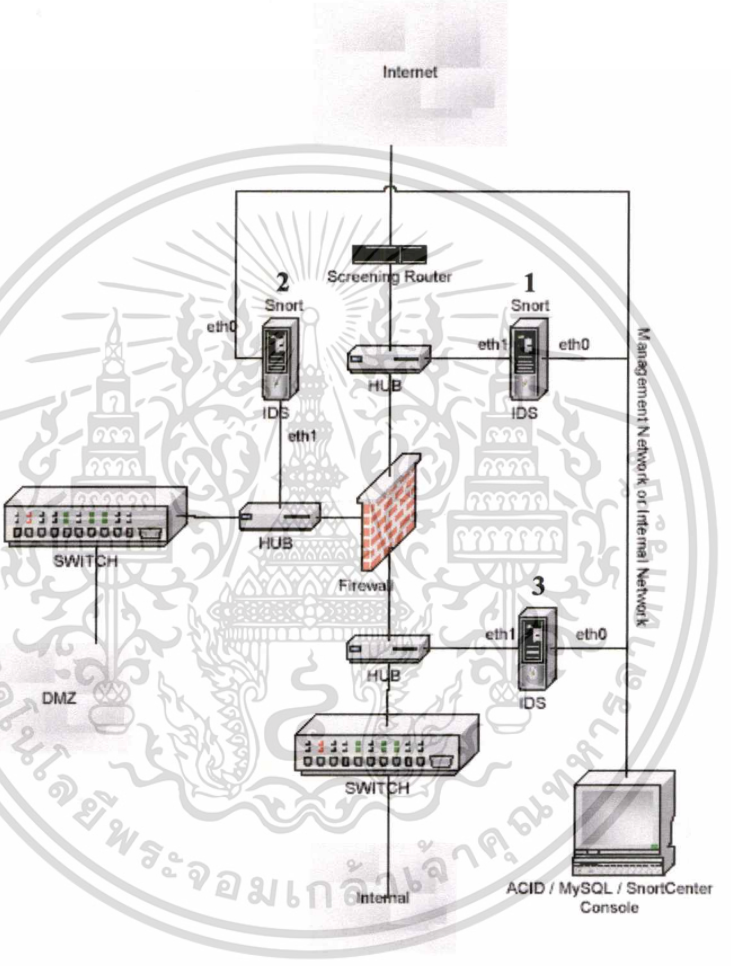
2.7 การวางระบบตรวจจับการบุกรุกบนเครือข่าย

จากรูปที่ 2.3 แสดงตำแหน่งที่เป็นที่นิยมในการวางระบบตรวจจับการบุกรุก โดยแบ่งเครือข่ายออกเป็น 3 ส่วนดังนี้

1. เป็นส่วนที่มีความเสี่ยงสูงที่สุดเพราะเป็นปราการด้านแรกที่จะถูกโจมตีจากการบุกรุก โดยที่ระบบตรวจจับการบุกรุกจะถูกปรับแต่งให้มีความไวต่อการตรวจจับการโจมตีมากที่สุด เพราะว่าระบบเครือข่ายคอมพิวเตอร์ในส่วนนี้จะมีข้อมูลไหลผ่านมากที่สุดและเป็นส่วนที่มีการแจ้งเตือนมากที่สุด

2. เป็นส่วนที่ระบบตรวจจับการบุกรุกจะถูกปรับแต่งให้มีความไวต่อการตรวจจับการโจมตีน้อยกว่าส่วนที่ 1 เพราะว่าเครือข่ายในส่วนนี้จะทำงานอยู่หลังไฟร์วอลล์ที่ได้ถูกปรับแต่งให้เหมาะสมแล้ว ผู้ที่จะผ่านเข้ามาได้ต้องเป็นผู้ที่ได้รับอนุญาตเท่านั้น

3. เป็นส่วนของระบบเครือข่ายภายใน ซึ่งต้องคำนึงถึงการโจมตีที่อาจเกิดจากผู้บุกรุกเครือข่ายภายใน เช่น คนในองค์กร หรือหุ้นส่วนทางธุรกิจที่อนุญาตให้สามารถดูข้อมูลบางอย่างได้ เป็นต้น ในส่วนนี้จะมีการแจ้งเตือนถึงการบุกรุกน้อยมากถ้าระบบเครือข่ายมีความมั่นคงเพียงพอ (Rehman. 2003)



รูปที่ 2.3 การวางระบบตรวจจับการบุกรุกบนเครือข่าย

บทที่ 3

ระบบตรวจจับการบุกรุกโดยใช้ Snort

ในช่วงหลายปีที่ผ่านมาได้มีบริษัทต่างๆ พยายามพัฒนาซอฟต์แวร์ซึ่งจะทำการตรวจจับการบุกรุก แต่เนื่องจากราคาที่ค่อนข้างสูงของซอฟต์แวร์ดังกล่าว ดังนั้น Open source community จึงได้พัฒนาซอฟต์แวร์ที่มีราคาถูกกว่า ในปัจจุบันมีการใช้ระบบตรวจจับการบุกรุกที่เป็น open source software อย่างจริงจัง ซึ่งซอฟต์แวร์ดังกล่าวสามารถดาวน์โหลดได้จากอินเทอร์เน็ตและต้องการการลงทุนด้านฮาร์ดแวร์เพียงเล็กน้อยเท่านั้น

ในการเลือกใช้เครื่องมือที่เหมาะสมจำเป็นที่จะต้องมีการทราบรายละเอียดเกี่ยวกับวิธีการของการตรวจจับการบุกรุก ซึ่งมี 2 วิธีคือ Content Analysis และ Traffic Analysis

- **Content Analysis**

จะต้องมีการ capture packets ทั้งหมด ซึ่งโดยปกติแล้วขนาดของ Ethernet Frames สามารถมีขนาดได้ถึง 1500 bytes เพราะฉะนั้นจำเป็นต้องมี disk space และ CPU time ในการจัดการข้อมูลดังกล่าว ข้อดีของการวิเคราะห์แบบนี้คือ ง่าย รวดเร็วกว่า และเป็นการตรวจจับแบบ real-time มากกว่าแต่ข้อเสียคือ มีโอกาสของความผิดพลาดในการแจ้งเตือนสูงกว่าและต้องการทรัพยากรของระบบในการทำงานมากกว่า

- **Traffic Analysis**

เป็นการแปลความหมายจาก patterns ใน packet header ซึ่งจะแสดงถึงความผิดปกติของเครือข่ายเพราะฉะนั้นจึงมีความจำเป็นที่ผู้วิเคราะห์จะต้องมีความรู้และทักษะในการแปลความหมายจากข้อมูลดังกล่าวเนื่องจากการวิเคราะห์จะดูเฉพาะส่วนที่เป็น header ฉะนั้นจึงมีการ capture เฉพาะ header ของข้อมูล โดยปกติแล้ว header ที่จะต้อง capture จะมีขนาดประมาณ 68 ไบต์ และหากต้องการความถูกต้องในการวิเคราะห์จึงจำเป็นต้องมีการ capture ทุกๆ header ที่ผ่านในสายสื่อสาร ข้อดีของ traffic analysis คือ ความถูกต้องของการแปลความหมายของข้อมูล แต่ข้อเสียคือผู้วิเคราะห์จะต้องผ่านการฝึกฝนมาเป็นอย่างดี และการทำงานไม่สามารถเป็นแบบ real time ได้

ระบบตรวจจับการบุกรุกส่วนใหญ่จะใช้ content analysis เนื่องจากผู้ดูแลระบบเครือข่ายคงไม่มีเวลาในการคอยตรวจเช็คข้อมูลซึ่งมีขนาดมหาศาลทั้งหมดได้ ดังนั้นการวิเคราะห์ content ของข้อมูลจะเป็นการตรวจหา signature ใน payload การทำงานแบบนี้คล้ายกับการทำงานของ

ซอฟต์แวร์ป้องกันไวรัสคอมพิวเตอร์ ซึ่งจำเป็นที่ต้องกำหนด signature หรือกฎ (rules) ให้กับระบบตรวจจับการบุกรุก การเขียนกฎนั้นจำเป็นที่ต้องใช้ความละเอียด เพราะเราคงไม่ต้องการการแจ้งเตือนที่ไม่จำเป็นจากระบบตรวจจับการบุกรุก หรือ false alarm และก็ไม่ต้องการให้เกิดการตรวจจับที่ผิดพลาดเช่นเดียวกัน (Caswell. 2002)

การใช้ Snort เป็นระบบตรวจจับการบุกรุก (IDS) นั้นมีมาตั้งแต่ปี 1998 และในปัจจุบันได้รับความนิยมอย่างมาก ซึ่ง snort เป็น open source, rule-based และ content analysis ซึ่งเขียนโดยใช้ภาษา C เป็น stand-alone program

Snort เป็นเครื่องมือที่ใช้ตรวจจับการบุกรุกทางเครือข่าย (network intrusion detection) โดย Martin Roesch การทำงานของ Snort จะใช้ไลบรารี (library) พื้นฐานชื่อ WinPCAP ซึ่งใช้กันโดยทั่วไปในบรรดา network sniffer และ network analyzer ทั้งหลาย สำหรับ Snort นั้นสามารถทำ protocol analysis, content searching/matching, ตรวจจับการบุกรุกและ probe เช่น buffer overflow, stealth port scan, CGI attack และอื่นๆ (Harper. 2003)

3.1 โหมดการทำงานของ Snort

การทำงานของ Snort จะมี 3 โหมดการทำงานดังนี้

3.1.1 Sniff Mode เป็นการดักข้อมูลบนเครือข่ายแล้วนำมาแสดงบนหน้าจอว่ามีข้อมูลอะไรวิ่งอยู่บนเครือข่ายบ้าง

3.1.2 Packet Logger Mode เป็นการบันทึกข้อมูลจากการดักข้อมูลที่วิ่งบนเครือข่ายลงดิสก์ไว้

3.1.3 Network Intrusion System Mode เป็นการวิเคราะห์ข้อมูลบนเครือข่ายเพื่อตรวจหาว่าตรงกับรูปแบบการโจมตี ซึ่งใช้กฎ (Rule) ในการตัดสินใจว่าข้อมูลนั้นเป็นการโจมตีหรือไม่ โดยปกติกฎจะถูกเก็บไว้ในไฟล์ snort.conf (Roesch and Green. 2002)

3.2 การสร้างกฎการตรวจจับผู้บุกรุก (Snort Rules)

ส่วนใหญ่ Snort rules ในเวอร์ชัน 1.8 จะถูกเขียนใน 1 บรรทัด แต่ในเวอร์ชันที่สูงขึ้นสามารถเขียนได้หลายบรรทัดโดยเพิ่ม \ (backslash) ในตอนท้ายของแต่ละบรรทัด Snort rules ถูกแบ่งออกเป็น 2 ส่วนคือ

3.2.1 Rule header ประกอบด้วย

3.2.1.1 Rule actions เป็นรายการแรกใน Rule header ซึ่งจะบอก Snort ว่าต้องทำอะไรเมื่อตรวจพบ packet ที่ตรงกับเงื่อนไขที่กฎระบุไว้ โดยมีการตั้งค่ากระทำไว้ 5 รูปแบบคือ Alert, Log, Pass, Activate และ Dynamic

- Alert สร้างการแจ้งเตือนโดยเลือกวิธีการเตือน

- Log ทำการ log packet

- Pass ละเลยไม่สนใจ packet

- Activate แจ้งเตือนและเรียกใช้ dynamic rule อื่นๆ

- Dynamic ไม่ต้องทำอะไรจนกว่าจะถูกเรียกโดย Activate

ruletype redalert

{

typealert output alert_syslog: LOG_AUTH LOG_ALERT \

output database: log, mysql, user=snort dbname=snort host=localhost

} (การสร้างกฎซึ่งจะเก็บ Log ลง Syslog และ MySQL)

3.2.1.2 Protocol เป็นฟิลด์ถัดมาของ Rule โดยปัจจุบัน Snort สามารถวิเคราะห์ได้ 4 โพรโทคอล คือ TCP, UDP, ICMP และ IP ในอนาคตอาจสามารถวิเคราะห์โพรโทคอลได้มากขึ้น เช่น ARP, IGRP, GRE, OSPF, RIP, IPX เป็นต้น

3.2.1.3 IP addresses และ Netmasks เป็นข้อมูล IP address ที่ให้กับกฎ โดยคำว่า any หมายถึง IP address ใดๆ

3.2.1.4 Port Number จะระบุถึงหมายเลขเส้นทางที่ใช้สื่อสารโดยสามารถระบุแบบเจาะจงหรือช่วงของหมายเลขพอร์ตได้ เช่น 111 สำหรับ portmapper, 23 สำหรับ telnet, 80 สำหรับ http เป็นต้น และหากต้องการระบุทุกพอร์ตก็ใช้ wildcard แทน

log udp any any -> 192.168.1.0/24 1:1024 log udp

(ข้อมูลมาจากพอร์ตใดๆ แต่พอร์ตปลายทางต้องอยู่ในช่วงหมายเลข 1 ถึง 1024 เท่านั้น)

log tcp any any -> 192.168.1.0/24 :6000

(Log TCP traffic จากพอร์ตใดๆที่จะไปยังพอร์ตที่น้อยกว่าหรือเท่ากับ 6000)

3.2.1.5 Direction Operation คือส่วนที่บอกทิศทางของการติดต่อสื่อสาร ได้แก่

-> หมายถึงพิจารณาเพียงทิศทางเดียว ฝั่งซ้ายของเครื่องหมายคือ ต้นทาง ส่วนฝั่งขวาคือ ปลายทาง

<> หมายถึงพิจารณาทั้งสองทิศทาง

3.2.2 Rule options ประกอบด้วยข้อความเตือนและข้อมูลที่เป็นส่วนหนึ่งของ packet เพื่อตรวจสอบว่ากฎนั้นได้มีการปฏิบัติหรือไม่ ซึ่งถือเป็นหัวใจของกลไกการตรวจจับการบุกรุกของ Snort ทุกๆ Rule options ถูกแยกจากกันโดยใช้ ; (semicolon) และ Rule option keyword ถูกแยกจากนิพจน์อื่นด้วย : (colon) ตัวอย่างของ Keyword เช่น

- **msg** พิมพ์ข้อความแจ้งเตือน
- **logto** บันทึก packet ไปยังไฟล์ที่ระบุไว้
- **ttl** ตรวจสอบค่า time-to-live ของ packet
- **seq** ทดสอบฟิลด์ TCP sequence number โดยระบุค่า
- **ack** ทดสอบฟิลด์ TCP acknowledgement โดยระบุค่า
- **tos** ทดสอบฟิลด์ TOS ของ packet
- **id** ตรวจสอบค่าในฟิลด์ fragment ID ของ IP header
- **ip_proto** ค่าส่วนหัวของ IP โพรโทคอล
- **priority** เป็นการบอกระดับความสำคัญของกฎซึ่งค่ายิ่งน้อยยิ่งสำคัญมาก
- **content** เป็นการค้นหาในส่วนของ packet payload ว่ามีข้อมูลตรงกับที่ระบุไว้หรือไม่ ซึ่งในส่วน content string สามารถระบุค่าที่เป็น text รวมกับ binary ได้โดยค่าที่เป็น binary จะปิดหัวท้ายด้วยเครื่องหมาย "|" และใช้เครื่องหมาย "!" อยู่หลัง ":" หมายถึงการค้นหาว่าไม่มีข้อความนี้ในส่วน payload โดยแสดงตัวอย่างดังรูปที่ 3.1

```
alert tcp any any -> 192.168.1.0/24 143 (content:"|90C8 C0FF FFFF|/bin/ \
sh";msg:"IMAP buffer overflow!";)
```

รูปที่ 3.1 ตัวอย่างของ Snort Rule ที่ใช้ content

- **content-list** เป็นการค้นหาในส่วน packet payload โดยใช้ข้อมูลในไฟล์ซึ่งบรรจุคำที่ใช้ในการค้นหา
- **reference** อ้างถึง IDS ภายนอกดังแสดงในตารางที่3.1โดยสามารถเข้าไปอ่านรายละเอียดของกฎ

ตารางที่ 3.1 การอ้างถึง IDS ภายนอกที่สนับสนุน

System	URL Prefix
McAfee	http://vil.nai.com/vil/dispVirus.asp?virus_k=
Bugtraq	http://www.securityfocus.com/bid/
CVE	http://cve.mitre.org/cgi-bin/cvename.cgi ?name=

```
alert tcp any any -> 192.168.1.0/24 111 (content:"|00 01 86 a5|"; \
msg:"mountd access";)
```

รูปที่ 3.2 ตัวอย่างของ Snort Rule

จากรูปที่ 3.2 ข้อความก่อนถึงวงเล็บจะเป็น Rule header และข้อความในวงเล็บจะเรียกว่า rule options โดยคำที่อยู่ก่อนเครื่องหมายจุดคู่ (colons) ในส่วนของ rule options เรียกว่า option keywords โดย rule options ไม่จำเป็นต้องมีอยู่ในแต่ละกฎก็ได้ แต่ถ้ามีก็จะเป็นประโยชน์ในการระบุ packets เพื่อรวบรวม, แจ้งเตือนหรือไม่ต้องสนใจ ทุกส่วนประกอบที่จะสร้างเป็นกฎต้องเป็นจริงเพื่อให้กฎนั้นได้รับการปฏิบัติ

ใน rule หนึ่งๆ ของ Snort นั้นจะมี action ให้เลือก 3 ชนิดคือ log, alert, pass ถ้าเลือกเป็น log ข้อมูลจะถูกเก็บลงล็อกไฟล์ (ถ้าไม่ระบุเป็นพิเศษ จะเก็บไว้ที่ /var/log/snort) และถ้าเลือกเป็น alert และทำงานใน daemon mode ข้อมูลนั้นๆ จะถูก alert ผ่านทางช่องทาง alert ที่กำหนดไว้ เช่น ผ่านทาง syslog แต่โดยปกติแล้ว จะถูกเก็บไว้ที่ /var/log/snort/alert และกรณีสุดท้ายถ้าเลือกเป็น pass นั้น packet นั้นจะถูกปล่อยทิ้งไป

Snort มีการแจ้งเตือนเมื่อตรวจพบการบุกรุก โดยสามารถกำหนดในไฟล์ของกฎดังนี้

- **Alert_syslog** ระบบแจ้งเตือนไปยัง syslog ของระบบซึ่งสามารถใช้โปรแกรม logcheck เป็นตัวที่จะส่ง email ให้ผู้ดูแลระบบได้
- **Alert_full** เก็บรายละเอียดของส่วนหัวของ packet ทั้งหมด
- **Alert_fast** เก็บรายละเอียดข้อความและทำการบันทึกใน 1 บรรทัด ซึ่งจะเร็วกว่า Alert_full
- **Alert_smb** ระบบส่งข้อความเตือนไปยัง WinPopUp ผ่านโพรโทคอล SMB ซึ่งในเครื่องที่ได้รับจะ popup ข้อความขึ้นมา
- **Alert_unixsock** ระบบส่งข้อความเตือนไปยัง unix socket ซึ่งสามารถรันโปรแกรมอื่นที่รอรับอยู่ที่ socket นี้ได้ทันที
- **Log_tcpdump** จะเก็บบันทึก packet บนเครือข่ายในรูปแบบของ tcpdump
- **Database** สามารถบันทึก log ไปยังฐานข้อมูล SQL ได้ซึ่งฐานข้อมูลที่สนับสนุนได้แก่ MySQL, PostgreSQL, Oracle และ unixODBC-compliant database
- **SNMP Trap** ระบบจะส่งข้อความเตือนไปยัง Network Management Station (NMS)

- **XML** ให้ XML plug-in เก็บ log ในรูปของ SNML (Simple Network Markup Language หรือ Snort Markup Language) ซึ่งสามารถส่งรายงานไปยังฐานข้อมูลหรือ Snort ตัวอื่นได้
- **Unified** ถูกออกแบบให้มีความเร็วมากที่สุดในการบันทึก log ของ Snort โดยบันทึกไว้ 2 ไฟล์คือ ไฟล์ Alert และไฟล์ Log (Roesch and Green. 2002)

Sensor: Coco23	**IP:** 127.2.44.2	**Mask:** 255.255.255.0	**GW:** 127.2.44.1
Network Placement: Internet / Pre-Firewall / (External)		**Source Address Category:** External Internet Address	
Destination Address Category: Proxy (10.77.3.4)			
Relationship to other sensors: Momo44 – To find the real destination address correlate events with Momo44 sensor.			
Contact:			
Comments:			
Allowable Protocols			
Source Address	**Direction (→ or ←)**	**Destination**	**Protocol**
Any	→	10.77.3.4	FTP
Any	←	10.77.0.0/16	HTTP
Public Servers			
Source Address	**Running Services**	**Contact**	
10.77.3.4	FTP	Jimmy John (444)-555-1111	

รูปที่ 3.3 เอกสารคุณสมบัติของ Sensor แต่ละจุดบนเครือข่าย

ตารางที่ 3.2 ฟิลด์ต่างๆของ Sensor

ชื่อฟิลด์	คำอธิบาย
Sensor	ชื่อของ Sensor
IP	IP address ของ Sensor
Mask	Subnet mask ของ Sensor
GW	Default Gateway ของ Sensor
Network Placement	Internet / Pre-Firewall / (ภายนอก) , Internet / Post-Firewall / (ภายใน) Extranet / Post-Firewall / (ภายใน)
Source Address	External Internet Address, Internal Address, Extranet Address, Proxy, Firewall
Destination Address	External Internet Address, Internal Address, Extranet Address, Proxy, Firewall
Relationship to other sensors	แสดงความสัมพันธ์ระหว่าง Sensors เช่น Sensor ก่อนและหลัง proxy ถ้ามีการแจ้งเตือนของ IDS หลัง proxy หากต้องการทราบ IP address ของต้นทาง จำเป็นต้องอ้างถึง Sensor ก่อน proxy
Comments	คำอธิบายเพิ่มเติม, คำอธิบายพิเศษ
Contact	ข้อมูลบุคคลที่ติดต่อ
Allowed Protocol	โพรโทคอลที่อนุญาตให้ข้ามผ่านได้
Public Servers	เครื่องแม่ข่ายที่เข้าถึงจากที่สาธารณะได้

Sensor ในแต่ละจุดที่วางบนเครือข่ายสามารถแสดงเป็นเอกสารได้ดังรูปที่ 3.3 ซึ่งแสดงถึงคุณสมบัติของ Sensor โดยจุดประสงค์ของเอกสารคุณสมบัติของ Sensor เพื่อให้เข้าใจถึงข้อมูลที่ไหลผ่านกันตรงตำแหน่งที่ Sensor วางอยู่ ซึ่งสามารถใช้ข้อมูลเหล่านี้คัดแยก false positives, ปรับแต่ง Sensors และตรวจจับพฤติกรรมของ packet ที่ผิดปกติ (Scott. 2003) ตารางที่ 3.2 แสดงรูปแบบที่ใช้อย่างแพร่หลายของฟิลด์ต่างๆ ในแต่ละ Sensor

3.3 ปัญหาของระบบตรวจจับการบุกรุกด้วย Snort

ปัญหาของระบบตรวจจับการบุกรุกด้วย Snort ปัญหาแรกคือกฎการตรวจจับของ Snort ซึ่งเป็นหัวใจหลักของระบบตรวจจับการบุกรุกด้วย Snort ด้วยเหตุที่ Snort มีกฎเป็นจำนวนมาก Snort จึงเกิดการตรวจจับที่ผิดพลาดได้มากเช่นเดียวกัน บางครั้งกฎบางข้อไม่ได้ใช้งานเพราะไม่มีการใช้งานระบบสารสนเทศนั้น เช่น องค์กรที่ไม่มีการใช้งานซอฟต์แวร์ Oracle ก็ไม่มีความจำเป็นที่ต้องเอาข้อมูลที่ต้องการตรวจจับมาเปรียบเทียบกับกฎที่เกี่ยวกับ Oracle ดังนั้นจึงไม่มีความจำเป็นที่จะต้องโหลดกฎที่เกี่ยวกับ Oracle ใน Snort การสร้างระบบตรวจจับการบุกรุกโดยใช้ Snort Rule ให้เหมาะกับระบบสารสนเทศในแต่ละองค์กร จึงเป็นสิ่งจำเป็นซึ่งแต่ละองค์กรอาจมีระบบสารสนเทศที่แตกต่างกันไป อีกปัญหาที่พบคือปัจจุบันกฎของ Snort และคอนฟิกไฟล์นั้นเป็นเพียงเท็กซ์ไฟล์ ทำให้การเพิ่มและแก้ไขกฎดังกล่าวทำได้ไม่สะดวก

จากปัญหาของระบบตรวจจับการบุกรุกด้วย Snort ทั้งสอง ผู้พัฒนาจึงได้จัดทำโครงการพัฒนาระบบตรวจจับการบุกรุกด้วย Snort Rules ที่เหมาะสม เพื่อให้ระบบตรวจจับการบุกรุกทำงานได้อย่างมีประสิทธิภาพมากขึ้น ช่วยลดการแจ้งเตือนที่ผิดพลาด ใช้หน่วยความจำลดลงและหน่วยประมวลผลทำงานได้มากขึ้น โดยสร้างโปรแกรมจัดการกับกฎและคอนฟิกไฟล์ของ Snort ให้สะดวกขึ้นด้วย Graphic User Interface (GUI)

บทที่ 4

การออกแบบและพัฒนาระบบงาน

4.1 ตัวอย่างคอนฟิกไฟล์ snort.conf

```
#--------------------------------------------------
#   Snort Rule Management     Snort RuleSet
#   Faculty Information Technology KMITL
#--------------------------------------------------
# $Id: snort.conf, 02/09/2004 00:32:57 cazz Exp $
##################################################
# This file contains a sample snort configuration.
# You can take the following steps to create your
#
#  1) Set the network variables for your network
#  2) Configure preprocessors
#  3) Configure output plugins
#  4) Customize your rule set
#
##################################################
# Step #1: Set the network variables:

# Specify lists of IP addresses for HOME_NET.
var HOME_NET 172.30.68.0/24

# Set up the external network addresses as well.
var EXTERNAL_NET any

# List of DNS servers on your network.
var DNS_SERVERS $HOME_NET

# List of SMTP servers on your network.
var SMTP_SERVERS $HOME_NET

# List of web servers on your network.
var HTTP_SERVERS $HOME_NET

# List of sql servers on your network.
var SQL_SERVERS $HOME_NET

# List of telnet servers on your network.
var TELNET_SERVERS $HOME_NET

# Ports you run web servers on.
var HTTP_PORTS 80
```

รูปที่ 4.1 ตัวอย่างคอนฟิกไฟล์ snort.conf

```
# Ports you want to look for SHELLCODE on.
var SHELLCODE_PORTS !80

# Ports you do oracle attacks on.
var ORACLE_PORTS 1521

# AIM servers.
var AIM_SERVERS
[64.12.24.0/24,64.12.25.0/24,64.12.26.14/24,64.12.28.0/24,64.12.29
.0/24,64.12.161.0/24,64.12.163.0/24,205.188.5.0/24,205.188.9.0/24]

# Path to your rules files.
var RULE_PATH c:\snort\rules

##################################################
# Step #2: Configure preprocessors
# General configuration for preprocessors is of the form
# preprocessor <name_of_processor>: <configuration_options>
preprocessor frag2

# stream4: stateful inspection/stream reassembly for Snort
preprocessor stream4: detect_scans, disable_evasion_alerts
preprocessor stream4_reassemble: both

# http_decode: normalize HTTP requests
preprocessor http_decode: 80 8877 unicode iis_alt_unicode
double_encode iis_flip_slash full_whitespace

# rpc_decode: normalize RPC traffic
preprocessor rpc_decode: 111 32771

# bo: Back Orifice detector
preprocessor bo

# telnet_decode: Telnet negotiation string normalizer
preprocessor telnet_decode

# Portscan: detect a variety of portscans
#preprocessor portscan: $HOME_NET 4 3 portscan.log
#preprocessor portscan-ignorehosts: 0.0.0.0

# Conversation
preprocessor conversation: allowed_ip_protocols all, timeout 60,
max_conversations 65535, alert_odd_protocols

##################################################################
# Step #3: Configure output plugins and Classification & Reference
output database: alert, mysql, dbname=snort host=localhost
port=7788 user=root password=snortids detail=full

# Include classification & priority settings
include c:\snort\etc\classification.config
```

รูปที่ 4.2 ตัวอย่างคอนฟิกไฟล์ snort.conf (ต่อ)

```
# Include reference systems
include c:\snort\etc\reference.config

##################################################################
# Step #4: Customize your rule set
include $RULE_PATH/attack-responses.rules
include $RULE_PATH/backdoor.rules
include $RULE_PATH/bad-traffic.rules
include $RULE_PATH/chat.rules
include $RULE_PATH/ddos.rules
include $RULE_PATH/deleted.rules
include $RULE_PATH/dns.rules
include $RULE_PATH/dos.rules
include $RULE_PATH/experimental.rules
include $RULE_PATH/exploit.rules
include $RULE_PATH/finger.rules
include $RULE_PATH/ftp.rules
include $RULE_PATH/icmp.rules
include $RULE_PATH/icmp-info.rules
include $RULE_PATH/imap.rules
include $RULE_PATH/info.rules
include $RULE_PATH/local.rules
include $RULE_PATH/misc.rules
include $RULE_PATH/multimedia.rules
include $RULE_PATH/mysql.rules
include $RULE_PATH/netbios.rules
include $RULE_PATH/nntp.rules
include $RULE_PATH/oracle.rules
include $RULE_PATH/other-ids.rules
include $RULE_PATH/p2p.rules
include $RULE_PATH/policy.rules
include $RULE_PATH/pop2.rules
include $RULE_PATH/pop3.rules
include $RULE_PATH/porn.rules
include $RULE_PATH/rpc.rules
include $RULE_PATH/rservices.rules
include $RULE_PATH/scan.rules
include $RULE_PATH/shellcode.rules
include $RULE_PATH/smtp.rules
include $RULE_PATH/snmp.rules
include $RULE_PATH/sql.rules
include $RULE_PATH/telnet.rules
include $RULE_PATH/tftp.rules
include $RULE_PATH/virus.rules
include $RULE_PATH/web-attacks.rules
include $RULE_PATH/web-cgi.rules
include $RULE_PATH/web-client.rules
include $RULE_PATH/web-coldfusion.rules
include $RULE_PATH/web-frontpage.rules
include $RULE_PATH/web-iis.rules
include $RULE_PATH/web-misc.rules
include $RULE_PATH/web-php.rules
include $RULE_PATH/x11.rules
```

รูปที่ 4.3 ตัวอย่างคอนฟิกไฟล์ snort.conf (ต่อ)

จากรูปที่ 4.1, 4.2 และ 4.3 เป็นตัวอย่างของ snort.conf ซึ่งเป็นคอนฟิกไฟล์หลักที่ใช้เก็บตัวแปรต่างๆ โดย snort.conf ประกอบด้วย 4 ขั้นตอนดังนี้

1. กำหนดค่าตัวแปรในเครือข่ายที่ต้องการตรวจจับการบุกรุก

เป็นขั้นตอนการกำหนดค่าให้กับเครือข่ายที่ต้องการตรวจจับการบุกรุก โดยตัวแปรต่างๆ แสดงดังตารางที่ 4.1

ตารางที่ 4.1 ตัวแปรในเครือข่ายที่ต้องการตรวจจับการบุกรุก

ตัวแปร	คำอธิบาย	ค่าตัวแปร (default value)
HOME_NET	Network address ภายใน	
EXTERNAL_NET	Network address ภายนอก	any
DNS_SERVERS	IP Address ของ DNS server	$HOME_NET
SMTP_SERVERS	IP Address ของ SMTP server	$HOME_NET
HTTP_SERVERS	IP Address ของ HTTP server	$HOME_NET
SQL_SERVERS	IP Address ของ SQL server	$HOME_NET
TELNET_SERVERS	IP Address ของ Telnet server	$HOME_NET
HTTP_PORTS	หมายเลขพอร์ตของ HTTP	80
SHELLCODE_PORTS	หมายเลขพอร์ตของ Shell code	!80
ORACLE_PORTS	หมายเลขพอร์ตของ Oracle	1521
AIM_SERVERS	IP Address List ของ AIM server	[64.12.24.0/24,64.12.25.0/24,64.12.26.14/24,64.12.28.0/24,64.12.29.0/24,64.12.161.0/24,64.12.163.0/24,205.188.5.0/24,205.188.9.0/24]
RULE_PATH	ตำแหน่งที่เก็บรูลไฟล์	c:\snort\rules

2. กำหนดค่าตัวแปรในการแปลความหมายของกระแสข้อมูล

เป็นขั้นตอนการสั่งให้ Snort แปลความหมายของกระแสข้อมูลที่ไหลผ่านบนเครือข่ายที่ต้องการตรวจจับ โดยโครงการพัฒนาระบบงานนี้ไม่มีการเปลี่ยนแปลงค่าใดๆในขั้นตอนนี้

3. กำหนดค่าตัวแปรในการเก็บการแจ้งเตือนของ Snort ลงฐานข้อมูล

เป็นขั้นตอนกำหนดค่าให้ Snort เก็บข้อมูลการแจ้งเตือนลงฐานข้อมูล โดยโครงการพัฒนาระบบงานนี้เก็บข้อมูลลงฐานข้อมูล MySQL, ชื่อฐานข้อมูล snort, เก็บข้อมูลลงเครื่องตัวเอง (localhost), พอร์ตที่ใช้เชื่อมต่อใช้พอร์ตหมายเลข 7788, ชื่อผู้ใช้ที่ใช้ล็อกอินเข้าใช้ฐานข้อมูลนี้คือ root, รหัสผ่านที่ใช้ล็อกอินเข้าใช้ฐานข้อมูลนี้คือ snortids และเก็บข้อมูลการแจ้งเตือนทุกอย่าง นอกจากนี้ยังมีการอ้างอิงถึงข้อมูลการจัดประเภทของกฎ ระดับความสำคัญของกฎ เพื่อสร้างคอนฟิกไฟล์ classification.config และมีการอ้างอิงถึงแหล่งที่มาของกฎแต่ละข้อ เพื่อสร้างคอนฟิกไฟล์ reference.config ดังรูปที่ 4.4

```
output database: alert, mysql, dbname=snort host=localhost
port=7788 user=root password=snortids detail=full

# Include classification & priority settings
include c:\snort\etc\classification.config
# Include reference systems
include c:\snort\etc\reference.config
```

รูปที่ 4.4 ตัวแปรในการเก็บการแจ้งเตือนของ Snort ลงฐานข้อมูล

4. กำหนดชื่อรูลไฟล์ที่ใช้ในการตรวจจับการบุกรุก

ขั้นตอนนี้เป็นการกำหนดให้ Snort ใช้รูลไฟล์ใดบ้าง รูลไฟล์เหล่านี้เก็บอยู่ที่ตำแหน่งใด ถือเป็นหัวใจของโครงการพัฒนาระบบงาน โดยฐานข้อมูลของโครงการพัฒนาระบบงานนี้มีกลุ่มกฎ (Rule Group) ทั้งหมด 48 กลุ่มกฎ และมีกฎ (Signature) ทั้งหมด 2,060 กฎดังตารางที่ 4.2 - 4.4

ตารางที่ 4.2 ชื่อรูลไฟล์ที่ใช้ในการตรวจจับการบุกรุก

ชื่อรูลไฟล์ (Rule Group)	จำนวนกฎ (Signatures)
attack-responses.rules	16
backdoor.rules	58
bad-traffic.rules	10
chat.rules	18
ddos.rules	33
deleted.rules	169

ตารางที่ 4.3 ชื่อรูลไฟล์ที่ใช้ในการตรวจจับการบุกรุก (ต่อ)

ชื่อรูลไฟล์ (Rule Group)	จำนวนกฎ (Signatures)
dns.rules	19
dos.rules	18
experimental.rules	0
exploit.rules	36
finger.rules	13
ftp.rules	50
icmp.rules	22
icmp-info.rules	93
imap.rules	16
info.rules	7
local.rules	0
misc.rules	44
multimedia.rules	6
mysql.rules	2
netbios.rules	33
nntp.rules	2
oracle.rules	25
other-ids.rules	3
p2p.rules	16
policy.rules	22
pop2.rules	4
pop3.rules	19
porn.rules	27
rpc.rules	128
rservices.rules	13
scan.rules	25
shellcode.rules	22
smtp.rules	27
snmp.rules	17
sql.rules	43

ตารางที่ 4.4 ชื่อรูลไฟล์ที่ใช้ในการตรวจจับการบุกรุก (ต่อ)

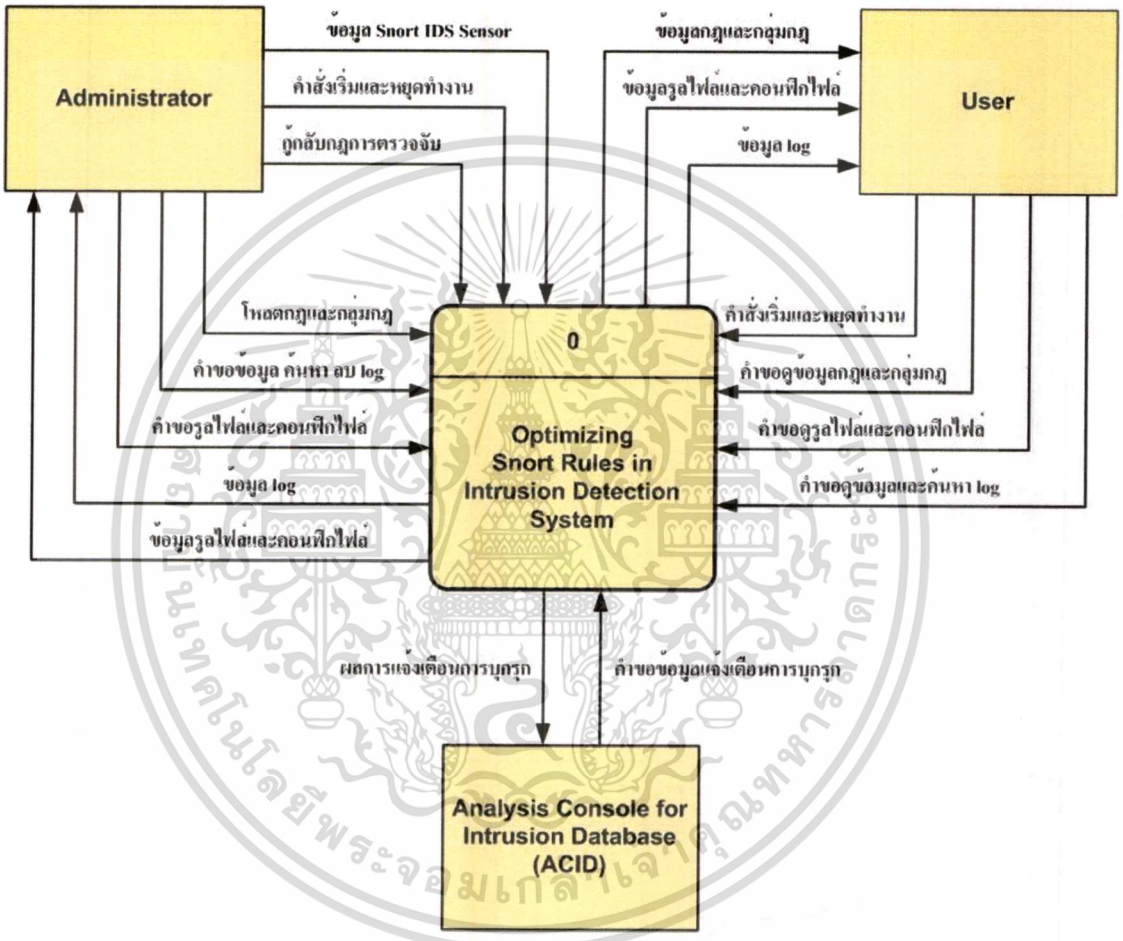
ชื่อรูลไฟล์ (Rule Group)	จำนวนกฎ (Signatures)
telnet.rules	14
tftp.rules	9
virus.rules	19
web-attacks.rules	47
web-cgi.rules	344
web-client.rules	6
web-coldfusion.rules	35
web-frontpage.rules	34
web-iis.rules	111
web-misc.rules	294
web-php.rules	89
x11.rules	2
รวมกฎทั้งหมด	**2,060**

4.2 ออกแบบระบบงาน

เมื่อได้ศึกษาถึงรูลไฟล์และคอนฟิกไฟล์ที่ใช้ในระบบตรวจจับการบุกรุกด้วย Snort แล้วจึงสร้างเป็นคอนเท็กซ์ไดอะแกรม เพื่อให้เห็นภาพรวมและขอบเขตการทำงานของระบบดังรูปที่ 4.5 จากรูปคอนเท็กซ์ไดอะแกรม มีส่วนที่เกี่ยวข้องกับระบบงานอยู่ 3 องค์ประกอบคือ Administrator, User และ Analysis Console for Intrusion Database (ACID) ซึ่งสามารถอธิบายออกเป็นส่วนงานหลักได้ 2 ส่วนดังนี้

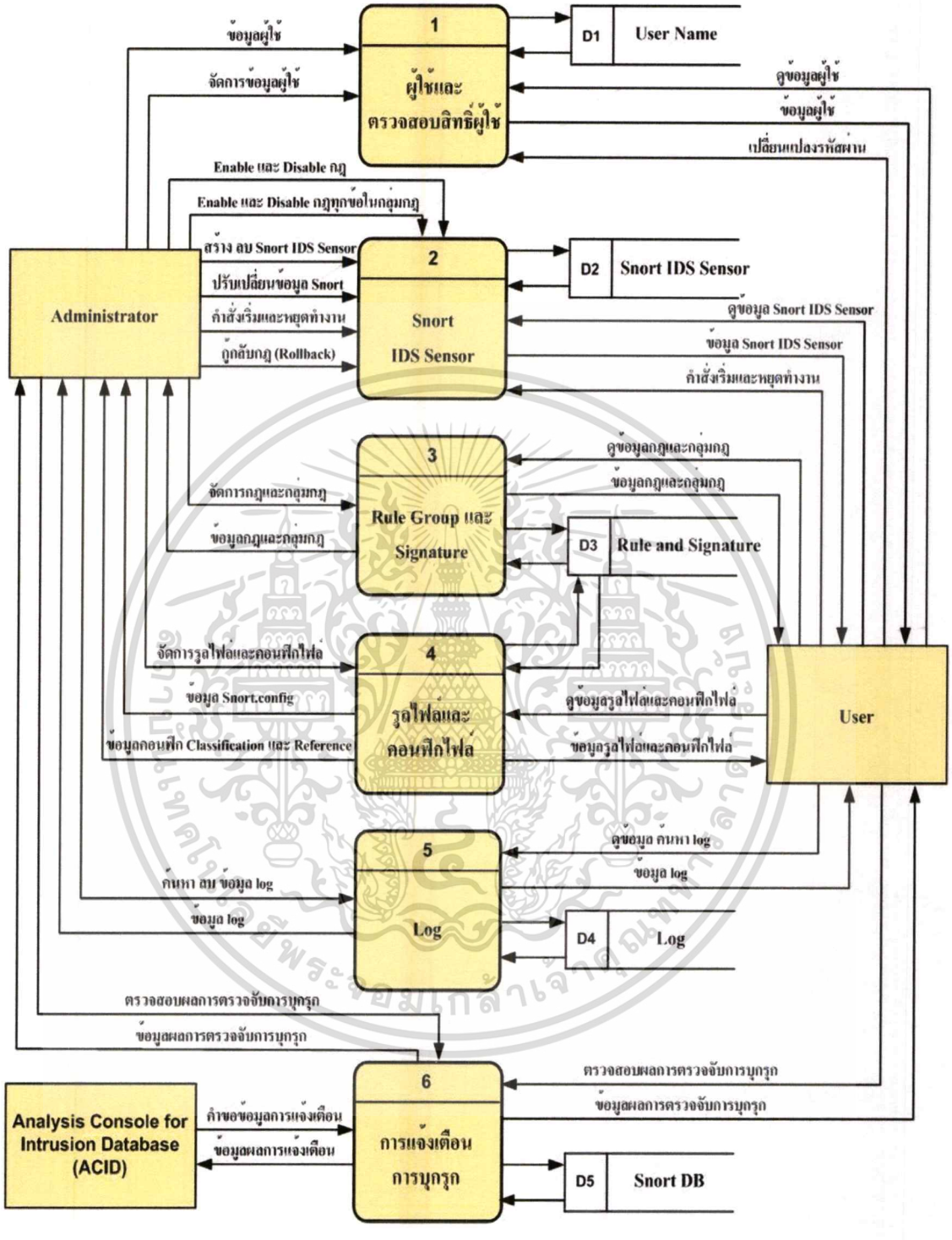
1. ส่วนนำเข้าระบบ ได้แก่
   - สร้าง Snort IDS Sensor
   - สร้างกลุ่มกฎและกฎการตรวจจับ
   - สร้างรูลไฟล์และคอนฟิกไฟล์
   - สร้างข้อมูลการกู้กฎกลับ
   - สร้างข้อมูลการตรวจจับ
   - สร้างข้อมูลผู้ใช้งานระบบ

2. ส่วนนำออกจากระบบ ได้แก่
   - รายงานผลการตรวจจับ
   - รายงานการทำงานของผู้ใช้
   - รูลไฟล์และคอนฟิกไฟล์



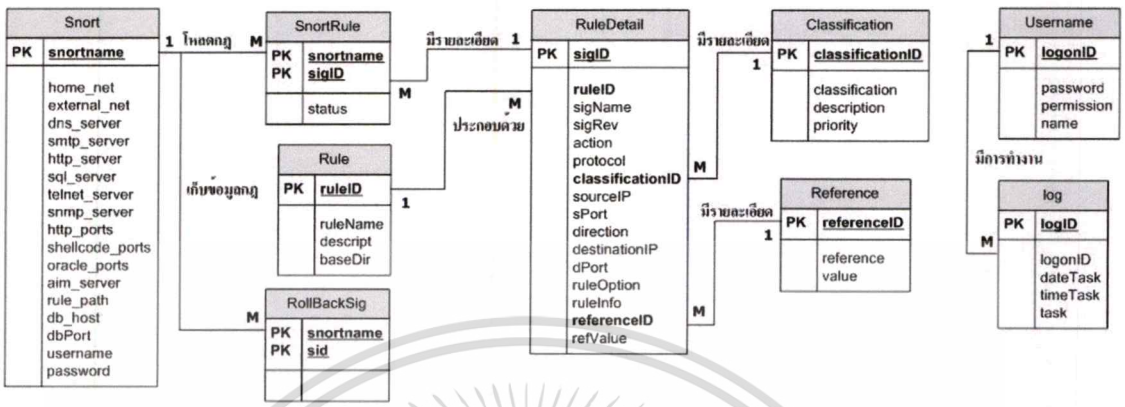
รูปที่ 4.5 ภาพรวมและขอบเขตงานของระบบด้วยคอนเท็กซ์ไดอะแกรม

เมื่อทราบถึงภาพรวมของระบบแล้ว ก็ต้องศึกษาในขั้นตอนการปฏิบัติงานและวิเคราะห์ความต้องการของระบบงานสารสนเทศต่างๆ โดยมีวิธีจัดทำแผนภาพกระแสข้อมูลระดับที่ 1 ซึ่งจะช่วยให้วิเคราะห์ระบบโดยแบ่งเป็นระบบย่อยได้สะดวกยิ่งขึ้นแสดงในรูปที่ 4.6 ซึ่งสามารถแบ่งงานออกเป็น 6 กระบวนการ ในแต่ละงานจะมีความสัมพันธ์กันและมีการติดต่อทั้งผู้ใช้และฐานข้อมูลเดียวกัน



รูปที่ 4.6 แผนภาพกระแสข้อมูลระดับที่ 1

4.3 ฐานข้อมูลสำหรับโครงการพัฒนาระบบงาน



รูปที่ 4.7 ฐานข้อมูลสำหรับโครงการพัฒนาระบบงาน

จากตัวอย่างไฟล์ snort.conf นำมาสร้างฐานข้อมูล ดังรูปที่ 4.7 ฐานข้อมูลสำหรับโครงการพัฒนาระบบตรวจจับการบุกรุกด้วย Snort Rule ที่เหมาะสม และตารางที่ 4.5-4.13 พจนานุกรมข้อมูลแสดงรายละเอียดของแต่ละตาราง โดยมีทั้งหมด 9 ตารางดังนี้

1. Snort เป็นตารางที่เก็บค่าตัวแปรทั้งหมดเพื่อนำมาสร้างคอนฟิกไฟล์ snort.conf
2. SnortRule เป็นตารางที่เก็บข้อมูลของ Snort IDS Sensor แต่ละตัวว่าชื่ออะไร มีการใช้กฎหรือ Signature ใดบ้าง เพื่อนำมาสร้างรูลไฟล์ของ Snort IDS Sensor แต่ละตัว
3. Rule เป็นตารางที่เก็บกลุ่มกฎ (Rule Group) ของ Snort ทั้งหมด
4. RuleDetail เป็นตารางที่เก็บรายละเอียดของกฎ (Signature) แต่ละข้อ
5. Classification เป็นตารางที่เก็บประเภทและระดับความสำคัญของกฎ เพื่อนำมาสร้างคอนฟิกไฟล์ classification.config
6. Reference เป็นตารางที่เก็บ URL เพื่ออ้างอิงไปยัง website ของผู้สร้างกฎในแต่ละข้อ เพื่อนำมาสร้างคอนฟิกไฟล์ reference.config
7. RollBackSig เป็นตารางที่เก็บ Signature ที่อนุญาตให้ใช้หรือยกเลิกการใช้ เพื่อการกู้คอนฟิกเก่ากลับมาใช้ใหม่
8. Username เป็นตารางที่เก็บข้อมูลผู้ใช้ระบบและสิทธิ์ผู้ใช้
9. Log เป็นตารางที่เก็บข้อมูลการทำงานของผู้ใช้ว่ามีการทำงานอะไรบ้าง โดยจะเก็บวัน เวลา และงานที่ผู้ใช้ได้ทำไว้เพื่อให้ผู้ดูแลระบบตรวจสอบได้

ตารางที่ 4.5 พจนานุกรมข้อมูลตาราง Snort

Attribute Name	Description	Type	Key	Reference Table
snortname	ชื่อ Snort	Text (50)	PK	
home_net	Network address ภายใน	Text (50)		
external_net	Network address ภายนอก	Text (50)		
dns_server	IP Address ของ DNS servers	Text (50)		
smtp_server	IP Address ของ SMTP server	Text (50)		
http_server	IP Address ของ HTTP server	Text (50)		
sql_server	IP Address ของ SQL server	Text (50)		
telnet_server	IP Address ของ Telnet server	Text (50)		
snmp_server	IP Address ของ SNMP server	Text (50)		
http_ports	หมายเลขพอร์ตของ HTTP	Text (50)		
shellcode_ports	หมายเลขพอร์ตของ Shell code	Text (50)		
oracle_ports	หมายเลขพอร์ตของ Oracle	Text (50)		
aim_server	IP Address ของ AIM server	Text (50)		
rule_path	ตำแหน่งที่เก็บรูลไฟล์	Text (50)		
db_host	คอมพิวเตอร์ที่เก็บล็อกลง MySQL	Text (50)		
username	ชื่อผู้ใช้ฐานข้อมูล MySQL	Text (50)		
password	รหัสผ่านฐานข้อมูล MySQL	Text (50)		
dbPort	พอร์ตสื่อสารฐานข้อมูล MySQL	Text (50)		

ตารางที่ 4.6 พจนานุกรมข้อมูลตาราง Rule

Attribute Name	Description	Type	Key	Reference Table
ruleID	หมายเลขกลุ่มกฎ	AutoNumber	PK	
ruleName	ชื่อกลุ่มกฎ	Text (100)		
descript	คำอธิบายกลุ่มกฎ	Memo		
baseDir	ไดเรคทอรีที่เก็บกลุ่มกฎ	Text (100)		

ตารางที่ 4.7 พจนานุกรมข้อมูลตาราง RollBackSig

Attribute Name	Description	Type	Key	Reference Table
snortname	ชื่อ Snort IDS Sensor	Text (50)	PK, FK	Snort
SID	หมายเลขกฎและสถานะของกฎ	Text (50)	PK	

ตารางที่ 4.8 พจนานุกรมข้อมูลตาราง SnortRule

Attribute Name	Description	Type	Key	Reference Table
snortname	ชื่อ Snort	Text (50)	PK, FK	Snort
sigID	หมายเลขกฎ	Number	PK, FK	RuleDetail
status	สถานะกฎ	Yes/No		

ตารางที่ 4.9 พจนานุกรมข้อมูลตาราง RuleDetail

Attribute Name	Description	Type	Key	Reference Table
sigID	หมายเลขกฎ	Number	PK	
ruleID	หมายเลขกลุ่มกฎ	Number	FK	Rule
sigName	ชื่อกฎ	Text (255)		
sigRev	เวอร์ชันของกฎ	Number		
action	แอคชัน	Text (50)		
protocol	โพรโทคอล	Text (50)		
classificationID	หมายเลข classification	Number	FK	Classification
sourceIP	IP Address ต้นทาง	Text (50)		
sPort	หมายเลขพอร์ตต้นทาง	Text (50)		
direction	ทิศทางกฎ	Text (50)		
destinationIP	IP Address ปลายทาง	Text (50)		
dPort	หมายเลขพอร์ตปลายทาง	Text (50)		
ruleOption	รูล option	Memo		
ruleinfo	ข้อมูลรูล	Memo		
referenceID	หมายเลข reference	Number	FK	Reference
refValue	ค่า reference	Text (255)		

ตารางที่ 4.10 พจนานุกรมข้อมูลตาราง Classification

Attribute Name	Description	Type	Key	Reference Table
classificationID	หมายเลข classification	AutoNumber	PK	
classification	ชื่อ classification	Memo		
description	คำอธิบาย classification	Memo		
priority	ค่าระดับความสำคัญ classification	Number		

ตารางที่ 4.11 พจนานุกรมข้อมูลตาราง Reference

Attribute Name	Description	Type	Key	Reference Table
referenceID	หมายเลข reference	AutoNumber	PK	
reference	ชื่อ reference	Text (255)		
value	ค่าเริ่มต้น reference	Memo		

ตารางที่ 4.12 พจนานุกรมข้อมูลตาราง Username

Attribute Name	Description	Type	Key	Reference Table
logonID	ชื่อผู้เข้าใช้ระบบ	Text (50)	PK	
name	ชื่อผู้ใช้ระบบ	Text (50)		
password	รหัสผ่าน	Text (50)		
permission	สิทธิ์ผู้เข้าใช้ระบบ	Text (50)		

ตารางที่ 4.13 พจนานุกรมข้อมูลตาราง Log

Attribute Name	Description	Type	Key	Reference Table
logID	ชื่อ Snort IDS Sensor	AutoNumber	PK	
logonID	ชื่อผู้เข้าใช้ระบบ	Text (50)	FK	Username
dateTask	วันที่ทำงาน	Date/Time		
timeTask	เวลาทำงาน	Date/Time		
task	งานที่ทำ	Memo		

4.4 ซอฟต์แวร์ที่เกี่ยวข้องในโครงการพัฒนาระบบงาน

ระบบตรวจจับการบุกรุกด้วย Snort Rule ที่เหมาะสมมีการใช้ซอฟต์แวร์ทั้งหมด 9 ประเภทดังนี้

4.4.1 **Snort Software** เป็นซอฟต์แวร์ระบบตรวจจับการบุกรุกเครือข่าย (NIDS) ซึ่งจะคอยเฝ้าดูข้อมูลบนเครือข่าย โดยทำการรับข้อมูลทั้งหมดที่อยู่บนส่วนของเครือข่ายที่รับผิดชอบ sensor จะมองเห็นเฉพาะ packet ที่ผ่านส่วนของเครือข่ายที่ sensor นั้นติดอยู่ packet ต่างๆ จะเป็นที่สนใจของ sensor ก็ต่อเมื่อ packet นั้นเข้ากับ signature ที่กำหนดในรูลไฟล์ โดยมีการอ้างอิงกับคอนฟิกไฟล์ 3 ไฟล์คือ snort.conf, classification.config, reference.config

4.4.2 **Analysis Console for Intrusion Database (ACID)** เป็น PHP-based analysis engine สำหรับค้นหาและกระบวนการทำงานของฐานข้อมูลสำหรับเหตุการณ์ด้านความปลอดภัยที่ถูกสร้างจากระบบตรวจจับการบุกรุกแบบต่างๆ, ไฟร์วอลล์ และเครื่องมือการเฝ้าดูเน็ตเวิร์ค โดยโครงการพัฒนาระบบงานนี้ใช้ ACID สำหรับดูการแจ้งเตือนของ Snort ซึ่งผู้ใช้สามารถค้นหา IP Address ต้นทาง, IP Address ปลายทาง, ชนิดการแจ้งเตือน, เวลาการแจ้งเตือน, หมายเลขพอร์ต และหรือโพรโทคอล

4.4.3 **Apache** เป็น Web Server สำหรับ ACID โดยโครงการพัฒนาระบบงานนี้ใช้หมายเลข HTTP พอร์ต 8877 สามารถเข้าถึง web page ได้ด้วย URL: http://localhost:8877/acid

4.4.4 **MySQL** เป็นฐานข้อมูลแบบเปิดที่ออกแบบมาเพื่อความรวดเร็ว มีประสิทธิภาพ และความถูกต้อง โดยโครงการพัฒนาระบบงานนี้นำมาใช้เก็บข้อมูลการแจ้งเตือนของ Snort และใช้งานร่วมกับ ACID

4.4.5 **PHP** (PHP:Hypertext Processor) เป็นภาษาสคริปต์ระบบเปิดแบบ server-side ใช้ในการสร้างไดนามิคเว็บเพจ ไวยากรณ์ของ PHP คล้ายกับภาษา C, JAVA และ Perl โดยโครงการพัฒนาระบบงานนี้นำมาใช้ร่วมกับ ACID เพื่อแสดงเว็บเพจ

4.4.6 **ADODB** เป็นกลุ่มคลาสขั้นสูงสำหรับ PHP ในการเชื่อมต่อฐานข้อมูล เช่น MySQL

4.4.7 **WinPCAP** เป็นไลบรารี (library) ซึ่งใช้กันโดยทั่วไปในบรรดา network sniffer และ network analyzer

4.4.8 **Microsoft Access** ใช้เก็บข้อมูลสำหรับโครงการพัฒนาระบบงาน โดยกล่าวมาแล้วในหัวข้อที่ 4.3

4.4.9 **Microsoft Visual Basic 6** ใช้เขียนโปรแกรมสำหรับโครงการพัฒนาระบบงาน

4.5 การออกแบบหน้าจอการทำงานของโครงการพัฒนาระบบงาน

4.5.1 การล็อกอินเข้าใช้งานระบบ

หน้าจอล็อกอินเป็นหน้าจอแรกที่ปรากฏเมื่อผู้ใช้งานเริ่มใช้ระบบ เพื่อระบุตัวตนว่ามีสิทธิ์เข้าใช้งานหรือไม่

รูปที่ 4.8 หน้าจอการล็อกอินเพื่อเข้าใช้งานระบบ

จากรูปที่ 4.8 เป็นหน้าจอแรกที่ผู้ใช้ป้อนชื่อผู้ใช้ (username) และรหัสผ่าน (password) เพื่อแสดงสิทธิ์การใช้งาน โดยระบบจะแบ่งประเภทสิทธิ์ผู้ใช้งานออกเป็น 2 ประเภทคือ

1. สิทธิ์ User สามารถดูค่าตัวแปรต่างๆ ได้โดยไม่สามารถเปลี่ยนแปลงแก้ไขค่าใดๆได้ นอกจากนี้ยังสามารถสั่งให้ระบบตรวจจับการบุกรุกด้วย Snort เริ่มทำงานหรือหยุดทำงานได้
2. สิทธิ์ Admin เป็นสิทธิ์ที่เหนือกว่าสิทธิ์ User โดยสามารถทำงานได้ทุกฟังก์ชันในระบบตรวจจับการบุกรุกด้วย Snort Rule ที่เหมาะสม

4.5.2 หน้าจอหลักของโครงการพัฒนาระบบงาน

หลังจากผ่านขั้นตอนการล็อกอินเรียบร้อยแล้ว ก็จะเข้าหน้าจอหลักของระบบตรวจจับการบุกรุก ต่อไปจะเรียกโปรแกรม Snort Rule Management ดังรูปที่ 4.9 ซึ่งหน้าจอออกแบบไว้ 4 ส่วน ดังนี้

1. **Toolbar** เป็นแถบที่มีปุ่มและตัวเลือกที่ใช้ในการออกคำสั่ง โดยมีทั้งหมด 8 ปุ่มดังนี้
   1.1 Snort IDS ใช้สำหรับสร้าง Snort IDS Sensor ใหม่ในระบบ
   1.2 Rule Group ใช้สำหรับเพิ่มและแก้ไขรูลไฟล์
   1.3 Classification ใช้สำหรับเพิ่มและแก้ไขคอนฟิกไฟล์ Classification.config
   1.4 Reference ใช้สำหรับเพิ่มและแก้ไขคอนฟิกไฟล์ Reference.config
   1.5 User Name ใช้สำหรับเพิ่ม ลบและแก้ไขผู้ใช้งานระบบ

1.6 Logging ใช้สำหรับมอนิเตอร์ผู้ใช้งานว่ามีการทำงานอะไรบ้างในระบบ

1.7 Help ใช้สำหรับขอความช่วยเหลือ

1.8 Exit ใช้สำหรับออกจากระบบตรวจจับการบุกรุก

2. **Navigation Menu** แบ่งออกเป็น 2 แท็บดังนี้

2.1 แท็บ Snort IDS จะเหมือนกับการเรียกใช้งานผ่าน Toolbar โดยด้านล่างจะแสดงรายการ Snort IDS Sensor ซึ่งแสดงรายชื่อของ Snort IDS Sensor ทั้งหมดในฐานข้อมูล โดยนำเสนอข้อมูลแบบ Tree View ภายใต้ชื่อ Snort IDS Sensor ก็จะแสดง Rule Group ทั้งหมดที่มีการใช้งานด้วย

2.2 แท็บ Rule แสดง Rule Group ทั้งหมดในฐานข้อมูล

3. **Snort Panel** แสดงฟังก์ชันที่มีในระบบตรวจจับการบุกรุก โดยมีทั้งหมด 8 แท็บดังนี้

3.1 แท็บ Snort แสดงเมื่อมีการกดปุ่ม Snort IDS หรือเมนู Snort IDS

3.2 แท็บ Rule Group แสดงกฎทั้งหมดที่อยู่ใน Rule Group นั้นๆ

3.3 แท็บ Signature แสดงรายละเอียดทั้งหมดของกฎที่ถูกเลือก

3.4 แท็บ Snort DB แสดง Web page ซึ่งเชื่อมโยงไปยัง www.snort.org โดยอ้างอิงกับ SID ของกฎแต่ละข้อ

3.5 แท็บ Reference แสดง Web page ซึ่งเชื่อมโยงไปยัง website ต่างๆ โดยอ้างอิงกับ ReferenceID ของกฎแต่ละข้อ

3.6 แท็บ Classification & Reference แสดงข้อมูล Classification และ Reference ในฐานข้อมูล

3.7 แท็บ User Name & Logging แสดงรายชื่อผู้ใช้และล็อกทั้งหมดในฐานข้อมูล

3.8 แท็บ ACID แสดง Web page ของ ACID เมื่อผู้ใช้งานสั่งให้ Snort ทำงาน

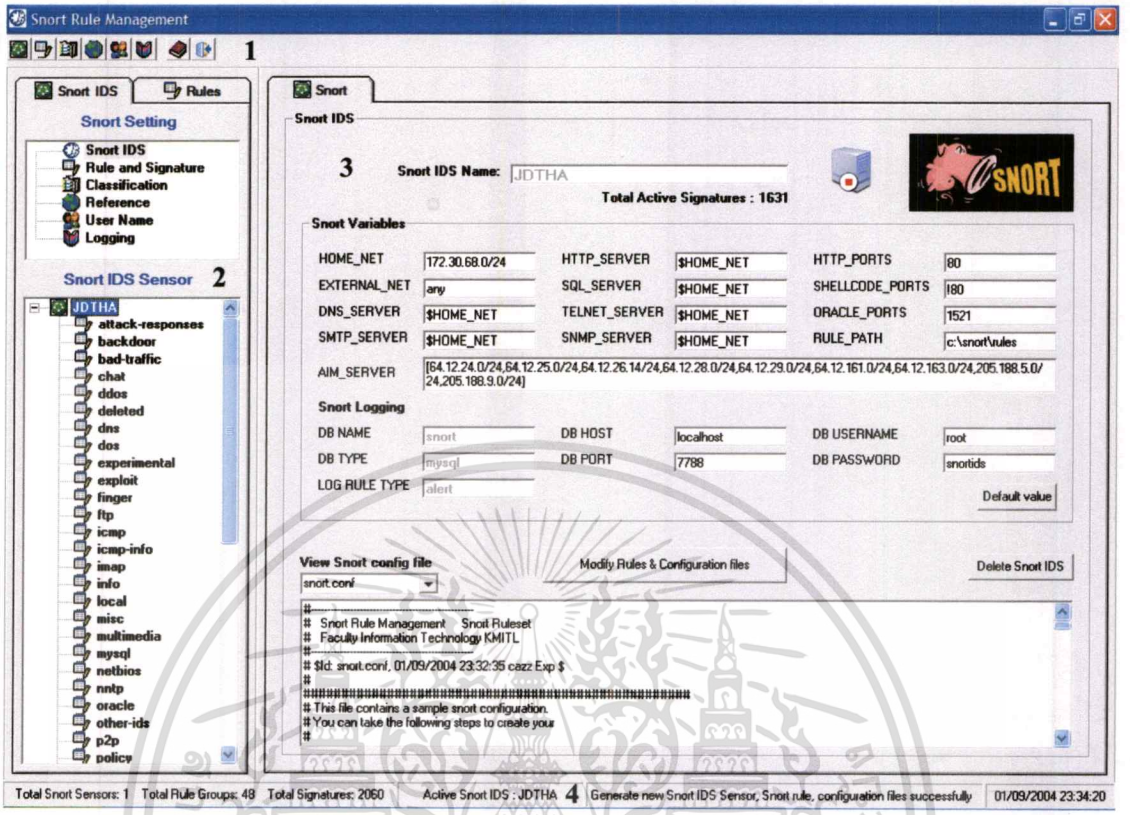
4. **Status Bar** แสดงสถานะของระบบตรวจจับการบุกรุก โดยแบ่งออกเป็น 4 ส่วน

4.1 แสดงจำนวน Snort Sensor, Rule Group และ Signature ทั้งหมด

4.2 แสดง Snort IDS Sensor ที่กำลังใช้งานอยู่

4.3 แสดงสถานะการทำงานของคำสั่งว่าถูกต้องสำเร็จหรือไม่

4.4 แสดงวันและเวลาปัจจุบัน



รูปที่ 4.9 หน้าจอหลักของระบบตรวจจับการบุกรุก

4.6 ฟังก์ชันการทำงานของโครงการพัฒนาระบบงาน

ฟังก์ชันการทำงานในระบบตรวจจับการบุกรุกด้วย Snort Rule ที่เหมาะสมมีพื้นฐานจากคอนฟิกไฟล์ 3 ไฟล์คือ snort.conf, classification.config, reference.config และรูปแบบรูลไฟล์ของ Snort โครงการพัฒนาระบบตรวจจับการบุกรุกด้วย Snort Rule ที่เหมาะสมแบ่งฟังก์ชันการทำงานออกเป็น 7 ฟังก์ชันดังนี้

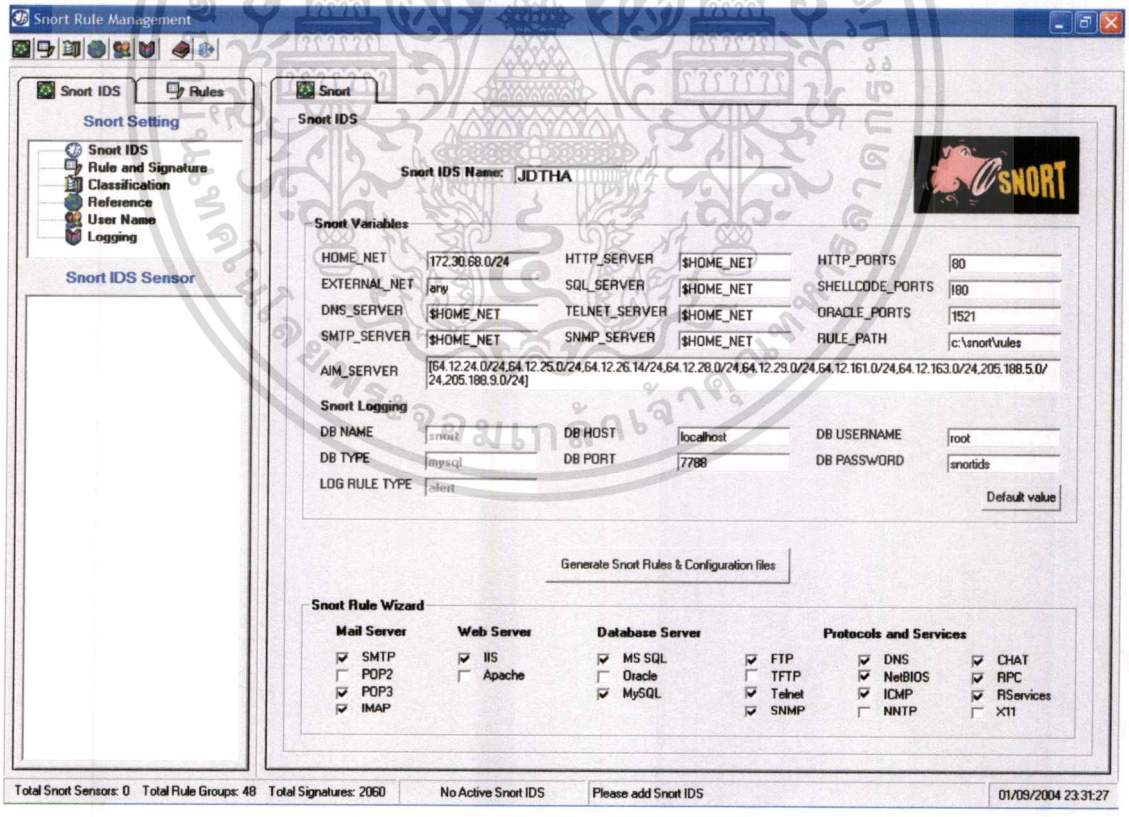
4.6.1 ฟังก์ชัน Snort IDS Sensor

เป็นฟังก์ชันที่เกี่ยวกับ Snort IDS Sensor ซึ่งประกอบด้วยฟังก์ชันย่อยดังนี้

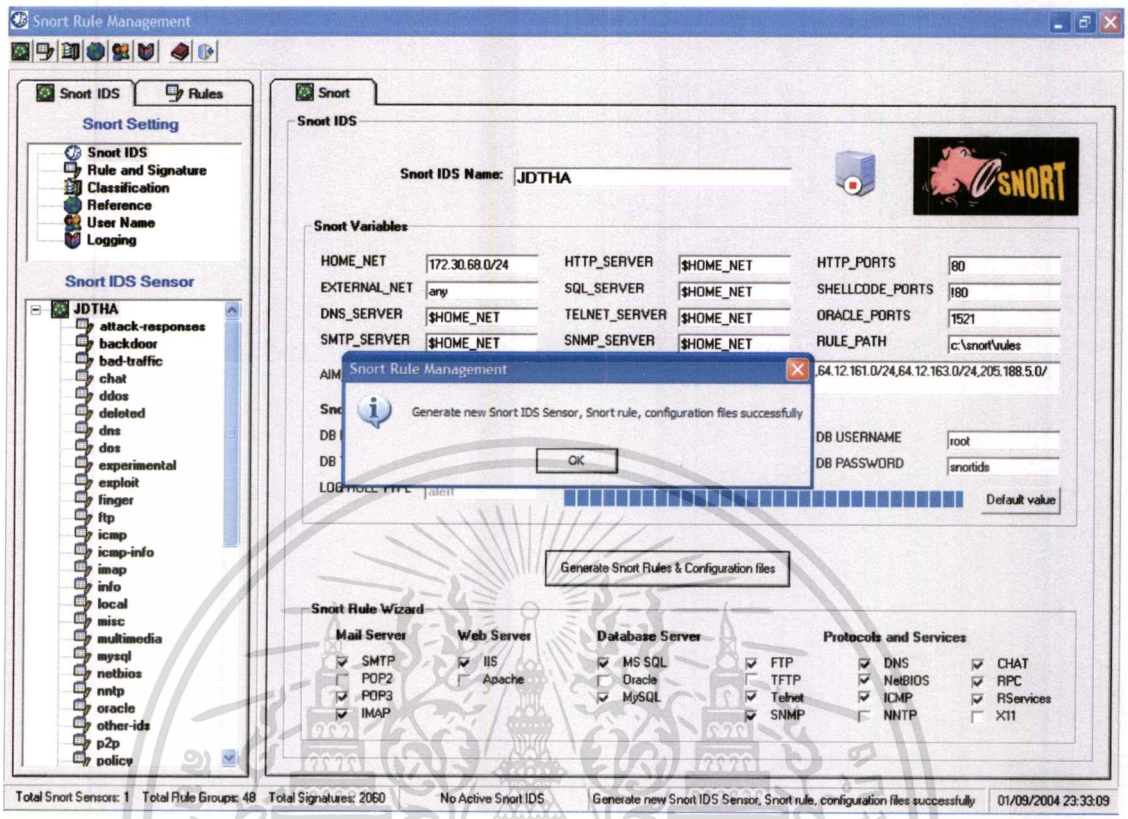
- **การสร้าง Snort IDS Sensor**

การสร้าง Snort IDS Sensor สามารถทำได้โดยการป้อนข้อมูลทั้งหมดในแท็บ Snort ดังรูปที่ 4.10 เนื่องจากคอนฟิกไฟล์ snort.conf ต้องการค่าของตัวแปร (Variables) เช่น HOME_NET, EXTERNAL_NET, HTTP_SERVER (ถ้ามี), HTTP_PORTS (ถ้ามี), SMTP_SERVER (ถ้ามี), DNS_SERVER (ถ้ามี), SQL_SERVER (ถ้ามี), RULE_PATH, DB NAME, DB HOST, DB PORT, DB USERNAME, DB PASSWORD เป็นต้น ซึ่งนำไปใช้ในระบบตรวจจับการบุกรุก โปรแกรมจึง

มีส่วนของตัวแปรต่างๆ ที่เกี่ยวข้องให้ผู้ใช้งานป้อนข้อมูล ซึ่งค่าตัวแปรต่างๆเหล่านี้จะถูกนำมาสร้างเป็นคอนฟิกและรูลไฟล์ นอกจากนี้โปรแกรมจะมีส่วนของ Snort Rule Wizard ที่ช่วยสร้างคอนฟิกไฟล์และรูลไฟล์จากระบบสารสนเทศที่มีอยู่ โดยระบบสามารถสร้างคอนฟิกไฟล์และรูลไฟล์ให้เหมาะกับองค์กร เช่น ถ้าองค์กรมี Mail Server ที่ให้บริการ SMTP, IMAP และ POP3 ก็สามารถเลือกบริการของ Mail Server เพียง SMTP, IMAP และ POP3 เท่านั้น ซึ่งโปรแกรมก็จะสร้างคอนฟิกไฟล์ที่อ้างถึงรูลไฟล์ SMTP, IMAP และ POP3 รวมถึงรูลไฟล์พื้นฐานอื่นๆ ที่จำเป็นในระบบให้ หรือถ้าองค์กรใดมี Web Server (IIS หรือ Apache), Database Server (MS SQL, Oracle หรือ MySQL) และให้บริการโพรโทคอล Ftp, Telnet, DNS, NetBIOS, SNMP, ICMP เป็นต้น ก็สามารถเลือกเฉพาะบริการที่มีอยู่ แล้วระบบก็จะสร้างคอนฟิกไฟล์และรูลไฟล์ที่เกี่ยวข้อง เมื่อผู้ใช้ป้อนข้อมูลเรียบร้อยแล้ว และพร้อมที่จะสร้าง Snort IDS Sensor ให้กดปุ่ม Generate Rules & Configuration Files หากไม่มีข้อผิดพลาดก็จะแสดงกรอบโต้ตอบการสร้าง Snort IDS Sensor สำเร็จ ดังรูปที่ 4.11



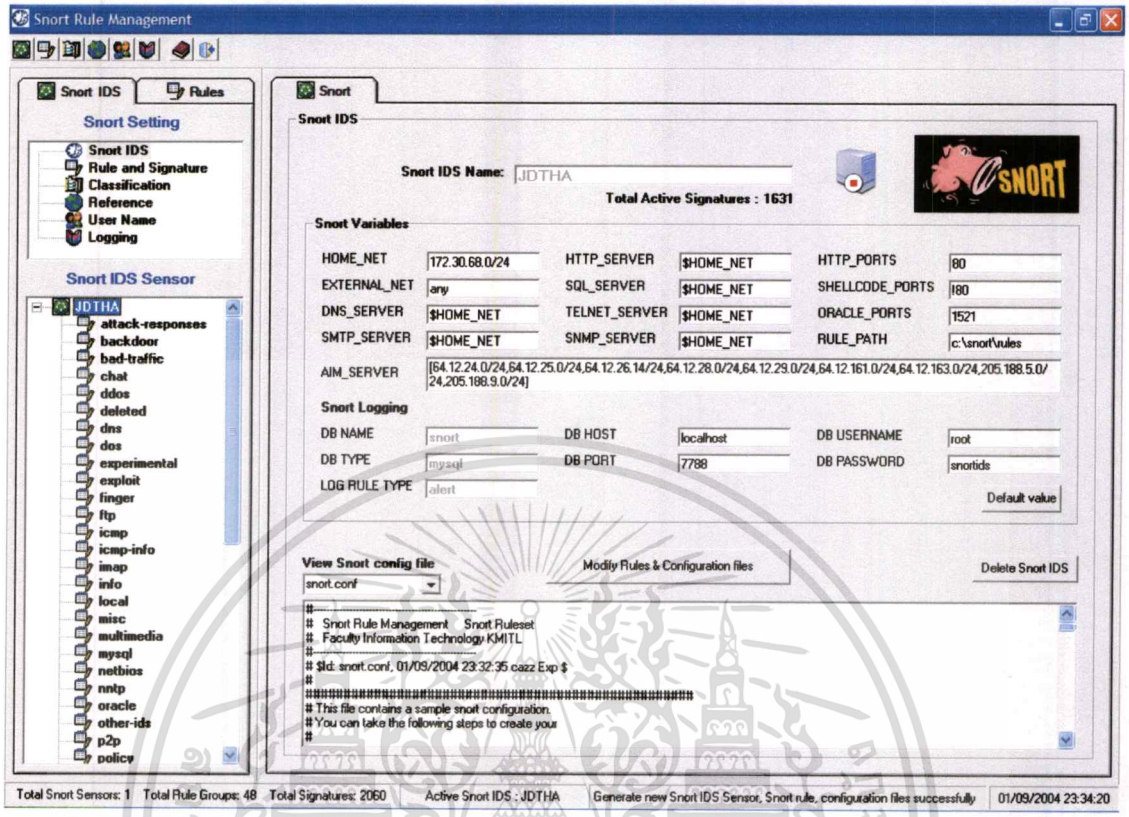
รูปที่ 4.10 การสร้าง Snort IDS Sensor



รูปที่ 4.11 กรอบโต้ตอบการสร้าง Snort IDS Sensor สำเร็จ

- **การเลือก Snort IDS Sensor**

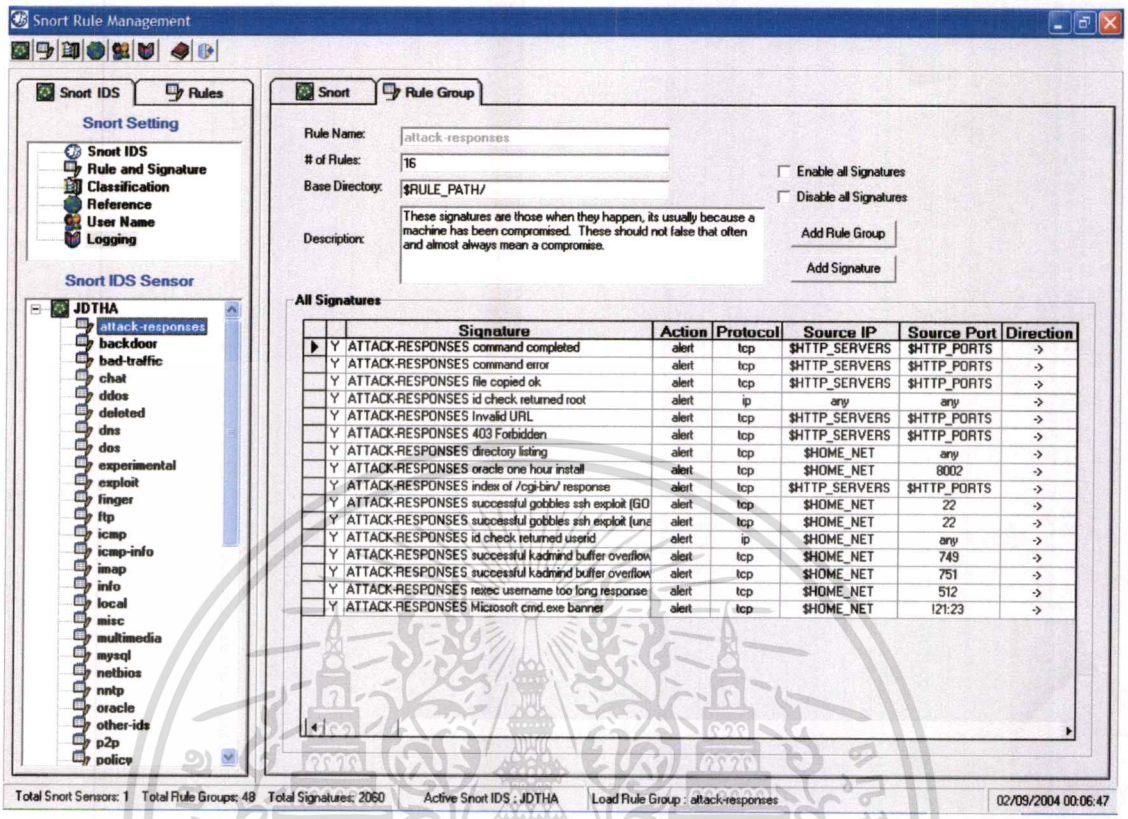
เมื่อผู้ใช้สร้าง Snort IDS แล้วผู้ใช้สามารถดูตัวแปรและแก้ไขค่าต่างๆ ได้โดยการเลือกไปที่ชื่อของ Snort IDS ที่ต้องการ โปรแกรมจะปรากฏแท็บ Snort บน Snort Panel ซึ่งแสดงรายละเอียดทั้งหมดของตัวแปร Snort IDS Sensor เช่น HOME_NET, EXTERNAL_NET, SMTP_SERVER, HTTP_SERVER, SQL_SERVER เป็นต้น ผู้ใช้สามารถดูได้ว่าขณะนี้สถานะของ Snort กำลังทำงานหรือหยุดทำงาน, จำนวนกฎที่ Snort IDS Sensor ทำงานมีจำนวนเท่าไร นอกจากนี้ด้านล่างของโปรแกรมผู้ใช้สามารถเปิดดูคอนฟิกไฟล์ทั้ง 3 ไฟล์ที่ระบบตรวจจับการบุกรุกอ้างอิงถึง ได้แก่ snort.conf, classification.config และ reference.config ที่ใช้ในระบบตรวจจับการบุกรุกดังรูปที่ 4.12 จากรูป Snort IDS Sensor ชื่อ JDTHA มีการโหลดกฎการตรวจจับการบุกรุกทั้งหมด 1,631 กฎจากทั้งหมด 2,060 กฎ



รูปที่ 4.12 การเลือก Snort IDS Sensor

ผู้ใช้สามารถดูสถานะของ Signature แต่ละข้อได้ว่า Signature ข้อใดที่มีการใช้งานเพื่อการตรวจจับการบุกรุก โดยการคลิกที่ Rule Group ภายใต้ชื่อ Snort IDS Sensor ก็จะปรากฏแท็บ Rule Group ซึ่งแสดงกฎทั้งหมดที่อยู่ภายใต้ Rule Group ที่เลือกพร้อมจำนวนกฎ และค่าตัวแปรของ Snort เช่น Signature Name, Action, Protocol, Source IP Address, Source Port, Direction, Destination IP Address, Destination Port, Rule Option

ผู้ใช้สามารถดูได้ว่า Signature ใดมีการใช้งาน (enable) ให้สังเกตจาก column แรกจะแสดงสัญลักษณ์ Y และ สัญลักษณ์ N แสดงว่า Signature นั้นไม่มีการใช้งาน (disable) ในระบบตรวจจับการบุกรุกดังรูปที่ 4.13 นอกจากนี้หากผู้ใช้ต้องการดูรายละเอียดของกฎแต่ละข้อภายใต้ Rule Group ที่เลือกก็ให้ผู้ใช้ดับเบิลคลิกกฎที่ต้องการ



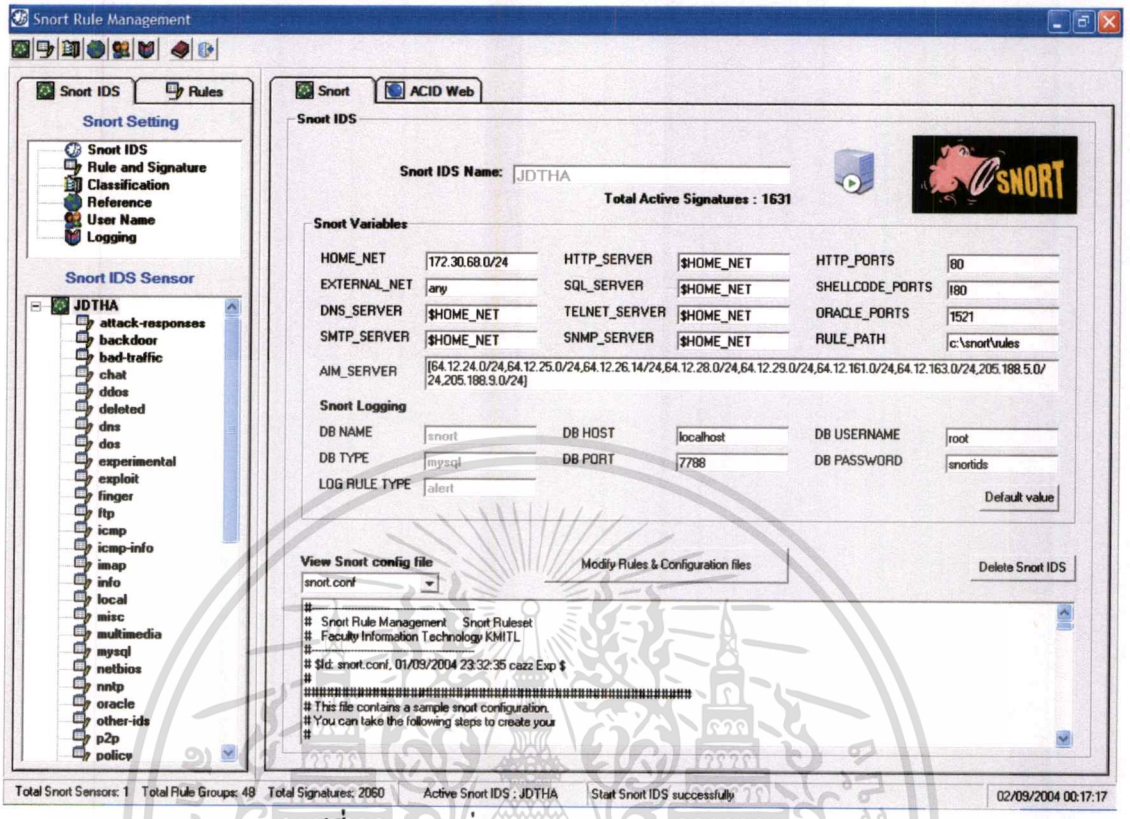
รูปที่ 4.13 สถานะ Signature แต่ละข้อของ Snort IDS Sensor

- **การเริ่มและหยุดทำงานของ Snort IDS Sensor**

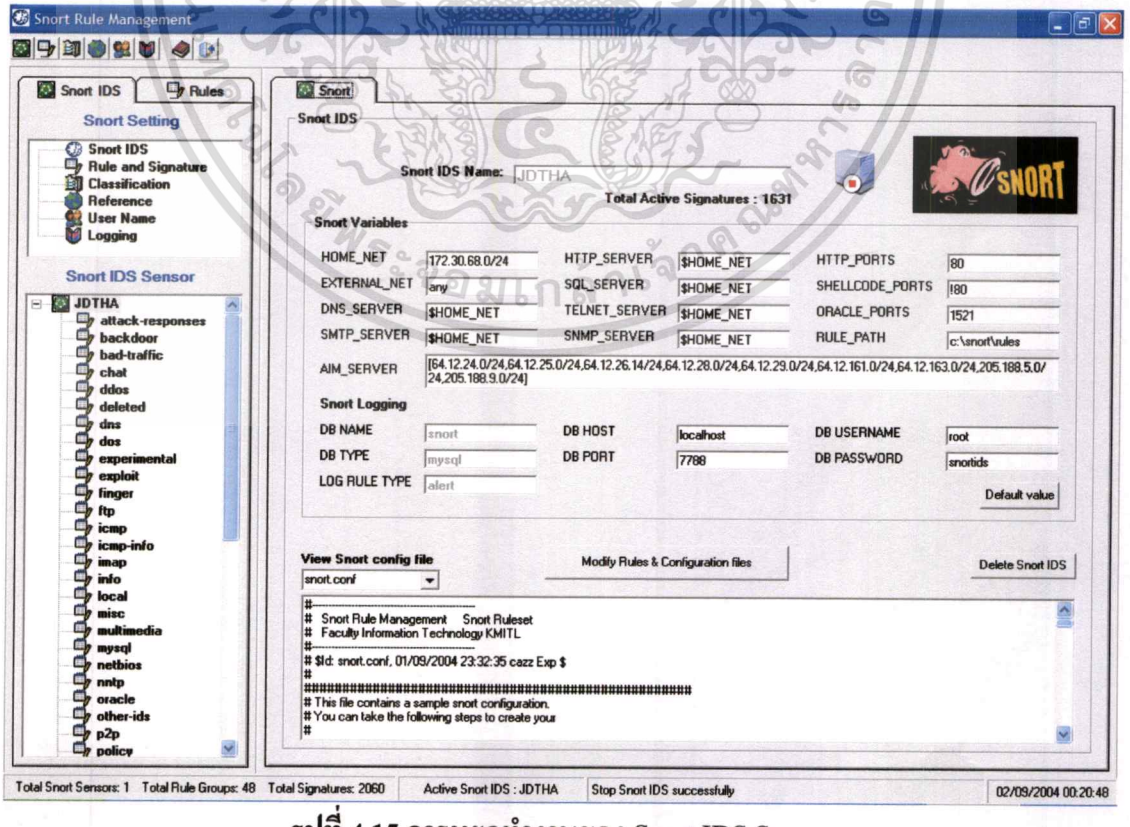
ระบบตรวจจับการบุกรุกด้วย Snort Rule ที่เหมาะสม จะเริ่มทำงานเมื่อผู้ใช้คลิกไปที่รูป Server ซึ่งจะเปลี่ยนรูปจาก stop server เป็น start server ดังรูปที่ 4.14 และหยุดทำงานก็จะเปลี่ยนรูปจาก start server เป็น stop server ดังรูปที่ 4.15 โดยคำสั่งการสั่งงานแสดงดังตารางที่ 4.14

ตารางที่ 4.14 คำสั่งการเริ่มและหยุดทำงานของ Snort

การทำงานของ Snort	คำสั่งในระบบ	สัญลักษณ์
Snort เริ่มทำงาน	net start snort	
Snort หยุดทำงาน	net stop snort	



รูปที่ 4.14 การเริ่มทำงานของ Snort IDS Sensor

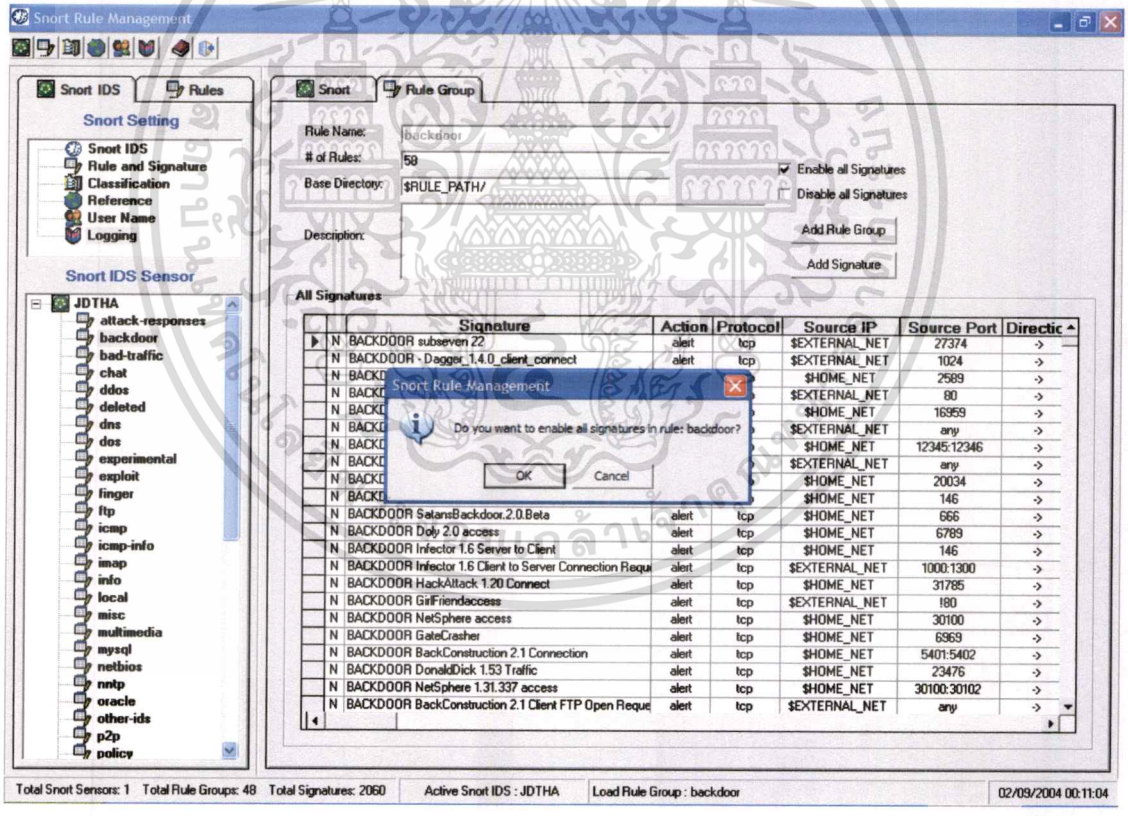


รูปที่ 4.15 การหยุดทำงานของ Snort IDS Sensor

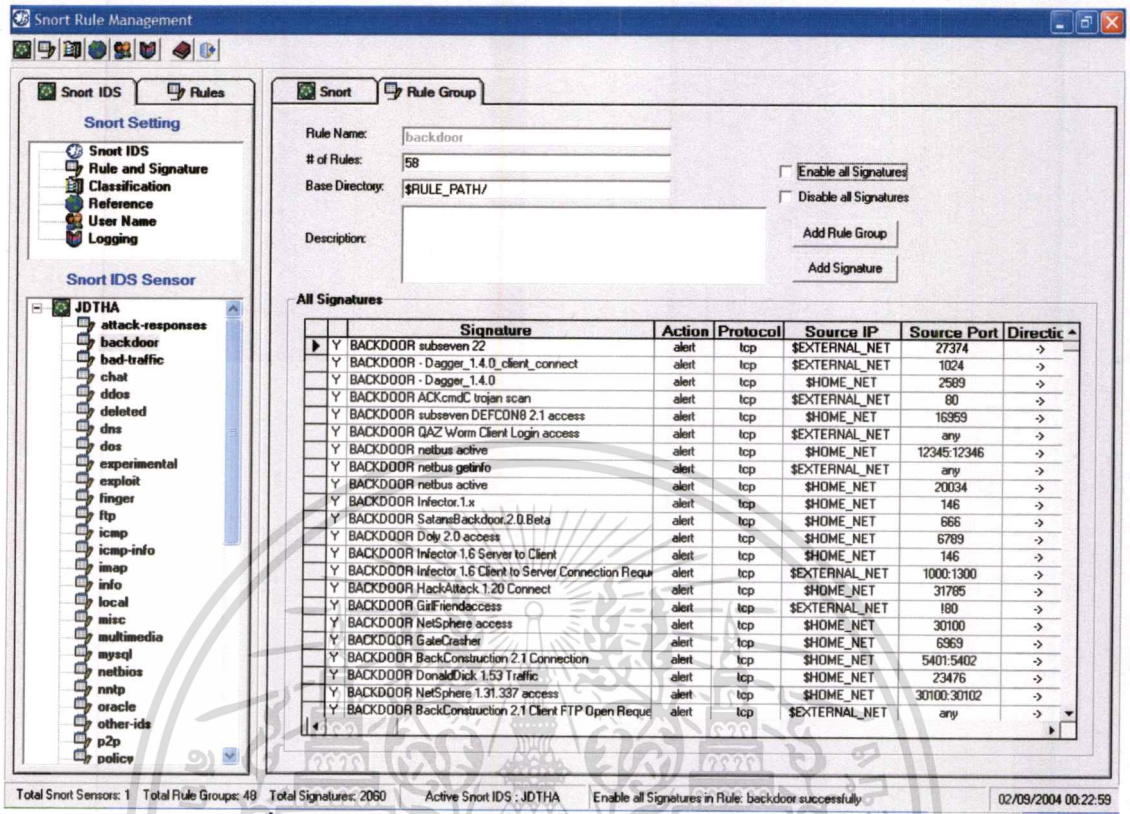
- **Enable และ Disable ทุก Signatures ในกลุ่มกฎ (Rule Group)**

โปรแกรมสามารถกำหนดการใช้งาน (enable) หรือไม่ใช้งาน (disable) Signature เป็นกลุ่มกฎได้ โดยไม่ต้องทำทีละกฎทำให้ไม่เสียเวลา ซึ่งทำได้ด้วยการ checkbox ที่ Enable all Signatures สำหรับการให้ใช้งาน (enable) หรือ Disable all Signatures สำหรับการไม่ให้ใช้งาน (disable) ของกฎทุกข้อภายใต้กลุ่มกฎนั้นๆ ในการ enable และ disable ทุก Signatures ในกลุ่มกฎจะมีกรอบโต้ตอบปรากฏเพื่อให้ผู้ใช้ยืนยันว่าต้องการทำคำสั่งนั้นๆ หรือไม่ ดังรูปที่ 4.16 และ 4.18

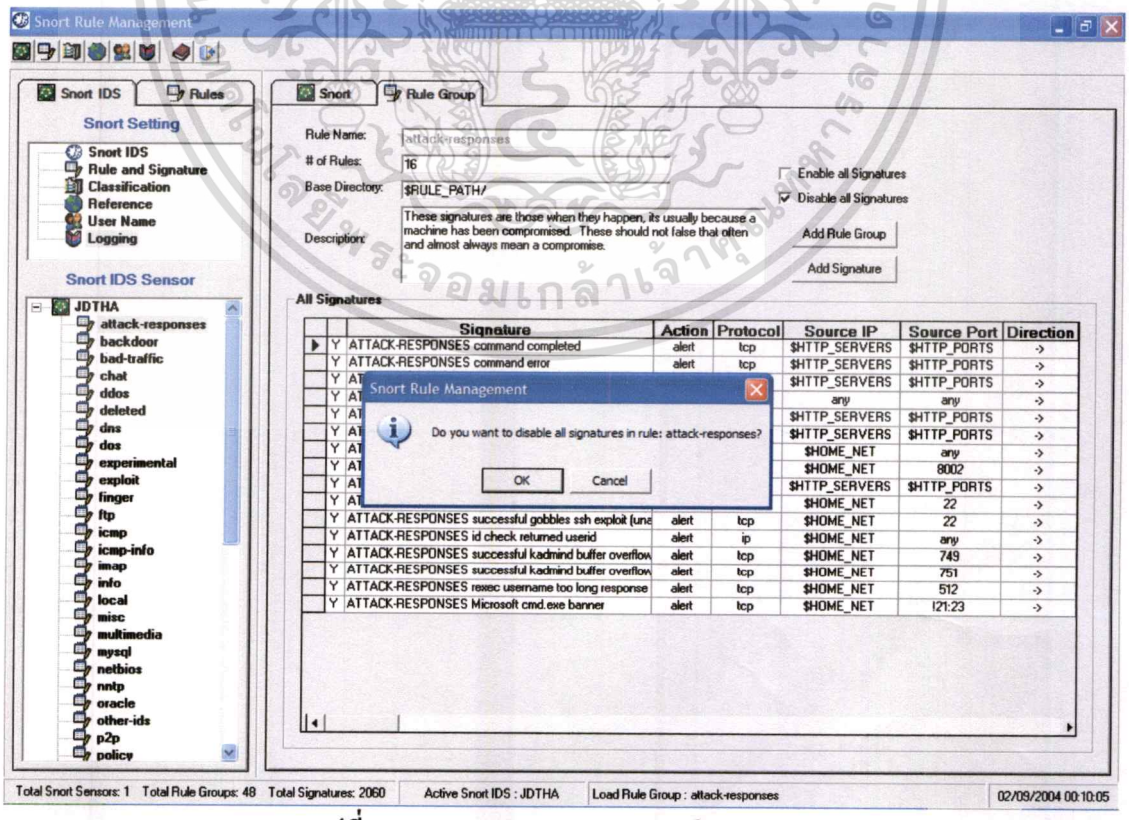
จากรูปที่ 4.17 เมื่อผู้ใช้ยืนยันคำสั่ง enable ทุก signatures ในกลุ่มกฎ BACKDOOR ระบบจะทำการ enable ทุก Signatures ในกลุ่มกฎ BACKDOOR โดยสังเกตจาก column แรกแสดงเปลี่ยนจาก สัญลักษณ์ N เป็น สัญลักษณ์ Y เพื่อบอกว่า Signature นั้นพร้อมใช้งานในระบบเพื่อตรวจจับการบุกรุก



รูปที่ 4.16 Enable ทุก Signatures ในกลุ่มกฎ



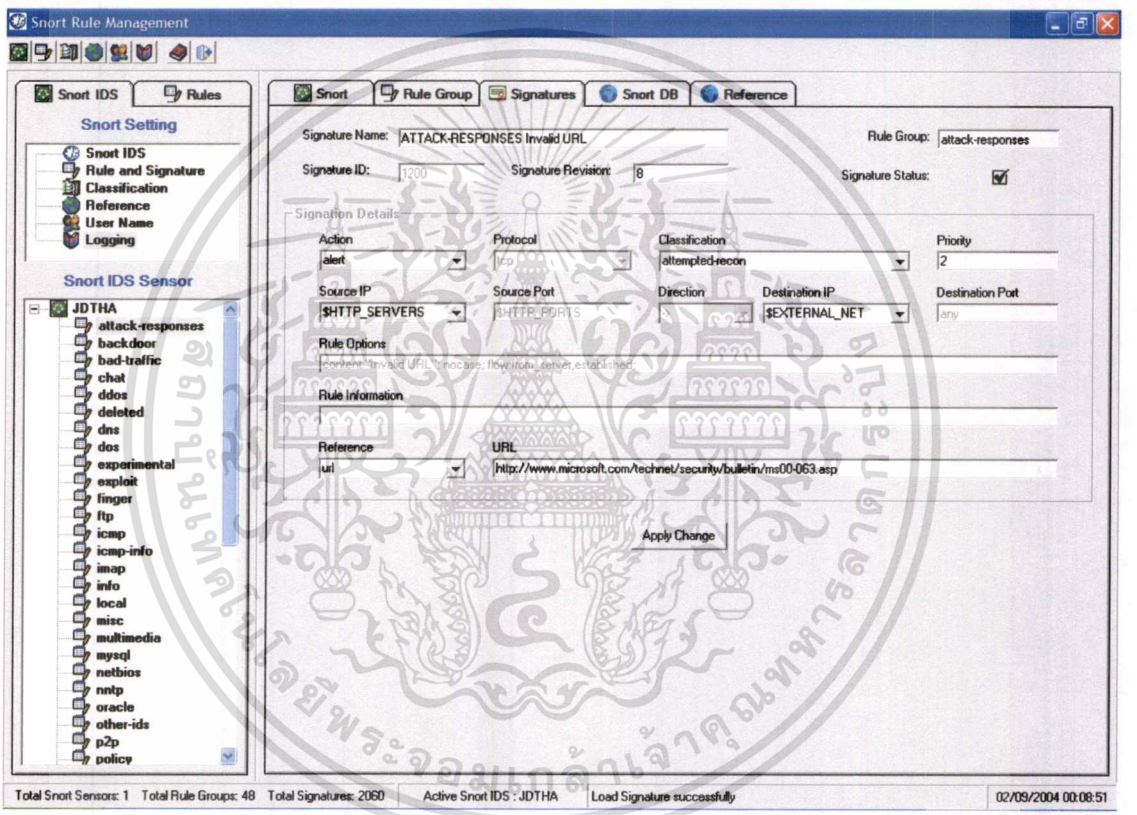
รูปที่ 4.17 ผลลัพธ์ของการ Enable ทุก Signatures ในกลุ่มกฎ



รูปที่ 4.18 Disable ทุก Signatures ในกลุ่มกฎ

- **Enable และ Disable Signature แต่ละข้อในกลุ่มกฎ (Rule Group)**

นอกจากระบบสามารถกำหนดการให้ใช้งาน (enable)หรือไม่ให้ใช้งาน (disable) Signature เป็นกลุ่มกฎได้ ระบบยังรองรับให้ผู้ใช้กำหนดให้ใช้งาน (enable) หรือไม่ใช้งาน (disable) Signature เป็นบางกฎได้ โดยการดับเบิลคลิกเลือก Signature ที่ต้องการ ก็จะปรากฏแท็บ Signature ให้ผู้ใช้คลิก checkbox โดยถ้ามีการ checkbox (☑) กฎนั้นก็จะ enable แต่ถ้าไม่มีการ checkbox (☐) กฎนั้นก็จะ disable ดังรูปที่ 4.19

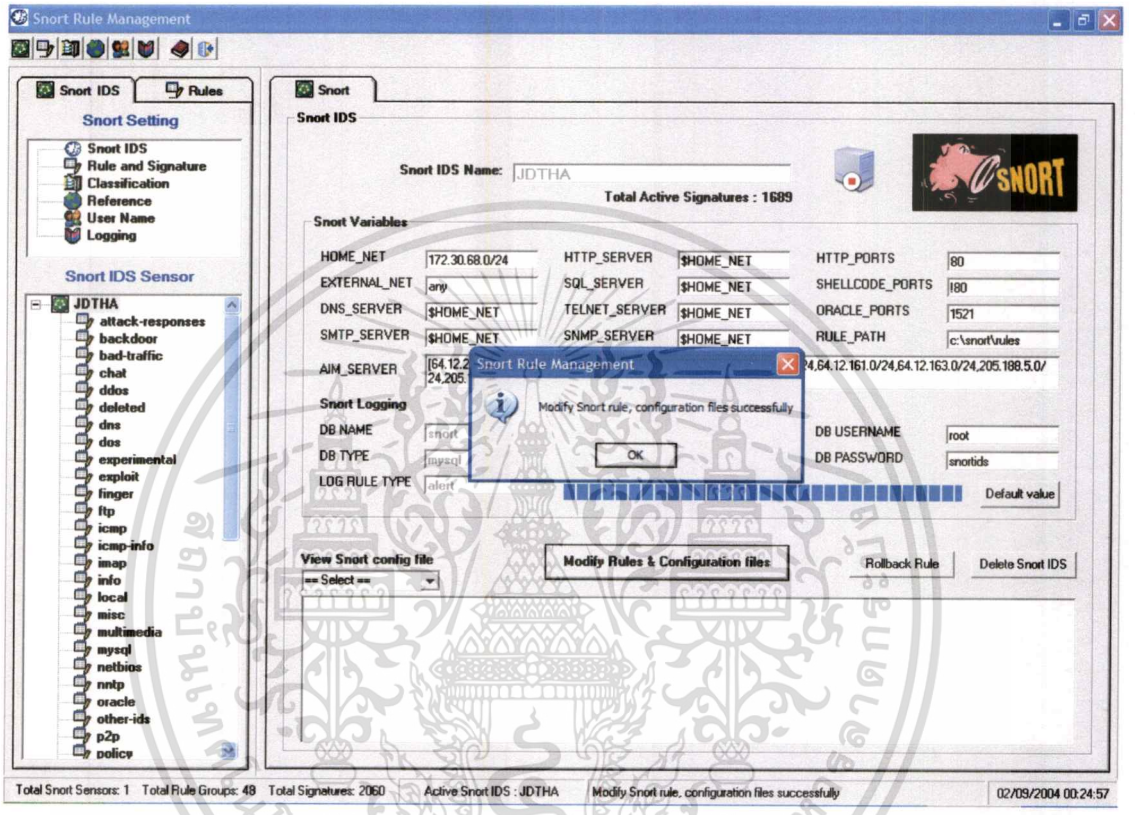


รูปที่ 4.19 Enable และ Disable Signature แต่ละข้อในกลุ่มกฎ

- **การปรับเปลี่ยนค่าต่างๆ ใน Snort IDS Sensor**

หลังจากที่ผู้ใช้มีการแก้ไขค่าตัวแปรต่างๆ Enable หรือ Disable Signature แล้วระบบจะยังไม่ได้รับการปรับเปลี่ยนในทันที เนื่องจากการปรับเปลี่ยนแต่ละครั้งต้องหยุดการทำงานของระบบตรวจจับการบุกรุก หากเปลี่ยนค่าใดค่าหนึ่งแล้วต้องหยุดการทำงานของระบบตรวจจับทุกครั้งก็ไม่เหมาะสม จึงให้ผู้ใช้ปรับเปลี่ยนจนเสร็จ แล้วกดปุ่ม Modify Rules & Configuration files (ตัวอักษรบนปุ่มมีลักษณะตัวหนา) เพียงครั้งเดียว ระบบก็จะหยุดการตรวจจับการบุกรุก แล้วรับค่าการปรับ

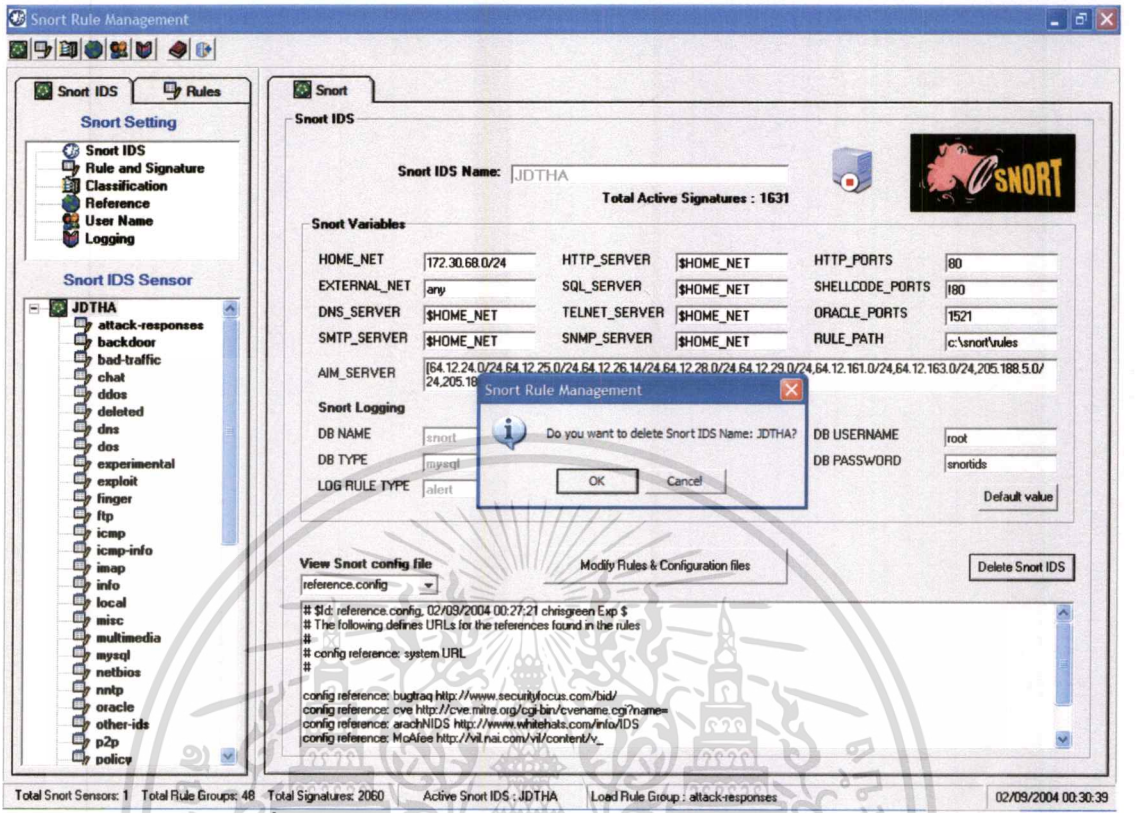
เปลี่ยนเหล่านั้น จากนั้นก็สร้างรูลไฟล์และคอนฟิกไฟล์ใหม่ เมื่อปรับเปลี่ยนเสร็จเรียบร้อยแล้ว ระบบจะปรากฏกรอบโต้ตอบเพื่อแจ้งให้ผู้ใช้ทราบดังรูปที่ 4.20 ระบบตรวจจับการบุกรุกจะไม่เริ่มทำงานต่อทันทีเมื่อปรับเปลี่ยนค่าเสร็จ ผู้ใช้ต้องคลิกที่รูป server เพื่อสั่งให้ระบบทำงาน



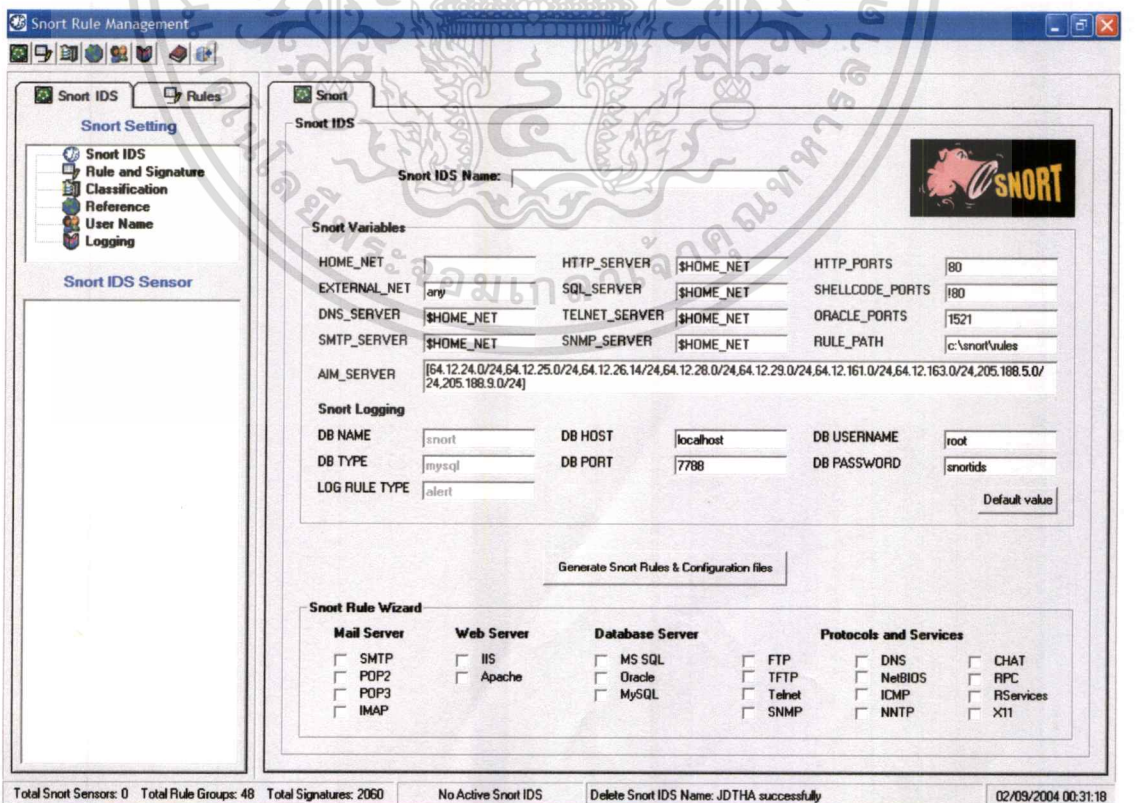
รูปที่ 4.20 การปรับเปลี่ยนค่าต่างๆ ใน Snort IDS Sensor

- **การลบ Snort IDS Sensor**

ระบบตรวจจับการบุกรุกด้วย Snort Rule ที่เหมาะสม มีฟังก์ชันการลบ Snort IDS Sensor ให้ผู้ใช้ใช้งานเพื่อลบข้อมูล Snort IDS Sensor ออกจากฐานข้อมูล โดยให้ผู้ใช้เลือกไปที่ชื่อ Snort IDS Sensor จากนั้นกดปุ่ม Delete Snort IDS ระบบก็จะปรากฏกรอบโต้ตอบให้ผู้ใช้ยืนยันการลบ ดังรูปที่ 4.21 หากผู้ใช้ยืนยันการลบ ระบบก็จะลบข้อมูลที่เกี่ยวข้องกับชื่อ Snort IDS Sensor นั้นจากฐานข้อมูลทั้งหมด โดยลบในทุกตารางที่เกี่ยวข้อง เมื่อระบบลบข้อมูลที่เกี่ยวข้องหมดแล้ว ระบบก็เข้าสู่หน้าจอการสร้าง Snort IDS Sensor ใหม่ดังรูปที่ 4.22



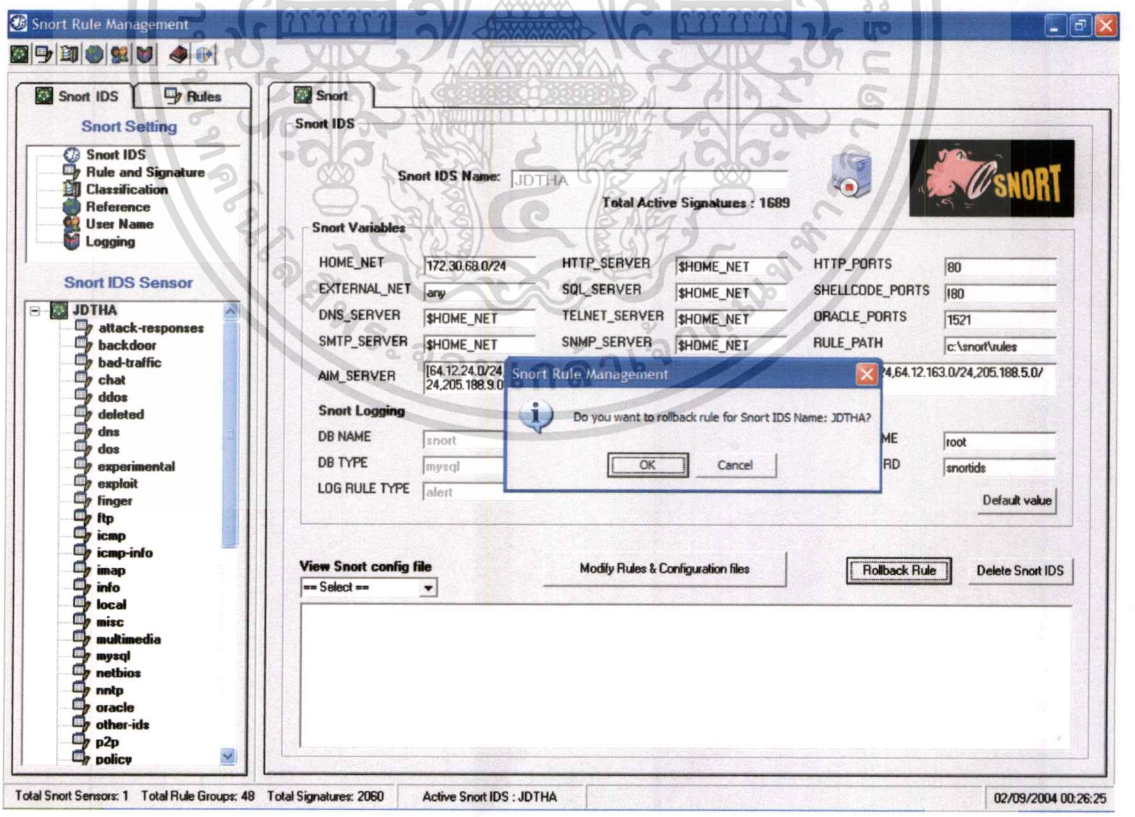
รูปที่ 4.21 กรอบโต้ตอบเพื่อยืนยันการลบ Snort IDS Sensor



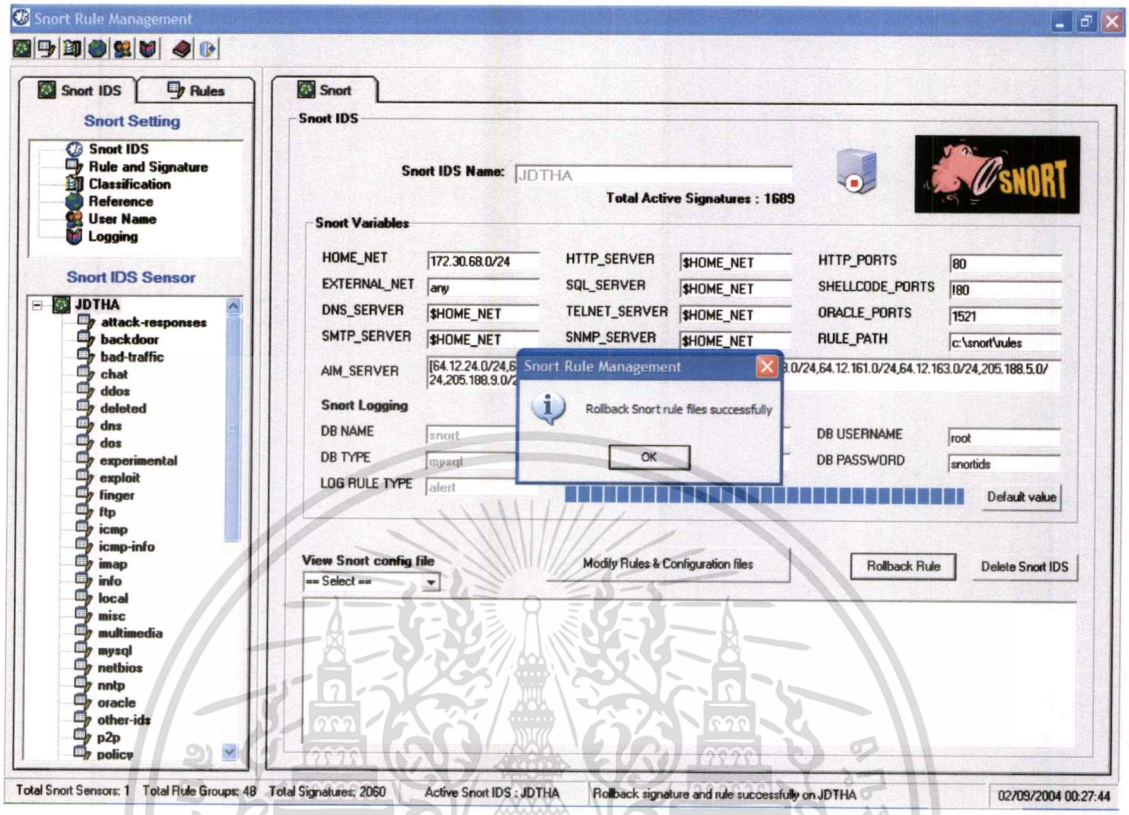
รูปที่ 4.22 หน้าจอการสร้าง Snort IDS Sensor ใหม่เมื่อลบ Snort IDS Sensor สำเร็จ

- **การกู้ Signature กลับ (Rollback Signature)**

เมื่อผู้ใช้มีการเปลี่ยนแปลงสถานะ Signature โดย Enable หรือ Disable ทุก Signatures ในกลุ่มกฎ (Rule Group) หรือเฉพาะบาง Signature ในกลุ่มกฎ ระบบจะเก็บข้อมูลสถานะ Signature เดิมลงฐานข้อมูลในตาราง RollbackSig โดยเก็บชื่อ Snort IDS Sensor และ Signature ID ที่มีการเปลี่ยนแปลง พร้อมทั้งปรากฏปุ่ม Rollback Rule เพื่อให้ผู้ใช้สามารถกู้สถานะเดิมของ Signature กลับคืนมาได้ เมื่อผู้ใช้ต้องการกู้สถานะเดิมของ Signature ให้กดปุ่ม Rollback Rule ระบบก็จะปรากฏกรอบโต้ตอบเพื่อให้ผู้ใช้ยืนยันการกู้ Signature กลับ ดังรูปที่ 4.23 หากผู้ใช้ยืนยันการกู้กลับระบบก็จะหยุดการตรวจจับการบุกรุก แล้วนำข้อมูลจากตาราง RollbackSig มาใช้ในการกู้ Signature กลับ จากนั้นระบบก็สร้างคอนฟิกไฟล์และรูลไฟล์ใหม่ แล้วระบบทำการลบข้อมูลในตาราง RollbackSig ทั้งหมดทิ้งไป เมื่อการกู้ Signature กลับสำเร็จจะปรากฏกรอบโต้ตอบเพื่อแจ้งให้ผู้ใช้ทราบ ดังรูปที่ 4.24 หากระบบมีข้อมูลในตาราง RollbackSig แต่ผู้ใช้ไม่ต้องการการกู้ Signature กลับ แต่มีการเปลี่ยนแปลงสถานะ Signature อีก ระบบจะลบข้อมูลเดิมในตาราง RollbackSig ทั้งหมด ก่อนที่ระบบจะเพิ่มข้อมูลการกู้ Signature กลับใหม่ลงตาราง RollbackSig



รูปที่ 4.23 กรอบโต้ตอบเพื่อยืนยันการกู้ Signature เก่ากลับคืน



รูปที่ 4.24 กรอบโต้ตอบแสดงการกู้ Signature เก่ากลับคืนสำเร็จ

4.6.2 ฟังก์ชัน Rule และ Signature

จากตัวอย่างคอนฟิกไฟล์ snort.conf ในขั้นตอนที่ 4 มีการอ้างอิงถึงรูลไฟล์ต่างๆ ซึ่งภายในรูลไฟล์ประกอบด้วย Signature โดยรูปแบบของ Snort Signature แสดงดังรูปที่ 4.25

Action	Protocol	Source Address	Source Port	Direction	Destination Address	Destination Port

Signature Name	Rule Option & Rule Information	Reference	Class Type	Signature ID	Revision

รูปที่ 4.25 รูปแบบของ Snort Signature

จากรูปที่ 4.25 รูปแบบของ Snort Signature แต่ละข้อสามารถแบ่งออกได้ 13 ฟิลด์ รายละเอียดของแต่ละฟิลด์แสดงดังตารางที่ 4.15

ตารางที่ 4.15 รูปแบบของ Snort Signature

ฟิลด์	คำอธิบาย	จำเป็นต้องมีข้อมูล
Action	เหตุการณ์ที่เกิดขึ้นเมื่อตรวจพบการบุกรุก	☑
Protocol	โพรโทคอลที่ตรวจจับ	☑
Source Address	IP Address ต้นทาง	☑
Source Port	พอร์ตต้นทาง	☑
Direction	ทิศทาง	☑
Destination Address	IP Address ปลายทาง	☑
Destination Port	พอร์ตปลายทาง	☑
Signature Name	ชื่อ Signature	☑
Rule Option & Rule Information	รูปแบบการตรวจจับการบุกรุก (Detection Patterns)	☑
Reference	อ้างอิงไปยัง Website ผู้สร้างกฎ	-
Class Type	ประเภทของ Signature	☑
Signature ID	หมายเลข Signature	☑
Revision	เวอร์ชันที่ปรับเปลี่ยน	☑

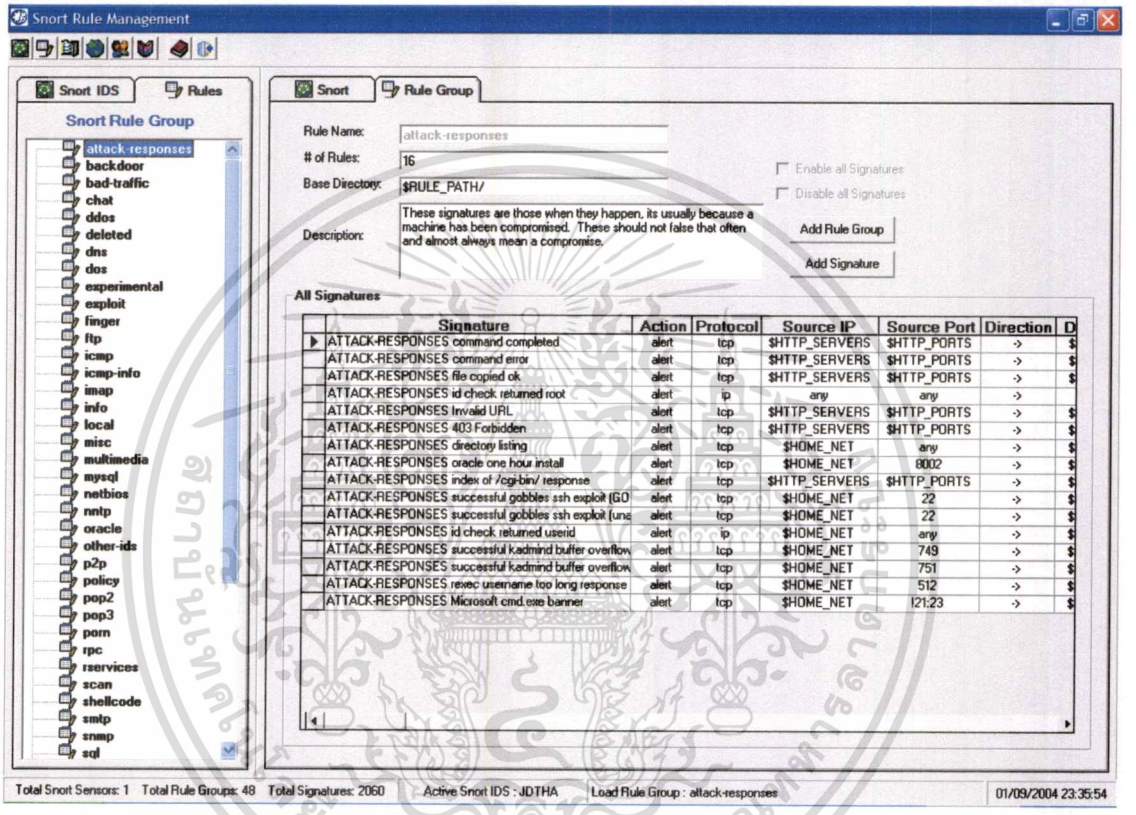
จากรูปที่ 4.26 แสดงตัวอย่าง Snort Rule สามารถนำมาแสดงผลบนโปรแกรมได้ดังรูปที่ 4.27 ซึ่งผู้ใช้งานสามารถแก้ไขค่าตัวแปรต่างๆ ของกฎได้

```
alert tcp $EXTERNAL_NET any -> $HOME_NET 21 (msg:"FTP iss scan";
flow:to_server,established; content:"pass -iss@iss";
reference:arachNIDS,331; classtype:suspicious-login; sid:354;
rev:4;)
```

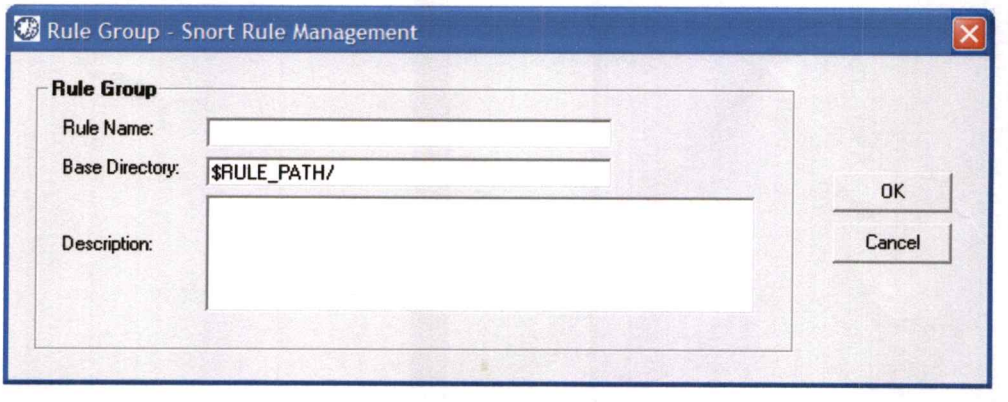
รูปที่ 4.26 ตัวอย่าง Snort Rule

โปรแกรมออกแบบมาให้ผู้ใช้ทำงานกับกลุ่มกฎ (Rule Group) ได้สะดวกมากขึ้น โดยสามารถเข้าไปดู แก้ไขกฎแต่ละข้อใน Rule Group ได้ สามารถแสดงจำนวนกฎทั้งหมดภายใน Rule Group ผู้ใช้สามารถเพิ่ม Rule Group ได้โดยกดปุ่ม Add Rule Group จะปรากฏหน้าต่าง Rule Group

ดังรูปที่ 4.28 โปรแกรมมีความสามารถในการตรวจสอบได้ว่า Rule Group ที่เพิ่มมาใหม่นั้นซ้ำกับ Rule Group ที่มีอยู่เดิมหรือไม่ หากมีการซ้ำก็จะมีกรอบโต้ตอบปรากฏขึ้นมาแจ้งให้ผู้ใช้ทราบ แต่หากไม่ซ้ำโปรแกรมก็จะปิดหน้าต่าง Rule Group แล้วเพิ่มชื่อ Rule Group ใหม่ในแท็บ Rules ให้โดยอัตโนมัติ

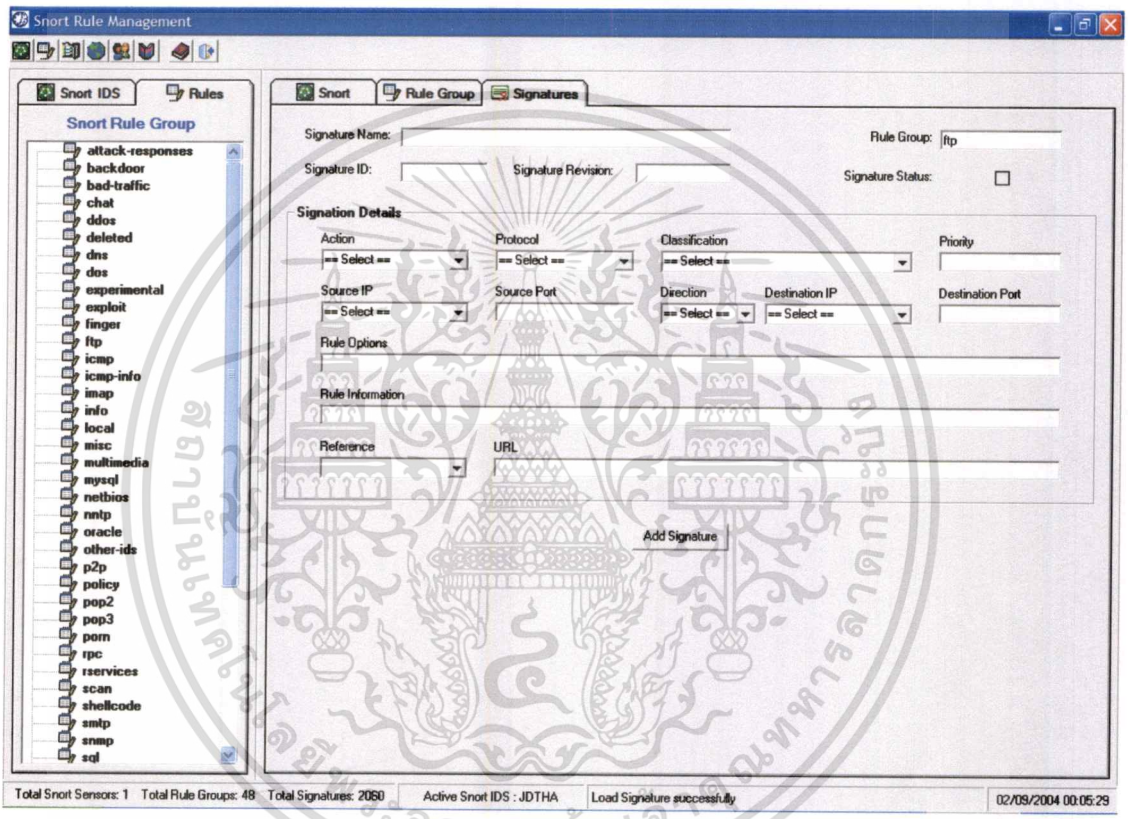


รูปที่ 4.27 หน้าจอ Rule Group



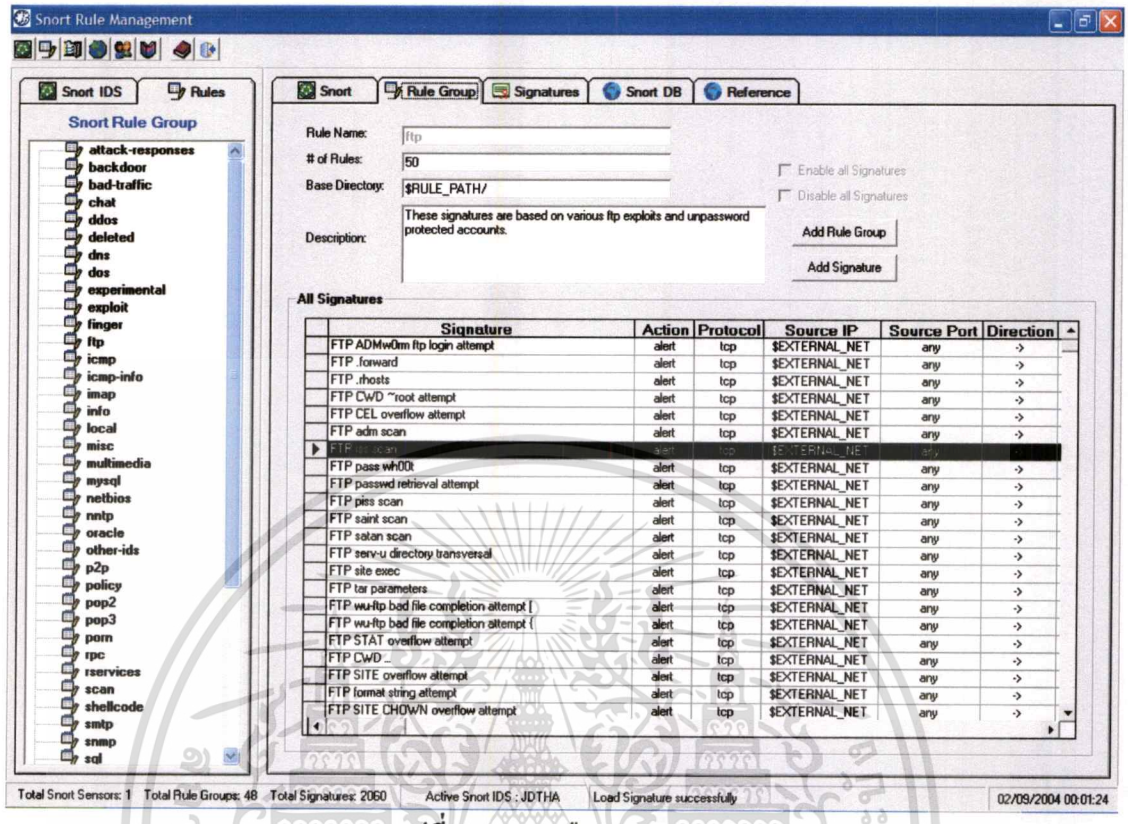
รูปที่ 4.28 การเพิ่ม Rule Group

โปรแกรมรองรับให้ผู้ใช้สามารถเพิ่ม Signature ใหม่ใน Rule Group ใดๆได้โดยกดปุ่ม Add Signature จะปรากฏแท็บ Signature ดังรูปที่ 4.29 ให้ผู้ใช้ป้อนข้อมูล Signature ใหม่แล้วกดปุ่ม Add Signature หากไม่มีข้อผิดพลาดจะปรากฏ Signature ใหม่ในแผ่นข้อมูล (Data grid) ภายใต้ Rule Group นั้น แต่หากมีข้อผิดพลาดเกิดขึ้นก็จะมีกรอบโต้ตอบปรากฏแจ้งให้ผู้ใช้ทราบถึงข้อผิดพลาดเช่น ผู้ใช้ป้อนข้อมูลไม่ครบ, Signature ID ซ้ำกัน เป็นต้น

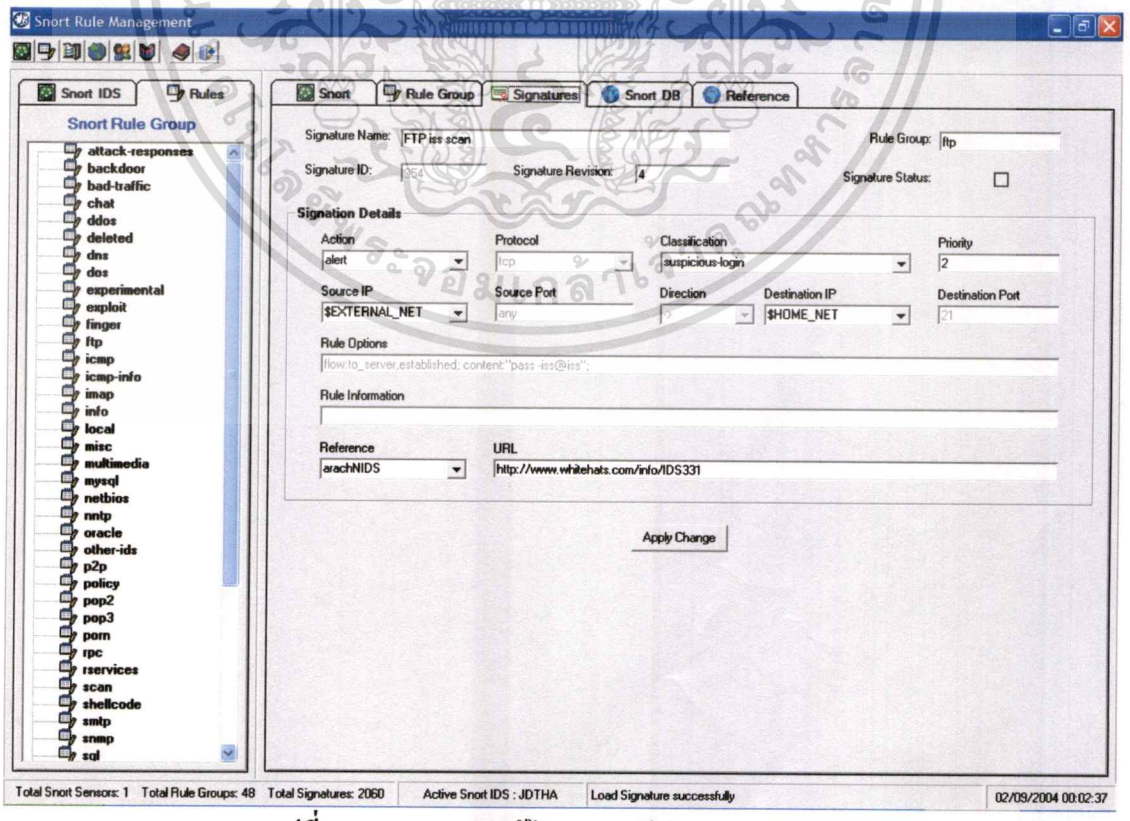


รูปที่ 4.29 การเพิ่ม Signature

ผู้ใช้สามารถดูและแก้ไขรายละเอียดบางฟิลด์ของ Signature แต่ละข้อได้ จากรูปที่ 4.30 เป็นการเลือก Signature : FTP iis sacn ภายใต้ Rule Group : ftp เมื่อผู้ใช้ต้องการดูหรือแก้ไขรายละเอียดของ Signature นี้ให้ดับเบิลคลิก Signature นี้ โปรแกรมจะปรากฏแท็บ Signature ขึ้นมาให้ผู้ใช้โดยแสดงรายละเอียดของ Signature นี้และพร้อมที่จะให้ผู้ใช้เปลี่ยนแปลงค่าได้ ดังรูปที่ 4.31 เมื่อผู้ใช้แก้ไขรายละเอียดของ Signature นี้เสร็จ ให้กดปุ่ม Apply Change เพื่อบันทึกการเปลี่ยนแปลงลงฐานข้อมูลในระบบ

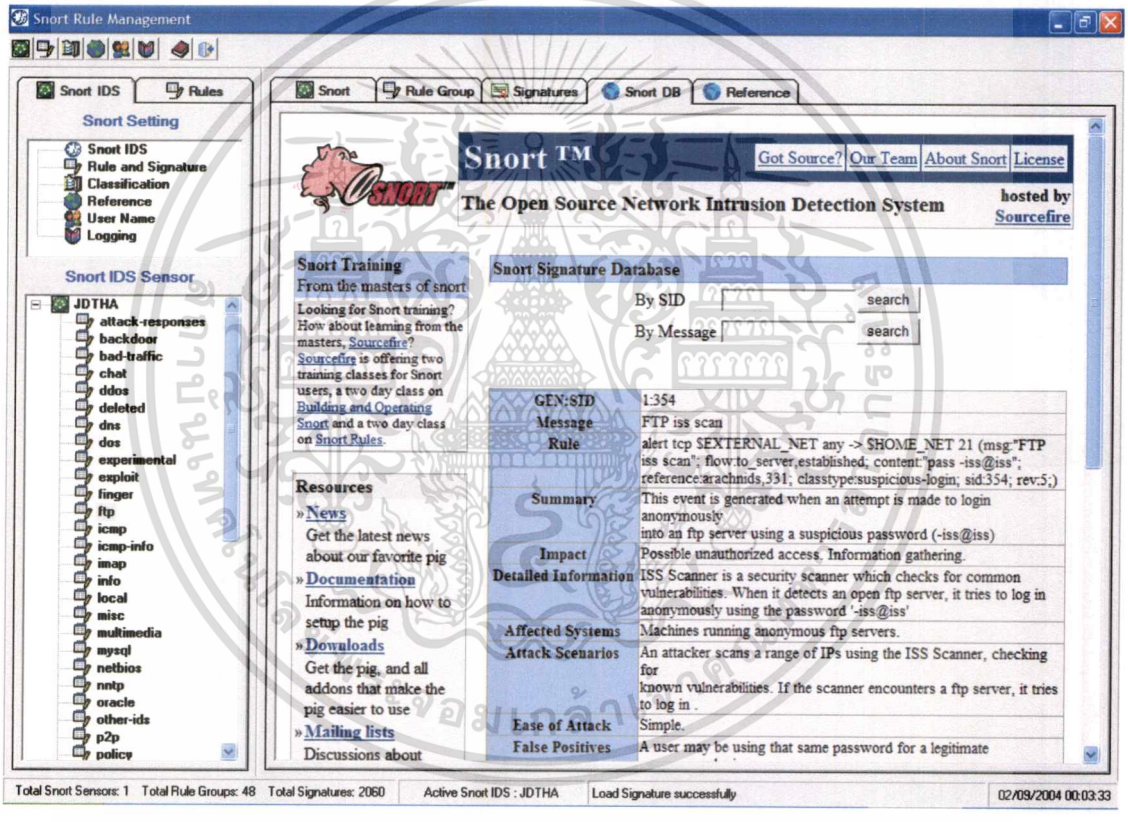


รูปที่ 4.30 การเลือก Signature



รูปที่ 4.31 การดูและแก้ไขรายละเอียดของ Signature

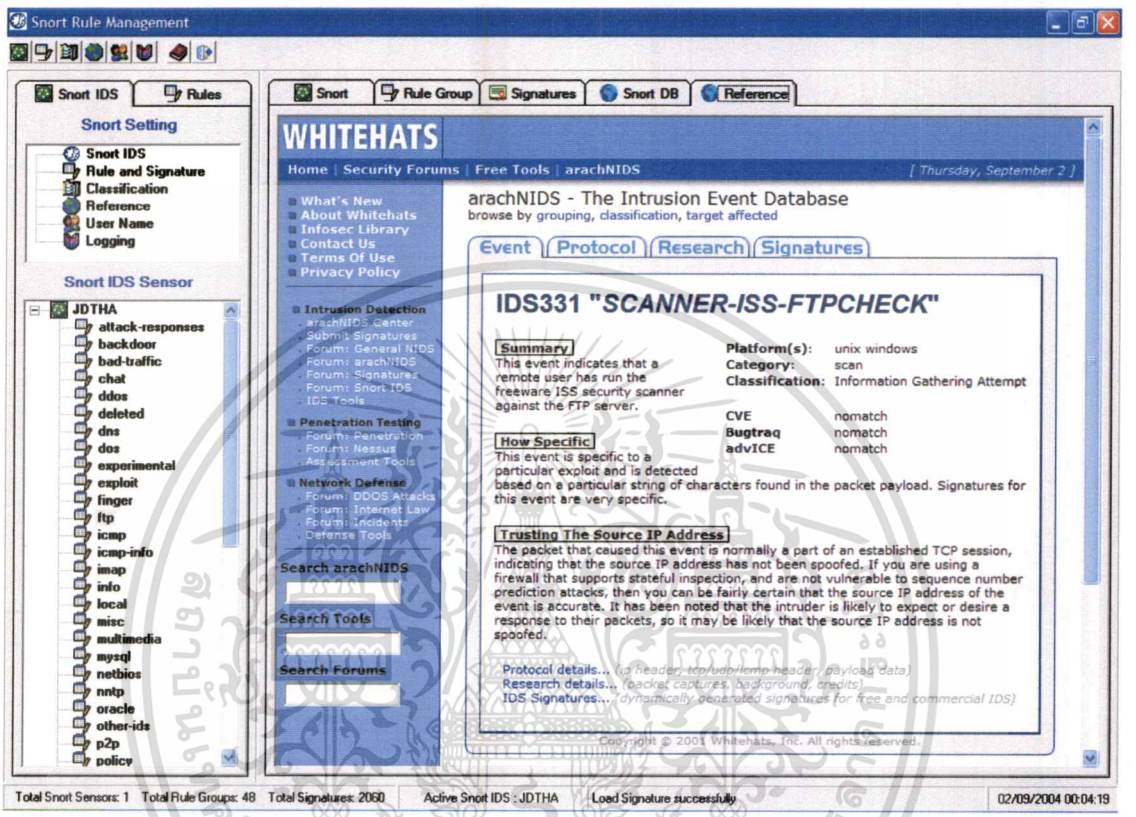
นอกจากนี้โปรแกรมปรากฏแท็บ Signature แล้วยังมีอีก 2 แท็บที่ปรากฏขึ้นมาพร้อมกัน ด้วยคือ แท็บ Snort DB ซึ่งเปิด Web page เพื่อแสดงรายละเอียดทั้งหมดของ Signature โดยอ้างอิงไปยัง Website ของ Snort ดังรูปที่ 4.32 แสดงหน้าจอ Snort DB ซึ่ง Signature แต่ละข้อจะมีค่า Signature ID หรือ SID โดยโปรแกรมสร้างการเชื่อมโยงไปยัง Snort Website ด้วย URL: http://www.snort.org/snort-db/?sid= แล้วตามด้วยค่า SID ของ Signature แต่ละข้อ จากรูปที่ 4.32 Signature : FTP iis scan มีค่า SID เป็น 354 ดังนั้นโปรแกรมจึงสร้างการเชื่อมโยงไปยัง Website ด้วย URL : http://www.snort.org/snort-db/?sid=**354**



รูปที่ 4.32 Snort Database Web Site (http://www.snort.org/snort-db/?sid=354)

อีกหนึ่งแท็บที่ปรากฏขึ้นคือแท็บ Reference ซึ่งแท็บนี้ปรากฏก็ต่อเมื่อ Signature นั้นมีข้อมูล Reference อยู่ในฐานข้อมูล เพื่ออ้างอิงไปยัง Website อื่นๆในการให้ข้อมูลเพิ่มเติมจากแท็บ Snort DB โดยข้อมูลมีความละเอียดมากขึ้น จากรูปที่ 4.33 แสดงหน้าจอ Reference ของ Signature : FTP iis scan ที่มีค่า ReferenceID : arachnids ซึ่งมี Base URL : http://www.whitehats.com/info/IDS

และ Signature นี้มีค่า Reference Value ในฐานข้อมูลเท่ากับ 331 ดังนั้นโปรแกรมจึงสร้างการเชื่อมโยงไปยัง URL : http://www.whitehats.com/info/IDS**331**



รูปที่ 4.33 Snort URL Reference ของ Signature (http://www.whitehats.com/info/IDS331)

4.6.3 ฟังก์ชัน Classification

Snort มีคอนฟิกไฟล์อีก 2 ไฟล์ที่มีการเรียกใช้ใน snort.conf คือ classification.config และ reference.config

```
#---------------------------------------------------
#          Classification Config
#---------------------------------------------------
config classification: not-suspicious,Not Suspicious Traffic,3
config classification: unknown,Unknown Traffic,3
config classification: bad-unknown,Potentially Bad Traffic,2
config classification: successful-recon-limited,Information Leak,2
config classification: attempted-dos,Attempted Denial of Service,2
config classification: successful-dos,Denial of Service,2
```

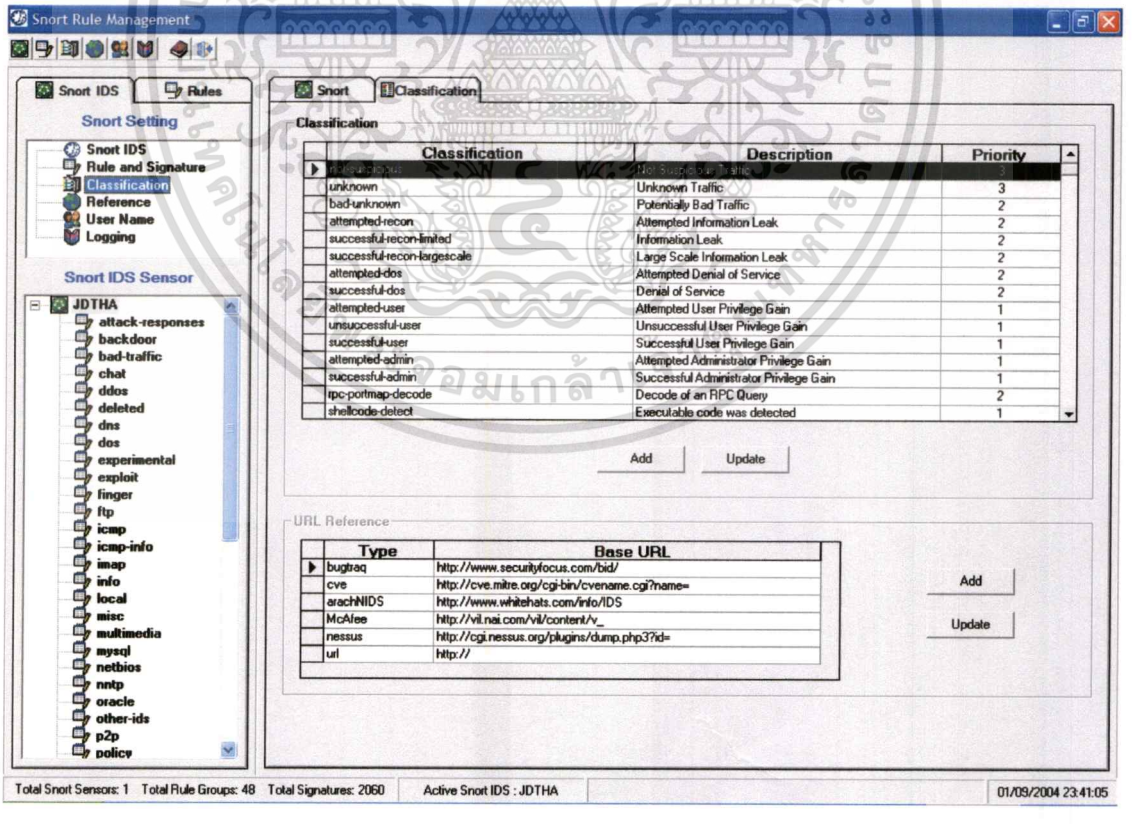
รูปที่ 4.34 ตัวอย่างไฟล์ classification.config

จากรูปที่ 4.34 ตัวอย่างของไฟล์ classification.config ที่ Snort ใช้เพื่อจัดแบ่งประเภทของกฎแต่ละข้อว่าอยู่ในประเภทใด และกฎมีระดับความสำคัญเท่าใด ตัวอย่างเช่น

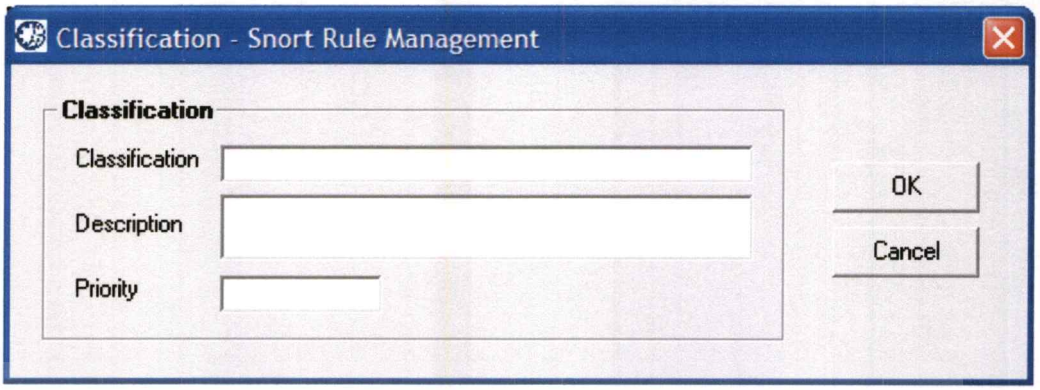
config classification: attempted-dos,Attempted Denial of Service,2

หมายถึงประเภทของกฎที่ชื่อ attempted-dos โดยมีคำอธิบายของกฎประเภทนี้คือ Attempted Denial of Service ซึ่งมีระดับความสำคัญของกฎเท่ากับ 2

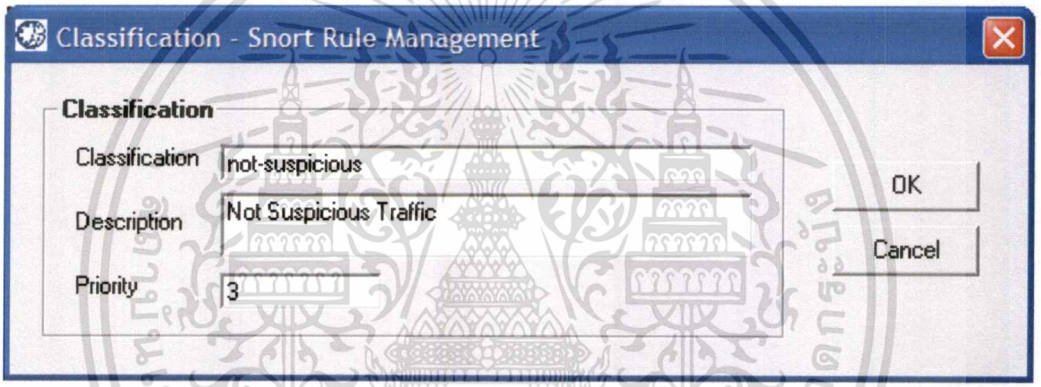
โปรแกรม Snort Rule Management มีส่วนการเพิ่มและแก้ไข Classification ในฐานข้อมูล โดยเลือกไปที่เมนู Classification ก็จะปรากฏแท็บ Classification ดังรูปที่ 4.35 หากผู้ใช้ต้องการเพิ่มหรือแก้ไขก็กดปุ่ม Add หรือ Update ตามลำดับ ดังรูปที่ 4.36 และ 4.37 โดยในการเพิ่มหรือแก้ไขค่านั้นโปรแกรมออกแบบมาให้มีความสามารถในการตรวจสอบชื่อ Classification ว่าซ้ำกับข้อมูลเดิมในฐานข้อมูลหรือไม่ หากซ้ำก็จะปรากฏกรอบโต้ตอบเพื่อแจ้งให้ผู้ใช้ทราบ แล้วให้ผู้ใช้ป้อนชื่อ Classification ใหม่



รูปที่ 4.35 หน้าจอ Classification



รูปที่ 4.36 การสร้าง Classification



รูปที่ 4.37 การแก้ไข Classification

4.6.4 ฟังก์ชัน Reference

คอนฟิกไฟล์อีกไฟล์ที่ Snort.conf อ้างถึงคือ reference.config ดังรูปที่ 4.38 ซึ่งเป็นตัวอย่างของไฟล์ reference.config ที่ระบบตรวจจับการบุกรุกด้วย Snort ใช้เพื่อทำการอ้างอิงสำหรับกฎแต่ละข้อที่มีการอ้างถึงแหล่งที่มาของกฎ ซึ่งกฎแต่ละข้อที่มีการอ้างอิงถึงนั้นจะมีค่าที่ชื่อว่า reference ปรากฏอยู่ในกฎ ตัวอย่างเช่น

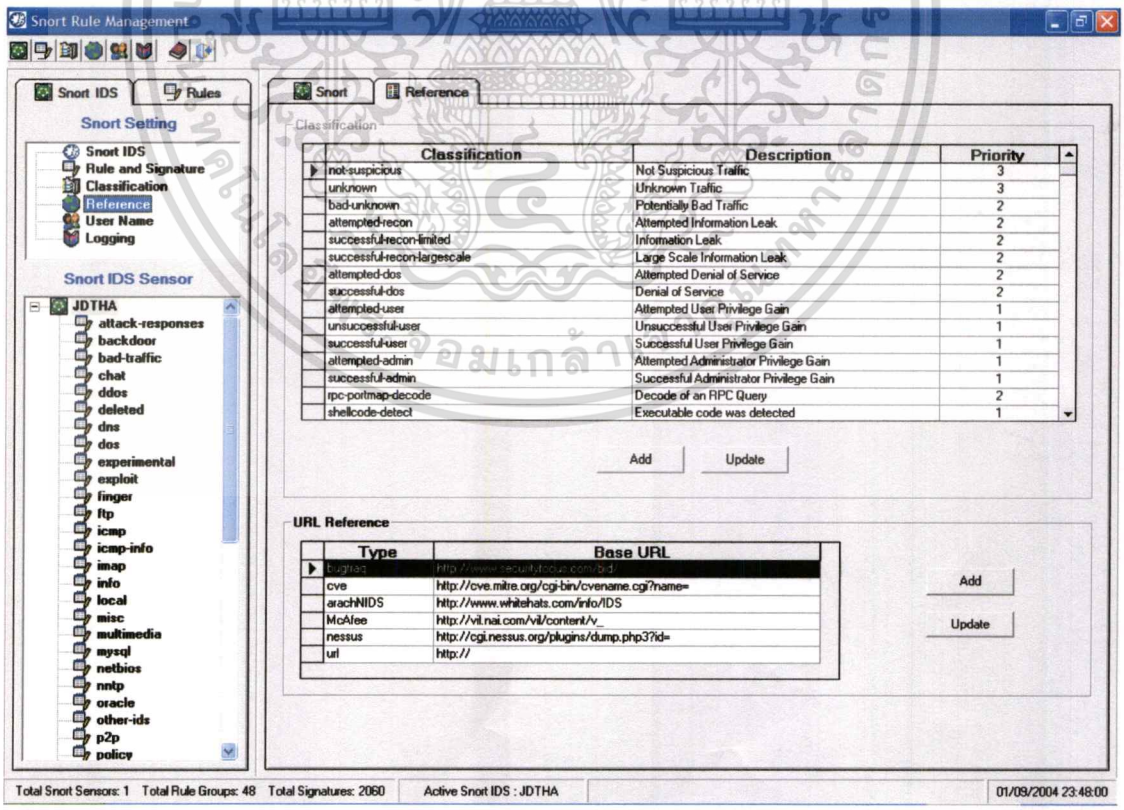
reference:cve,can-2002-1235

หมายถึง กฎข้อนี้อ้างอิงกับ reference ชื่อ cve ที่มีค่า reference เป็น can-2002-1235 ทำให้กฎดังกล่าวนี้สามารถแสดงหน้า Web page ที่มี URL เป็น http://cve.mitre.org/cgi-bin/cvename.cgi?name=**can-2002-1235** ซึ่งผู้ใช้สามารถเข้าไปดูรายละเอียดทั้งหมดของกฎข้อนี้ได้

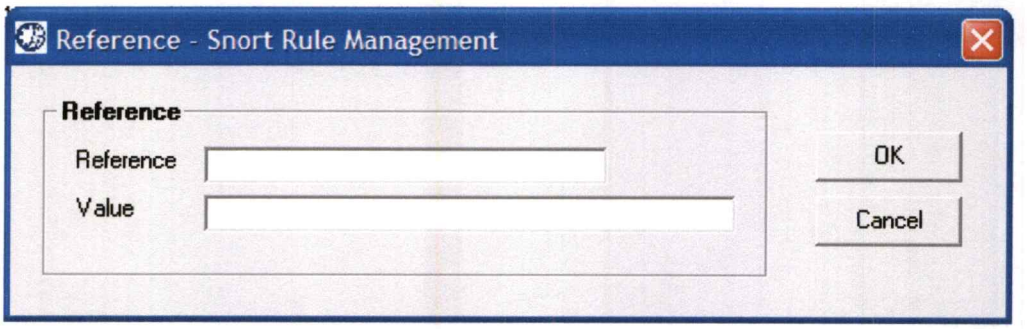
โปรแกรม Snort Rule Management ก็มีส่วนการเพิ่มและแก้ไข Reference ในฐานข้อมูล โดยเลือกไปที่เมนู Reference ก็จะปรากฏแท็บ Reference ดังรูปที่ 4.39 หากผู้ใช้ต้องการเพิ่มหรือแก้ไขก็กดปุ่ม Add หรือ Update ตามลำดับ ดังรูปที่ 4.40 และ 4.41 โดยในการเพิ่มหรือแก้ไขค่านั้น โปรแกรมออกแบบมาให้มีความสามารถในการตรวจสอบชื่อ Reference ว่าซ้ำกับข้อมูลเดิมในฐานข้อมูลหรือไม่ หากซ้ำก็จะปรากฏกรอบโต้ตอบเพื่อแจ้งให้ผู้ใช้ทราบ แล้วให้ผู้ใช้ป้อนชื่อ Reference ใหม่

```
#-------------------------------------------------
#           Reference Config
#-------------------------------------------------
config reference: bugtraq http://www.securityfocus.com/bid/
config reference: cve http://cve.mitre.org/cgi-bin/cvename.cgi?name=
config reference: arachNIDS http://www.whitehats.com/info/IDS
config reference: McAfee http://vil.nai.com/vil/content/v_
config reference: nessus http://cgi.nessus.org/plugins/dump.php3?id=
config reference: url http://
```

รูปที่ 4.38 ตัวอย่างไฟล์ reference.config



รูปที่ 4.39 หน้าจอ Reference

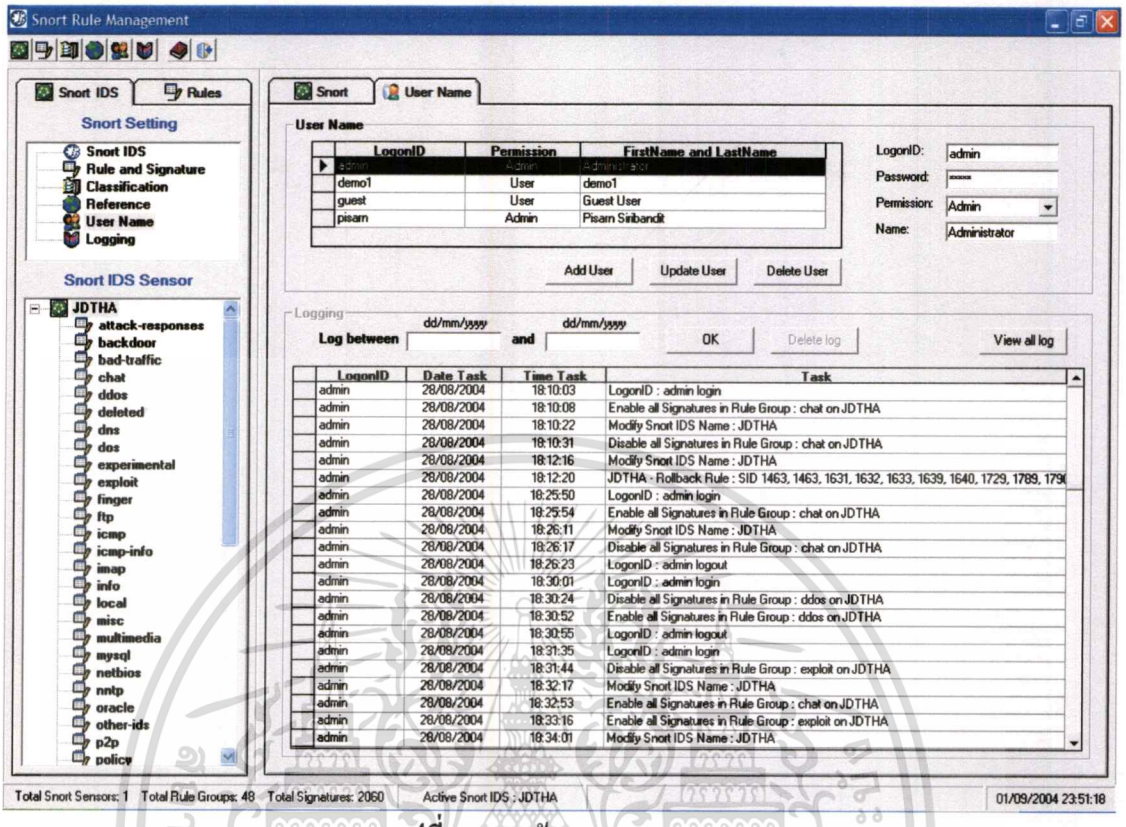


รูปที่ 4.40 การเพิ่ม Reference

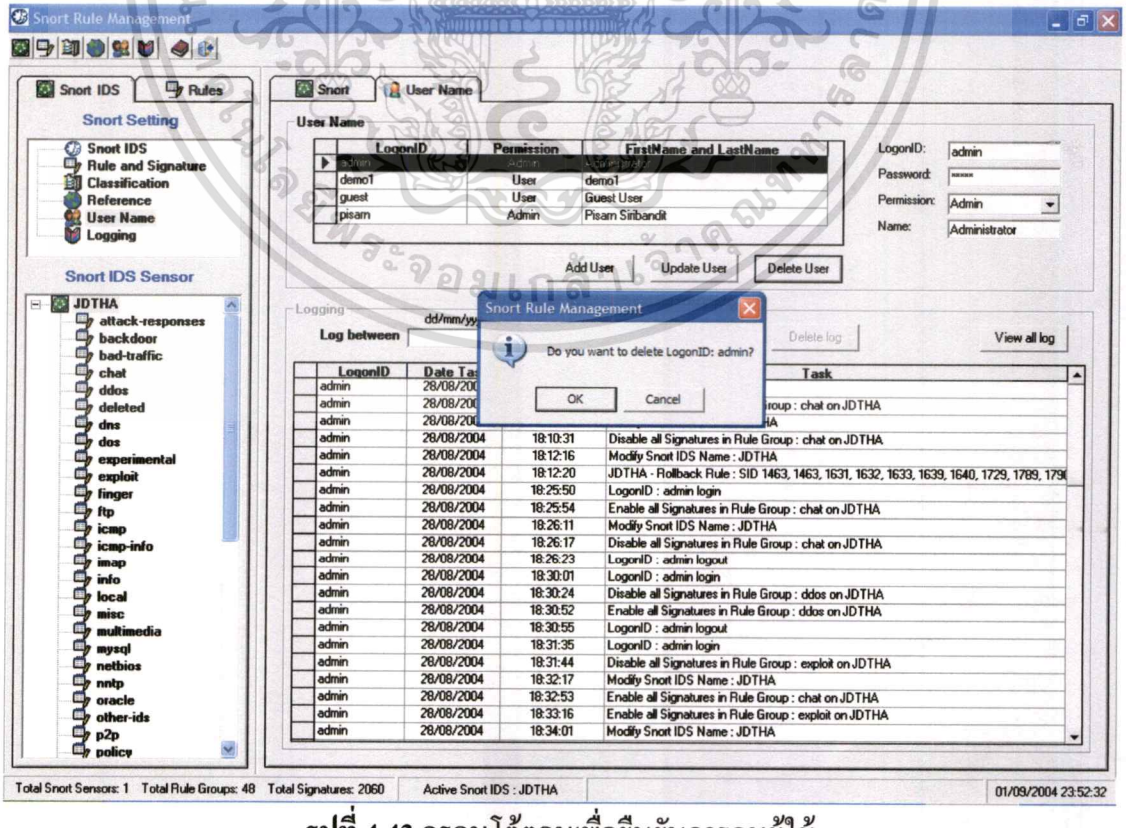
รูปที่ 4.41 การแก้ไข Reference

4.6.5 ฟังก์ชัน User Name

ก่อนที่ผู้ใช้เริ่มใช้งานโปรแกรม Snort Rule Management จะต้องมีการระบุตัวตนของผู้ใช้ก่อนว่ามีสิทธิ์เข้าใช้งานโปรแกรมหรือไม่ โดยการให้ผู้ใช้ล็อกอิน ซึ่งข้อมูลชื่อผู้ใช้ รหัสผ่าน และสิทธิ์ของผู้ใช้ ถูกเก็บอยู่ในฐานข้อมูลในตาราง Username โปรแกรม Snort Rule Management จึงมีแท็บ Username ให้กับผู้ใช้ได้ใช้งาน ดังรูปที่ 4.42 เมื่อผู้ใช้ล็อกอินเข้ามาและมีสิทธิ์ประเภท Admin โปรแกรมจะแสดงรายชื่อผู้ใช้งานทั้งหมดในฐานข้อมูล เพื่อให้ผู้ใช้สามารถเพิ่ม แก้ไข และลบข้อมูลผู้ใช้ได้ หากเป็นการลบผู้ใช้โปรแกรมจะปรากฏกรอบโต้ตอบเพื่อให้ผู้ใช้ยืนยันว่าต้องการลบผู้ใช้หรือไม่ ดังรูปที่ 4.43 ถ้าผู้ใช้ล็อกอินเข้ามาและมีสิทธิ์ประเภท User โปรแกรมจะแสดงรายชื่อเฉพาะผู้ใช้งานที่ล็อกอินเข้ามาเท่านั้น และผู้ใช้งานจะไม่สามารถเพิ่มและลบข้อมูลผู้ใช้ได้ แต่สามารถแก้ไขข้อมูลตัวเองได้เช่น แก้ไขรหัสผ่าน ยกเว้นสิทธิ์ไม่สามารถเปลี่ยนแปลงได้



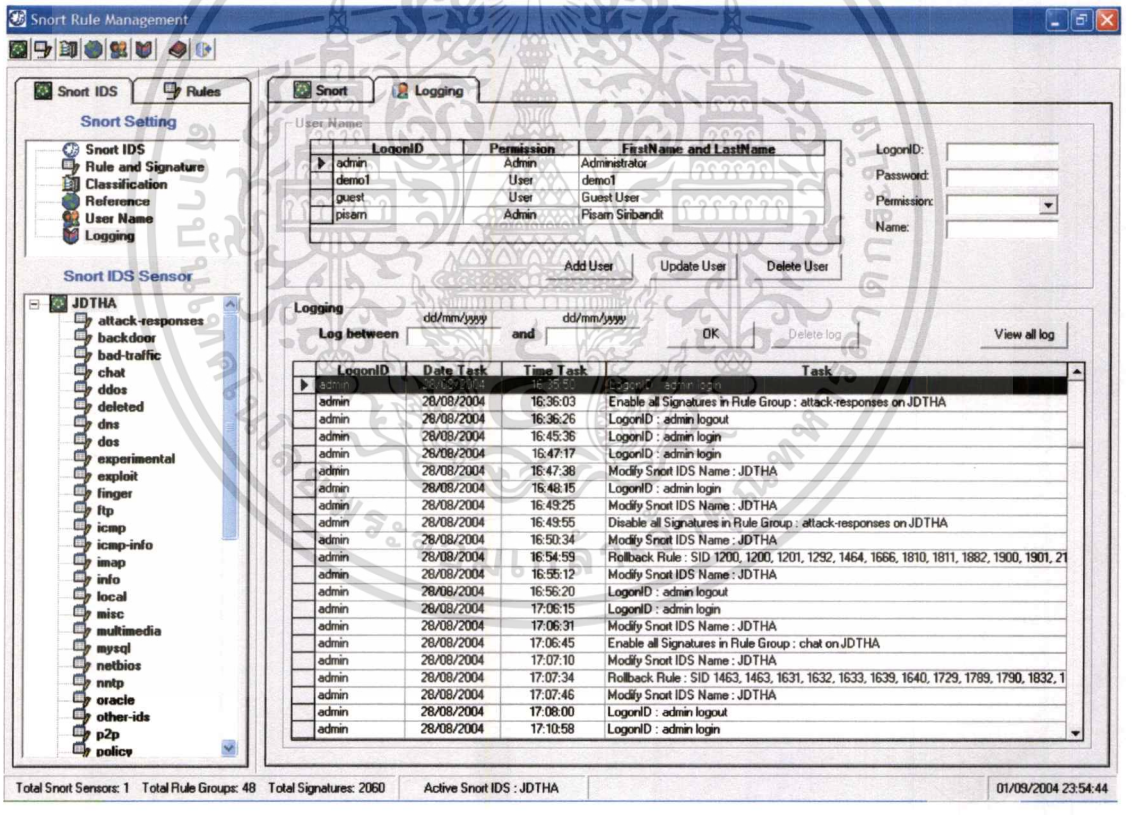
รูปที่ 4.42 หน้าจอ User Name



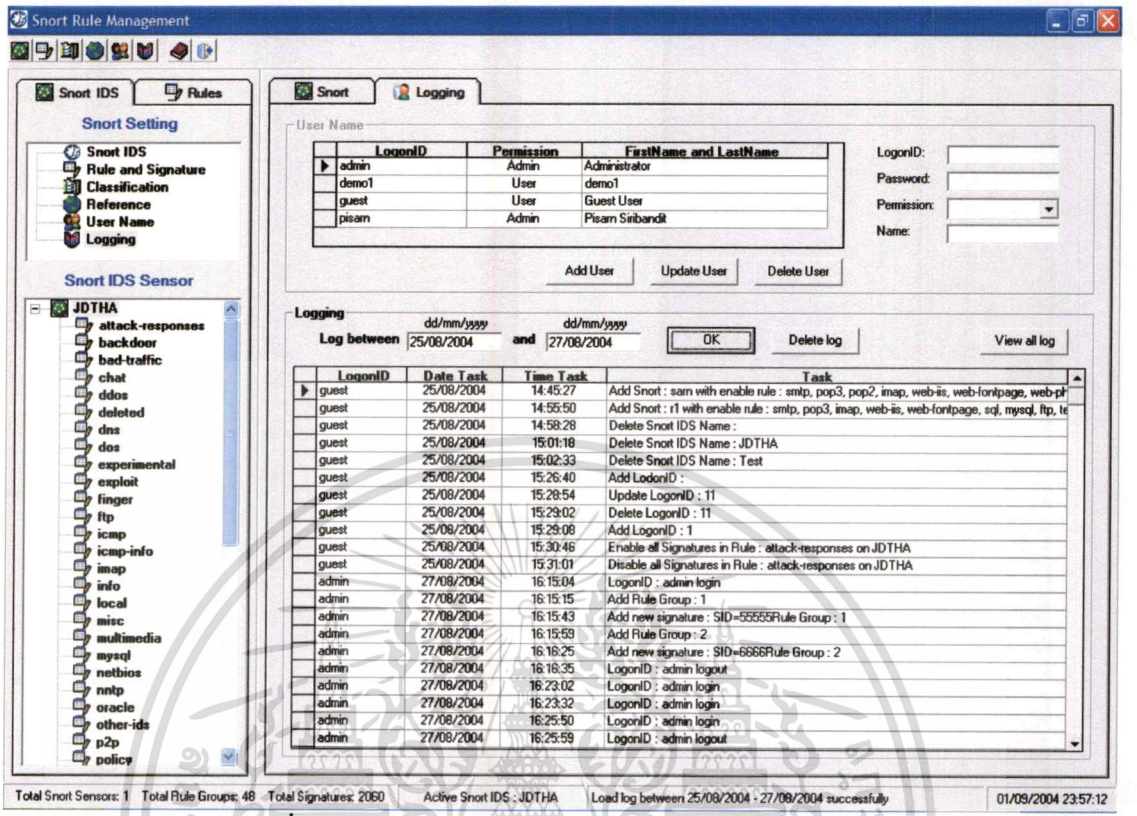
รูปที่ 4.43 กรอบโต้ตอบเพื่อยืนยันการลบผู้ใช้

4.6.6 ฟังก์ชัน Logging

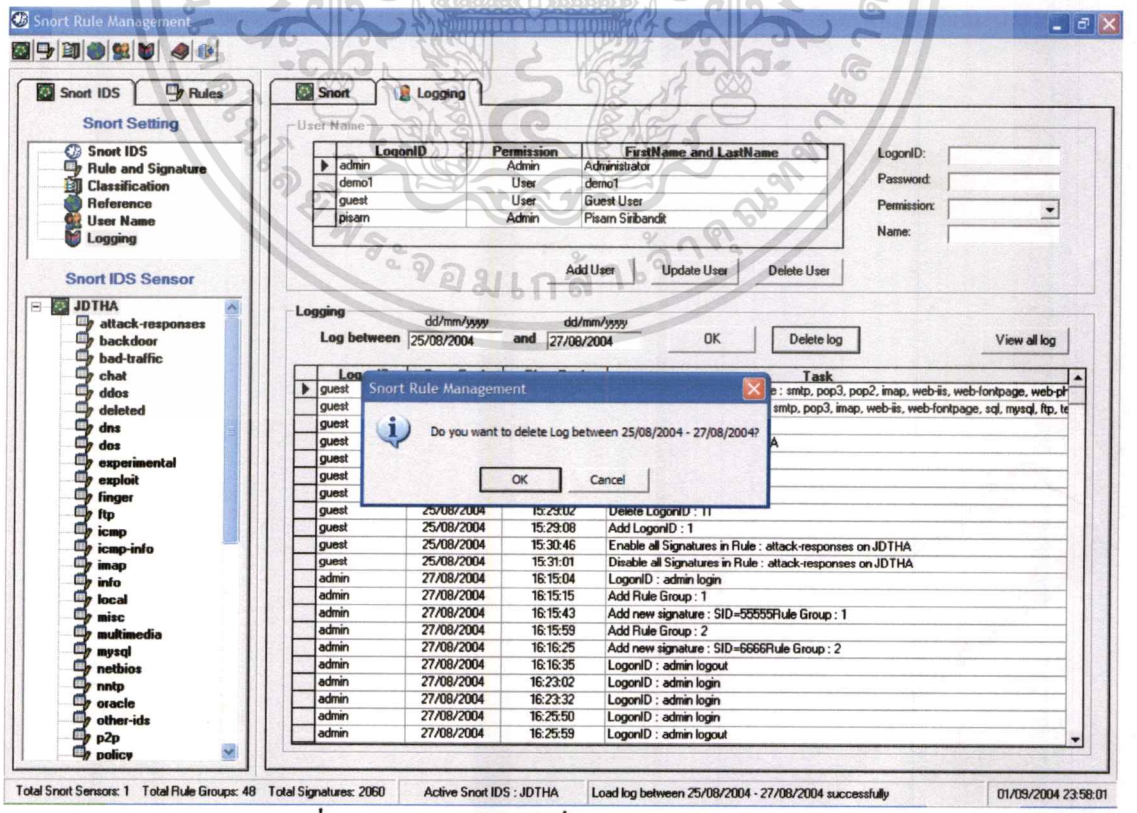
โปรแกรม Snort Rule Management มีส่วนการเก็บ log ที่ผู้ใช้สั่งงานทั้งหมดโดยบันทึกลงฐานข้อมูลในตาราง Logging โปรแกรมจึงมีแท็บ Logging ให้ผู้ใช้ได้ใช้งาน ดังรูปที่ 4.44 ผู้ใช้สามารถค้นหา log โดยระบุเป็นช่วงวันที่ต้องการได้โดยป้อนช่วงวันที่ที่ต้องการค้นหาแล้วกดปุ่ม OK โปรแกรมก็จะแสดงผลการค้นหา log ตามช่วงวันที่ที่ผู้ใช้ป้อน ดังรูปที่ 4.45 เป็นการค้นหา log ในช่วงวันที่ 25/08/2547 ถึงวันที่ 28/08/2547 นอกจากนี้โปรแกรมออกแบบมาให้ผู้ใช้ที่มีสิทธิ์ประเภท Admin สามารถลบ log ได้ โดยสามารถลบ log ระบุเป็นช่วงวันที่ที่ต้องการลบ โดยทำการค้นหาช่วงวันที่ที่ต้องการก่อนแล้วจึงกดปุ่ม Delete หากต้องการลบ log ทั้งหมดให้กดปุ่ม View all log ก่อนแล้วจึงกดปุ่ม Delete ในการลบ log นั้นโปรแกรมจะปรากฏกรอบโต้ตอบเพื่อให้ผู้ใช้ยืนยันว่าต้องการลบ log หรือไม่ ดังรูปที่ 4.46



รูปที่ 4.44 หน้าจอ Logging



รูปที่ 4.45 การค้นหา log โดยระบุเป็นช่วงวันที่ที่ต้องการ

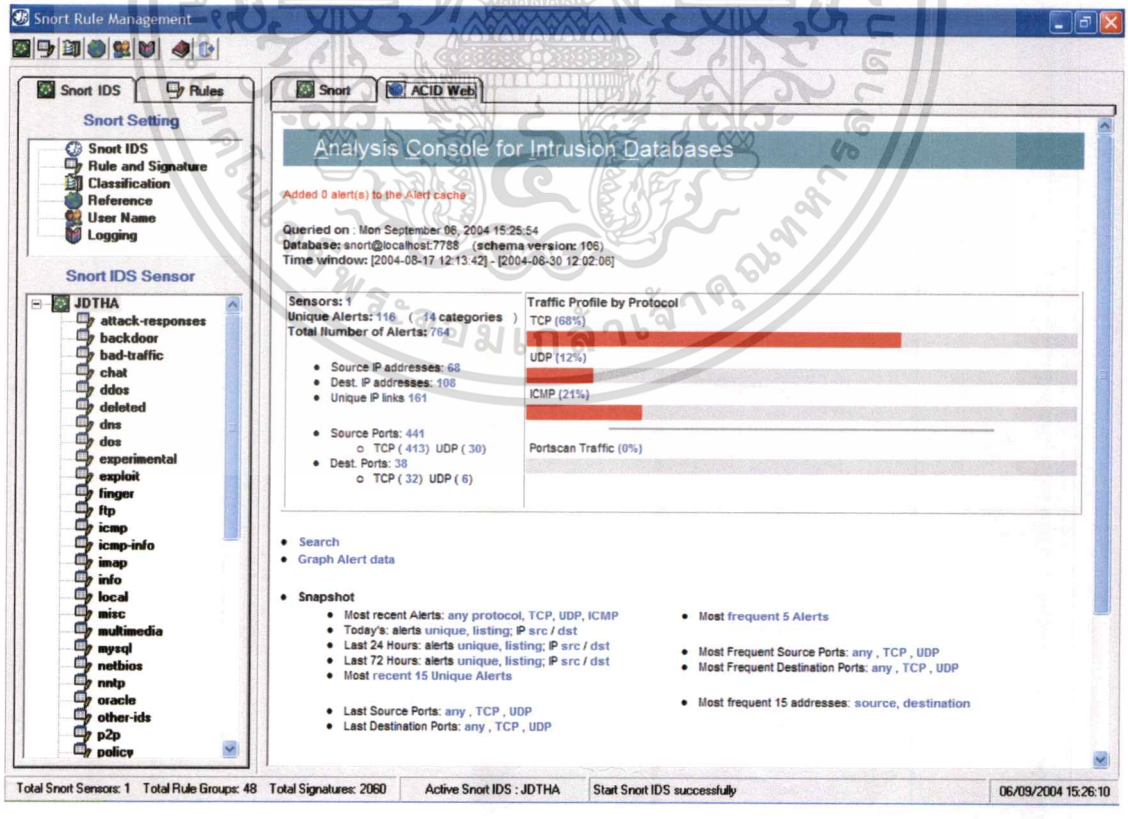


รูปที่ 4.46 กรอบโต้ตอบเพื่อยืนยันการลบข้อมูล log

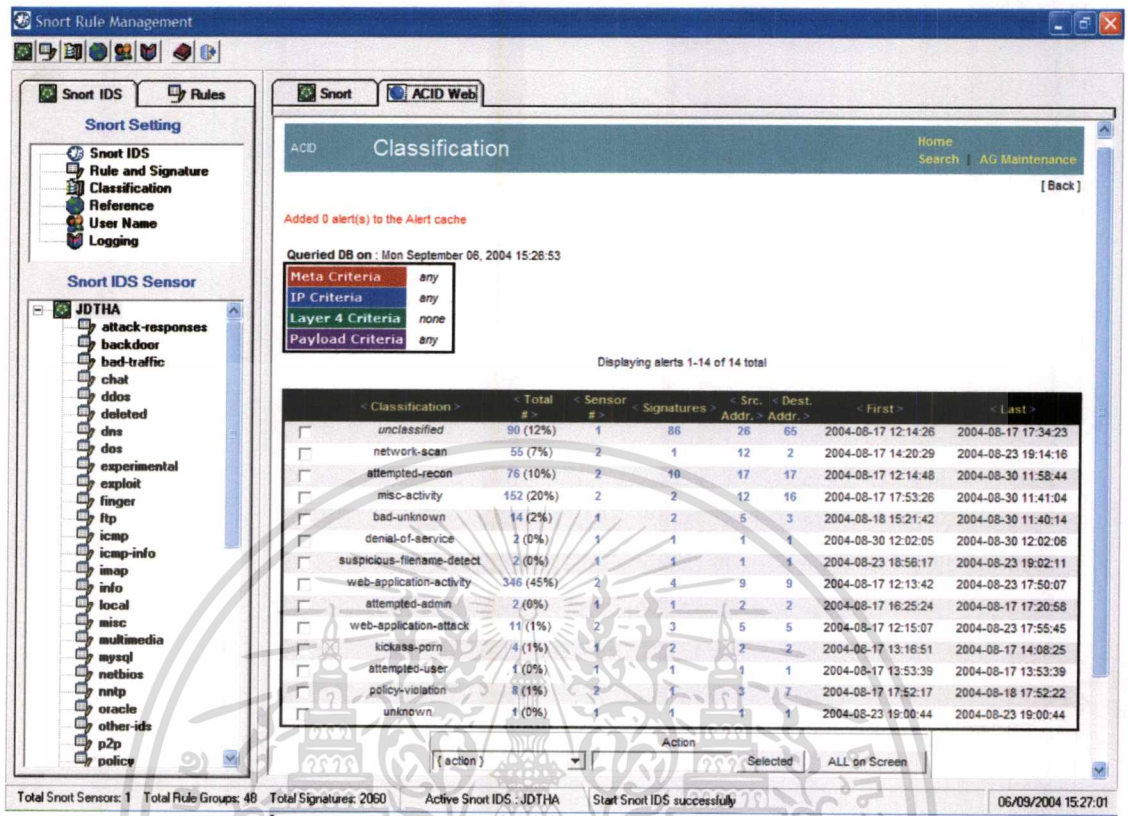
4.6.7 ฟังก์ชัน Analysis Console for Intrusion Database (ACID)

โปรแกรม Snort Rule Management ได้รวมเอาความสามารถของ Analysis Console for Intrusion Database (ACID) เวอร์ชัน 0.9.6b23 เข้ามาเป็นฟังก์ชันหนึ่งของระบบตรวจจับการบุกรุกด้วย Snort Rule ที่เหมาะสม ACID ถูกพัฒนาโดย Roman Danyliw ซึ่งเป็นส่วนหนึ่งของโครงการ AirCERT

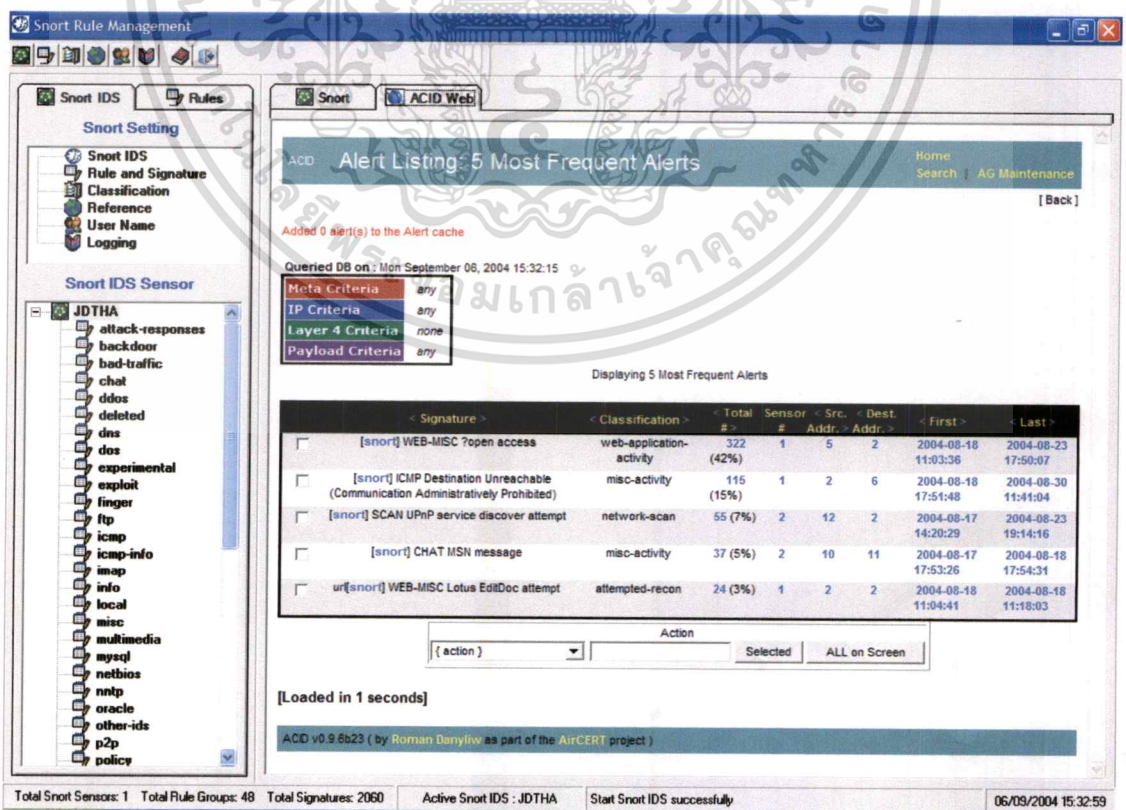
Analysis Console for Intrusion Database (ACID) เป็น PHP-based analysis engine ใช้สำหรับค้นหาและกระบวนการทำงานของฐานข้อมูลสำหรับเหตุการณ์ด้านความปลอดภัยที่ถูกสร้างจากระบบตรวจจับการบุกรุกแบบต่างๆ, ไฟร์วอลล์ และเครื่องมือการเฝ้าดูเน็ตเวิร์ค ปัจจุบัน ACID มีความสามารถสร้างการสอบถามและค้นหา, แสดงข้อมูล (ถอดรหัสข้อมูล), จัดการการแจ้งเตือน และจัดทำสถิติ จากรูปที่ 4.47 แท็บ ACID Web ปรากฏให้ผู้ใช้ใช้งานเมื่อผู้ใช้เริ่มให้ระบบตรวจจับการบุกรุกด้วย Snort Rule ที่เหมาะสมทำงาน โดยผู้ใช้สามารถดูการแจ้งเตือนของ Snort สามารถค้นหา IP Address ต้นทาง, IP Address ปลายทาง, ชนิดการแจ้งเตือน, เวลาการแจ้งเตือน, หมายเลขพอร์ต, โพรโทคอล, การแจ้งเตือนตามประเภทของ Signature ดังรูปที่ 4.48, Signature ที่มีการแจ้งเตือนบ่อยที่สุด 5 อันดับแรก ดังรูปที่ 4.49 เป็นต้น



รูปที่ 4.47 ACID Web page



รูปที่ 4.48 ACID ค้นหาประเภทของ Signature ในฐานข้อมูล

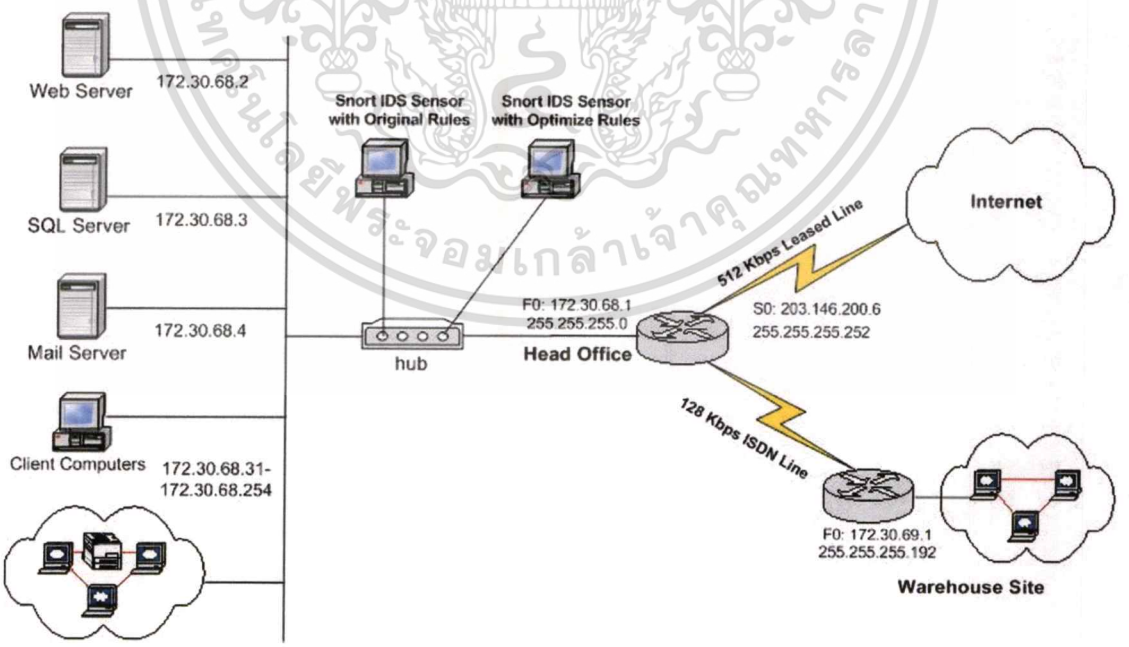


รูปที่ 4.49 ACID ค้นหาการแจ้งเตือนบ่อยที่สุด 5 อันดับแรก

บทที่ 5
การทดสอบโครงการพัฒนาระบบงาน

5.1 การออกแบบการทดสอบโครงการพัฒนาระบบงาน

การทดสอบระบบตรวจจับการบุกรุกด้วย Snort Rules ที่เหมาะสม ได้ทดสอบระบบในสภาพแวดล้อมที่มีการใช้สารสนเทศ SQL Server, Web Server และ Mail Server โดยมีการสร้าง Snort IDS Sensor จำนวน 2 เครื่อง โดยเครื่องแรกเป็น Snort IDS Sensor ที่ไม่ได้รับการปรับเปลี่ยน Snort Rule (Original Snort Rule) อีกเครื่องเป็น Snort IDS Sensor ที่ได้รับการปรับเปลี่ยน Snort Rule (Optimizing Snort Rule) จากโครงการพัฒนาระบบงานนี้ ซึ่งเหมาะกับสารสนเทศที่ทดสอบ การทดสอบนี้มีระยะเวลาตั้งแต่วันที่ 1 กันยายน 2547 – 6 กันยายน 2547 โดยมีการเชื่อมต่อระบบเครือข่ายดังรูปที่ 5.1 มีการกำหนดค่าตัวแปรเครือข่ายในไฟล์ snort.conf ดังรูปที่ 5.2 และใช้รูลไฟล์ดังตารางที่ 5.1 และ 5.2 แล้วเปรียบเทียบผลการตรวจจับการบุกรุกด้วย Snort Rule ที่ไม่ได้ปรับเปลี่ยนกับ Snort Rule ที่ปรับเปลี่ยนให้เหมาะกับสารสนเทศในองค์กร



รูปที่ 5.1 การเชื่อมต่อเครือข่ายสำหรับการทดสอบโครงการพัฒนาระบบงาน

```
var HOME_NET 172.30.68.0/24
var EXTERNAL_NET any
var DNS_SERVERS $HOME_NET
var SMTP_SERVERS 172.30.68.4/32
var HTTP_SERVERS 172.30.68.2/32
var SQL_SERVERS 172.30.68.3/32
var TELNET_SERVERS $HOME_NET
var HTTP_PORTS 80
var SHELLCODE_PORTS !80
var ORACLE_PORTS 1521
var AIM_SERVERS
[64.12.24.0/24,64.12.25.0/24,64.12.26.14/24,64.12.28.0/24,64.12.29
.0/24,64.12.161.0/24,64.12.163.0/24,205.188.5.0/24,205.188.9.0/24]
var RULE_PATH c:\snort\rules
```

รูปที่ 5.2 ค่าตัวแปรเครือข่ายที่ใช้ทดสอบในไฟล์ snort.conf

ตารางที่ 5.1 ชื่อรูลไฟล์ที่ใช้ทดสอบในการตรวจจับการบุกรุก (จากรูลไฟล์ทั้งหมด 48 รูลไฟล์)

ชื่อรูลไฟล์ (Rule Group)	ไม่ปรับเปลี่ยน Rule	ปรับเปลี่ยน Rule
attack-responses.rules	☑	☑
backdoor.rules	☑	☑
bad-traffic.rules	☑	☑
chat.rules	-	☑
ddos.rules	☑	☑
deleted.rules	☑	☑
dns.rules	☑	☑
dos.rules	☑	☑
experimental.rules	☑	☑
exploit.rules	☑	☑
finger.rules	☑	☑
ftp.rules	☑	☑
icmp.rules	☑	-
icmp-info.rules	-	-
imap.rules	☑	☑
info.rules	☑	☑
local.rules	☑	☑
misc.rules	☑	☑
multimedia.rules	-	-

ตารางที่ 5.2 ชื่อรูลไฟล์ที่ใช้ทดสอบในการตรวจจับการบุกรุก (ต่อ)

ชื่อรูลไฟล์ (Rule Group)	ไม่ปรับเปลี่ยน Rule	ปรับเปลี่ยน Rule
mysql.rules	☑	-
netbios.rules	☑	-
nntp.rules	☑	-
oracle.rules	☑	-
other-ids.rules	☑	☑
p2p.rules	-	-
policy.rules	-	-
pop2.rules	-	-
pop3.rules	☑	☑
porn.rules	-	-
rpc.rules	☑	☑
rservices.rules	☑	-
scan.rules	☑	☑
shellcode.rules	-	-
smtp.rules	☑	☑
snmp.rules	☑	-
sql.rules	☑	☑
telnet.rules	☑	☑
tftp.rules	☑	-
virus.rules	☑	☑
web-attacks.rules	-	-
web-cgi.rules	☑	☑
web-client.rules	☑	☑
web-coldfusion.rules	☑	☑
web-frontpage.rules	☑	☑
web-iis.rules	☑	☑
web-misc.rules	☑	☑
web-php.rules	☑	-
x11.rules	☑	-
รวมรูลไฟล์ที่ใช้ทั้งหมด	**39**	**30**

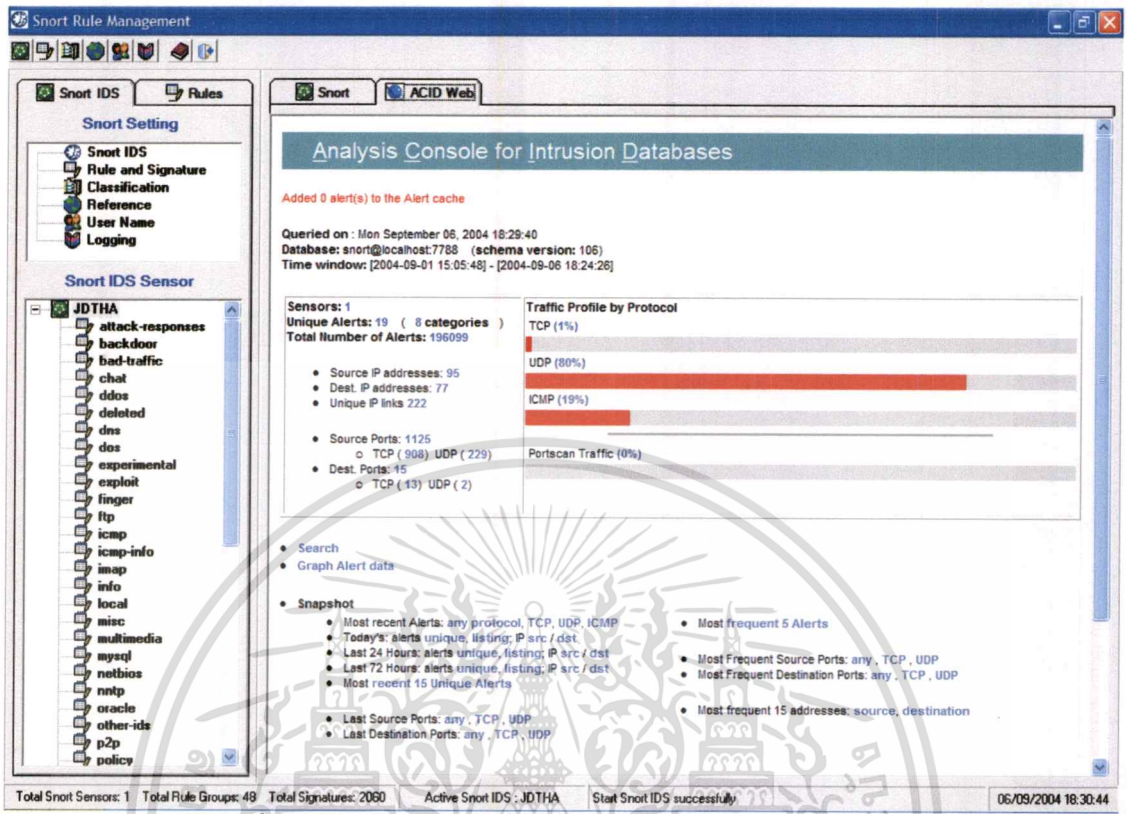
5.2 ผลการทดสอบโครงการพัฒนาระบบงาน

5.2.1 ผลของ Snort Rule ที่ไม่ได้รับการปรับเปลี่ยน (Original Snort Rule)

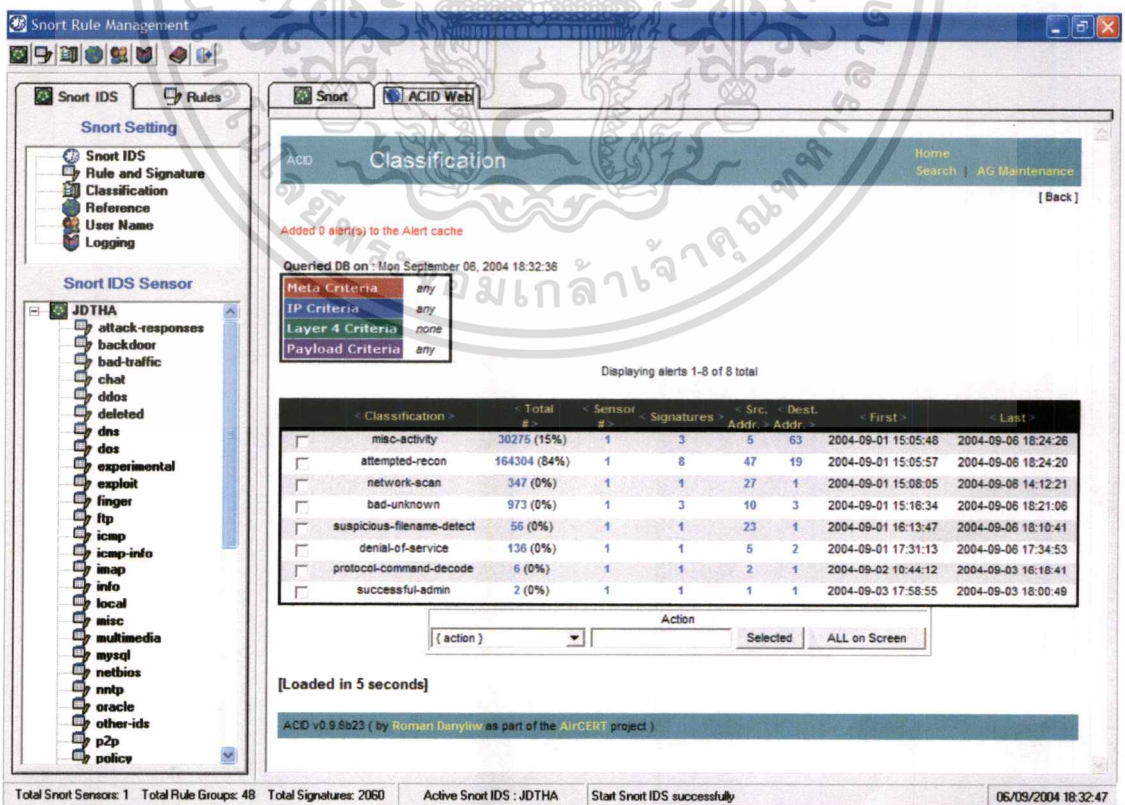
ผลการทดสอบแสดงดังตารางที่ 5.3 และรูปที่ 5.3, 5.4, 5.5 และ 5.6

ตารางที่ 5.3 ผลการทดสอบ Snort Rule ที่ไม่ได้รับการปรับเปลี่ยน

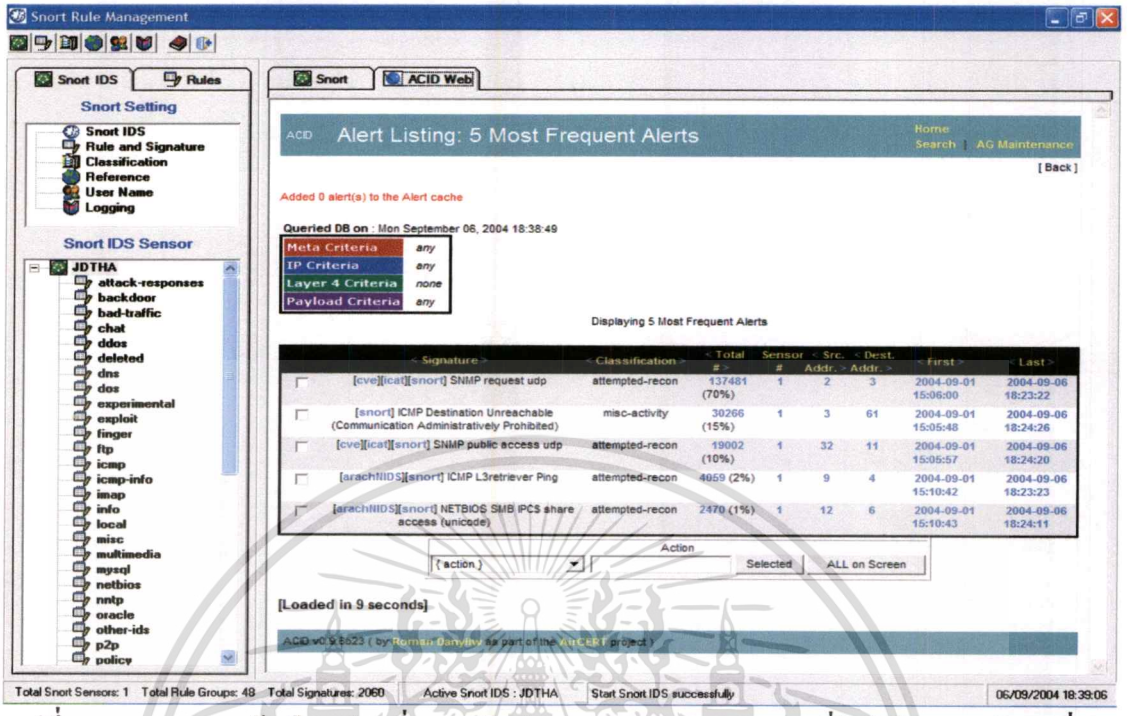
จำนวนการแจ้งเตือนทั้งหมด 196,099 ครั้ง		
ประเภทของ Signature (เรียงจากมากไปน้อย)	**5 อันดับแรกที่มีการแจ้งเตือนมากที่สุด**	**Signature ที่มีการแจ้งเตือน (เรียงจากมากไปน้อย)**
1. attempted-recon 2. misc-activity 3. bad-unknown 4. network-scan 5. denial-of-service 6. suspicious-filename-detect 7. protocol-command-decode 8. successful-admin	1. SNMP request udp 2. ICMP Destination Unreachable 3. SNMP public access udp 4. ICMP L3retriever Ping 5. NETBIOS SMB IPC$ share access (unicode)	1. SNMP request udp 2. ICMP Destination Unreachable 3. SNMP public access udp 4. ICMP L3retriever Ping 5. NETBIOS SMB IPC$ share access (unicode) 6. ICMP PING NMAP 7. ICMP Large ICMP Packet 8. SCAN UPnP service discover attempt 9. NETBIOS SMB SMB_COM_TRANSACTION Max Parameter and Max Count of 0 DOS Attempt 10. VIRUS OUTBOUND .doc file attachment 11. NETBIOS SMB winreg access (unicode) 12. NETBIOS SMB IPC$ share access 13. Virus - Possible MyRomeo Worm 14. MISC MS Terminal server request 15. ATTACK-RESPONSES id check returned root 16. SNMP Broadcast request 17. ATTACK-RESPONSES directory listing 18. ATTACK-RESPONSES Microsoft cmd.exe banner 19. INFO Outbound GNUTella client request



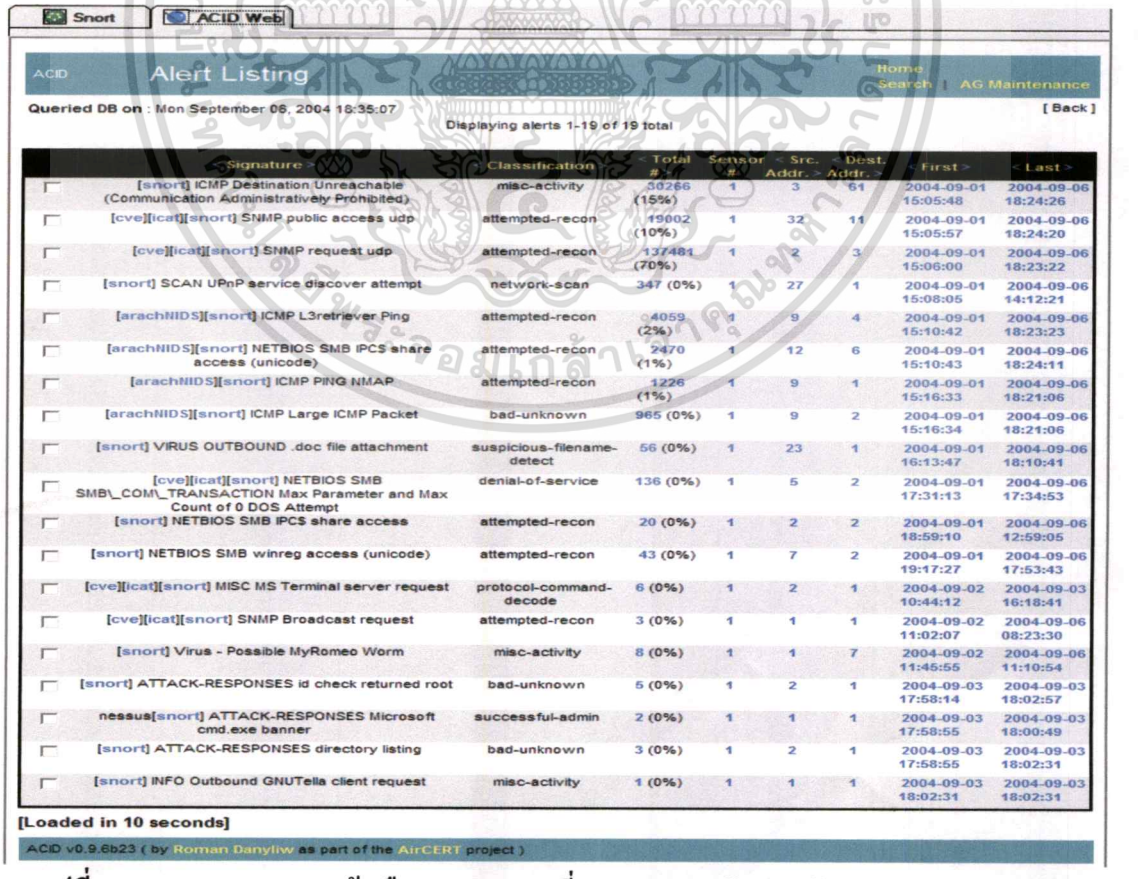
รูปที่ 5.3 ACID สำหรับ Snort Rule ที่ไม่ได้รับการปรับเปลี่ยน



รูปที่ 5.4 ACID ค้นหาตามประเภทของ Signature สำหรับ Snort Rule ที่ไม่ได้รับการปรับเปลี่ยน



รูปที่ 5.5 ACID การแจ้งเตือนบ่อยที่สุด 5 อันดับแรกสำหรับ Snort Rule ที่ไม่ได้รับการปรับเปลี่ยน



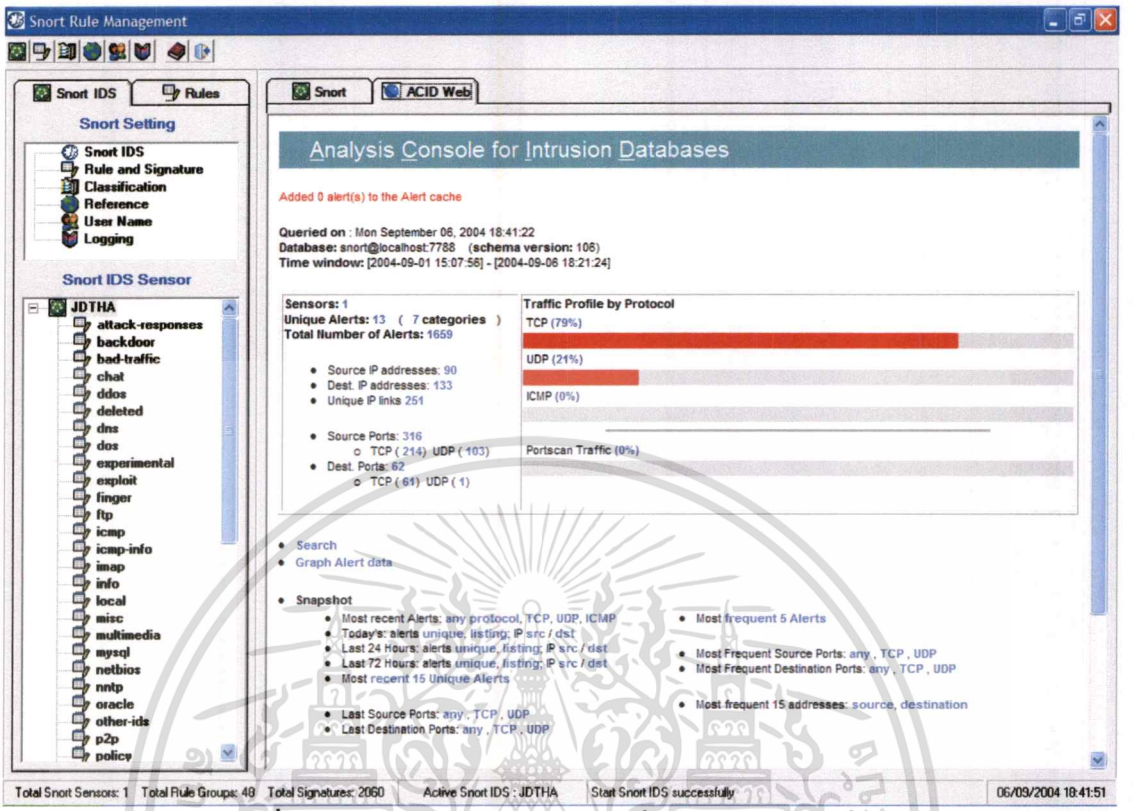
รูปที่ 5.6 ACID แสดงการแจ้งเตือน Signature ที่พบการตรวจจับสำหรับ Original Snort Rule

5.2.2 ผลของ Snort Rule ที่ได้รับการปรับเปลี่ยน (Optimizing Snort Rule)

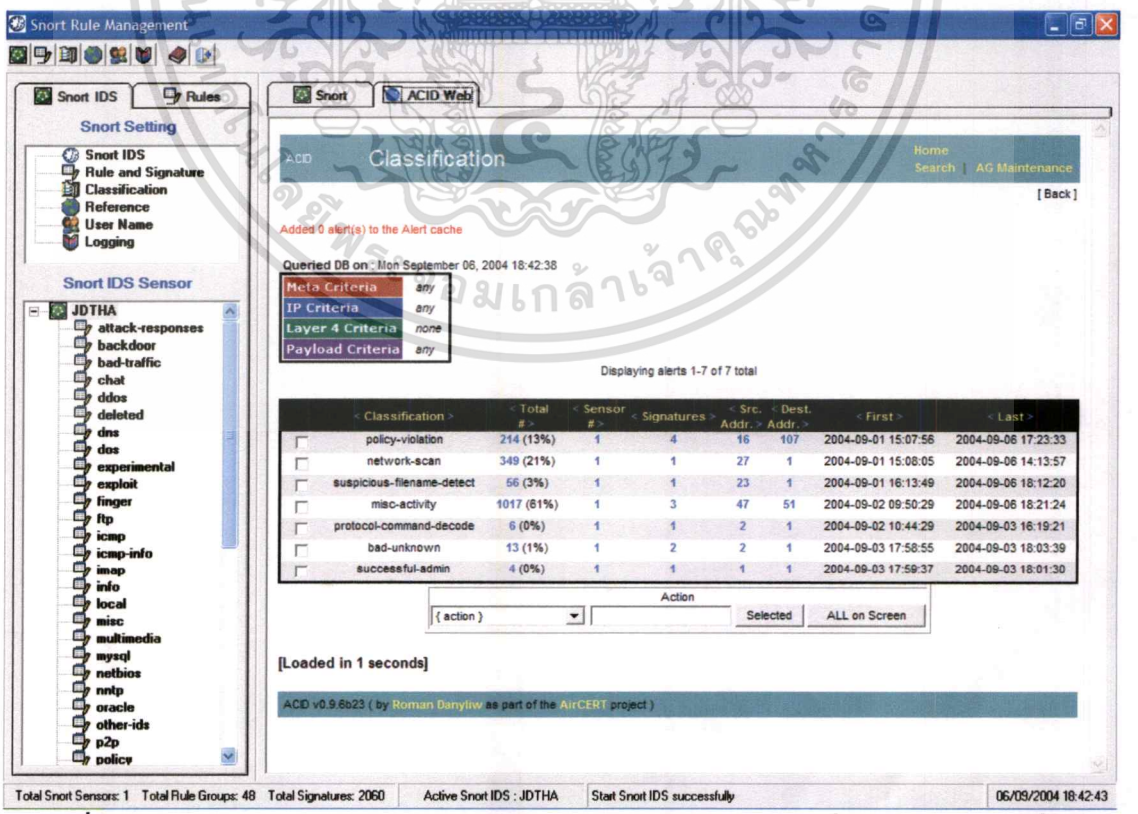
ผลการทดสอบแสดงดังตารางที่ 5.4 และรูปที่ 5.7, 5.8, 5.9 และ 5.10

ตารางที่ 5.4 ผลการทดสอบ Snort Rule ที่ได้รับการปรับเปลี่ยน

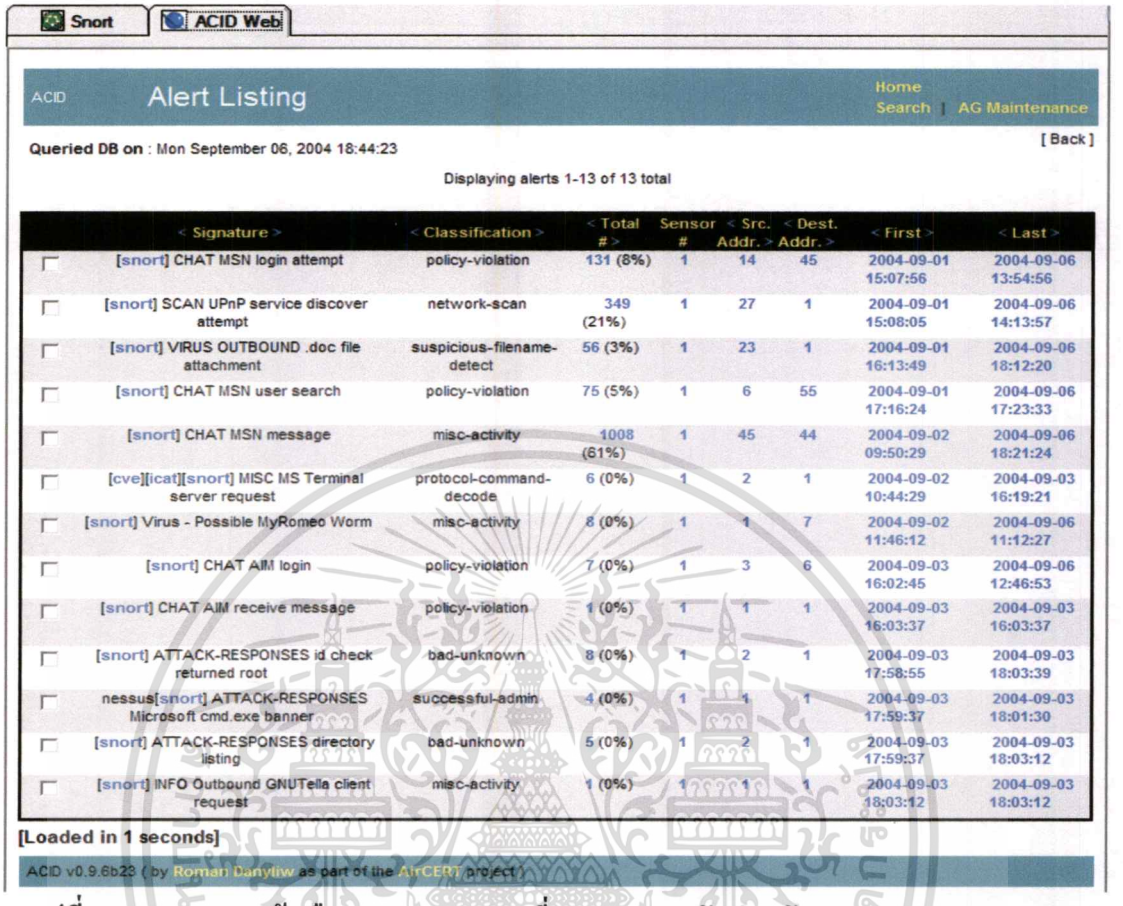
จำนวนการแจ้งเตือนทั้งหมด 1,659 ครั้ง		
ประเภทของ Signature (เรียงจากมากไปน้อย)	**5 อันดับแรกที่มีการแจ้งเตือนมากที่สุด**	**Signature ที่มีการแจ้งเตือน (เรียงจากมากไปน้อย)**
1. misc-activity 2. network-scan 3. policy-violation 4. suspicious-filename-detect 5. bad-unknown 6. protocol-command-decode 7. successful-admin	1. CHAT MSN message 2. SCAN UPnP service discover attempt 3. CHAT MSN login attempt 4. CHAT MSN user search 5. VIRUS OUTBOUND .doc file attachment	1. CHAT MSN message 2. SCAN UPnP service discover attempt 3. CHAT MSN login attempt 4.CHAT MSN user search 5. VIRUS OUTBOUND .doc file attachment 6. Virus - Possible MyRomeo Worm 7. ATTACK-RESPONSES id check returned root 8. CHAT AIM login 9. MISC MS Terminal server request 10. ATTACK-RESPONSES directory listing 11. ATTACK-RESPONSES Microsoft cmd.exe banner 12. INFO Outbound GNUTella client request 13. CHAT AIM receive message



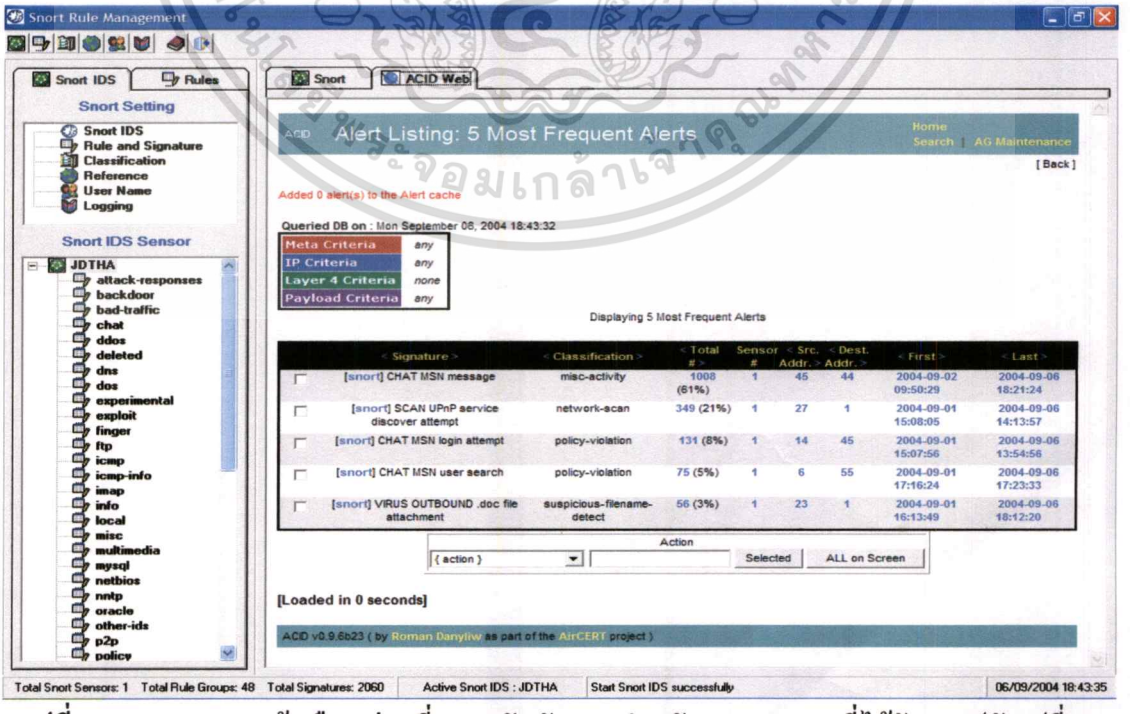
รูปที่ 5.7 ACID สำหรับ Snort Rule ที่ได้รับการปรับเปลี่ยน



รูปที่ 5.8 ACID ค้นหาตามประเภทของ Signature สำหรับ Snort Rule ที่ได้รับการปรับเปลี่ยน



รูปที่ 5.9 ACID การแจ้งเตือนตาม Signature ที่พบการตรวจจับสำหรับ Optimizing Snort Rule



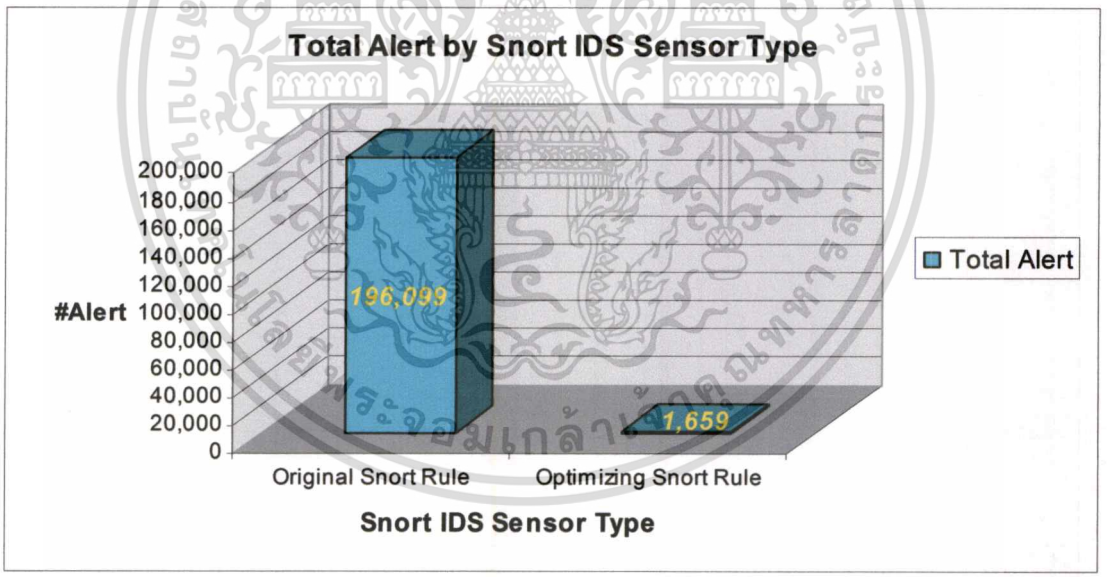
รูปที่ 5.10 ACID การแจ้งเตือนบ่อยที่สุด 5 อันดับแรกสำหรับ Snort Rule ที่ได้รับการปรับเปลี่ยน

5.2.3 เปรียบเทียบผลการทดสอบ Snort IDS Sensor ทั้ง 2 ประเภท

จากผลการทดสอบ Snort IDS Sensor ทั้ง 2 ประเภทคือ Snort IDS Sensor ที่มีการใช้ Snort Rule ให้เหมาะกับสารสนเทศในองค์กร (Optimizing Snort Rule) และ Snort IDS Sensor ที่ไม่ได้ปรับเปลี่ยน Snort Rule (Original Snort Rule) นำมาสร้างกราฟ 4 แบบดังนี้

- กราฟแสดงจำนวนการแจ้งเตือนแบ่งตามประเภทของ Snort IDS Sensor ดังรูปที่ 5.11

Original Snort Rule มีการแจ้งเตือน 196,099 ครั้ง ส่วน Optimizing Snort Rule มีการแจ้งเตือน 1,659 ครั้ง ซึ่งต่างกันอย่างมากถึง 118 เท่า เนื่องจากเครือข่ายที่ทดสอบมีเครื่องพิมพ์เครือข่าย (Network Printer) ทั้งหมด 9 เครื่องที่ใช้โพรโทคอล SNMP ซึ่งเครื่องลูกข่ายทั้งหมด 85 เครื่องมีการเชื่อมต่ออยู่ เพื่อคอยตรวจสอบสถานะของเครื่องพิมพ์อยู่ตลอดเวลา ซึ่ง Original Snort Rule มีการใช้กลุ่มกฎ SNMP จึงมีการแจ้งเตือนในระบบเป็นจำนวนมาก ตรงข้ามกับ Optimizing Snort Rule ที่ไม่มีการใช้กลุ่มกฎนี้ แต่ Optimizing Snort Rule มีการใช้กลุ่มกฎ CHAT ในระบบ ซึ่งผู้ใช้ในเครือข่ายใช้โปรแกรม MSN Messenger ทำให้มีการแจ้งเตือนกฎกลุ่มนี้เป็นส่วนใหญ่

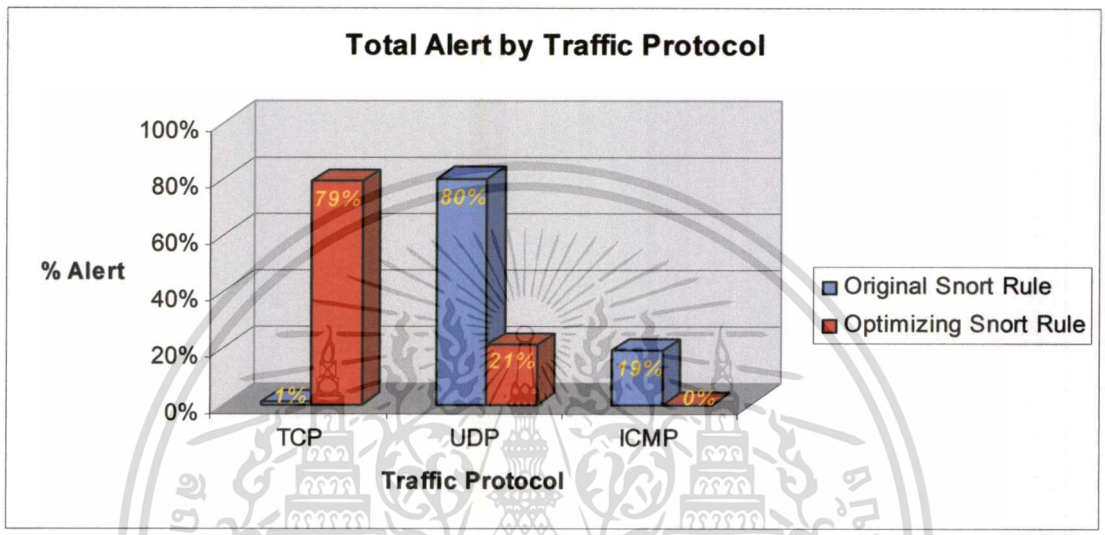


รูปที่ 5.11 กราฟแสดงจำนวนการแจ้งเตือนแบ่งตามประเภทของ Snort IDS Sensor

- กราฟแสดงจำนวนการแจ้งเตือนแบ่งตามประเภทของ Traffic Protocol ดังรูปที่ 5.12

Original Snort Rule มีการแจ้งเตือนโพรโทคอล TCP 2,750 ครั้ง (1%), UDP 156,833 ครั้ง (80%), ICMP 36,516 ครั้ง (19%) จะเห็นว่า UDP ประเภท SNMP มีการแจ้งเตือนมากที่สุด ซึ่งเครือข่ายที่ทดสอบมีการใช้งานเครื่องพิมพ์เครือข่ายที่ใช้บริการ SNMP daemon (public communities)

Optimizing Snort Rule มีการแจ้งเตือนโพรโทคอล TCP 1,310 ครั้ง (79%), UDP 349 ครั้ง (21%), ICMP ไม่มีการแจ้งเตือน จะเห็นว่า TCP มีการแจ้งเตือนมากที่สุดโดยการแจ้งเตือนส่วนใหญ่เป็นการใช้งาน CHAT เช่น MSN Messenger แต่ก็ถือว่าการแจ้งเตือนทั้งหมดนั้นน้อยมากเมื่อเทียบกับการแจ้งเตือนใน Original Snort Rule



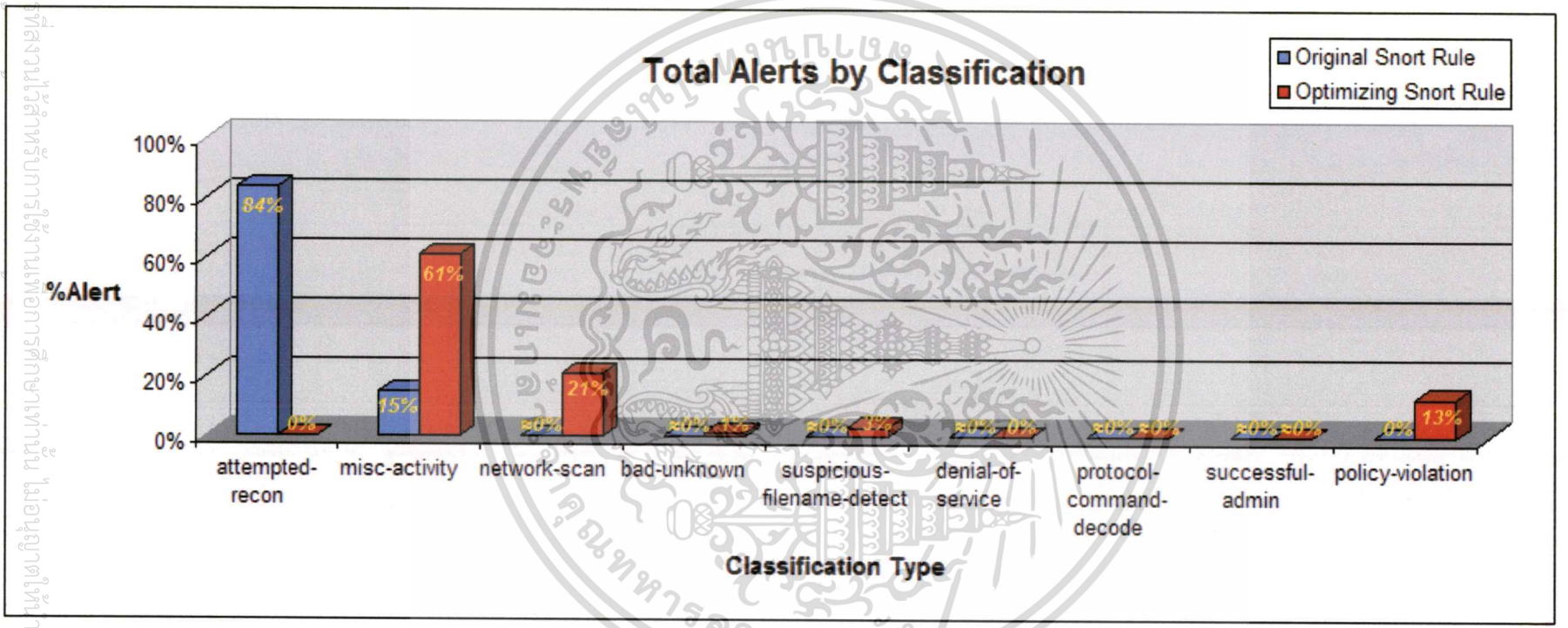
รูปที่ 5.12 กราฟแสดงจำนวนการแจ้งเตือนแบ่งตามประเภทของ Traffic Protocol

- กราฟแสดงจำนวนการแจ้งเตือนแบ่งตาม Classification 9 ประเภท ดังรูปที่ 5.13

Original Snort Rule มีการแจ้งเตือนประเภท attempted-recon มากที่สุดถึง 164,304 ครั้ง (84%) เนื่องจากกลุ่มกฎตรวจจับ SNMP ซึ่งเป็น UDP จัดอยู่ในการแจ้งเตือนประเภทนี้ โดยกฎที่แจ้งเตือนมากที่สุด 3 อันดับแรกคือ SNMP request udp, ICMP Destination Unreachable และ SNMP public access udp ตามลำดับ ซึ่งสอดคล้องกับผลของกราฟในรูปที่ 5.12 และ 5.14

Optimizing Snort Rule มีการแจ้งเตือนประเภท misc-activity มากที่สุด 1,017 ครั้ง (61%) เนื่องจากกลุ่มกฎตรวจจับ CHAT ซึ่งเป็น TCP จัดอยู่ในการแจ้งเตือนประเภทนี้ โดยกฎที่แจ้งเตือนมากที่สุดคือ CHAT MSN message ซึ่งสอดคล้องกับผลของกราฟในรูปที่ 5.12 และ 5.14

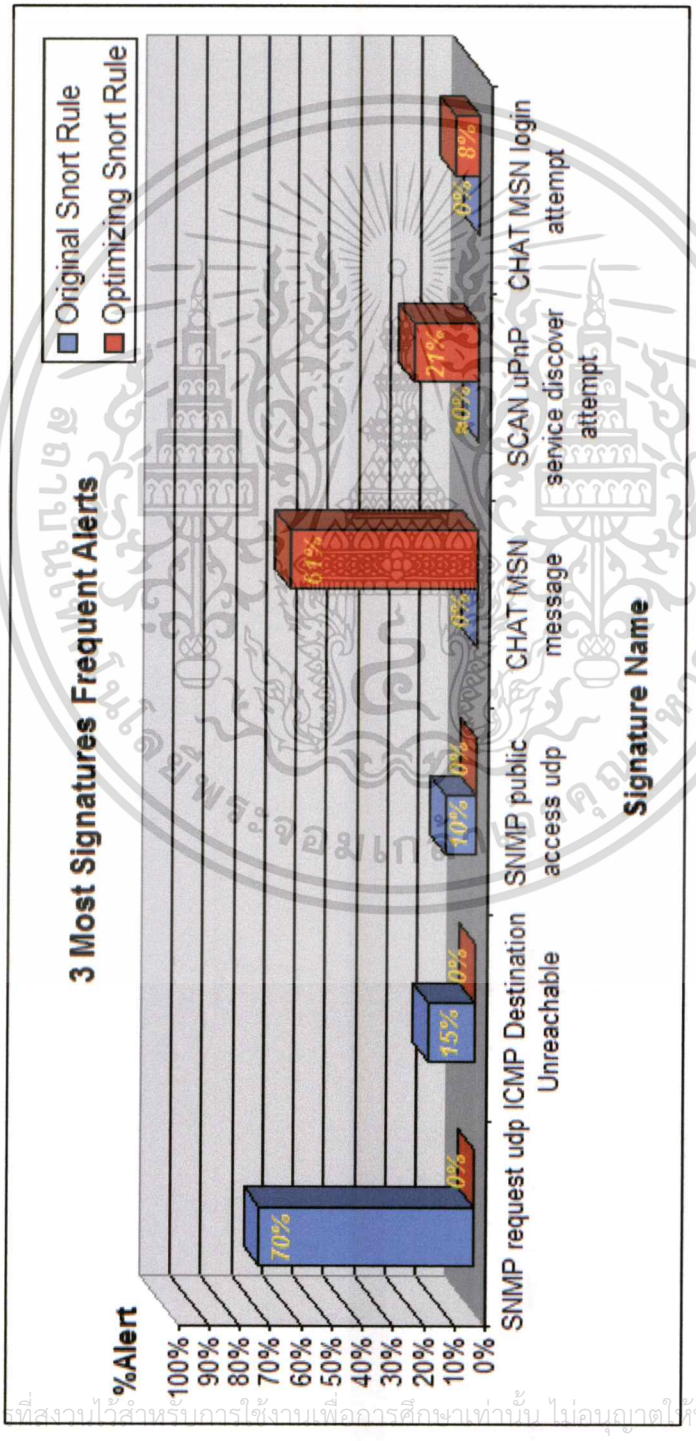
จากกราฟรูปที่ 5.13 มีข้อสังเกตอีกประการหนึ่งคือ ถ้า Snort IDS Sensor มีการใช้กฎเพื่อตรวจจับเหมือนกันก็จะได้ผลจำนวนการแจ้งเตือนที่เท่ากัน เช่นการแจ้งเตือนประเภท protocol-command-decode (6 ครั้ง) และ successful-admin (2 ครั้ง) ของ Snort IDS Sensor ทั้งสองเท่ากัน และถ้า Snort IDS Sensor ใดไม่มีการใช้กฎ เช่น กฎ SNMP request udp ซึ่งจัดอยู่ในประเภท attempted-recon ก็จะไม่มีการแจ้งเตือนประเภท attempted-recon ในระบบ ซึ่งสอดคล้องกับกราฟในรูปที่ 5.13



รูปที่ 5.13 กราฟแสดงจำนวนการแจ้งเตือนแบ่งตาม Classification

- กราฟแสดงจำนวนการแจ้งเตือนมากที่สุด 3 อันดับแรกแบ่งตามชื่อ Signature ดังรูปที่ 5.14

จากกราฟจะเห็นได้ว่า Original Snort Rule มีการแจ้งเตือนของกฎ SNMP request udp มากที่สุด แต่ Optimizing Snort Rule ไม่มีการแจ้งเตือนเพราะไม่มีการใช้กฎดังกล่าวในการตรวจจับ ในทำนองเดียวกัน Optimizing Snort Rule มีการแจ้งเตือนของกฎ CHAT MSN message มากที่สุด แต่ Original Snort Rule ไม่มีการแจ้งเตือนเพราะไม่มีการใช้กฎดังกล่าวในการตรวจจับ



รูปที่ 5.14 กราฟแสดงจำนวนการแจ้งเตือนมากที่สุด 3 อันดับแรกแบ่งตามชื่อ Signatures

บทที่ 6
สรุปผลและข้อเสนอแนะโครงการพัฒนาระบบงาน

6.1 สรุปผลโครงการพัฒนาระบบงาน

จากผลการทดสอบการตรวจจับการบุกรุกด้วย Snort Rule ที่เหมาะกับระบบสารสนเทศในองค์กรเกิดการแจ้งเตือนที่ไม่จำเป็นน้อยกว่าการตรวจจับการบุกรุกด้วย Snort Rule ที่ไม่ได้ปรับเปลี่ยนอย่างมากประมาณ 118 เท่า ดังนั้นหัวใจหลักของระบบตรวจจับการบุกรุกด้วย Snort คือ กฎการตรวจจับ ซึ่งกฎการตรวจจับการบุกรุกนั้นมีมากมาย บางครั้งกฎบางข้อก็ไม่ได้ใช้งานเพราะไม่มีการใช้งานสารสนเทศนั้น เช่น บางองค์กรไม่มีการใช้งานซอฟต์แวร์ระบบฐานข้อมูล MS-SQL ก็ไม่มีความจำเป็นที่จะต้องเอาข้อมูลที่ต้องการตรวจจับมาเปรียบเทียบกับกฎที่เกี่ยวกับ MS-SQL ดังนั้นจึงไม่มีความจำเป็นที่จะต้องโหลดกฎที่เกี่ยวกับ MS-SQL ในระบบตรวจจับการบุกรุกด้วย Snort ส่งผลให้ระบบลดการแจ้งเตือนที่ผิดพลาด การทำงานมีประสิทธิภาพมากขึ้น หน่วยประมวลผลทำงานได้ดีขึ้นและใช้หน่วยความจำน้อยลง ดังนั้นจึงสรุปได้ว่าการใช้ระบบตรวจจับการบุกรุกด้วย Snort Rule ที่เหมาะกับสารสนเทศที่มีอยู่ในองค์กร ทำให้ลดการแจ้งเตือนที่ไม่จำเป็นหรือ false alarm จากระบบตรวจจับการบุกรุกได้

6.2 ข้อเสนอแนะโครงการพัฒนาระบบงาน

- ระบบควรมีการกำหนดค่า Threshold ให้กับกฎการตรวจจับ เมื่อมีการแจ้งเตือนมากกว่าค่า Threshold ที่กำหนดไว้ล่วงหน้าแล้ว กฎที่ตรวจจับนั้นก็จะถูกยกเลิกการใช้งาน (Disable Signature) พร้อมทั้งมีการแจ้งเตือนทางอีเมลให้กับผู้ดูแลระบบทราบ เพื่อลดการแจ้งเตือนที่ผิดพลาดหรือไม่จำเป็นออกจากระบบ
- การกำหนดคอนฟิกยังผูกติดกับการออกแบบ การเพิ่มข้อมูลใหม่ๆ ใน Snort Rule Wizard ต้องไปแก้ไข GUI ให้รองรับกับข้อมูลใหม่นั้น แนวทางแก้ไขโดยให้ Snort Rule Wizard อ่านข้อมูลจากตาราง Rule เมื่อมีการเพิ่มข้อมูลในตาราง Rule ก็จะปรากฏใน Snort Rule Wizard ด้วย
- การอัพเดตกฎใหม่ๆ ผู้ใช้ต้องอัพเดตด้วยตัวเองจากโปรแกรม แนวทางแก้ไขโดยสร้างส่วนการอัพเดตกฎให้อ่านกฎใหม่ๆ จากเท็กซ์ไฟล์แล้วอัพเดตลงฐานข้อมูล

บรรณานุกรม

Amoroso, Edward. 1999. **Intrusion Detection An Introduction to Internet Surveillance, Correlation, Trap, Trace back and Response.** New Jersey : Intrusion Net book Pub.

Bace, Rebecca. 1998. "An Introduction to Intrusion Detection and Assessment for System and Network Security Management." **ICSA White Paper.**

Caswell, Brian. 2002. **SNORT 2.0 Intrusion Detection Systems.** Rockland : Syngress Pub.

Harper, Patrick. 2003. **Snort, Apache, PHP, MySQL, ACID on Redhat 9.0 Installation Guide.** [Online]. Available : http://www.snort.org/docs/snort_acid_rh9.pdf.

Innella, Paul and McMillan, Oba. 2002. **An Introduction to Intrusion Detection Systems. [Online]. Available : http://www.securityfocus.com/infocus/1520.**

Kayacik, Hilmi and Nur, Zincir-Heywood. 2003. "A case study of three open source security management tools." pp. 101-104. in **IFIP/IEEE International Symposium Integrated Network Management VIII.** Colorado, USA, March 24-28.

Koziol, Jack. 2003. **Intrusion Detection with SNORT.** 1st ed. Pearson Education Pub.

Proctor, Paul. 2000. **Practical Intrusion Detection Handbook.** 1st ed. Prentice Hall PTR Pub.

Rehman, Rafeeq. 2003. **Intrusion Detection with SNORT: Advanced IDS Techniques Using SNORT, Apache, MySQL, PHP, and ACID.** 1st ed. Prentice Hall PTR Pub.

Roesch, Martin and Green, Chris. 2002. **Snort Users Manual.** [Online]. Available : http://www.snort.org/docs/SnortUsers Manual-2.0.1.pdf.

SANS Institute. 2002. **Intrusion Detection FAQ.** [Online]. Available : http://www.sans.org/newlook/resources/IDFAQ/ID_FAQ.htm.

Scott, Steven. 2003. **Snort Enterprise Implementation Snort, MySQL, SnortCenter and ACID on Redhat 9.0.** [Online]. Available : http://www.superhac.com/docs/ snort_enterprise.pdf.

Stevens, Richard. 1997. **Advanced Programming in UNIX Environment.** Addison Wesley.

Whitman, Michael and Mattord, Herbert. 2003. **Principles of Information Security.** Massachusetts : Thomson Course Technology.

ภาคผนวก

ภาคผนวก ก.

ตัวอย่างรูลไฟล์และคอนฟิกไฟล์ของ Snort

ตัวอย่างรูลไฟล์ของกลุ่มกฎ (Rule Group) การตรวจจับบริการ FTP (ftp.rules)

```
# $Id: ftp.rules, 05/09/2004 02:01:15 cazz Exp $
# ----------------------------------
# Snort IDS Sensor Name : JDTHA
# FTP
# ----------------------------------

alert tcp $EXTERNAL_NET any -> $HOME_NET 21 (msg:"FTP ADMw0rm ftp
login attempt"; flow:to_server,established; content:"USER
w0rm|0D0A|"; reference:arachNIDS,1; classtype:suspicious-login;
sid:144; rev:6;)
alert tcp $EXTERNAL_NET any -> $HOME_NET 21 (msg:"FTP .forward";
content: ".forward"; flow:to_server,established;
reference:arachNIDS,319; classtype:suspicious-filename-detect;
sid:334; rev:4;)
alert tcp $EXTERNAL_NET any -> $HOME_NET 21 (msg:"FTP .rhosts";
flow:to_server,established; content:".rhosts";
reference:arachNIDS,328; classtype:suspicious-filename-detect;
sid:335; rev:4;)
alert tcp $EXTERNAL_NET any -> $HOME_NET 21 (msg:"FTP CWD ~root
attempt"; content:"CWD "; content:" ~root"; nocase;
flow:to_server,established; reference:cve,cve-1999-0082;
classtype:bad-unknown; sid:336; rev:5;)
alert tcp $EXTERNAL_NET any -> $HOME_NET 21 (msg:"FTP CEL overflow
attempt"; flow:to_server,established; content:"CEL "; nocase;
content:!"|0a|"; within:100; reference:bugtraq,679;
classtype:attempted-admin; sid:337; rev:5;)
alert tcp $EXTERNAL_NET any -> $HOME_NET 21 (msg:"FTP adm scan";
flow:to_server,established; content:"PASS ddd@|0a|";
reference:arachNIDS,332; classtype:suspicious-login; sid:353; rev:4;)
alert tcp $EXTERNAL_NET any -> $HOME_NET 21 (msg:"FTP iss scan";
flow:to_server,established; content:"pass -iss@iss";
reference:arachNIDS,331; classtype:suspicious-login; sid:354; rev:4;)
alert tcp $EXTERNAL_NET any -> $HOME_NET 21 (msg:"FTP pass wh00t";
flow:to_server,established; content:"pass wh00t"; nocase;
reference:arachNIDS,324; classtype:suspicious-login; sid:355; rev:4;)
alert tcp $EXTERNAL_NET any -> $HOME_NET 21 (msg:"FTP passwd
retrieval attempt"; flow:to_server,established; content:"RETR";
nocase; content:"passwd"; reference:arachNIDS,213;
classtype:suspicious-filename-detect; sid:356; rev:5;)
alert tcp $EXTERNAL_NET any -> $HOME_NET 21 (msg:"FTP piss scan";
flow:to_server,established; content:"pass -cklaus";
classtype:suspicious-login; sid:357; rev:4;)
alert tcp $EXTERNAL_NET any -> $HOME_NET 21 (msg:"FTP saint scan";
flow:to_server,established; content:"pass -saint";
reference:arachNIDS,330; classtype:suspicious-login; sid:358; rev:4;)
```

```
alert tcp $EXTERNAL_NET any -> $HOME_NET 21 (msg:"FTP satan scan";
flow:to_server,established; content:"pass -satan";
reference:arachNIDS,329; classtype:suspicious-login; sid:359; rev:4;)
alert tcp $EXTERNAL_NET any -> $HOME_NET 21 (msg:"FTP serv-u
directory transversal"; flow:to_server,established; content: ".%20.";
nocase; reference:bugtraq,2052; classtype:bad-unknown; sid:360;
rev:5;)
alert tcp $EXTERNAL_NET any -> $HOME_NET 21 (msg:"FTP site exec";
flow:to_server,established; content:"SITE "; nocase; content:"EXEC ";
distance:0; nocase; reference:bugtraq,2241; classtype:bad-unknown;
sid:361; rev:7;)
alert tcp $EXTERNAL_NET any -> $HOME_NET 21 (msg:"FTP tar
parameters"; flow:to_server,established; content:" --use-compress-
program"; nocase; reference:arachNIDS,134; classtype:bad-unknown;
sid:362; rev:7;)
alert tcp $EXTERNAL_NET any -> $HOME_NET 21 (msg:"FTP USER overflow
attempt"; flow:to_server,established,no_stream; content:"USER ";
nocase; content:!"|0a|"; within:100; reference:cve,can-2000-0479;
classtype:attempted-admin; sid:1734; rev:7;)
alert tcp $EXTERNAL_NET any -> $HOME_NET 21 (msg:"FTP CWD …";
flow:to_server,established; content:"CWD"; nocase; content:"...";
classtype:bad-unknown; sid:1229; rev:5;)
alert tcp $EXTERNAL_NET any -> $HOME_NET 21 (msg:"FTP wu-ftp bad file
completion attempt ["; flow:to_server,established; content:"~";
content:"["; distance:1; reference:cve,cve-2001-0550; classtype:misc-
attack; sid:1377; rev:10;)
alert tcp $EXTERNAL_NET any -> $HOME_NET 21 (msg:"FTP wu-ftp bad file
completion attempt {"; flow:to_server,established; content:"~";
content:"{"; distance:1; reference:cve,cve-2001-0550; classtype:misc-
attack; sid:1378; rev:10;)
alert tcp $EXTERNAL_NET any -> $HOME_NET 21 (msg:"FTP STAT overflow
attempt"; flow:to_server,established; content:"STAT "; nocase;
content:!"|0a|"; within:100;
reference:url,labs.defcom.com/adv/2001/def-2001-31.txt;
classtype:attempted-admin; sid:1379; rev:5;)
alert tcp $EXTERNAL_NET any -> $HOME_NET 21 (msg:"FTP SITE overflow
attempt"; flow:to_server,established; content:"SITE "; nocase;
content:!"|0a|"; within:100; reference:cve,can-2001-0770;
classtype:attempted-admin; sid:1529; rev:7;)
alert tcp $EXTERNAL_NET any -> $HOME_NET 21 (msg:"FTP format string
attempt"; flow:to_server,established; content:"%p"; nocase;
classtype:attempted-admin; sid:1530; rev:4;)
alert tcp $EXTERNAL_NET any -> $HOME_NET 21 (msg:"FTP SITE CHOWN
overflow attempt"; flow:to_server,established; content:"SITE ";
nocase; content:" CHOWN "; nocase; content:!"|0a|"; within:100;
reference:cve,can-2001-0065; classtype:attempted-admin; sid:1562;
rev:6;)
alert tcp $EXTERNAL_NET any -> $HOME_NET 21 (msg:"FTP CMD overflow
attempt"; flow:to_server,established; content:"CMD "; nocase;
content:!"|0a|"; within:100; classtype:attempted-admin; sid:1621;
rev:8;)
alert tcp $EXTERNAL_NET any -> $HOME_NET 21 (msg:"FTP RNFR ././
attempt"; flow:to_server,established; content:"RNFR "; nocase;
content:" ././"; nocase; classtype:misc-attack; sid:1622; rev:5;)
alert tcp $EXTERNAL_NET any -> $HOME_NET 21 (msg:"FTP invalid MODE";
flow:to_server,established; content:"MODE "; nocase; content:!" B";
nocase; content:!" A"; nocase; content:!" S"; nocase; content:!" C";
nocase; classtype:protocol-command-decode; sid:1623; rev:4;)
```

```
alert tcp $EXTERNAL_NET any -> $HOME_NET 21 (msg:"FTP large PWD
command"; flow:to_server,established; content:"PWD"; nocase;
dsize:10; classtype:protocol-command-decode; sid:1624; rev:3;)
alert tcp $EXTERNAL_NET any -> $HOME_NET 21 (msg:"FTP large SYST
command"; flow:to_server,established; content:"SYST"; nocase;
dsize:10; classtype:protocol-command-decode; sid:1625; rev:3;)
alert tcp $EXTERNAL_NET any -> $HOME_NET 21 (msg:"FTP CWD ~<NEWLINE>
attempt"; content:"CWD "; content:" ~|0A|";
flow:to_server,established; reference:cve,can-2001-0421;
classtype:denial-of-service; sid:1672; rev:2;)
alert tcp $EXTERNAL_NET any -> $HOME_NET 21 (msg:"FTP command
overflow attempt"; flow:to_server,established,no_stream; dsize:>100;
reference:bugtraq,4638; classtype:protocol-command-decode; sid:1748;
rev:4;)
alert tcp $EXTERNAL_NET any -> $HOME_NET 21 (msg:"FTP EXPLOIT STAT *
dos attempt"; flow:to_server,established; content:"STAT"; nocase;
content:"*"; distance:1; reference:bugtraq,4482; classtype:attempted-
dos; sid:1777; rev:2;)
alert tcp $EXTERNAL_NET any -> $HOME_NET 21 (msg:"FTP EXPLOIT STAT ?
dos attempt"; flow:to_server,established; content:"STAT"; nocase;
content:"?"; distance:1; reference:bugtraq,4482; classtype:attempted-
dos; sid:1778; rev:2;)
alert tcp $EXTERNAL_NET any -> $HOME_NET 21 (msg:"FTP SITE NEWER
attempt"; flow:to_server,established; content:"SITE "; nocase;
content:" NEWER "; nocase; reference:cve,cve-1999-0880;
classtype:attempted-dos; sid:1864; rev:2;)
alert tcp $EXTERNAL_NET any -> $HOME_NET 21 (msg:"FTP SITE CPWD
overflow attempt"; flow:established,to_server; content:"SITE ";
nocase; content:" CPWD "; nocase; content:!"|0a|"; within:100;
reference:bugtraq,5427; classtype:misc-attack; sid:1888; rev:3;)
alert tcp $EXTERNAL_NET any -> $HOME_NET 21 (msg:"FTP CWD overflow
attempt"; flow:to_server,established; content:"CWD "; nocase;
content:!"|0a|"; within:100; reference:cve,can-2000-1035;
classtype:attempted-admin; sid:1919; rev:3;)
alert tcp $EXTERNAL_NET any -> $HOME_NET 21 (msg:"FTP SITE NEWER
overflow attempt"; flow:to_server,established; content:"SITE ";
nocase; content:" NEWER "; nocase; content:!"|0a|"; within:100;
reference:cve,cve-1999-0800; classtype:attempted-admin; sid:1920;
rev:1;)
alert tcp $EXTERNAL_NET any -> $HOME_NET 21 (msg:"FTP SITE ZIPCHK
attempt"; flow:to_server,established; content:"SITE "; nocase;
content:" ZIPCHK "; nocase; content:!"|0a|"; within:100;
reference:cve,cve-2000-0040; classtype:attempted-admin; sid:1921;
rev:1;)
alert tcp $EXTERNAL_NET any -> $HOME_NET 21 (msg:"FTP
authorized_keys"; flow:to_server,established;
content:"authorized_keys"; classtype:suspicious-filename-detect;
sid:1927; rev:2;)
alert tcp $EXTERNAL_NET any -> $HOME_NET 21 (msg:"FTP shadow
retrieval attempt"; flow:to_server,established; content:"RETR";
nocase; content:"shadow"; classtype:suspicious-filename-detect;
sid:1928; rev:3;)
alert tcp $EXTERNAL_NET any -> $HOME_NET 21 (msg:"FTP RMDIR overflow
attempt"; flow:to_server,established; content:"RMDIR "; nocase;
content:!"|0a|"; within:100; classtype:attempted-admin; sid:1942;
rev:1;)
alert tcp $EXTERNAL_NET any -> $HOME_NET 21 (msg:"FTP SITE EXEC
format string attempt"; flow:to_server,established; content:"SITE";
```

```
nocase; content:"EXEC"; nocase; distance:0; content:"%"; distance:1;
content:"%"; distance:1; classtype:bad-unknown; sid:1971; rev:2;)
alert tcp $EXTERNAL_NET any -> $HOME_NET 21 (msg:"FTP PASS overflow
attempt"; flow:to_server,established,no_stream; content:"PASS ";
nocase; content:!"|0a|"; within:100; reference:cve,can-2000-1035;
classtype:attempted-admin; sid:1972; rev:1;)
alert tcp $EXTERNAL_NET any -> $HOME_NET 21 (msg:"FTP MKD overflow
attempt"; flow:to_server,established; content:"MKD "; nocase;
content:!"|0a|"; within:100; reference:cve,can-1999-0911;
classtype:attempted-admin; sid:1973; rev:1;)
alert tcp $EXTERNAL_NET any -> $HOME_NET 21 (msg:"FTP REST overflow
attempt"; flow:to_server,established; content:"REST "; nocase;
content:!"|0a|"; within:100; reference:cve,can-2001-0826;
classtype:attempted-admin; sid:1974; rev:1;)
alert tcp $EXTERNAL_NET any -> $HOME_NET 21 (msg:"FTP DELE overflow
attempt"; flow:to_server,established; content:"DELE "; nocase;
content:!"|0a|"; within:100; reference:cve,can-2001-0826;
classtype:attempted-admin; sid:1975; rev:1;)
alert tcp $EXTERNAL_NET any -> $HOME_NET 21 (msg:"FTP RMD overflow
attempt"; flow:to_server,established; content:"RMD "; nocase;
content:!"|0a|"; within:100; reference:cve,can-2001-0826;
classtype:attempted-admin; sid:1976; rev:1;)
alert tcp $EXTERNAL_NET any -> $HOME_NET 21 (msg:"FTP LIST directory
traversal attempt"; flow:to_server,established; content:"LIST";
content:".."; distance:1; content:".."; distance:1;
reference:cve,cve-2001-0680; classtype:protocol-command-decode;
sid:1992; rev:2;)
alert tcp $EXTERNAL_NET any -> $HOME_NET 21 (msg:"FTP CWD Root
directory transversal attempt"; flow:to_server,established;
content:"CWD"; nocase; content:"C\:\\"; distance:1;
reference:nessus,11677; classtype:protocol-command-decode; sid:2125;
rev:4;)
alert tcp $EXTERNAL_NET any -> $HOME_NET 21 (msg:"FTP USER format
string attempt"; flow:to_server,established; content:"USER"; nocase;
content:"%"; distance:1; content:"%"; distance:1; within:10;
reference:bugtraq,7474; classtype:misc-attack; sid:2178; rev:1;)
alert tcp $EXTERNAL_NET any -> $HOME_NET 21 (msg:"FTP PASS format
string attempt"; flow:to_server,established; content:"PASS"; nocase;
content:"%"; distance:1; content:"%"; distance:1; within:10;
reference:bugtraq,7474; classtype:misc-attack; sid:2179; rev:1;)
```

ตัวอย่าง Snort Configuration files

- **snort.conf**

```
#--------------------------------------------------
#   Snort Rule Management     Snort RuleSet
#   Faculty Information Technology KMITL
#--------------------------------------------------
# $Id: snort.conf, 05/09/2004 02:01:13 cazz Exp $
##################################################
# This file contains a sample snort configuration.
# You can take the following steps to create your
#
#  1) Set the network variables for your network
#  2) Configure preprocessors
#  3) Configure output plugins
#  4) Customize your rule set
#
##################################################
# Step #1: Set the network variables:

# Specify lists of IP addresses for HOME_NET.
var HOME_NET 172.30.68.0/24

# Set up the external network addresses as well.
var EXTERNAL_NET any

# List of DNS servers on your network.
var DNS_SERVERS $HOME_NET

# List of SMTP servers on your network.
var SMTP_SERVERS $HOME_NET

# List of web servers on your network.
var HTTP_SERVERS $HOME_NET

# List of sql servers on your network.
var SQL_SERVERS $HOME_NET

# List of telnet servers on your network.
var TELNET_SERVERS $HOME_NET

# Ports you run web servers on.
var HTTP_PORTS 80

# Ports you want to look for SHELLCODE on.
var SHELLCODE_PORTS !80

# Ports you do oracle attacks on.
var ORACLE_PORTS 1521

# AIM servers.
var AIM_SERVERS
[64.12.24.0/24,64.12.25.0/24,64.12.26.14/24,64.12.28.0/24,64.12.29.0/
24,64.12.161.0/24,64.12.163.0/24,205.188.5.0/24,205.188.9.0/24]

# Path to your rules files.
var RULE_PATH c:\snort\rules
```

```
#################################################
# Step #2: Configure preprocessors

# General configuration for preprocessors is of the form
# preprocessor <name_of_processor>: <configuration_options>
preprocessor frag2

# stream4: stateful inspection/stream reassembly for Snort
preprocessor stream4: detect_scans, disable_evasion_alerts
preprocessor stream4_reassemble: both

# http_decode: normalize HTTP requests
preprocessor http_decode: 80 8877 unicode iis_alt_unicode
double_encode iis_flip_slash full_whitespace

# rpc_decode: normalize RPC traffic
preprocessor rpc_decode: 111 32771

# bo: Back Orifice detector
preprocessor bo

# telnet_decode: Telnet negotiation string normalizer
preprocessor telnet_decode

# Portscan: detect a variety of portscans
#preprocessor portscan: $HOME_NET 4 3 portscan.log
#preprocessor portscan-ignorehosts: 0.0.0.0

# Conversation
preprocessor conversation: allowed_ip_protocols all, timeout 60,
max_conversations 65535, alert_odd_protocols

#################################################################
# Step #3: Configure output plugins

# General configuration for output plugins is of the form:
output database: alert, mysql, dbname=snort host=localhost port=7788
user=root password=snortids detail=full

# Include classification & priority settings
include c:\snort\etc\classification.config

# Include reference systems
include c:\snort\etc\reference.config

#################################################################
# Step #4: Customize your rule set

include $RULE_PATH/attack-responses.rules
include $RULE_PATH/backdoor.rules
include $RULE_PATH/bad-traffic.rules
include $RULE_PATH/chat.rules
include $RULE_PATH/ddos.rules
include $RULE_PATH/deleted.rules
include $RULE_PATH/dns.rules
include $RULE_PATH/dos.rules
include $RULE_PATH/experimental.rules
include $RULE_PATH/exploit.rules
```

```
include $RULE_PATH/finger.rules
include $RULE_PATH/ftp.rules
include $RULE_PATH/icmp.rules
include $RULE_PATH/icmp-info.rules
include $RULE_PATH/imap.rules
include $RULE_PATH/info.rules
include $RULE_PATH/local.rules
include $RULE_PATH/misc.rules
include $RULE_PATH/multimedia.rules
include $RULE_PATH/mysql.rules
include $RULE_PATH/netbios.rules
include $RULE_PATH/nntp.rules
include $RULE_PATH/oracle.rules
include $RULE_PATH/other-ids.rules
include $RULE_PATH/p2p.rules
include $RULE_PATH/policy.rules
include $RULE_PATH/pop2.rules
include $RULE_PATH/pop3.rules
include $RULE_PATH/porn.rules
include $RULE_PATH/rpc.rules
include $RULE_PATH/rservices.rules
include $RULE_PATH/scan.rules
include $RULE_PATH/shellcode.rules
include $RULE_PATH/smtp.rules
include $RULE_PATH/snmp.rules
include $RULE_PATH/sql.rules
include $RULE_PATH/telnet.rules
include $RULE_PATH/tftp.rules
include $RULE_PATH/virus.rules
include $RULE_PATH/web-attacks.rules
include $RULE_PATH/web-cgi.rules
include $RULE_PATH/web-client.rules
include $RULE_PATH/web-coldfusion.rules
include $RULE_PATH/web-frontpage.rules
include $RULE_PATH/web-iis.rules
include $RULE_PATH/web-misc.rules
include $RULE_PATH/web-php.rules
include $RULE_PATH/x11.rules
```

- **classification.config**

```
# $Id: classification.config, 05/09/2004 02:01:13 cazz Exp $
# The following includes information for prioritizing rules
# config classification:shortname,short description,priority
config classification: not-suspicious,Not Suspicious Traffic,3
config classification: unknown,Unknown Traffic,3
config classification: bad-unknown,Potentially Bad Traffic,2
config classification: attempted-recon,Attempted Information Leak,2
config classification: successful-recon-limited,Information Leak,2
config classification: successful-recon-largescale,Large Scale
Information Leak,2
config classification: attempted-dos,Attempted Denial of Service,2
config classification: successful-dos,Denial of Service,2
config classification: attempted-user,Attempted User Privilege Gain,1
config classification: unsuccessful-user,Unsuccessful User Privilege
Gain,1
config classification: successful-user,Successful User Privilege
Gain,1
config classification: attempted-admin,Attempted Administrator
Privilege Gain,1
config classification: successful-admin,Successful Administrator
Privilege Gain,1
config classification: rpc-portmap-decode,Decode of an RPC Query,2
config classification: shellcode-detect,Executable code was
detected,1
config classification: string-detect,A suspicious string was
detected,3
config classification: suspicious-filename-detect,A suspicious
filename was detected,2
config classification: suspicious-login,An attempted login using a
suspicious username was detected,2
config classification: system-call-detect,A system call was
detected,2
config classification: tcp-connection,A TCP connection was detected,4
config classification: trojan-activity,A Network Trojan was
detected,1
config classification: unusual-client-port-connection,A client was
using an unusual port,2
config classification: network-scan,Detection of a Network Scan,3
config classification: denial-of-service,Detection of a Denial of
Service Attack,2
config classification: non-standard-protocol,Detection of a non-
standard protocol or event,2
config classification: protocol-command-decode,Generic Protocol
Command Decode,3
config classification: web-application-activity,access to a
potentially vulnerable web application,2
config classification: web-application-attack,Web Application
Attack,1
config classification: misc-activity,Misc activity,3
config classification: misc-attack,Misc Attack,2
config classification: icmp-event,Generic ICMP event,3
config classification: kickass-porn,SCORE! Get the lotion!,1
config classification: policy-violation,Potential Corporate Privacy
Violation,1
config classification: default-login-attempt,Attempt to login by a
default username and password,2
```

- **reference.config**

```
# $Id: reference.config, 05/09/2004 02:01:13 chrisgreen Exp $
# The following defines URLs for the references found in the rules
#
# config reference: system URL
#
config reference: bugtraq http://www.securityfocus.com/bid/
config reference: cve http://cve.mitre.org/cgi-bin/cvename.cgi?name=
config reference: arachNIDS http://www.whitehats.com/info/IDS
config reference: McAfee http://vil.nai.com/vil/content/v_
config reference: nessus http://cgi.nessus.org/plugins/dump.php3?id=
config reference: url http://
```

ประวัติผู้เขียน

ชื่อ-นามสกุล นายพิศาล ศิริบัณฑิตย์

ประวัติการศึกษา คณะวิศวกรรมศาสตร์ สาขาวิศวกรรมคอมพิวเตอร์
มหาวิทยาลัยสงขลานครินทร์ วิทยาเขตหาดใหญ่

Email spisarn@hotmail.com