ระบบตรวจจับผู้บุกรุกระบบคอมพิวเตอร์

INTRUSION DETECTION SYSTEM

นางสาวสุธิดา วัฒนชัย
นางสาวหนึ่งฤทัย ชัยเสวิกุล

ปริญญานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิศวกรรมศาสตรบัณฑิต
ภาควิชาวิศวกรรมคอมพิวเตอร์
คณะวิศวกรรมศาสตร์
สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง
ปีการศึกษา 2542

ระบบตรวจจับผู้บุกรุกระบบคอมพิวเตอร์
INTRUSION DETECTION SYSTEM

โดย

นางสาวสุธิดา วัฒนชัย

นางสาวหนึ่งฤทัย ชัยเสวิกุล

อาจารย์ที่ปรึกษา

อาจารย์ธนา หงษ์สุวรรณ

อาจารย์อัครเดช วัชระภูพงษ์

ปริญญานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิศวกรรมศาสตรบัณฑิต

ภาควิชาวิศวกรรมคอมพิวเตอร์

คณะวิศวกรรมศาสตร์

สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

ปีการศึกษา 2542

ปริญญานิพนธ์ปีการศึกษา 2542

ภาควิชา วิศวกรรมคอมพิวเตอร์

คณะวิศวกรรมศาสตร์ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

เรื่อง ระบบตรวจจับผู้บุกรุกระบบคอมพิวเตอร์

INTRUSION DETECTION SYSTEM

ผู้จัดทำ

1. นางสาวสุธิดา วัฒนชัย รหัสประจำตัว 39014594
2. นางสาวหนึ่งฤทัย ชัยเสวิกุล รหัสประจำตัว 39014633

_____ อาจารย์ที่ปรึกษา

(นายธนา หงษ์สุวรรณ)

_____ อาจารย์ที่ปรึกษา

(นายอัครเดช วัชระภูพงษ์)

ระบบตรวจจับผู้บุกรุกระบบคอมพิวเตอร์

นางสาวสุธิดา วัฒนชัย 39014594
นางสาวหนึ่งฤทัย ชัยเสวิกุล 39014633
อ.ธนา หงษ์สุวรรณ อาจารย์ที่ปรึกษา
อ.อัครเดช วัชระภูพงษ์ อาจารย์ที่ปรึกษา
ปีการศึกษา 2542

บทคัดย่อ

ระบบปฏิบัติการลีนุกซ์เป็นระบบปฏิบัติการที่มีแนวโน้มที่จะถูกนำมาใช้อย่างแพร่หลายในอนาคต เพราะไม่ต้องซื้อและสามารถทำงานกับระบบที่มีทรัพยากรไม่มาก แต่ลีนุกซ์ยังมีความปลอดภัยไม่เพียงพอสำหรับการใช้งานในองค์กรที่ต้องการความปลอดภัยสูง เนื่องจากเป็นระบบปฏิบัติการที่เปิดเผยซอร์สโค้ด ทำให้ผู้ดูแลระบบต้องทำงานหนักในการดูแลเรื่องความปลอดภัยของระบบ

โครงงานวิจัยนี้จัดทำขึ้นเพื่อสร้างซอฟต์แวร์สำหรับตรวจจับการบุกรุก เพื่อเป็นการลดภาระของผู้ดูแลระบบ โดยระบบตรวจจับผู้บุกรุกระบบคอมพิวเตอร์นี้จะตรวจสอบพฤติกรรมต่าง ๆ ของผู้ใช้แต่ละคน และวิเคราะห์ว่าผู้ใช้คนใดมีพฤติกรรมที่มีโอกาสเป็นผู้บุกรุกระบบได้ แล้วแจ้งเตือนแก่ผู้ดูแลระบบพร้อมกับบันทึกชื่อผู้ใช้ที่เป็นผู้บุกรุกนั้นให้ผู้ดูแลระบบติดตามพฤติกรรมต่อไป

Intrusion Detection System

Ms. Suthida Wattanachai
Ms. Nuengruetai Chaisewikul
Mr. Thana Hongsuwan Advisor
Mr. Akkradach Watcharapupong Advisor

ABSTRACT

Linux, a well-known free operating system, shows its trend to be used more widely in the future, as it requires very small resources. However its opened source code makes a number of troubles to system administrators of organizations that a strong data security is essential.

The objective of this thesis is to develop software, which is able to detect system intrusions. This may also reduce the system administrator tasks in administrating the systems. The software monitors users' behaviours and analyses whether they have behaviours in intruding the system. It can also alert and report situation to the system administrators. The intruding behaviours are recorded.

กิตติกรรมประกาศ

ปริญญานิพนธ์ฉบับนี้จะไม่สามารถเสร็จสมบูรณ์ได้ถ้าไม่ได้รับคำแนะนำ คำเตือน ทั้งหลายจาก อาจารย์ ธนา หงษ์สุวรรณ, อาจารย์ อัครเดช วัชระภูพงษ์ และ อาจารย์ สุมณฑา หลิมศิริวงษ์ คณะผู้จัดทำขอขอบพระคุณยิ่งสำหรับทุกสิ่งทุกอย่างที่ได้รับจากท่านทั้งสาม

นอกจากนี้ต้องขอขอบพระคุณอาจารย์ทุกท่านในสถาบันนี้ที่ได้สอนสั่งคณะผู้จัดทำจนมีความรู้ความสามารถจนถึงทุกวันนี้ รวมทั้งคณาจารย์ทุกท่านในภาควิชาวิศวกรรมคอมพิวเตอร์ที่ทำให้คณะผู้จัดทำได้เป็นวิศวกรคอมพิวเตอร์อย่างเต็มภาคภูมิ

ขอขอบคุณภาควิชาวิศวกรรมคอมพิวเตอร์โดยเฉพาะห้องวิจัยและพัฒนาการรักษาความปลอดภัยข้อมูล (ISAG) ที่ได้เอื้อเฟื้อสถานที่ ให้คณะผู้จัดทำได้ทำการวิจัย และช่วยอำนวยความสะดวกต่าง ๆ

ขอขอบคุณพี่ก้อง, พี่ระ, พี่เชาว์, พี่ชาติ, พี่เอิง, พี่อรรถ, น้องกระบี่, น้องโจ้, น้องตั้ม, ต่อ, บุ๋ม, เดียร์, ตั้ม, หลิว และเจี๊ยบ ที่คอยให้ความช่วยเหลือคณะผู้จัดทำในการทำงานตลอดเวลา เป็นที่ปรึกษายามมีปัญหา รวมทั้งให้ยืมใช้ทรัพยากรจำเป็นต่าง ๆ

ที่สำคัญและขาดมิได้ คือ ต้องขอขอบพระคุณบิดา มารดาที่ได้ให้กำเนิด คอยสั่งสอน และให้การสนับสนุนการศึกษา กิจกรรมต่าง ๆ นับเป็นพระคุณที่หาใดเปรียบมิได้ ทางคณะผู้จัดทำขอกราบขอบพระคุณมา ณ ที่นี้ด้วย

สุธิดา วัฒนชัย
หนึ่งฤทัย ชัยเสวิกุล

สารบัญ

สารบัญ (ต่อ)

สารบัญ (ต่อ)

สารบัญตาราง

สารบัญภาพ

หน้าที่

สารบัญภาพ (ต่อ)

หน้าที่

บทที่ 1

บทนำ

1.1 ความสำคัญและที่มา

ลีนุกซ์ (Linux) เป็นระบบปฏิบัติการที่มีแนวโน้มความนิยมใช้ในอนาคตสูง เพราะไม่ต้องซื้อและต้องการทรัพยากรระบบน้อยเมื่อเทียบกับระบบปฏิบัติการตัวอื่นอย่างระบบปฏิบัติการตระกูลวินโดว์ส (Windows) ทำให้ผู้ทำโครงงานเกิดความสนใจในการเขียนแอพพลิเคชันบนระบบปฏิบัติการนี้ นอกจากนั้นระบบปฏิบัติการลีนุกซ์ยังมีส่วนคล้ายกับระบบปฏิบัติการยูนิกซ์(Unix) การศึกษาการทำงานของลีนุกซ์สามารถใช้เป็นพื้นฐานที่ดีในการศึกษาการทำงานของระบบปฏิบัติการยูนิกซ์ต่อไป

เนื่องจากลีนุกซ์เปิดเผยซอร์สโค้ด (Open source) ทำให้ทุกคนสามารถแกะซอร์สโค้ดและผู้บุกรุกสามารถหาช่องโหว่ของระบบปฏิบัติการตัวนี้ได้ ลีนุกซ์จึงไม่ค่อยมีความปลอดภัยนัก ผู้ดูแลระบบต้องทำงานหนักเพื่อป้องกันระบบจากการบุกรุกทั้งการบุกรุกจากภายนอกและจากภายใน ซึ่งการบุกรุกจากภายนอกอาจใช้ไฟร์วอลล์ในการป้องกันได้ แต่ก็ป้องกันได้เพียงบางส่วนและไม่สามารถตรวจจับการบุกรุกจากภายในได้เลย จึงควรมีระบบอีกระบบหนึ่งสำหรับตรวจจับการบุกรุกจากภายในและการบุกรุกที่ไฟร์วอลล์ไม่สามารถป้องกันได้

จากสาเหตุที่กล่าวมานี้ ผู้จัดทำโครงงานจึงคิดทำระบบตรวจจับผู้บุกรุกระบบคอมพิวเตอร์ซึ่งทำงานอยู่บนระบบปฏิบัติการลีนุกซ์นี้ขึ้น

1.2 วัตถุประสงค์

1.2.1 สร้างระบบตรวจจับผู้บุกรุกระบบคอมพิวเตอร์บนระบบปฏิบัติการลีนุกซ์

1.2.2 ศึกษาและเขียนแอพพลิเคชันบนระบบปฏิบัติการลีนุกซ์

1.2.3 ศึกษาการทำงานของระบบปฏิบัติการลีนุกซ์

1.3 ขอบเขตของงานวิจัย

ในงานวิจัยนี้ได้สร้างระบบตรวจจับผู้บุกรุกระบบคอมพิวเตอร์สำหรับระบบปฏิบัติการ Slackware 4.0 ถ้านำไปใช้กับลีนุกซ์ระบบอื่นอาจมีปัญหาในการกรองข้อมูล เนื่องจากที่อยู่ของไฟล์ล็อกและรูปแบบในการเก็บล็อกไม่เหมือนกัน และต้องมีการตั้งค่าบางอย่างตามที่ได้ระบุไว้

ระบบตรวจจับผู้บุกรุกนี้สามารถตรวจจับผู้บุกรุก 2 แบบ คือ ผู้บุกรุกที่มีพฤติกรรมตรงกับรูปแบบการบุกรุกที่รู้จัก และ ผู้บุกรุกที่ใช้งานทรัพยากรระบบอย่างผิดปรกติ

การตรวจจับการบุกรุกระบบคอมพิวเตอร์แบบแรกสามารถตรวจจับผู้บุกรุกที่มีลำดับพฤติกรรมตรงตามรูปแบบของการบุกรุกที่เป็นที่ศึกษาไว้ ทั้งนี้ไม่รวมถึงการพัฒนาส่วนปัญญาประดิษฐ์ และเนื่องจากมีการคิดค้นวิธีบุกรุกแบบใหม่ ๆ ขึ้นตลอดเวลา ทำให้ไม่สามารถตรวจพบการบุกรุกกรณีนอกเหนือไปจากที่ได้ศึกษาไว้ ซึ่งในงานวิจัยนี้พยายามลดช่องโหว่ของการตรวจจับผู้บุกรุกระบบ

คอมพิวเตอร์แบบนี้ โดยเพิ่มส่วนตรวจจับผู้บุกรุกแบบตรวจสอบการใช้งานทรัพยากรระบบที่ผิดปรกติเข้ามา

การตรวจจับการบุกรุกแบบที่สองเป็นการตรวจจับการใช้งานทรัพยากรระบบที่ผิดปรกติ ซึ่งอาจเป็นพฤติกรรมของการบุกรุกวิธีใหม่ ๆ ที่ไม่ได้ศึกษาไว้ การตรวจจับแบบนี้มีข้อจำกัดตรงที่ไม่สามารถระบุได้แน่นอนว่าการใช้งานทรัพยากรระบบที่ผิดปรกตินั้น เกิดจากการใช้งานของผู้ใช้ธรรมดา หรือเกิดจากการพยายามบุกรุกเข้ามาในระบบ เพียงแต่สามารถรายงานผลว่าเกิดการใช้งานทรัพยากรระบบอย่างผิดปรกติขึ้น

นอกจากข้อจำกัดที่ได้กล่าวมาข้างต้น ยังมีข้อจำกัดของงานวิจัยอีกอย่างหนึ่งคือข้อจำกับเกี่ยวกับไฟล์ล็อกของระบบปฏิบัติการที่สร้างขึ้นมาอาจมีข้อมูลไม่เพียงพอสำหรับการวิเคราะห์การบุกรุก เช่น การเข้าใช้งานระบบโดยใช้แอพพลิเคชันใหม่ ๆ

1.4 วิธีการดำเนินงาน

การดำเนินงานเริ่มต้นจากการศึกษาทฤษฎีและแนวความคิดต่าง ๆ เกี่ยวกับระบบตรวจจับผู้บุกรุกระบบคอมพิวเตอร์และการทำงานของระบบปฏิบัติการลีนุกซ์ซึ่งแสดงรายละเอียดไว้ในบทที่ 2

จากนั้นนำความรู้ที่ได้มาออกแบบระบบตรวจจับผู้บุกรุกระบบคอมพิวเตอร์สำหรับระบบปฏิบัติการลีนุกซ์ขึ้น รายละเอียดของการออกแบบแสดงไว้ในบทที่ 3

หลังจากนั้นเข้าสู่ขั้นตอนของการพัฒนาแอพพลิเคชัน เริ่มจากการเลือกภาษาคอมพิวเตอร์ที่ใช้ในการพัฒนาระบบและศึกษาการเขียนภาษานั้น ๆ ต่อมาจึงพัฒนาแอพพลิเคชันโดยใช้วิธีพัฒนาแบบวนรอบ (Iterative process model)

เมื่อได้ทำการทดลองและแก้ไขข้อผิดพลาดต่าง ๆ เรียบร้อยแล้ว ก็นำมาสรุปผลการดำเนินงานของงานวิจัยนี้ รวมถึงหาแนวทางในการปรับปรุงและพัฒนาในอนาคตซึ่งอธิบายรายละเอียดในบทที่ 4-5

บทที่ 2
ทฤษฎีและหลักการเบื้องต้น

2.1 ระบบตรวจจับผู้บุกรุกระบบคอมพิวเตอร์ (Intrusion Detection System)

2.1.1 ความหมายของระบบตรวจจับผู้บุกรุกระบบคอมพิวเตอร์

ระบบตรวจจับผู้บุกรุกระบบคอมพิวเตอร์ (Intrusion Detection System: IDS) เป็นส่วนให้ความช่วยเหลือระบบคอมพิวเตอร์ ในการเตรียมการรับมือกับการบุกรุก ระบบตรวจจับผู้บุกรุกระบบคอมพิวเตอร์รวบรวมข้อมูลจากแหล่งข้อมูลหลาย ๆ แหล่ง ทั้งจากภายในระบบและจากเครือข่ายคอมพิวเตอร์ แล้วทำการวิเคราะห์ข้อมูลเหล่านั้นเพื่อหาลักษณาการที่บ่งบอกว่ามีการบุกรุกระบบเกิดขึ้น ในบางกรณีระบบตรวจจับผู้บุกรุกระบบคอมพิวเตอร์อาจอนุญาตให้ผู้ใช้กำหนดวิธีการตอบสนองต่อการบุกรุกเองได้

กล่าวโดยสรุปแล้วระบบตรวจจับผู้บุกรุกระบบคอมพิวเตอร์ คือ ระบบที่ทำหน้าที่ติดตามดูการทำงานที่เกิดขึ้นบนระบบคอมพิวเตอร์ เพื่อค้นหาร่องรอยที่บอกว่ามีผู้กำลังพยายามบุกรุกระบบคอมพิวเตอร์ หรือค้นหาการกระทำที่เกินขอบเขตสิทธิ์ของผู้ใช้ระบบ

ผู้บุกรุกระบบคอมพิวเตอร์หรือ Intruder หมายถึงบุคคลที่พยายามบุกรุกหรือได้บุกรุกเข้ามาในระบบโดยที่ไม่ได้รับอนุญาต หมายความถึงบุคคลที่เรียกว่าแฮ็กเกอร์ (Hacker) คือผู้ที่ชอบเข้าไปศึกษาบางสิ่งบางอย่างในระบบ เช่นเข้าไปศึกษาหลักการทำงานของระบบและโปรแกรม

2.1.2 ความจำเป็นของระบบตรวจจับผู้บุกรุกระบบคอมพิวเตอร์

ผู้ดูแลระบบอาจคิดว่าในระบบที่ตนดูแลอยู่ไม่มีข้อมูลสำคัญ ถึงมีผู้บุกรุกเข้ามาก็ไม่เป็นไร แต่ความเป็นจริงแล้วสามารถเกิดกรณีที่มีผู้บุกรุกเข้ามายังระบบ แล้วใช้เป็นทางผ่านในการเจาะระบบของธนาคาร หรือหน่วยงานที่มีข้อมูลที่สำคัญ และเข้าไปทำความเสียหายให้กับระบบนั้น ๆ ถ้ามีการตรวจสอบกลับมา เจ้าของระบบต้องเป็นผู้รับผิดชอบกับเหตุการณ์ที่เกิดขึ้น ดังนั้นการรักษาความปลอดภัยของระบบจึงเป็นเรื่องจำเป็นที่ละเลยไม่ได้

การบุกรุกถ้าแบ่งจากสถานที่ที่ติดต่อเข้ามาของผู้บุกรุก สามารถแบ่งได้เป็น 2 ประเภท คือ การบุกรุกจากภายในเครือข่ายเอง และบุกรุกจากภายนอกเครือข่าย

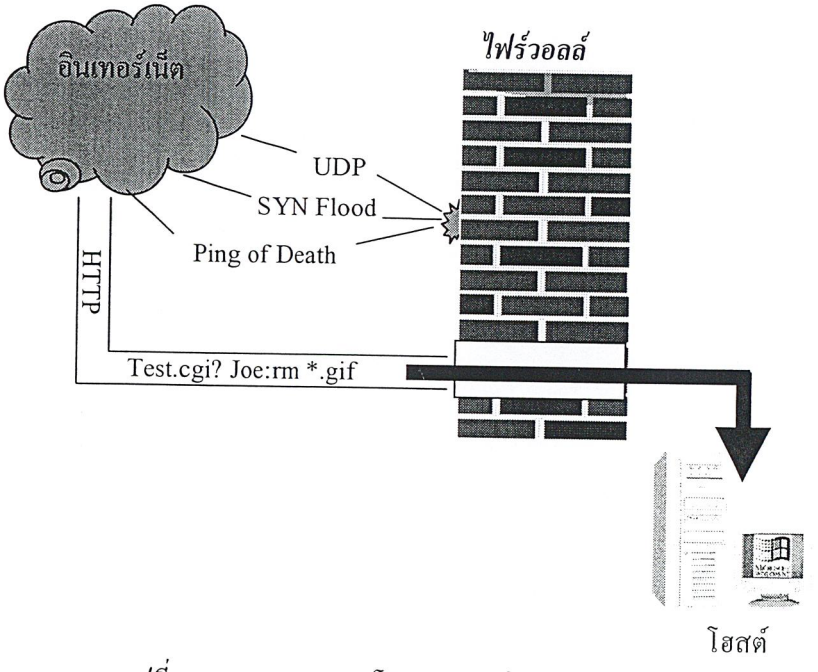
การบุกรุกจากภายนอก เป็นการบุกรุกที่มาจากภายนอกเน็ตเวิร์กของระบบ และเข้ามาทำความเสียหายให้แก่ระบบ เช่น เข้ามาเปลี่ยนข้อมูลในโฮมเพจ หรือส่งสแปมเมล์ไปให้ผู้อื่นโดยผ่านระบบของหรือพยายามบุกรุกผ่านไฟร์วอลล์เข้ามาทำความเสียหายให้กับเครื่องที่อยู่ภายในเน็ตเวิร์ก ผู้บุกรุกจากภายนอกสามารถเข้ามาโดยผ่านบริการ (Services) ของระบบ หรือติดต่อผ่านโมเด็มเข้ามา

การบุกรุกจากภายใน เป็นการบุกรุกโดยผู้ที่มีสิทธิ์อันชอบธรรมที่เข้ามาใช้ทรัพยากรในระบบ แต่ใช้งานทรัพยากรอย่างไม่ถูกต้อง หรือพยายามแอบอ้างไปใช้สิทธิ์ของผู้ใช้อื่นที่มีสิทธิ์ในการใช้งานเหนือกว่า

ตามสถิติของการบุกรุกที่ตรวจพบ 80 เปอร์เซ็นต์ เกิดจากการบุกรุกจากภายใน ถึงแม้ว่าระบบมีการติดตั้งไฟร์วอลล์เพื่อป้องกันการบุกรุก แต่การมีไฟร์วอลล์เพียงอย่างเดียวก็เหมือนกับการมีเครื่องกีดขวางมากั้นระหว่างระบบคอมพิวเตอร์ภายในเครือข่าย (Internal networks) กับระบบภายนอก (Internet) เท่านั้น ไม่สามารถป้องกันการบุกรุกจากภายในระบบได้

นอกจากนี้ไฟร์วอลล์ยังมีจุดอ่อนสำคัญ 2 ประการ คือ

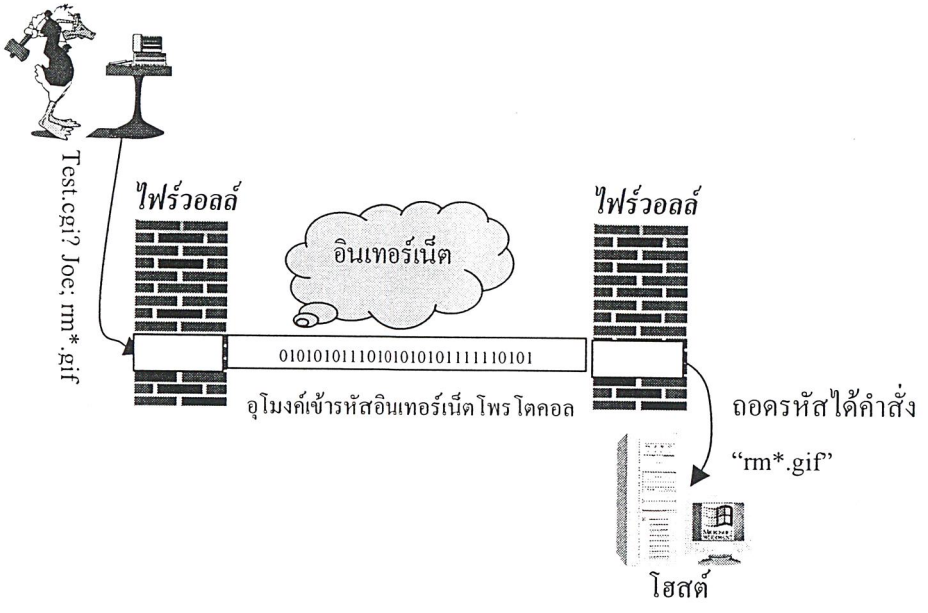
1. ไม่สามารถป้องกันการบุกรุกที่อาศัยช่องโหว่ของแอพพลิเคชันได้ ดังแสดงในรูปที่ 2-1 พบว่าไฟร์วอลล์กรองแพ็กเก็ตตามที่ถูกระบุไว้ สามารถป้องกันการส่ง SYN เป็นจำนวนมาก (SYN Flood) ได้ แต่ยอมให้แพ็กเก็ตของ HTTP ผ่าน ทำให้สามารถส่งไฟล์ใด ๆ ที่มีนามสกุล cgi หรือนามสกุล phf (phf เป็นไลบรารีที่มากับ httpd ของ NCSA) เข้ามาในระบบได้ ซึ่งอาจมีผู้ไม่ประสงค์ดีส่งไฟล์ที่มีนามสกุลเหล่านี้เข้ามาเพื่อทำอันตรายต่อระบบ หรือเพื่อบุกรุกเข้ามาในระบบ เช่นในตัวอย่าง ผู้บุกรุกส่งไฟล์ชื่อ test.cgi เข้ามาในระบบเพื่อลบไฟล์ทั้งหมดที่มีนามสกุล gif ในเว็บเซิร์ฟเวอร์ซึ่งทำให้โฮมเพจในนั้นเสียหาย



รูปที่ 2-1 การบุกรุกระบบโดยอาศัยจุดอ่อนของแอพพลิเคชัน

2. ไฟร์วอลล์กรองแพ็กเก็ตต่าง ๆ โดยพิจารณาจากโพรโตคอลในชั้นเน็ตเวิร์ก ถ้าข้อมูลที่ผู้บุกรุกส่งมามีการเข้ารหัสถูกต้องตามโพรโตคอล ไฟร์วอลล์ยอมให้แพ็กเก็ตนั้นผ่านเข้ามาได้ เมื่อแพ็กเก็ตนั้นเข้ามาในระบบจะถูกถอดรหัสออก ซึ่งผู้บุกรุกสามารถซ่อนข้อมูลใด ๆ เข้ามาในระบบได้ ดังแสดงในรูปที่ 2-2 ผู้บุกรุกส่ง "test.cgi ?Joe;rm *.gif" ผ่านอุโมงค์การเข้ารหัสในอินเทอร์เน็ตโพรโตคอล ทำ

ให้แพ็กเก็ตนี้สามารถผ่านไฟร์วอลล์เข้ามาได้ เมื่อถอดรหัสออกแล้วได้คำสั่ง "rm *.gif" แก่เซิร์ฟเวอร์ ทำให้ไฟล์รูปภาพที่มีนามสกุล gif ถูกลบออกจากระบบ



รูปที่ 2-2 การบุกรุกระบบโดยอาศัยอุโมงค์ของอินเทอร์เน็ตโพรโตคอล

ดังนั้นการใช้ไฟร์วอลล์แต่เพียงอย่างเดียวจึงไม่เพียงพอสำหรับการรักษาความปลอดภัยในระบบคอมพิวเตอร์ จำเป็นต้องมีระบบตรวจจับผู้บุกรุกระบบคอมพิวเตอร์มาช่วยเสริมการทำงาน เปรียบเหมือนกับการที่มีรั้วกั้น และมียามเฝ้าประตูคอยตรวจสอบกันบุคคลไม่พึงประสงค์เข้ามาก็ตาม ผู้บุกรุกอาจใช้วิธีปลอมแปลง หรือหาช่องทางพิเศษเล็ดลอดเข้ามาได้ และตัวไฟร์วอลล์เอง ไม่สามารถบอกได้เลยว่าภายในระบบกำลังมีอะไรเกิดขึ้นบ้าง จึงต้องมีโทรทัศน์วงจรปิด ทำงานร่วมกับสัญญาณกันขโมย คอยรวบรวมข้อมูลจากแหล่งต่าง ๆ มาวิเคราะห์ ช่วยให้สามารถตรวจจับการบุกรุกที่มาจากภายในระบบ รวมทั้งผู้บุกรุกที่เล็ดลอดผ่านไฟร์วอลล์ เข้ามาได้ ซึ่งก็คือระบบตรวจจับผู้บุกรุกระบบคอมพิวเตอร์นั่นเอง

2.1.3 ชนิดของระบบตรวจจับผู้บุกรุกระบบคอมพิวเตอร์

ระบบตรวจจับผู้บุกรุกผู้บุกรุกระบบคอมพิวเตอร์แบ่งตามแหล่งข้อมูลได้ 3 ชนิด ดังนี้

1. ระบบตรวจจับผู้บุกรุกทางเน็ตเวิร์ก (Network Intrusion Detection System : NIDS) เป็นระบบตรวจจับแพ็กเก็ตที่วิ่งอยู่ในเน็ตเวิร์ก พยายามตรวจสอบว่ามีผู้บุกรุกพยายามบุกรุกเข้าสู่ระบบหรือไม่ ตัวอย่างของระบบนี้คือ ระบบที่ตรวจสอบว่ามีแพ็กเก็ตของการเชื่อมต่อแบบทีซีพี/ไอพี (TCP/IP) ชื่อว่า SYN ส่งเข้ามายังระบบเป็นจำนวนมากอย่างผิดปกติในเครื่องเป้าหมาย ซึ่งการกระทำแบบนี้สามารถตรวจสอบว่ามีผู้บุกรุกพยายามสแกนพอร์ตทีซีพี (TCP port) ของเครื่องอยู่หรือไม่ ซึ่งระบบตรวจจับผู้บุกรุกทางเน็ตเวิร์กสามารถทำงานที่เครื่องเป้าหมาย หรือเครื่องที่ทำหน้าที่เฉพาะในการเฝ้าดูเหตุการณ์ไม่ปกติในเน็ตเวิร์กได้ ดังนั้นระบบตรวจจับผู้บุกรุกทางเน็ตเวิร์กสามารถตรวจสอบเครื่องใด ๆ ก็ได้ในเน็ตเวิร์ก

2. ระบบตรวจสอบความถูกต้องของข้อมูลในระบบ (System Integrity Verifiers : SIV) ตรวจสอบความถูกต้องของข้อมูลในระบบ เพื่อค้นหาว่ามีผู้บุกรุกพยายามเปลี่ยนแปลงข้อมูลของไฟล์ระบบ (system file) หรือ ส่วนประกอบอื่นๆ (component) เช่น ไฟล์ที่เป็นรีจิสตรี (registry) ของวินโดว์ส (Windows) หรือครอน (Cron) ในระบบปฏิบัติการยูนิกซ์หรือไม่

3. ระบบมอนิเตอร์ไฟล์ล็อก (Log File Monitor : LFM) ตรวจสอบไฟล์ล็อกที่สร้างขึ้นมาโดยเซอร์วิสในเน็ตเวิร์ก ทำงานคล้ายกับระบบตรวจจับผู้บุกรุกทางเน็ตเวิร์ก ระบบนี้เฝ้าดูรูปแบบของไฟล์ล็อกที่เกิดขึ้นว่า ตรงกับพฤติกรรมการบุกรุกที่เคยเกิดขึ้นแล้วหรือไม่ ถ้าตรงก็สามารถตั้งข้อสงสัยได้ว่าเข้าข่ายการบุกรุกระบบ ตัวอย่างของระบบตรวจจับผู้บุกรุกแบบนี้ เช่น SWATCH โดยสร้างเซอร์วิสปลอมขึ้นมาหลอกว่าเป็นช่องโหว่ของระบบ ทำหน้าที่เป็นกับดักล่อให้ผู้บุกรุกเข้ามาติดกับ (สามารถดาว์นโหลดได้ที่ http://www.all.net/dtk)

2.1.4 พฤติกรรมทั่วไปของผู้บุกรุก

อันดับแรก ผู้บุกรุกพยายามหาข้อมูลของเครื่องเป้าหมายให้ได้มากที่สุดเท่าที่จะหาได้ ผู้บุกรุกอาจเข้าระบบโดยได้สิทธิ์ของผู้ใช้ปกติ แล้วเรียกใช้โปรแกรม เช่น whois (เป็นโปรแกรมที่ใช้หาข้อมูลว่ามีใครใช้งานอยู่ในระบบบ้าง) ดูโดเมนเนมของระบบ และค้นหาค่ากำหนดของเครื่องที่ทำงานอยู่ในเน็ตเวิร์ก เช่น ชื่อเครื่อง รุ่นของระบบปฏิบัติการ ชื่อผู้ใช้ทั้งหมดในระบบ

สำหรับภายในระบบ ผู้บุกรุกมักสแกนข้อมูลต่าง ๆ เพื่อหาช่องโหว่ เช่น เข้าไปตรวจสอบการใช้ CGI ในเว็บเพจ หรือใช้โปรแกรม ping หรือโปรแกรมที่ใช้โพรโตคอลคล้ายกัน เช่น SNMP ในการค้นหาว่ามีเครื่องใดที่ยังเปิดให้บริการ หรือสแกนพอร์ตที่มีเซอร์วิสที่ใช้โพรโตคอล TCP หรือ UDP เพื่อค้นหาว่ามีเซอร์วิสอะไรที่เปิดให้บริการบ้าง ซึ่งข้อมูลเหล่านี้ใช้สำหรับการบุกรุกระบบ

หลังจากนั้น เมื่อผู้บุกรุกพยายามล้ำเส้นเข้ามาในเครื่องเป้าหมาย โดยอาจใช้ช่องโหว่ในสคริปต์ซีจีไอ (CGI script) แล้วส่งคำสั่งหรือให้เรียกโปรแกรมใด ๆ ส่งค่าไปเป็นอินพุตโดยผ่านทางคำสั่งเชลล์ หรือผู้บุกรุกพยายามหารหัสผ่านที่เดาได้ง่าย ๆ หรือการเข้าใช้งานระบบโดยล็อกอินที่ไม่มีรหัสผ่าน เมื่อผู้บุกรุกสามารถเข้าไปเป็นผู้ใช้ทั่วไปแล้วก็สามารถหาช่องทางที่จะทำให้ได้สิทธิ์ของผู้ดูแลระบบ

ผู้บุกรุกได้ข้อมูลต้องการหรือได้ใช้ทรัพยากรใด ๆ แล้ว สิ่งที่ทำต่อไปคือพยายามกลบเกลื่อนหลักฐานทั้งหมดในการบุกรุก หรือสร้างช่องทางให้สามารถกลับไปใช้ระบบนั้น ๆ อีกโดยการสร้างประตูหลัง (Back door) หรือทำความเสียหายให้แก่ระบบนั้นโดยการวางโปรแกรมที่เป็นโทรจัน (Trojan) ในระบบ ในกรณีนี้สามารถใช้ระบบตรวจจับผู้บุกรุกแบบตรวจสอบความถูกต้องของข้อมูลในระบบ เพื่อตรวจสอบว่ามีการเปลี่ยนแปลงใด ๆ ในไฟล์หรือองค์ประกอบอื่นในระบบหรือไม่ อีกพฤติกรรมหนึ่งของผู้บุกรุกคือบุกรุกระบบใดระบบหนึ่งได้ จะใช้เป็นช่องทางในการบุกรุกระบบอื่น ๆ ต่อไป

2.1.5 ประเภทของการบุกรุก

การบุกรุกเข้าสู่ระบบแบ่งออกเป็นประเภทหลัก ๆ 3 ประเภทดังนี้

1. การบุกรุกทางกายภาพ (Physical Intrusion) ผู้บุกรุกพยายามบุกรุกที่เครื่องโดยตรง อาจเข้ามาใช้สิทธิ์พิเศษจากการทำงานที่คอนโซล หรือถอดย้ายอุปกรณ์ เช่น ฮาร์ดดิสก์ ซึ่งอาจนำไปเขียนหรืออ่านภายหลัง หรือบายพาสไบออสได้

2. การบุกรุกทางระบบ (System Intrusion) ผู้บุกรุกเข้ามาในระบบ โดยปกติมักเป็นผู้ใช้ที่มีสิทธิ์ต่ำ ถ้าระบบไม่ได้ใส่แพตช์ (patch) ที่สามารถแก้บักของแอพพลิเคชันแล้ว จุดนี้ก็เป็นช่องโหว่ที่ทำให้ผู้ใช้คนนั้นสร้างสิทธิ์ของตัวเองให้มากขึ้น จนเทียบเท่าผู้ดูแลระบบได้ เนื่องจากแอพพลิเคชันที่ใช้งานเกือบทุกอย่างมีบักอยู่ ถ้ายังไม่สามารถทำให้บักนั้นหมดไปหรือลดลงไปได้ จุดนี้ก็เป็นช่องทางสำหรับการบุกรุกระบบ

3. การบุกรุกระยะไกล (Remote Intrusion) ผู้บุกรุกติดต่อผ่านทางเน็ตเวิร์ก มีหลายเทคนิคในการบุกรุกระบบแบบนี้ ปัจจุบันมีแอพพลิเคชันประเภทไฟร์วอลล์ (Firewall) ทำหน้าที่เป็นด่านแรกในการป้องกันการบุกรุกทางเน็ตเวิร์ก

2.1.6 ช่องโหว่ภายในระบบ

เมื่อผู้บุกรุกต้องการบุกรุกระบบสามารถทำได้โดยหาช่องโหว่ภายในระบบ แล้วพยายามเข้าสู่ระบบทางช่องโหว่นั้น ๆ ช่องโหว่ของระบบมีดังนี้

ข้อบกพร่องของโปรแกรม

เกิดจากบักที่อยู่ในโปรแกรมซึ่งทำงานในเครื่องเซิร์ฟเวอร์, เ ครื่องไคลเอนต์, บ นระบบปฏิบัติการ, หรือสแต็กของเน็ตเวิร์ก ช่องโหว่ของระบบซึ่งพบที่โปรแกรมจำแนกเป็นประเภทต่าง ๆ ดังนี้

- บัฟเฟอร์โอเวอร์โฟลว์ (Buffer Overflow) ช่องโหว่ที่พบในปัจจุบันเกิดจากบัฟเฟอร์โอเวอร์โฟลว์แทบทั้งสิ้น การเกิดบัฟเฟอร์โอเวอร์โฟลว์ เช่น สมมุติว่าโปรแกรมเมอร์เขียนโปรแกรมรับอินพุตเป็นชื่อผู้ใช้ และจองพื้นที่ 256 ตัวอักษรสำหรับเก็บอินพุตนี้ โปรแกรมเมอร์คาดไว้ว่าไม่มีผู้ใช้คนในที่มีชื่อยาวกว่านี้แน่ ส่วนในมุมมองของผู้บุกรุกนั้นจะพิจารณาว่าหากใส่อินพุตที่ยาวกว่า 256 ตัวอักษร แล้วตัวอักษรที่เกินมาจะถูกวางไว้ที่ส่วนใดในหน่วยความจำ ผู้บุกรุกจึงพยายามส่งตัวอักษรมากกว่า 256 ตัวอักษรติดกัน พร้อมกับแทรกโค้ดที่สามารถทำงานได้ไว้ในอินพุตนั้นด้วย ถ้าโปรแกรมเกิดแครชขึ้น ผู้บุกรุกสามารถนำมาใช้เป็นจุดที่เข้าไปบุกรุกระบบได้ ซึ่งผู้บุกรุกสามารถหาช่องโหว่นี้ได้หลายทาง เช่น จากซอร์สโค้ดของเซอร์วิสต่าง ๆ ที่มีแจกให้ในอินเทอร์เน็ต ผู้บุกรุกเพียงแค่ค้นหาโปรแกรมตัวที่มีช่องโหว่นี้ จากนั้นก็ศึกษาซอร์สโค้ดแอสเซมบลีในการค้นหาช่องโหว่และทดลองใส่ข้อมูลสุ่มเพื่อหาข้อบกพร่อง มีข้อสังเกตว่าปัญหานี้มักเกิดกับโปรแกรมที่เขียนด้วย C หรือ C++ และพบได้น้อยมากในโปรแกรมที่เขียนด้วยจาวา (JAVA) เนื่องจากจาวาไม่อนุญาตให้โปรแกรมเมอร์ไปเข้าถึงหน่วยความจำได้โดยตรง
- การใช้โปรแกรมหลายโปรแกรมทำงานร่วมกันทำให้เกิดสิ่งที่ไม่คาดคิดขึ้นในการเขียนโปรแกรม (Unexpected combinations) โปรแกรมเมอร์เขียนโดยใช้โค้ดหลาย ๆ ระดับสร้าง

โปรแกรมขึ้นมา โดยมีระดับระบบปฏิบัติการเป็นระดับล่างสุด ตัวอย่างช่องโหว่แบบนี้ที่สามารถเห็นได้คือโปรแกรมที่เขียนด้วยภาษาเพิร์ล ซึ่งสามารถส่งอินพุตไปยังโปรแกรมอื่นได้ เช่น " | mail < /etc/passwd " เมื่อโปรแกรมทำงานที่คำสั่งนี้ ทำให้ระบบส่งไฟล์ /etc/passwd ไปให้ผู้บุกรุกผ่านทางอีเมล์

- อินพุตที่ไม่สามารถควบคุมได้ (Unhandled input) โปรแกรมเมอร์โดยส่วนใหญ่พิจารณาถึงเฉพาะอินพุตที่ใส่อย่างถูกต้องเท่านั้น โดยไม่ได้คิดถึงการใส่อินพุตที่เป็นไปไม่ได้ด้วย นี่เป็นช่องโหว่อีกทางหนึ่งที่สามารถใช้ในการบุกรุกระบบได้
- สภาพที่มีการแข่งขัน (Race condition) ระบบปัจจุบันเป็นระบบแบบมัลติทาร์กิง (multitasking) และมัลติเธรด (multithread) คือในขณะใดขณะหนึ่งสามารถมีโปรแกรมมากกว่าหนึ่งโปรแกรมทำงานอยู่ได้ ซึ่งเป็นอันตรายต่อระบบถ้าสองโปรแกรมกำลังเข้าถึงข้อมูลเดียวกันในเวลาเดียวกัน กรณีนี้อาจทำให้ข้อมูลของโปรแกรมใดโปรแกรมหนึ่งไม่สามารถเขียนได้อย่างสมบูรณ์ เหตุการณ์นี้เกิดขึ้นน้อยมาก ผู้บุกรุกต้องใช้เวลานานสำหรับการบุกรุกด้วยช่องทางนี้

ข้อบกพร่องของการกำหนดค่าของระบบ

ข้อบกพร่องจากการกำหนดค่าของระบบเกิดได้จากหลายสาเหตุดังนี้

- การตั้งค่าโดยใช้ค่าเดิมที่ระบบกำหนดมาให้ (Default configure) โปรแกรมส่วนใหญ่ที่ลูกค้าซื้อมาได้กำหนดค่าการทำงานต่าง ๆ มาแล้ว และเป็นค่าที่ทำให้โปรแกรมใช้งานได้ง่าย ซึ่งการใช้งานง่ายนำไปสู่การง่ายต่อการถูกบุกรุกระบบด้วย
- เกิดจากผู้ดูแลระบบเกียจคร้าน ไม่ได้ใส่รหัสผ่านของรูต (root) หรือผู้ใช้ใดๆ ในระบบ ทำให้เป็นช่องโหว่ที่ผู้บุกรุกใช้บุกรุกเข้าระบบโดยง่าย
- โปรแกรมอาจมีช่องโหว่จากเซอร์วิสที่ทำงานอยู่ในระบบ ผู้ดูแลระบบควรปิดเซอร์วิสของระบบทุกตัวที่ไม่ได้ใช้งาน เพื่อหลีกเลี่ยงช่องโหว่ที่อาจเกิดขึ้นได้ในภายหลัง ในส่วนนี้มีโปรแกรมสำหรับตรวจสอบความปลอดภัย ซึ่งสามารถตรวจสอบและแจ้งเตือนผู้ดูแลระบบให้ไปแก้ไขได้
- เครื่องที่เชื่อถือกัน (Trust relationships) ผู้บุกรุกอาศัยช่องโหว่จากเครื่องที่ติดต่อกันแบบทรัสต์ โดยสามารถเข้าไปยังเครื่องอื่น ๆ ที่ยกเว้นการตรวจสอบสิทธิ์ของกันและกันได้ ตัวอย่างการบุกรุกทางช่องโหว่ตรงนี้คือ การบุกรุกโดยใช้ .rhost

ช่องโหว่ของรหัสผ่าน

ส่วนใหญ่เกิดจากผู้ใช้ส่วนใหญ่จะใช้ชื่อที่ผู้ใช้คุ้นเคย เช่นชื่อตัวเอง ชื่อเพื่อน หรือสัตว์เลี้ยง เป็นรหัสผ่าน ทำให้ผู้บุกรุกสามารถเดารหัสผ่านได้ง่าย

การบุกรุกจากเดารหัสผ่านจากคำในพจนานุกรม มักเป็นขั้นตอนที่ผู้บุกรุกทำหลังจากไม่สามารถเดารหัสผ่านได้ โดยลองใช้รหัสผ่านซึ่งได้จากการเข้ารหัสคำที่อยู่ในพจนานุกรม แล้วนำมาเปรียบเทียบ

กับรหัสผ่านที่เข้ารหัสในไฟล์ของระบบ ซึ่งผู้บุกรุกอาจใช้คำที่อยู่ในฐานข้อมูลพจนานุกรมคำศัพท์ภาษาอังกฤษ หรือภาษาต่างประเทศอื่น ๆ ก็ได้

การบุกรุกโดยพลการ (Brute Force Attack) เป็นอีกวิธีหนึ่งที่ผู้บุกรุกใช้ในการเดารหัสผ่าน ผู้บุกรุกเดารหัสที่เป็นไปได้ที่เกิดขึ้นจากการสร้างรหัสผ่าน เช่นสมมุติว่ารหัสผ่านที่เป็นไปได้ของระบบเป็นตัวอักษรภาษาอังกฤษพิมพ์เล็กจำนวน 4 ตัว ผู้บุกรุกก็พยายามล็อกอินเข้าสู่ระบบโดยใช้รหัสผ่านที่เป็นไปได้ทั้งหมดจากการผสมคำ ในกรณีนี้ รหัสผ่านที่เป็นไปได้คือ 26x26x26x26 ตัว ซึ่งถ้าตั้งรหัสผ่านมีการผสมกันระหว่าง ตัวอักษรทั้งตัวเล็ก ตัวใหญ่ ตัวเลขและเครื่องหมายต่างๆ จะทำให้คำที่เป็นไปได้ทั้งหมดมีจำนวนมากขึ้น ดังนั้นหากผู้บุกรุกใช้วิธีนี้ในการเดารหัสผ่านก็จะใช้เวลานานยิ่งขึ้น

นอกจากนี้ผู้บุกรุกสามารถได้รหัสผ่านโดยวิธีต่อไปนี้

- การดักจับข้อมูลที่ไม่ได้เข้ารหัส (clear text sniffing) เซอร์วิสที่รันบนโพรโตคอล TCP/IP เช่น telnet มีการส่งรหัสผ่านที่ไม่เข้ารหัส ซึ่งอาจมีการดักจับโดยใช้ตัววิเคราะห์โพรโตคอล (protocol analyzer) ระหว่างทางของการส่งแพ็กเก็ตผ่านไปในเน็ตเวิร์ก ผู้บุกรุกสามารถเอารหัสผ่านที่ได้ไปล็อกอินเข้าระบบในภายหลัง
- การดักจับข้อมูลเข้ารหัส (Encrypt sniffing) ถึงแม้ว่ารหัสผ่านถูกเข้ารหัสไว้ ผู้บุกรุกสามารถทราบรหัสผ่านเหล่านั้นได้โดยนำรหัสผ่านจากคำในพจนานุกรม หรือจากการเดาคำไปเข้ารหัสเพื่อมาเปรียบเทียบกับรหัสผ่านที่ถูกเข้ารหัสไว้ (Brute force) หากผู้บุกรุกสามารถทราบรหัสผ่านเข้าสู่ระบบแล้ว ผู้นั้นก็เหมือนกับผู้ใช้ทั่วๆ ไป โดยที่ไม่อาจทราบได้เลยว่าผู้ที่ล็อกอินเข้ามานั้นเป็นผู้ที่มีสิทธิ์คนนั้นจริงๆ หรือไม่
- การบุกรุกโดยการส่งข้อมูลซ้ำ (Replay attack) ผู้บุกรุกไม่จำเป็นต้องถอดรหัสรหัสผ่านเพียงแต่ดักจับแพ็กเก็ตนั้น และสร้างโปรแกรมที่สามารถส่งแพ็กเก็ตของรหัสผ่านที่เข้ารหัสของผู้ที่มีสิทธิ์ในการใช้งานที่ดักจับได้ไว้ก่อนหน้านั้น แล้วส่งแพ็กเก็ตนั้นอีกครั้งไปยังเซิร์ฟเวอร์ขณะที่กำลังตรวจสอบสิทธิ์ ทำให้การติดต่อครั้งนั้นสำเร็จด้วยโดยใช้สิทธิ์ของผู้ใช้คนอื่น
- การขโมยไฟล์รหัสผ่าน (Password file stealing) ในระบบของเครื่องเซิร์ฟเวอร์ทั่วๆ ไปจะเก็บฐานข้อมูลรหัสผ่านของผู้ใช้ให้อยู่ในไฟล์ ซึ่งในระบบปฏิบัติการลีนุกซ์อยู่ที่ไฟล์ /etc/password หรือสำหรับระบบปฏิบัติการ WindowsNT เก็บอยู่ในไฟล์ที่ชื่อว่า SAM เมื่อผู้บุกรุกกระทำการใด ๆ ก็ตามทำให้ได้ไฟล์รหัสผ่านเหล่านี้แล้ว ผู้บุกรุกสามารถนำไฟล์นี้ไปถอดรหัสโดยให้โปรแกรมถอดรหัส (Crack) หารหัสผ่านที่ใช้เข้าสู่ระบบ
- การเฝ้าสังเกต (Observation) ปัญหาพื้นฐานของระบบการรักษาความปลอดภัย คือการขโมยใช้รหัสผ่าน หากผู้ใช้ในระบบมีการกำหนดรหัสผ่านของตนให้เป็นรหัสผ่านที่ยากต่อการเดา ปัญหาที่จะเกิดกับผู้ใช้ผู้นั้นคือ ผู้ใช้ต้องจำรหัสผ่านที่ตัวเองตั้งขึ้นมาด้วย ผู้ใช้บางคนอาจเผอเรอเขียนรหัสผ่านไว้บนกระดาษแล้วทิ้งไว้ ทำให้ผู้บุกรุกได้เอกสารที่มีรหัสผ่านนั้นไปได้ อีกวิธีหนึ่งคือถามรหัสผ่านผู้ที่รู้โดยใช้วิธีหลอกถามรหัสผ่านจากผู้ที่มีสิทธิ์จริง ๆ โดยอ้างเหตุผลต่าง ๆ

2.1.7 ช่องทางพื้นฐานสำหรับการบุกรุกระบบคอมพิวเตอร์

สคริปต์ซีจีไอ

เป็นที่รู้จักกันดีว่าการใช้งานสคริปต์ซีจีไอไม่ปลอดภัย เนื่องจากผู้บุกรุกสามารถสอดแทรกโปรแกรมแปลกปลอมผ่านเข้าไปกับฟิลด์ที่รับอินพุตให้ทำงานจากเชลล์ได้ ซึ่งโปรแกรมแปลกปลอมนั้นสามารถซ่อนไว้โดยการกำหนดตัวแปรแทนได้ ช่องโหว่ที่เป็นที่รู้จักกันดีตัวหนึ่งคือ phf ซึ่งเป็นไลบรารีที่มากับ httpd ของ NCSA

การบุกรุกผ่านทาง เว็บเซิร์ฟเวอร์

เว็บเซิร์ฟเวอร์หลายตัวที่สามารถเขียนตัวเองได้ เช่น IIS มีจุดบกพร่องในการระบุชื่อของไฟล์สามารถเรียกได้โดยใช้ "../" (เป็นการย้อนกลับไปในไดเรกทอรีนอกหนึ่งชั้น) ทำให้สามารถเข้าไปเรียกใช้ได้ทุกไดเรกทอรีในระบบไฟล์ ช่องทางการบุกรุกอื่น เช่น ใช้บัฟเฟอร์โอเวอร์โฟลว์

การบุกรุกผ่านทางเว็บบราวเซอร์

บราวเซอร์ทั้งจากค่ายไมโครซอฟต์ (Microsoft) และค่ายเน็ตสเคป (Netscape) ต่างมีข้อบกพร่องเหมือนกัน ซึ่งพบได้จาก HTTP, HTML, Java script, Frame Java, ActiveX ดังนี้

- HTML เช่น ปัญหาจากบัฟเฟอร์โอเวอร์โฟลว์ของ MIME-type ในคำสั่ง EMBED ของ Netscape Communicator
- Java Script เป็นตัวที่สามารถสอดแทรกไฟล์หรือคำสั่งเข้าไป เนื่องจากมีฟังก์ชัน file upload ซึ่งสามารถแทรกไฟล์เอ็กซิคิวต์ (execute file) เข้าไปได้
- เฟรม (Frame) มักใช้ร่วมกับจาวาสคริปต์หรือจาวาสามารถทำให้เกิดช่องโหว่ได้ เช่น หากผู้บุกรุกสร้างลิงก์ไปที่ทรัสต์ไซต์ (Trust site) ที่ใช้เฟรมแล้วแก้ไข ให้แทนเฟรมด้วยเว็บเพจของผู้บุกรุก ทำให้เว็บเพจของผู้บุกรุกปรากฏเป็นส่วนหนึ่งของทรัสต์ไซต์นั้น
- ActiveX ทำงานโดยตรงกับเครื่องในรูปแบบเชื่อใจกัน (trust model) และเนื่องจากทำงานด้วยเนทีฟโค้ด (native code) ทำให้เป็นอันตรายต่อการรักษาความปลอดภัย เนื่องจากสามารถได้รับไวรัสโดยไม่ได้ตั้งใจจากโค้ดที่ได้มาจากผู้ขาย

การบุกรุกผ่านโพรโตคอล SMTP (Sendmail)

Sendmail เป็นโปรแกรมที่ใช้อย่างแพร่หลายและซับซ้อนโปรแกรมหนึ่ง ซึ่งมีข้อบกพร่องอยู่มากในประวัติศาสตร์มีผู้บุกรุกพบบักที่เกิดจากการใช้คำสั่ง DEBUG หรือฟีเจอร์ WIZ ที่ซ่อนไว้ เป็นตัวช่วยเจาะระบบผ่านทาง SMTP

การเดาหมายเลขซีเควนต์ของทีซีพี

การติดต่อสื่อสารระหว่างเครื่องในเน็ตเวิร์ก ทุกแพ็กเก็ตต้องมีหมายเลขระบุเรียกว่าซีเควนต์นัมเบอร์ (Sequence number) ในสแต็กรุ่นเก่าสามารถคาดเดาลำดับหมายเลขเหล่านี้ได้ ผู้บุกรุกอาจสร้าง

แพ็กเก็ตปลอมที่มีหมายเลขซีเควนที่คาดเดาได้ และปลอมแปลงการติดต่อไปยังเครื่องที่กำลังติดต่อกันอยู่นั้น ทำให้สามารถส่งแพ็กเก็ตแทรกเข้าไป โดยผู้ส่งไม่ต้องผ่านขั้นตอนการตรวจสอบสิทธิ์

การบุกรุกผ่านทางโดเมนเนมเซิร์ฟเวอร์ (DNS)

ช่องโหว่เกิดจากโดเมนเนมเซิร์ฟเวอร์ (Domain Name Server) มีการทำงานแบบรีเคอร์ซีฟ (recursive) โดยส่งคำถามไปยังโดเมนเนมเซิร์ฟเวอร์ที่อยู่ในลำดับขั้นที่สูงขึ้นไป ผู้บุกรุกอาจส่งคำร้องขอเข้าไปยังโดเมนเนมเซิร์ฟเวอร์ตัวแรก ซึ่งโดเมนเนมเซิร์ฟเวอร์ตัวนั้นจะส่งคำถามขึ้นไปยังโดเมนเนมเซิร์ฟเวอร์ในลำดับสูงขึ้นถัดไป ผู้บุกรุกสามารถปลอมที่อยู่ทางอินเทอร์เน็ตให้เป็นโดเมนเนมเซิร์ฟเวอร์ตัวที่ตอบสนองตัวแรก และส่งคำตอบผิด ๆ กลับมา ทำให้โปรแกรมที่เป็นผู้ถามในลำดับแรกได้รับคำตอบที่ผิด ๆ ไป ซึ่งอาจเกิดการติดต่อไปยังเครื่องที่ผู้บุกรุกจัดเตรียมไว้ก่อนแล้วได้

2.1.8 วิธีที่ผู้บุกรุกใช้สำรวจระบบ

Ping sweeps

การส่งคำสั่ง ping ไปยังเครื่องแบบสุ่ม เพื่อค้นหาว่ามีเครื่องใดเปิดให้บริการอยู่ การกระทำในทำนองนี้อาจใช้โพรโตคอลอื่น เช่น SNMP sweep

TCP scan

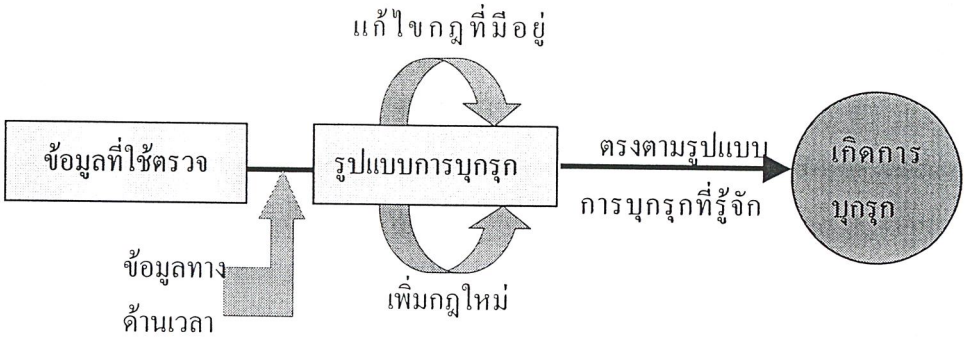
การเข้าไปตรวจดูพอร์ตทีซีพีว่ามีช่องทางใดที่เปิดให้บริการอยู่ และเป็นช่องที่ผู้บุกรุกสามารถใช้เจาะระบบเข้าไปได้ รูปแบบในการสแกนพอร์ตต่างๆ เป็นไปได้ทั้งการสแกนพอร์ตแบบต่อเนื่อง (เรียงตามหมายเลขพอร์ตที่เป็นไปได้) แบบสุ่ม และแบบกำหนดหมายเลขพอร์ตที่ต้องการสแกนไว้ล่วงหน้า

OS identification

กระทำโดยการส่งแพ็กเก็ต ICMP หรือ TCP ไปยังเครื่องเป้าหมาย ทำให้ผู้บุกรุกทราบชนิด และเวอร์ชันของ ระบบปฏิบัติการ และชนิดของเครื่อง

2.1.9 วิธีเปรียบเทียบพฤติกรรมผู้ใช้กับรูปแบบการบุกรุกที่รู้จัก (Misuse Intrusion Detection)

การแสดงถึงการบุกรุกระบบสามารถแสดงอยู่ในรูปแบบหรือสัญญาณซึ่งเคยมีการเกิดการบุกรุกแบบนั้นขึ้นแล้ว วิธีนี้ทำการตรวจจับการบุกรุกที่อยู่ในรูปแบบที่รู้จัก หลักสำคัญของการตรวจจับวิธีนี้คือต้องระบุถึงสัญญาณการเกิดการบุกรุกทั้งหมดที่เป็นไปได้

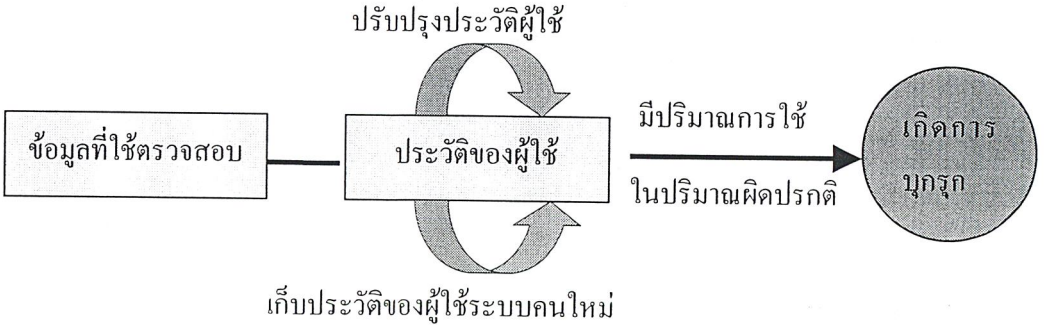


รูปที่ 2-3 การทำงานของการตรวจจับผู้บุกรุกโดยวิธีเปรียบเทียบพฤติกรรมผู้ใช้กับรูปแบบการบุกรุกที่รู้จัก

จากรูปที่ 2-3 ข้อมูลที่ใช้ตรวจสอบถูกนำมาเปรียบเทียบกับกฎ(Rules) ที่มีอยู่ ซึ่งมีการนำข้อมูลทางเวลาเข้ามาพิจารณาด้วย การตรวจจับการบุกรุกโดยวิธีนี้สามารถมีการแก้ไขกฎหรือเพิ่มกฎได้ ในระบบที่เป็นปัญญาประดิษฐ์ ระบบอาจทำการแก้ไขกฎหรือเพิ่มกฎได้ด้วยตัวเอง

2.1.10 วิธีตรวจสอบการใช้งานทรัพยากรระบบที่ผิดปรกติ (Anomaly Intrusion Detection)

วิธีตรวจจับการบุกรุกโดยตรวจสอบการใช้งานทรัพยากรระบบที่ผิดปรกตินี้ตั้งอยู่บนสมมติฐานที่ว่าการกระทำใด ๆ ที่เป็นการบุกรุกจะต้องมีการใช้งานทรัพยากรระบบอย่างผิดปรกติ หมายความว่าในการตรวจจับวิธีนี้ต้องมีการเก็บพฤติกรรมปรกติของผู้ใช้ระบบไว้ เพื่อเปรียบเทียบว่าพฤติกรรมใดเป็นพฤติกรรมที่ผิดปรกติ การเปรียบเทียบใช้สถิติศาสตร์เข้าช่วย ดังแสดงในรูปที่ 2-4



รูปที่ 2-4 การทำงานของการตรวจจับผู้บุกรุกโดยวิธีตรวจสอบการใช้งานระบบที่ผิดปรกติ

ข้อมูลที่ใช้ตรวจสอบ (Audit data) ซึ่งเก็บบันทึกโดยระบบจะถูกนำมาปรับปรุงไฟล์ประวัติ (Profile) ของผู้ใช้ พร้อมกันนั้นนำข้อมูลนี้มาเปรียบเทียบกับสถิติการใช้งานของผู้ใช้แต่ละคน หากพบว่ามีค่าผิดไปจากค่าปรกติก็ระบุว่าอยู่ในสถานะที่มีการบุกรุกเกิดขึ้น และถ้ามีการตรวจสอบโดยใช้ข้อมูลใหม่ ๆ ก็สามารถเพิ่มในไฟล์ประวัติได้

ปัญหาของการตรวจจับโดยวิธีนี้คือ

- อาจมีพฤติกรรมการใช้งานทรัพยากรระบบของผู้ใช้ที่ผิดปรกติเกิดขึ้นแต่ไม่ได้เป็นการบุกรุกระบบ ทำให้ระบุสถานะผิดพลาด
- ตรวจไม่พบการบุกรุกระบบเนื่องจากการบุกรุกนั้นไม่ได้ใช้ทรัพยากรยากรระบบอย่างผิดปรกติ

และมีข้อเสียคือต้องมีการเก็บการข้อมูลการกระทำของผู้ใช้และต้องทำการอัพเดตไฟล์ประวัติของผู้ใช้บ่อย ๆ

2.1.11 การตอบสนองเมื่อตรวจพบการบุกรุกของระบบตรวจจับผู้บุกรุก

- แก้ไขการตั้งค่าของไฟร์วอลล์ โดยการตั้งค่าไม่ให้แพ็กเก็ตจากเครื่องของผู้บุกรุกผ่านเข้ามายังเครือข่ายภายในได้ แต่ไม่สามารถกันได้ตลอดเวลา เนื่องจากผู้บุกรุกสามารถเปลี่ยนแปลงที่อยู่ของเครื่องให้เป็นอะไรก็ได้
- การส่งเสียงเตือน โดยการกำหนดให้แสดงไฟล์ .WAV เช่น ให้ส่งเสียงว่ากำลังถูกบุกรุกเมื่อพบพฤติกรรมที่น่าสงสัย
- ส่งดาต้าแกรมของ SNMP ไปยังหน้าจอควบคุมเน็ตเวิร์ก เช่น HP OpenView, Tivoli, Cabletron Spectrum และอื่นๆ (SNMP Trap)
- ส่งล็อกไปเก็บที่ระบบ syslogd
- ส่งอีเมล์ไปแจ้งเตือนผู้ดูแลระบบ
- ส่งข้อความทางเพจเจอร์ไปยังผู้ดูแลระบบเมื่อพบปัญหา
- เก็บข้อมูลเช่น เวลา หมายเลข พอร์ต และที่อยู่ทางอินเทอร์เน็ตของทั้งเครื่องที่ใช้บุกรุก และเครื่องที่ถูกบุกรุก
- ให้เก็บแพ็กเก็ตข้อมูลที่วิ่งอยู่ในเน็ตเวิร์กขณะที่ตรวจพบผู้บุกรุก เพื่อเป็นหลักฐานในการตรวจสอบต่อไป
- เรียกโปรแกรมที่เหมาะสมให้จัดการกับเหตุการณ์ที่เผชิญอยู่
- ปิดการเชื่อมต่อ โดยให้สร้างแพ็กเก็ต TCP FIN เพื่อปิดการเชื่อมต่อของผู้บุกรุกในขณะนั้นทันที

2.1.12 แนวทางการปฏิบัติเมื่อมีการบุกรุกระบบ

- จัดตั้งกลุ่มที่รับผิดชอบในการจัดการได้ทันทีที่ทราบว่าถูกบุกรุก
- กำหนดแนวทางในการรับมือ เช่น ควรให้ความสำคัญกับการจัดการให้เน็ตเวิร์กทำงานได้ปกติในสภาพเดิมเร็วที่สุด หรือ ให้ความสำคัญในการหาตัวผู้บุกรุก ผู้ดูแลระบบสามารถถอดสายเน็ตเวิร์กโดยทันทีหรือไม่หากพบว่ามีผู้บุกรุกกำลังทำความเสียหายกับระบบอยู่ เมื่อพบผู้บุกรุกแล้วจะคอยดูพฤติกรรมต่อไปหรือจะจัดการทันที ให้ลองคิดปัญหาที่เกิดขึ้น

แล้วหาแนวทางการดำเนินการก่อนที่จะเกิดจริง เพราะเมื่อเกิดเหตุการณ์ขึ้นแล้วจะไม่มีเวลาคิด

- ให้กำหนดแนวทางในการแจ้งปัญหา เช่น หากเกิดปัญหาแล้ว จะแจ้งผู้บังคับบัญชาในลำดับสูงขึ้นไปก่อน หรือ แจ้งหน่วยงานที่เกี่ยวข้องเลย ต้องแจ้งเหตุการณ์การบุกรุกที่เกิดขึ้นกับหน่วยงานที่คอยให้คำปรึกษาและช่วยเหลือใด (ในอเมริกา มีหลายหน่วยงานเช่น FIRST หรือ CERT) ต้องแจ้งตำรวจหรือไม่ จะแจ้งเหตุการณ์นี้กับหุ้นส่วนทางธุรกิจหรือไม่ จะปิดข่าวนี้กับหนังสือพิมพ์หรือไม่
- ให้จัดระบบและขั้นตอนในการดำเนินงานในการเก็บล็อก วิเคราะห์ และเฝ้าติดตามข้อมูลทันที ประเด็นสำคัญในการหาร่องรอยของผู้บุกรุกคือต้องมีการเก็บล็อกอย่างเพียงพอสำหรับการวิเคราะห์
- ให้คำแนะนำกับผู้ใช้ทุกคนให้ทราบเกี่ยวกับการป้องกันการบุกรุก

2.1.13 เครื่องมืออื่นๆในการตรวจจับผู้บุกรุก

- ไฟร์วอลล์ คนส่วนใหญ่รู้จักไฟร์วอลล์ว่าเป็นด่านแรกที่สามารถป้องกันผู้บุกรุกทางเน็ตเวิร์ก แต่ถ้าผู้บุกรุกสามารถผ่านไฟร์วอลล์ได้แล้ว ก็สามารถกระทำการใด ๆ ภายในเน็ตเวิร์กหลังไฟร์วอลล์ได้อย่างอิสระ ดังนั้นการกำหนดค่าการทำงานของไฟร์วอลล์จึงต้องกระทำให้ดี และควรจะใช้ระบบตรวจจับผู้บุกรุกระบบคอมพิวเตอร์ช่วยในการตรวจสอบด้วย
- ระบบตรวจสอบสิทธิ์ (Authentication) ควรตรวจสอบว่ามีผู้ใดที่ใช้งานอยู่บ้าง ต้องมีนโยบายในการกำหนดรหัสผ่านที่ดี และตั้งรหัสผ่านของผู้ใช้ทุกคน เพื่อป้องกันการเดารหัสผ่าน นอกจากนี้ควรหาระบบช่วยจัดการการตรวจสอบสิทธิ์ของผู้ใช้มาทำงานร่วมด้วย เช่น ใช้ RADIOUS/TACACS ที่มีอยู่ใน WinNT และ ยูนิกซ์ ในระบบ dial up หรือการรวมเอา Kerberos ใน Windows 2000 ระบบช่วยการตรวจสอบสิทธิ์ผู้ใช้ช่วยแก้ปัญหาการส่งรหัสผ่านของบริการของโพรโตคอล Telnet, FTP, IMAP, POP ที่ไม่ได้เข้ารหัสผ่านไปในเครือข่าย
- เวอร์ชวลไพรเวตเน็กเวิร์ก (Virtual Private Networks : VPNs) สร้างการเชื่อมต่อที่ปลอดภัยในอินเทอร์เน็ต โดยมีหลาย ๆ เทคโนโลยีที่เลือกมาสร้างได้ เช่น PPTP เป็นการเข้ารหัส การเชื่อมต่อแบบ PPP บนโพรโตคอล TCP เหมือนกับสร้างเส้นทางการเชื่อมต่อที่เป็นท่อระหว่างเน็ตเวิร์กที่ติดต่อด้วย แล้วเข้ารหัสข้อมูลทุกอย่างที่ส่งหากันภายในท่อนั้น มาตรฐานใหม่ของโพรโตคอล IP ที่ใช้ในเทคโนโลยีนี้เรียกว่า Ipsec
- การเข้ารหัส เป็นเทคโนโลยีที่ได้รับความนิยมเพิ่ม มากขึ้นเรื่อยๆ สามารถเข้ารหัสข้อมูลที่ต้องการเก็บเป็นความลับได้ทุกอย่างเช่นอีเมล์หรือ ไฟล์ในระบบ

2.1.14 ตัวอย่างแอพพลิเคชันสำหรับตรวจจับผู้บุกรุกระบบ

- Network Flight Recorder (NFR) เป็นตัวตรวจจับแพ็กเก็ตที่ ใช้ไลบรารี libcap ของ tcpdump ซึ่งทำงานอยู่ในระบบยูนิกซ์ ข้อดีคือโปรแกรมนี้ได้แจกซอร์สโค้ด ผู้พัฒนาสามารถสอดแทรกโค้ดในการตรวจสอบเพิ่มเติมได้ด้วย สามารถศึกษาเพิ่มเติมได้ที่ http:// www.nfr.net/forum/publications/LISA-97.htm
- Bro เป็นระบบของเวิร์น แพ็กซัน (Vern Paxson) ใช้ libcap เป็นพื้นฐานด้วย หาข้อมูลเพิ่มได้ที่ http://ftp.ee.lbl.gov/paper/bro-usenix98-revised.ps.Z
- AAFID เป็นผลงานของโคสต์ (COAST) ที่มหาวิทยาลัยเปอร์ดู (Purdue)
- Argus ไม่ได้เป็นระบบตรวจจับผู้บุกรุก แต่เป็นระบบมอนิเตอร์แพ็กเก็ตและเก็บข้อมูลลงไฟล์ล็อก สามารถอ่านไฟล์ล็อกขึ้นมาวิเคราะห์หาผู้บุกรุกได้ ศึกษาเพิ่มเติมได้ที่ ftp://coast.cs.purdue.edu หรือที่ ftp://ftp.sei.cmu.edu/pub/argus-1.5

2.1.15 ตัวอย่างเครื่องมือที่ใช้ในการบุกรุกระบบ

- netcat ช่วยในการเขียนการติดต่อกันแบบโต้ตอบได้ บนโพรโตคอลต่างๆ ที่เป็นโพรโตคอลแบบเท็กซ์
- crack, Ntcrack, L0phtCrack และอื่นๆ เป็นโปรแกรมเก็บฐานข้อมูลของคำ และวิธีการในการเดารหัสผ่าน นอกจากนั้นก็ยังทำหน้าที่ในการส่งรหัสผ่านที่สุ่มเดาขึ้นมา ไปตรวจสอบที่เครื่องเซิร์ฟเวอร์ด้วย
- Sniffing utilities ใช้เฝ้าดูและตรวจจับแพ็กเก็ตในเน็ตเวิร์ก เช่น Gobbler, tcpdump หรือ ตัวที่เรียกว่า honest-to-god Network Associaters Sniffer Network Analyzer
- TCP port scanner ใช้สำหรับ สแกนพอร์ตทีซีพี
- Ping sweeper สำหรับการส่งสัญญาณ ping ไปยังเครื่องจำนวนมากๆ เพื่อดูว่าเครื่องเหล่านั้นทำงานอยู่หรือไม่
- Exploit pack เป็นชุดของโปรแกรมที่ใช้ช่องโหว่ของแอปพลิเคชันต่างๆ ในการเจาะระบบ
- Remote security auditors เช่น SATAN เป็นโปรแกรมที่ตรวจสอบช่องโหว่ของระบบบางส่วน รวมทั้งตรวจสอบเครื่องอื่นๆ ในเน็ตเวิร์กเดียวกันได้ จากนั้นจะแจ้งผลลัพธ์ให้ทราบ
- NAT เป็นเครื่องมือในการค้นหาข้อมูล NetBIOS/SMB จากวินโดว์สและเซิร์ฟเวอร์แซมบา (SAMBA)
- Scanner เป็นโปรแกรมที่คล้ายกับ SATAN, ISS, CyberCop ตรวจสอบเน็ตเวิร์กเพื่อค้นหาช่องโหว่

2.2 ลีนุกซ์ (Linux)

2.2.1 ประวัติความเป็นมา

ลีนุกซ์ถือกำเนิดขึ้นในประเทศฟินแลนด์เมื่อปี ค.ศ.1980 โดยนายลีนุส โทรวัสด์ส (Linus Trovalds) นักศึกษาภาควิชาวิทยาการคอมพิวเตอร์ ในมหาวิทยาลัยเฮลซิงกิ มีความคิดว่าระบบปฏิบัติการยูนิกซ์ (Unix) บนพีซีในขณะนั้นยังมีความสามารถไม่เพียงพอกับความต้องการของผู้ใช้ ดังนั้นจึงเริ่มสร้างระบบปฏิบัติการยูนิกซ์ของตนเองขึ้น

ลักษณะการพัฒนาลีนุกซ์ได้ใช้อินเทอร์เน็ตช่วยโดยอาศัยความร่วมมือของนักพัฒนาจากสถานที่ต่าง ๆ จึงสังเกตได้ว่าการพัฒนาของระบบปฏิบัติการลีนุกซ์เป็นไปอย่างรวดเร็ว

2.2.2 ลีนุกซ์คืออะไร

ลีนุกซ์เป็นระบบปฏิบัติการแบบ 32 บิตอย่างแท้จริง โดยมีลักษณะเป็นยูนิกซ์โคลน (Unix clone) สำหรับเครื่องพีซี และแจกจ่ายให้ใช้ฟรี สนับสนุนการใช้งานแบบหลายงาน (Multitasking) และแบบหลายผู้ใช้ (Multiuser) มีระบบเอ็กซ์วินโดว์ (X-windows) ซึ่งเป็นระบบการติดต่อกับผู้ใช้แบบกราฟฟิกที่ไม่ขึ้นกับระบบปฏิบัติการ (Operating system) หรือฮาร์ดแวร์ใด ๆ และมีมาตรฐานการสื่อสาร TCP/IP ที่ใช้เป็นมาตรฐานการสื่อสารของอินเทอร์เน็ตมาให้ในตัว

ลีนุกซ์มีความเข้ากันได้ (Compatible) กับมาตรฐานโพสซิก (POSIX) ซึ่งเป็นมาตรฐานอินเทอร์เฟซที่ระบบยูนิกซ์ส่วนใหญ่ต้องมี และมีรูปแบบบางส่วนที่คล้ายกับระบบปฏิบัติการยูนิกซ์จากค่ายเบิร์กเลย์ (Berkeley) และซิสเต็มวี (System V)

โดยทางเทคนิคแล้วลีนุกซ์เป็นเพียงเคอร์เนล (kernel) ของระบบปฏิบัติการ ซึ่งทำหน้าที่ในด้านการจัดสรรและบริหารโพรเซสงาน การจัดการไฟล์และอุปกรณ์อินพุตเอาต์พุต (I/O)

นอกจากนี้ในปัจจุบัน ลีนุกซ์ยังสามารถใช้งานบนแพลตฟอร์ม (platform) ต่าง ๆ ได้ ตัวอย่างเช่น DEC Alpha, Motorola Power-PC เป็นต้น

ข้อดีของลีนุกซ์อีกข้อคือลีนุกซ์มีทีมพัฒนาโปรแกรมอย่างต่อเนื่อง และไม่จำกัดจำนวน เนื่องจากส่วนใหญ่ติดต่อกันผ่านอินเทอร์เน็ต โดยนายลีนุส โทรวัลด์ส เป็นผู้รวบรวมโปรแกรมที่ผู้พัฒนาได้ร่วมกันพัฒนาขึ้นมาและแจกจ่ายให้ทดลองใช้เพื่อทดสอบหาข้อบกพร่อง

2.2.3 สาเหตุที่เลือกใช้ลีนุกซ์ในโครงงาน

สาเหตุที่เลือกให้ลีนุกซ์ในโครงงานนี้คือลีนุกซ์เป็นระบบปฏิบัติการที่ไม่ต้องซื้อ สามารถดาวน์โหลดจากอินเทอร์เน็ตโดยไม่ผิดกฎหมาย และมีผู้นิยมใช้มาก มีผู้นำลีนุกซ์ไปแก้ไขให้สามารถใช้งานได้บนตัวประมวลผลหลากหลาย นอกจากนี้ยังมีผู้พัฒนาแอพพลิเคชันสำหรับลีนุกซ์ออกมาเรื่อย ๆ

เนื่องจากลีนุกซ์เป็นระบบปฏิบัติการ 32 บิตแบบแท้จริง ทำให้สามารถดึงการทำงานของคอมพิวเตอร์ออกมาได้เต็มกำลัง การทำงานของลีนุกซ์จึงมีประสิทธิภาพ และลีนุกซ์มีคุณลักษณะของ

ยูนิกซ์เต็มรูปแบบ ทำให้เป็นระบบหลายผู้ใช้ และหลายงาน ซึ่งสนับสนุนโพรโตคอลแบบ TCP/IP, SLIP, PPP, UUCP และอื่น ๆ

สำหรับการใช้ประโยชน์ของลีนุกซ์ในด้านอื่น ๆ สามารถใช้ลีนุกซ์ในการศึกษาระบบปฏิบัติการยูนิกซ์ หรือหากต้องการใช้แอพพลิเคชันบนระบบปฏิบัติการดอส (DOS) หรือวินโดว์ส ลีนุกซ์ก็มีดอสอีมูเลเตอร์ (DOSEMU) และ วินโดว์อีมูเลเตอร์ (WINE) ให้ แต่ยังอยู่ในขั้นทดสอบ

ลีนุกซ์ได้เตรียมเครื่องมือพัฒนาโปรแกรมไว้อย่างครบครัน มีตั้งแต่แอพพลิเคชันมาตรฐาน คือ คอมไพเลอร์ภาษาซีและซีพลัสพลัส คอมไพเลอร์ของ GNU และหากต้องการพัฒนาระบบบนเอ็กซ์ (X) ลีนุกซ์ก็มี TCL/TK ไว้ให้

คอมไพเลอร์ภาษาอื่น ๆ ที่ลีนุกซ์มีให้ เช่น เพิร์ล (Perl), สมอลล์ทอล์ก (Smalltalk), พาสคาล (Pascal), ลิสป์ (Lisp) เป็นต้น หรือถ้าต้องการเขียนโปรแกรมที่เกี่ยวข้องกับดาตาเบส ลีนุกซ์มีโปรแกรมแบบเอ็กซ์เบส (X-Base) หรือฟ็อกซ์โปร (FoxPro) ไว้ให้เช่นกัน และล่าสุด ลีนุกซ์ได้เพิ่มจาวาคอมไพเลอร์ให้สำหรับผู้ที่ต้องการเขียนแอพเพลตจาวาสำหรับการทำงานบนอินเทอร์เน็ตด้วย

2.2.4 ตำแหน่งของไฟล์ที่ใช้ในงานเขียนโปรแกรมบนลีนุกซ์

ตำแหน่งของโปรแกรม

โปรแกรมเก็บไว้ในไดเรกทอรีตามจุดประสงค์ของการใช้งาน เช่น

/usr/bin	แอพลิเคชันของระบบที่ใช้งานทั่วไป
/usr/local/bin	ส่วนโปรแกรมที่เพิ่มเข้ามาโดยผู้ควบคุมระบบ ใช้สำหรับคอมพิวเตอร์บางเครื่องโดยเฉพาะ หรือใช้งานสำหรับเครือข่ายโลคอลเน็ตเวิร์ก ผู้ควบคุมระบบมักเก็บโปรแกรมไว้ที่ไดเรกทอรี /usr/local ซึ่งมีประโยชน์ในกรณีที่ต้องติดตั้งหรืออัพเกรดระบบ ไดเรกทอรี /usr/local จะไม่ถูกลบ
/usr/bin หรือ/usr/local/bin	สำหรับโปรแกรมไดรเวอร์ของคอมไพเลอร์ GNU gcc แต่ตำแหน่งของคอมไพเลอร์นี้ อาจเปลี่ยนไปได้ กำหนดเมื่อตอนคอมไพล์คอมไพเลอร์เอง (หรืออาจเปลี่ยนแปลงไปตามระบบก็ได้) ลีนุกซ์เก็บคอมไพเลอร์ของ GNU C/C++ หรือ header file ไว้ที่ /usr/lib/gcc-lib/i486-unknow-linux/2.7.2

นอกจากนี้ยังมีโปรแกรมพิเศษบางชนิด ที่มีไดเรกทอรีเป็นของตนเอง หนึ่งในจำพวกนี้ก็คือระบบเอ็กซ์วินโดว์ส (X Windows) ซึ่งปรกติเก็บไว้ที่ไดเรกทอรี /usr/X11 หรือ /usr/X11R6 สำหรับ Revision 6 หรือ /usr/openwin สำหรับระบบกราฟฟิกอินเทอร์เฟซ OpenWindows ของ Sun Solaris

ไฟล์ Header

การเขียนโปรแกรมภาษาซี จำเป็นต้องใช้เฮดเดอร์ไฟล์ ซึ่งเป็นไฟล์ที่มีการกำหนดค่าคงที่ หรือตัวแปรไว้แล้ว ทำให้สามารถเรียกค่าคงที่หรือตัวแปรนั้นมาใช้งานได้เลย

/usr/include	สำหรับภาษา C (และไดเรกทอรีย่อยภายใต้ไดเรกทอรีนี้) เช่น /usr/include/sys สำหรับระบบยูนิกซ์บางชนิด หรือ /usr/include/linux สำหรับลีนุกซ์

สำหรับการเขียนโปรแกรมในภาษาอื่นเก็บไว้ในไดเรกทอรีที่คอมไพเลอร์ค้นหาโดยอัตโนมัติ เช่น

/usr/include/g++ สำหรับ GNU C++

/usr/include/X11 สำหรับการเขียนโปรแกรมบน X Windows

เมื่อคอมไพล์ สามารถระบุตำแหน่งที่ไม่เป็นมาตรฐานของ header file ได้โดยใช้แฟลก –I เช่น

```
$ gcc -i/usr/include/X11 richie.c
```

คอมไพเลอร์ไปค้นหาที่ตำแหน่ง /usr/include/X11 (รวมทั้งในตำแหน่งมาตรฐานที่ได้กำหนดไว้แล้ว) นอกจากนี้ถ้าต้องการค้นหาเฮดเดอร์ไฟล์ที่มีหน้าที่หรือสถานะของการทำงานบางอย่าง สามารถใช้คำสั่ง grep ให้ค้นหาภายในเฮดเดอร์ไฟล์ที่ต้องการ เช่น ถ้าต้องการรู้ว่าโพรเซสเปิดไฟล์ได้มากที่สุดเท่าใด เข้าไปยังไดเรกทอรี /usr/include ใช้คำสั่งต่อไปนี้

```
$ grep  POSIX_OPEN_MAX   *.h
posix1_lim.h:# define   POSIX_OPEN_MAX  16
```

2.3 การบันทึกเหตุการณ์บนระบบ (Logging)

ลีนุกซ์มีความสามารถในการบันทึกเหตุการณ์ต่าง ๆ เพื่อให้ผู้ดูแลระบบทราบเมื่อมีเหตุการณ์ต่าง ๆ เกิดขึ้นกับระบบ อย่างเช่น เมื่อมีการล็อกอินเข้ามาใช้งาน หรือล็อกเอาต์ออกจากระบบ, เมื่อมีการรับหรือส่งอีเมล์ หรือปัญหาใหญ่ที่เกิดกับระบบอย่างเช่น เกิดความผิดพลาดในการอ่านฮาร์ดดิสก์ หรืออุปกรณ์ต่าง ๆ เกิดความเสียหายเป็นต้น

ลีนุกซ์บันทึกเหตุการณ์ต่าง ๆ เหล่านี้ไว้ในไฟล์ ซึ่งไฟล์เหล่านี้เรียกว่า “ไฟล์ล็อก” (log files) แต่ละไฟล์บันทึกเหตุการณ์แต่ละเหตุการณ์ต่างกันออกไป ไฟล์ล็อกบันทึกโดยเคอร์เนล (kernel) เช่น ไฟล์ utmp, wtmp และบันทึกโดยโปรแกรมต่าง ๆ เช่น โปรแกรม syslogd ทำการเขียนไฟล์ messages เป็นต้น

สิ่งที่ได้จากไฟล์ล็อกสามารถสรุปได้ดังนี้

1. ความผิดพลาด (error) ของระบบที่เกิดขึ้นว่ามีอะไรบ้าง เมื่อ ผู้ดูแลระบบสามารถตรวจสอบความผิดพลาดของระบบ เช่น ความผิดพลาดของโปรแกรมที่กำลังทำงานอยู่ ผู้ดูแลระบบสามารถทำการหาสาเหตุและทำการแก้ไขข้อบกพร่องที่เกิดขึ้นได้

2. สมรรถนะ (performance) ของระบบ อัตราการใช้งานของระบบ มีการส่งผ่านข้อมูลมากแค่ไหน ระบบปัจจุบันสามารถรองรับการใช้งานได้เพียงพอหรือไม่ ช่วยให้ผู้ดูแลระบบสามารถตัดสินใจในการอัพเกรด (upgrade) เครื่องให้บริการให้สามารถรองรับการทำงานตามความเหมาะสมได้

3. ความปลอดภัยของระบบ (security) ผู้ใช้งานแต่ละคนทำอะไรกับระบบบ้าง มีใครที่ไม่ได้รับอนุญาตเข้ามาใช้งานระบบหรือเปล่า ถ้ามีสามารถตรวจสอบได้ว่าเป็นใครมาจากไหนได้

2.3.1 ประเภทของไฟล์ล็อก

สามารถแบ่งประเภทของไฟล์ล็อกเป็นประเภทใหญ่ 2 ประเภท คือ

1. ไฟล์ล็อกที่เกี่ยวกับผู้ใช้งาน ได้แก่

/usr/adm/lastlog	บันทึกการล็อกอินครั้งสุดท้าย
/usr/adm/acct	บันทึกคำสั่งที่ถูกรันโดยผู้ใช้งานในระบบทุกคน
/usr/adm/sulog	บันทึกเมื่อมีการเรียกใช้คำสั่ง su
/etc/utmp	บันทึกเวลาที่ผู้ใช้งานเริ่มล็อกออน (log on) เข้าสู่ระบบ
/usr/acm/wtmp	บันทึกเวลาของผู้ใช้งานที่เคย login และ logout จากระบบ
/usr/adm/xferlog	บันทึกเมื่อมีการ FTP เข้ามาในระบบ
$HOME/.history	บันทึกเมื่อมีการป้อนคำสั่งของผู้ใช้งานแต่ละคน

2. ไฟล์ล็อกที่เกี่ยวกับระบบ (system) และแอพพลิเคชัน (application) ได้แก่

/etc/syslog.conf	ไฟล์สำหรับตั้งค่า syslogd
/usr/adm/messages	บันทึกไฟล์ที่ได้ทำการตั้งค่าใน /etc/syslog.conf ซึ่งส่วนใหญ่บันทึกความผิดพลาดที่เกิดจากผู้ใช้งาน

2.3.2 ล็อกของระบบ

ไฟล์ล็อกที่ใช้ในโครงงานนี้มี 4 ไฟล์ คือ

1. SULOG

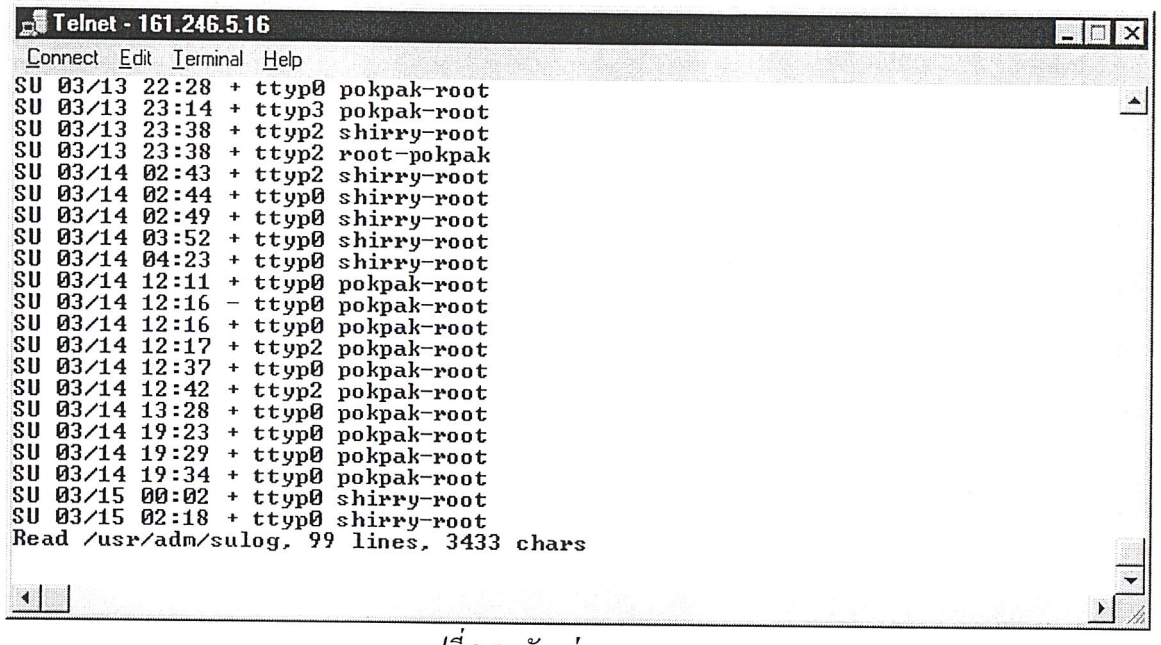
ในระบบลีนุกซ์ การใช้คำสั่ง telnet เข้ามาแล้วล็อกอินเป็น root ในระบบเลยนั้น เป็นสิ่งที่ไม่ปลอดภัยมาก เพราะอาจมีการดักจับรหัสผ่าน ไปได้ ดังนั้นจึงควรล็อกอินเป็น ผู้ใช้ ธรรมดาก่อน แล้วจึงเรียกใช้คำสั่ง su (คำสั่ง su ใช้เมื่อต้องการเปลี่ยนสิทธิ์เป็นผู้ใช้คนอื่น) เมื่อผู้ใช้งานคนใดคนหนึ่ง เรียกใช้คำสั่ง su ระบบมีการบันทึกลงในไฟล์ /usr/adm/sulog

ในไฟล์ SULOG มีการแสดงผลในรูปแบบดังนี้

SU DATE TIME RESULT PORT USER-NEWUSER

ซึ่ง

DATE	คือ เดือน และวัน ที่ทำการเรียกใช้คำสั่ง SU ซึ่งแสดงผลในรูปแบบ mm/dd ซึ่ง mm เป็นตัวเลขแสดงเดือน และ dd เป็นตัวเลขแสดงวัน
TIME	คือ เวลาที่คำสั่ง SU ประมวลผล แสดงในรูปแบบ hh/mm ซึ่ง hh เป็นตัวเลขแสดงชั่วโมง และ mm เป็นตัวเลขแสดงนาที
RESULT	ผลลัพธ์เมื่อใช้คำสั่ง SU "+" แสดงว่า สำเร็จ "-" แสดงว่า ไม่สำเร็จ
PORT	เป็นชื่อของเทอร์มินอล ที่คำสั่ง SU ประมวลผล
USER	ผู้ใช้ที่ใช้คำสั่ง SU
NEWUSER	ผู้ใช้ที่เปลี่ยนแปลงไปหลังจากใช้คำสั่ง SU



รูปที่ 2-5 ตัวอย่าง SULOG

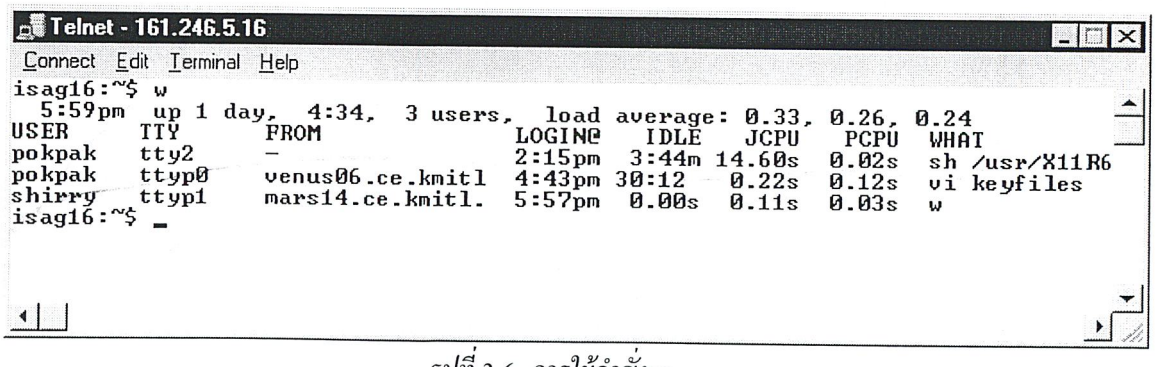
ผู้ดูแลระบบควรตรวจสอบ sulog ทุกวันเพราะอาจมีผู้ใช้งานบางคน ใช้คำสั่ง su และเดารหัสผ่านเพื่อที่เป็น root ถ้าผู้ใช้งานคนนั้นได้ใช้คำสั่ง su บ่อยมาก ๆ ควรทำการเตือน และตัดสิทธิ์ในการใช้งานระบบ

2. UTMP และ WTMP

เมื่อผู้ใช้งานคนใดก็ตามที่ล็อกอินเข้ามาในระบบ และล็อกเอาต์ออกจากระบบ มีการบันทึกในไฟล์ /usr/adm/utmp และ /usr/adm/wtmp

ไฟล์ /usr/adm/utmp ใช้บันทึกผู้ที่กำลังล็อกออน (log on)ในระบบอยู่ ข้อมูลในไฟล์นี้สามารถแสดงได้โดยใช้คำสั่ง finger, w, who, users, whodo

คำสั่ง w แสดงข้อมูลของผู้ใช้งานแต่ละคน ประกอบด้วย login name, terminal being used, remote host, login time, total cpu time, user time, system time, active process



รูปที่ 2-6 การใช้คำสั่ง w

ไฟล์ /usr/adm/wtmp บันทึกแต่ละครั้งที่มีผู้ใช้งานทำการล็อกอินเข้าสู่ระบบ และล็อกเอาต์ออกจากระบบ ข้อมูลในไฟล์นี้จะมีขนาดใหญ่ขึ้นเรื่อย ๆ เนื่องจากบันทึกทุก ๆ ครั้ง เมื่อมีการล็อกอินและ

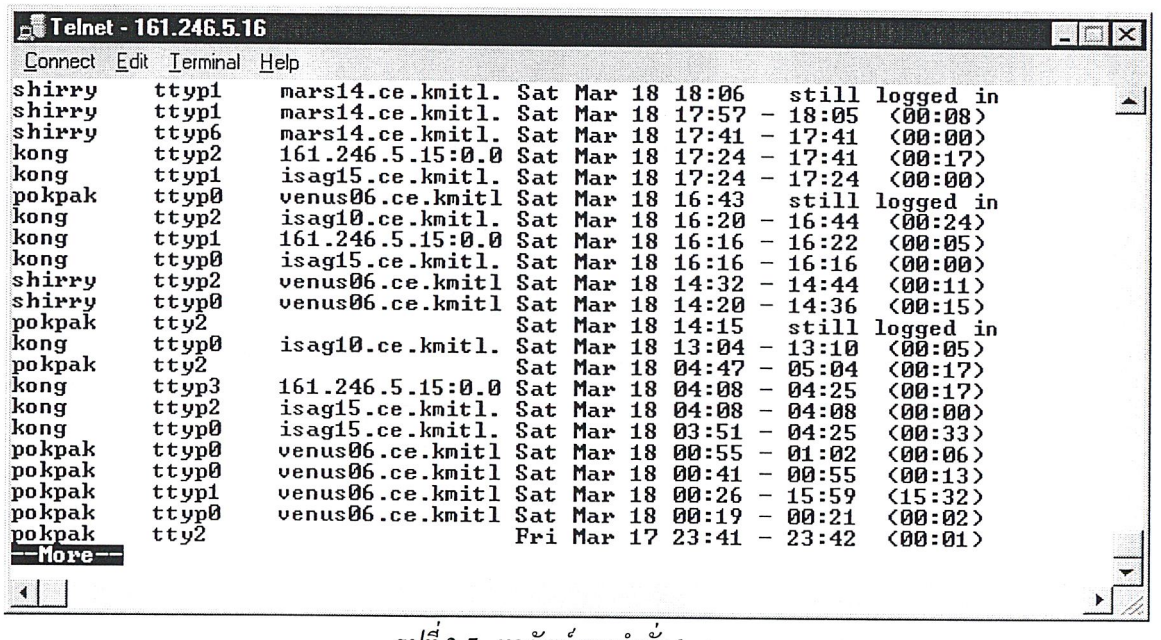
ล็อกเอาต์ ไม่เหมือนไฟล์ utmp คือเมื่อผู้ใช้ได้ล็อกเอาต์ออกไป ข้อมูลในไฟล์ก็ถูกลบออกไปด้วย ข้อมูลในไฟล์ wtmp นี้สามารถแสดงได้โดยใช้คำสั่ง last

ข้อมูลที่ได้จากการใช้คำสั่ง last นี้แสดง user name, terminal, remote host, เวลาที่เริ่มต้นใช้งาน และเวลาที่ออกจากระบบ หากเป็นข้อความ still logged in แสดงว่าผู้ใช้ยังล็อกอินอยู่ในระบบ และสุดท้ายแสดงเวลาทั้งหมดที่เข้ามาใช้งานในระบบ

ไฟล์ wtmp มีขนาดใหญ่ขึ้นเรื่อย ๆ ดังนั้นเพื่อป้องกันไม่ใช้ไฟล์มีขนาดใหญ่เกิดไปควรสำรองข้อมูลของไฟล์นี้ไว้โดยอาจทำเป็นต่อวันหรือต่อสัปดาห์ เมื่อต้องการตรวจสอบโดยใช้ไฟล์นี้ทำให้ง่ายยิ่งขึ้น และข้อมูลไม่มากเกินไป

ผู้ดูแลระบบควรใช้คำสั่ง last เพื่อดูว่าในวันหนึ่ง ๆ มีผู้ล็อกอินเข้ามาในระบบเป็นใครและมาจากไหนบ้าง ถ้าสังเกตเห็นถึงสิ่งผิดปกติที่เกิดขึ้นเช่นมีการล็อกอินเข้ามาใช้งานจากโฮสต์ (host) อื่นบ่อยครั้ง, มีการล็อกอินมาจากหลายที่ในเวลาเดียวกัน นั่นแสดงว่าอาจมีการบุกรุกเข้ามาใช้งานระบบของคุณแล้ว

คำสั่ง last แสดงผลดังรูปที่ 2-7



รูปที่ 2-7 ผลลัพธ์ของคำสั่ง last

3. SYSLOG

ในตอนแรกสร้างไว้สำหรับโปรแกรม sendmail โดย syslog บันทึกโพรเซสของอีเมล์ทั้งการรับ และการส่ง แต่ในปัจจุบันได้มีการเพิ่มฟังก์ชันให้มากขึ้น

syslog เป็นกลไกการจัดเก็บล็อกกิ้งแมสเซจ (logging messages) จากเคอร์เนล (kernel) และ แอพพลิเคชันที่รันอยู่บนระบบปฏิบัติการ ซึ่งโดยปกติเป็นมาตรฐานของระบบอยู่แล้ว และต้องมีการตั้งค่าว่าแมสเซจใดจะถูกเก็บลงไฟล์ หรือส่งต่อไปยังโฮสต์อื่น ซึ่งสามารถตั้งค่าได้ในไฟล์ /etc/syslog.conf แอพพลิเคชันบางตัวสามารถสร้างล็อกแมสเซจได้ แต่ละแมสเซจประกอบด้วย

- ชื่อของโปรแกรม (program name)
- แฟกซิลิตี้ (Facility) เป็นส่วนที่บอกว่าแมสเซจมาจากส่วนไหน เช่น ระบบเมล์ เป็นต้น

- ไพรออริตี้ (Priority) เป็นส่วนที่บอกระดับความสำคัญของแมสเซจนั้น เช่น เป็นการเตือน (warning)
- เท็กซ์แมสเซจ (Text message)

ตัวอย่างเช่น

```
login: Root LOGIN REFUSED on ttya
```

จากตัวอย่างเป็นความผิดพลาดจากโปรแกรมล็อกอิน (login) มีความหมายว่ามีบางคนพยายามล็อกอินเป็น root บนเทอร์มินอลที่ไม่ปลอดภัย

ในการตั้งค่า syslog นั้นทำได้โดยใช้ทั้งแฟกซิลิตี้และไพรออริตี้ควบคู่กันในการแบ่งแมสเซจต่างๆ ซึ่งมีรายละเอียดในตารางที่ 2-1 และ 2-2

แฟกซิลิตี้	คำอธิบาย
kern	แมสเซจถูกสร้างจากเคอร์เนล
user	เป็นแฟกซิลิตี้พื้นฐานสำหรับโปรแกรมใด ๆ
mail	ระบบเมล์
daemon	ซิสเต็มหรือเน็ตเวิร์กเดมอน เช่น in.ftpd
auth	ถูกใช้โดยระบบออโธไรเซชัน (authorization) หรือ โปรแกรมที่มีการถามชื่อผู้ใช้และรหัสผ่าน เช่น login, su, getty, ftpd เป็นต้น
lpr	ระบบการพิมพ์
news	สงวนไว้สำหรับระบบข่าว
uucp	สงวนไว้สำหรับระบบยูยูซีพี
cron	ใช้สำหรับระบบครอน (cron) และเอที (at)
local0..7	สงวนไว้สำหรับใช้งานโลคอล
mark	ใช้สำหรับการทำไทม์แสตมป์ (timestamp) ซึ่งส่งแมสเซจทุก ๆ 20 นาที
syslog	เกี่ยวกัน syslog
*	หมายถึงทุกแฟกซิลิตี้ ยกเว้น mask

ตารางที่ 2-1 รายละเอียดของแฟกซิลิตี้

ไพรออริตี้	คำอธิบาย
None	ไม่ต้องทำการส่งแมสเซจที่มาจากแฟกซิลิตี้ที่กำหนด
Debug	ใช้สำหรับการดีบัก (debug)
Info	เป็นแมสเซจที่ให้ข้อมูล (information)
Notice	แสดงสภาพที่ต้องจับตาดูเป็นพิเศษ
warning	คำเตือนต่าง ๆ
warn	เหมือนกับ warning (แต่เลิกใช้ไปแล้ว)
err	ความผิดพลาด (error) ต่าง ๆ
error	เหมือน err (แต่ได้เลิกใช้ไปแล้ว)
crit	สภาพที่มีปัญหารุนแรงอย่าง เช่น ปัญหาเกี่ยวกับฮาร์ดแวร์
alert	สภาพที่ต้องการการดูแลในทันที
emerg	สภาวะเร่งด่วนทุกชนิด เช่น ระบบเกิดความผิดพลาด

ตารางที่ 2-2 รายละเอียดของไพรออริตี้โดยเรียงลำดับตามความสำคัญจากน้อยไปหามาก

แอ็กชันเป็นตัวกำหนดว่าทำอย่างไร เมื่อมีแมสเซจที่ตรงกับแฟกซิลิตี้ และ ไพรออริตี้ที่กำหนดนั้น ๆ รายละเอียดของแอ็กชันแสดงไว้ในตารางที่ 2-3

แอ็กชัน	คำอธิบาย
/dev/console	ส่งข้อความออกทางอุปกรณ์
/var/adm/messages	เขียนแมสเซจลงในไฟล์ที่ชื่อ /var/adm/messages
@loghost	ส่งแมสเซจต่อไปยังโฮสต์อื่น
Fred,user1	ส่งแมสเซจไปยังผู้ใช้ชื่อ Fred และ user1
*	ส่งแมสเซจไปยังผู้ใช้ที่กำลังล็อกออน อยู่ทุกคน

ตารางที่ 2-3 ตัวอย่างของแอ็กชัน

ตัวอย่างของไฟล์ syslog.conf

```
*.err;kern.debug;auth.notice          /dev/console
*.crit;kern.none                      /var/adm/critical
```

ในแต่ละบรรทัดของไฟล์ประกอบด้วย 2 ส่วน คือ

- ตัวเลือก (selector) ระบุชนิดของแมสเซจที่ต้องการบันทึก
- การกระทำ (action) เป็นการบอกว่าให้กระทำกับแมสเซจอย่างไร เช่น บันทึกลงไฟล์ หรือส่งแมสเซจไปให้ผู้ใช้งาน

ในตัวเลือกแมสเซจแบ่งเป็น 2 ส่วนคือ แฟกซิลิตี้ และไพรออริตี้ ตัวอย่างเช่น kern.debug มีความหมายว่า เคอร์เนล (แฟกซิลิตี้) สร้างแมสเซจ เมื่อมีการดีบัก (ไพรออริตี้) ถ้ามีเครื่องหมายดอกจัน (*) วาง

ไว้ในตำแหน่งของแฟกซิลิตี้ หรือไพรออริตี้ มีความหมายว่า "ทั้งหมด หรือ ทุกอย่าง" เช่น .debug หมายความว่า แมสเซจดีบักทั้งหมด หรือ kern. หมายความว่าแมสเซจทั้งหมดที่ถูกสร้างโดยเคอร์เนล รายละเอียดดูในตารางที่ 2-4

สัญลักษณ์	คำอธิบาย
,	เครื่องหมายคอมม่า ใช้สำหรับการกำหนดหลายแฟกซิลิตี้ที่มีไพรออริตี้เดียวกัน เช่น mail, auth.info มีความหมายเหมือนกับ mail
;	เครื่องหมายเซมิโคลอน เป็นการกำหนดการเลือกแฟกซิลิตี้ และไพรออริตี้หลายอันเข้ากับแอ็กชันเดียว เช่น mail.info; kern.crit /dev/donsole
=	เครื่องหมายเท่ากับ หมายถึง เฉพาะไพรออริตี้นั้น ๆ เท่านั้น
!	เครื่องหมายอัศเจรีย์ หมายถึง ทุกไพรออริตี้ที่มีความสำคัญต่ำกว่า
#	เครื่องหมาย hash ใช้เป็นคอมเมนท์ (comment)
|	เครื่องหมายเส้นตรงแนวตั้ง หรือ ไปป์ (pipe) ใช้ในส่วนแอ็กชัน เพื่อสร้าง lilo ในการเชื่อม 2 คำสั่งเข้าด้วยกัน
@	เครื่องหมาย AT ใช้ในส่วนแอ็กชันเพื่อส่งต่อแมสเซจไปยัง loghost อื่น
*	เครื่องหมายดอกจัน หมายถึง ทุกไพรออริตี้หรือทุกแฟกซิลิตี้

ตารางที่ 2-4 สัญลักษณ์พิเศษที่ใช้ใน syslog.conf

ในส่วนของแอ็กชัน สามารถแบ่งได้เป็น 4 รูปแบบ และเลือกใช้งานในรูปแบบใดรูปแบบหนึ่ง

- บันทึกลงไฟล์ หรือดีไวซ์ (device) ซึ่งประกอบด้วยชื่อไฟล์หรือชื่อของดีไวซ์ และต้องนำหน้าด้วยเครื่องหมาย "/" (slash) เช่น /usr/adm/messages หรือ /dev/console
- ส่งข้อความให้ผู้ใช้งาน ซึ่งประกอบด้วย username เช่น pokpak ซึ่งสามารถที่ส่งข้อความให้ผู้ใช้งานหลาย ๆ คนโดยใช้เครื่องหมายคอมม่า เช่น pokpak,shirry
- ส่งข้อความถึงผู้ใช้งานทุกคน คือการใช้เครื่องหมายดอกจัน (*)
- ส่งข้อความไปยังโฮสต์อื่น คือ การใส่ชื่อโฮสต์ และต้องใช้เครื่องหมาย "@" นำหน้าชื่อโฮสต์นั้น เช่น @backup.ce.kmitl.ac.th

จากตัวอย่างของไฟล์ syslog.conf ข้างต้น สามารถทำความเข้าใจได้ดังนี้

1. *.err; kern.debug;auth.notice /dev/console

ข้อความผิดพลาดทั้งหมด, เคอร์เนลดีบักแมสเซจทั้งหมด และโนติซแมสเซจทั้งหมดที่สร้างโดยระบบออโธไรเซชัน ถูกส่งไปยังคอนโซล ถ้าคอนโซลเป็นเครื่องพิมพ์ก็เป็นการดีที่สามารถนำข้อผิดพลาดนั้น ๆ มาใช้อ้างอิงได้ภายหลัง

2. *.crit;kern.none /var/adm/critical

ทุกแฟกซิลิตี้ที่มีไพรออริตี้เป็น crit ยกเว้น แฟกซิลิตี้ kern ให้นำบันทึกในไฟล์ /usr/adm/critical

2.4 ครอนและครอนแท็ป (Cron and Crontab)

ครอนเป็นระบบจัดการเอ็กซิคิวต์คำสั่งตามเวลาที่กำหนดไว้ล่วงหน้าโดยตัวครอนเดมอน (crond) ถูกรันจากไฟล์ /etc/rc หรือ /ect/rc.local โดยตัวเดมอนนี้เมื่อเริ่มต้นทำงานจะอ่านค่าเริ่มต้นจากไฟล์ /var/spool/cron ซึ่งเป็นส่วนที่ผู้ใช้แต่ละคนสามารถสร้างตารางเวลาของแต่ละคนได้ ชื่อไฟล์มีชื่อเดียวกันกับชื่อผู้ใช้ ส่วนอีกไฟล์หนึ่งเป็นการกำหนดตารางเวลาของระบบคือไฟล์ /ect/crontab โดยผู้ที่จัดการกับไฟล์นี้ได้ต้องเป็น root เท่านั้น

Crontab เป็นไฟล์ที่ใช้กำหนดเวลาล่วงหน้าของระบบ อยู่ใน /etc/crontab คำสั่งที่รันจากครอนของระบบนี้จะมีสิทธิ์ของ root

ไฟล์ crontab มีข้อกำหนดการเขียนคือ บรรทัดที่ว่าง และบรรทัดที่ขึ้นต้นด้วย # ถือว่าเป็นคอมเมนต์ ในไฟล์ crontab สามารถกำหนดค่าตัวแปรในรูปแบบดังนี้

name = value

รูปแบบการกำหนดตารางการทำงาน

ฟิลด์	minute	hourday of mon	th	month	day of week	[user]	command
ช่วงเวลาที่ใช้	0-59	0-23	1-31	1-12	0-7		

ถ้าค่าในฟิลด์เวลาเป็น * หมายถึงทุกค่าที่เป็นไปได้ นอกจากนี้ยังสามารถกำหนดค่าเป็นช่วง โดยใช้เครื่องหมาย – ได้ เช่น 8-11 ก็หมายถึงเวลา 8,9,10,11 หรือกำหนดเป็นลิสท์ได้เช่น 1,2,5,9 หรือ 1-4,8-12 ก็ได้

นอกจากนี้สามารถกำหนดเป็นสเต็ปได้ เช่น สำหรับ hour กำหนดเป็น 0-23/2 หมายถึง 0,2,4,6,8,10,12,14,16,18,20,22 ซึ่งหมายถึงให้ทำคำสั่งนั้นทุก 2 ชั่วโมง อาจเขียนง่ายๆ ได้ว่า */2 และเครื่องหมาย % มีความหมายเท่ากับเครื่องหมายขึ้นบรรทัด (new line)

สำหรับฟิลด์ month และ day of week สามารถใช้ชื่อเป็นภาษาอังกฤษสามตัวแรก เช่น วันอาทิตย์ก็จะเป็น sun เดือนธันวาคมก็จะเป็น Dec โดยไม่สนใจว่าเป็นตัวพิมพ์เล็กหรือพิมพ์ใหญ่

ถ้าใช้เป็นตัวเลขในฟิลด์ day of week เลขแต่ละตัวมีความหมายดังนี้

เลข 0,7 หมายถึง วันอาทิตย์

เลข 1 หมายถึง วันจันทร์

เลข 2 หมายถึง วันอังคาร

เลข 3 หมายถึง วันพุธ

เลข 4 หมายถึง วันพฤหัส

เลข 5 หมายถึง วันศุกร์

เลข 6 หมายถึง วันเสาร์

ตัวอย่าง

30	4	1,15	*	5	- หมายถึงคำสั่งรันที่เวลา 4:30 ทุกวันที่ 1 และ 15 ของเดือนรวมถึงวันศุกร์ด้วย

*	*	*	*	*	- หมายถึงคำสั่งรันทุกๆ 1 นาทีของทุกวัน
23	0-23/2	*	*	*	- รันที่เวลา 23 นาทีหลังเที่ยงคืน แล้วรันทุกๆ สองชั่วโมง ทุกวัน
15	4	*	*	sun	- รันทุกวันอาทิตย์เวลา 4:15

2.5 ภาษาซีและภาษาซีพลัสพลัส (C and C++)

2.5.1 ภาษาซี

ภาษาซีได้ถูกคิดค้นขึ้นโดยนายเดนิส ริตชี (Denis Ritchie) ในปี ค.ศ. 1970 โดยใช้ระบบจัดการของยูนิกซ์ (UNIX) นับจากนั้นมาก็ได้รับความนิยมเพิ่มขึ้นจนถึงปัจจุบัน ภาษาซีสามารถติดต่อในระดับฮาร์ดแวร์ได้ดีกว่าภาษาระดับสูงเช่น เบสิก (Basic) ฟอร์แทรน (Fortran) ขณะเดียวกันก็มีคุณสมบัติของภาษาระดับสูงอยู่ด้วย ดังนั้นจึงได้จัดให้ภาษาซีเป็นภาษาระดับกลาง (middle-level language)

ภาษาซีเป็นภาษาคอมพิวเตอร์ชนิดคอมไพล์ (compiled language) ซึ่งมีคอมไพเลอร์ (compiler) คอมไพล์ (compile) คำสั่งทั้งหมดในซอร์สโค้ดเป็นภาษาเครื่อง (object code) ซึ่งเป็นภาษาที่คอมพิวเตอร์เข้าใจแล้วจึงทำตามคำสั่งต่างๆ ในโปรแกรม นอกจากนี้ภาษาซียังมีลักษณะเป็นภาษาโครงสร้าง (structured language) เหมือนกับปาสคาล

2.5.2 ภาษาซีพลัสพลัส

C++ (ซีพลัสพลัส) เป็นภาษาระดับสูง (High-level language) ซึ่งกำลังได้รับความนิยมอยู่ในปัจจุบัน เนื่องจาก C++ ได้รวบรวมเทคนคิต่าง ๆ ซึ่งมีการคิดค้นและสั่งสมต่อเนื่องกันมาในช่วงเวลาหลายปี ดังนั้น C++ จึงเป็นภาษาที่มีประสิทธิ์ภาพสูงสุดภาษาหนึ่งสำหรับการเขียนโปรแกรม

ผู้สร้างภาษา C++ คือ บจาร์น สโตรสรัป (Bjarne Stroustrup) ซึ่งทำงานอยู่ห้องปฏิบัติการเบล (Bell Laboratories) ประเทศสหรัฐอเมริกา ภาษานี้สร้างขึ้นประมาณปี ค.ศ.1980

C++ ประกอบด้วยทุกสิ่งที่มีในภาษา C และส่วนที่เพิ่มเติมเข้ามา ดังนั้นจึงสามารถเขียนโปรแกรมภาษา C ใน C++ ได้ แต่ทำกลับกันคือเขียนโปรแกรมภาษา C++ ใน C ไม่ได้ เป็นสาเหตุในมีผู้นิยมใช้ภาษา C++ มากขึ้นเป็นลำดับ ส่วนสำคัญที่เพิ่มเติมจากภาษา C ได้แก่ การเขียนโปรแกรมแบบออบเจ็กต์หรือ OOP (Object-Oriented Programming) ซึ่งเป็นวิธีการเขียนโปรแกรมที่สามารถใช้งานได้อย่างกว้างขวาง นิยมใช้ในการเขียนโปรแกรมขนาดใหญ่ที่มีการทำงานสลับซับซ้อน เพราะสะดวกและง่ายแก่การเขียน การตรวจสอบ การปรับปรุงและการนำมาใช้ใหม่

C++ ได้รับมาตรฐาน 4 มาตรฐานคือ ANSI (The American National Standards Institute - สถาบันมาตรฐานแห่งชาติอเมริกัน) มาตรฐาน ISO (International Organization fo Standardization - องค์การระหว่างประเทศเกี่ยวกับมาตรฐาน) มาตรฐาน BSI (The British Standards Institute - สถาบันมาตรฐานแห่งอังกฤษ) มาตรฐาน DIN (The German National Standards Organization - องค์การมาตรฐานแห่งชาติเยอรมัน) การกำหนดมาตรฐานมีส่วนดีทำให้โปรแกรมที่เขียนขึ้นสามารถนำไปใช้ในระบบ

คอมพิวเตอร์ที่แต่ต่างกันได้ทันทีโดยไม่ต้องปรับปรุงเปลี่ยนแปลงใด ๆ การกำหนดมาตรฐานอาจเป็นเครื่องบ่งชี้ให้ทราบว่า C++ เป็นภาษาที่ได้รับความนิยมมากภาษาหนึ่ง

2.5.3 องค์ประกอบของโปรแกรม

โปรแกรมมีองค์ประกอบหลักดังนี้คือ เฮดเดอร์ไฟล์ (header file), ชื่อฟังก์ชัน, ส่วนข้อกำหนดตัวแปร (declaration) และตัวโปรแกรม

```
#include  "stdio.h"          <-------  พรีโพรเซสเซอร์ไดเรกทีฟ
main()                       <-------  ชื่อฟังก์ชัน
{  int a;                    <-------  ส่วนข้อกำหนดตัวแปร
   a = 30;                   }
   printf("a = %d",a);       }  ตัวโปรแกรม
}
```

โปรแกรมที่ 2-1

1. บรรทัดที่เริ่มต้นด้วยเครื่องหมาย # (เครื่องหมาย number) เรียกว่า พรีโพรเซสเซอร์ไดเรกทีฟ (Preprocessor directive) คำที่อยู่จาก # เรียกว่า ไดเรกทีฟ (Directive) พรีโพรเซสเซอร์ไดเรกทีฟเป็นคำสั่งซึ่งสั่งให้คอมไพเลอร์ทำงานอย่างหนึ่งก่อนคอมไพล์โปรแกรม โดยในคอมไพเลอร์มีส่วนที่เรียกว่า พรีโพรเซสเซอร์ (Preprocessor) ทำหน้าที่ดำเนินการรับไดเรกทีฟที่มีอยู่ในโปรแกรม

ไดเรกทีฟ `#include "stdio.h"` หมายถึงการสั่งให้คอมไพเลอร์นำสิ่งที่อยู่ในไฟล์ที่กำหนดไว้รวมเข้ากับซอร์สโค้ด ในที่นี้คือไฟล์ stdio.h เรียกว่าเป็น เฮดเดอร์ไฟล์ (Header file) หรือ อินคลูดไฟล์ (Include file) เฮดเดอร์ไฟล์แบ่งเป็น 2 ประเภท คือ ประเภทที่มีให้มาพร้อมกับคอมไพเลอร์และประเภทที่ทำเขียนขึ้นเอง

เฮดเดอร์ไฟล์หรืออินคลูดไฟล์เป็นไฟล์ที่เก็บไลบรารีมาตรฐาน (standard library) ไลบรารีมาตรฐานถูกอ่านเข้ามารวม(include) ในโปรแกรมขณะที่ทำการคอมไพล์ ในโปรแกรมที่ 2-1 เฮดเดอร์ไฟล์คือ stdio.h ซึ่งเป็นไฟล์ที่เก็บไลบรารีมาตรฐานที่เกี่ยวข้องกับอินพุตและเอาต์พุต ซึ่งควรมีทุกโปรแกรม เฮดเดอร์ไฟล์เขียนด้วยอักษรตัวใหญ่หรือตัวเล็กก็ได้ แต่นิยมเขียนด้วยอักษรตัวเล็ก ในโปรแกรมหนึ่งๆ อาจมีเฮดเดอร์ไฟล์มากกว่า 1 ไฟล์ เฮดเดอร์ไฟล์มีส่วนขยายเป็น .h เสมอ

2. ชื่อฟังก์ชัน ต้องมีฟังก์ชัน main() เสมอ การทำงานเริ่มที่คำสั่งแรกในฟังก์ชัน main() ในโปรแกรมหนึ่งๆ อาจมีฟังก์ชันมากกว่า 1 ฟังก์ชันได้ แต่ฟังก์ชันแรกต้องเป็นฟังก์ชัน main() ฟังก์ชันต้องเริ่มต้นด้วยเครื่องหมาย { และปิดท้ายด้วยเครื่องหมาย } เสมอ

3. ส่วนข้อกำหนดตัวแปร ภายในฟังก์ชัน main() ต้องกำหนดชนิดข้อมูลให้กับตัวแปรก่อนที่อ้างถึงตัวแปรนั้นๆ การกำหนดชนิดข้อมูลดังกล่าวต้องกำหนดในส่วนข้อกำหนดตัวแปร

4. ส่วนตัวโปรแกรม ส่วนนี้อยู่ถัดจากส่วนข้อกำหนดตัวแปร ประกอบด้วยคำสั่งต่างๆ ต้องลงท้ายด้วยเครื่องหมาย ; เพื่อแสดงว่าจบคำสั่งหนึ่งๆ และมักนิยมเขียนด้วยอักษรตัวเล็ก คอมไพเลอร์แยก

ความแตกต่างของอักษรตัวใหญ่และตัวเล็ก (case sensitive) นั่นคือ BOY และ boy ถือว่าเป็นตัวแปรคนละตัว นอกจากนี้คอมไพเลอร์ไม่สนใจการขึ้นบรรทัดใหม่ หมายความว่าสามารถเขียนคำสั่งหลายๆ คำสั่งบนบรรทัดเดียวกันได้ โดยแต่ละคำสั่งต้องลงท้ายด้วยเครื่องหมาย ; ดังโปรแกรมที่ 2-2

```
#include "stdio.h"                    /* header file*/
main()                                /* function name */
{     int a;                          /* declaration */
      a = 30; printf("a = %d",a);
}
```

โปรแกรมที่ 2-2

เมื่อต้องการแสดงข้อความเพื่ออธิบายส่วนใดส่วนหนึ่งของโปรแกรมให้ปิดตัวและท้ายข้อความด้วยเครื่องหมาย /* และ */ ตามลำดับ หรือใช้ // ขึ้นต้นบรรทัด คอมไพเลอร์จะข้ามเฉพาะบรรทัดนั้นไป

ขั้นตอนการทำงานเริ่มโดยคอมไพล์คำสั่งทั้งหมดในโปรแกรมเป็นไฟล์ชนิด .OBJ ซึ่งเป็นไฟล์ที่พร้อมจะลิงก์กับไลบรารีมาตรฐานต่างๆ เพื่อทำเป็นไฟล์ชนิด .EXE

2.5.4 ชื่อ (Identifiers)

ตัวแปร ฟังก์ชัน ชื่อและอื่นๆ ในภาษานี้มีชื่อเฉพาะซึ่งกำหนดโดยผู้เขียนโปรแกรม การกำหนดชื่อดังกล่าวมีกฎเกณฑ์ดังนี้

1. ชื่อต้องขึ้นต้นด้วยตัวอักษรเสมอหรือเครื่องหมาย _
2. ไม่มีการเว้นวรรคภายในชื่อแต่อาจใช้เครื่องหมาย_ คั่นได้
3. ถัดจากอักษรตัวแรกอาจเป็นตัวเลข ตัวอักษรหรือเครื่องหมาย _
4. ความยาวของชื่อไม่จำกัดแค่เฉพาะตัวอักษร 32 ตัวแรกเท่านั้นที่ใช้กำหนดชื่อ
5. ชื่อที่เขียนด้วยอักษรตัวใหญ่และอักษรตัวเล็ก ถือว่าเป็นคนละตัวกันเช่น TOTAL_SALE, Total_sale และ total_sale ถือเป็นชื่อคนละตัว
6. ชื่อที่กำหนดขึ้นต้องไม่ซ้ำกับคำสงวนในเทอร์โบซี (reserved words)

2.5.5 คำสงวนของภาษาซีและซีพลัสพลัส

auto	double	int	struct
break	else	long	switch
case	enum	register	typedef
char	extern	return	union
const	float	short	unsigned
continue	for	signed	void
default	goto	sizeof	volatile
do	if	static	while

asm	_cs	_ds	_es
_ss	cdecl	far	huge
interrupt	near	pascal	

ในการเขียนโปรแกรมคอมพิวเตอร์ด้วยภาษาซีและซีพลัสพลัสต้องกำหนดชนิดข้อมูลให้กับตัวแปรก่อนที่นำตัวแปรนั้นมาใช้เหมือนภาษาปาสคาล ข้อมูลต่างชนิดกันมีคุณสมบัติและการใช้งานต่างกัน

2.5.6 ชนิดข้อมูล

ชนิดข้อมูลมีทั้งหมด 4 ชนิดดังนี้

1. ข้อมูลชนิดจำนวนเต็ม (integer type)
2. ข้อมูลชนิดทศนิยม (floating point type)
3. ข้อมูลชนิดทศนิยมละเอียดสองเท่า (double precision floating point)
4. ข้อมูลชนิดตัวอักษร (character type)
5. ข้อมูลชนิดไม่มีค่า (valueless type)

จะเห็นว่าไม่มีข้อมูลชนิดสตริงก์หรือข้อความต่างๆ ข้อมูลชนิดสตริงก์ประกอบด้วยตัวอักษรหลายๆ ตัวเรียงติดกันเป็นข้อความ ดังนั้นจึงใช้อาร์เรย์ (array) ในการเก็บข้อมูลชนิดสตริงก์ โดยขนาดของอาร์เรย์ต้องมากกว่าความยาวของสตริงก์หรือข้อความอย่างน้อย 1 ตัวอักษร

อาร์เรย์ที่เก็บข้อมูลที่สตริงก์ปิดท้ายด้วยสตริงก์ว่าง (null string) ซึ่งใช้ \0 แทนสตริงก์ว่าง

2.5.7 การกำหนดตัวแปรพอยน์เตอร์

การกำหนดตัวแปรพอยน์เตอร์ คล้ายกับการกำหนดตัวแปรชนิดต่างๆ เพียงแต่ต้องมีเครื่องหมาย * หน้าชื่อตัวแปร ตัวแปรพอยน์เตอร์เก็บแอดเดรส (address) ของตัวแปรแทนการเก็บข้อมูลต่างๆ เหมือนตัวแปรชนิดอื่น เช่น

```
int *pt;
```

ในที่นี้กำหนดให้ pt เป็นตัวแปรพอยน์เตอร์ซึ่งเก็บแอดเดรสของตัวแปรชนิดเลขจำนวนเต็ม ในเรื่องของพอยน์เตอร์มีเครื่องหมาย 2 ชนิดที่ใช้คือ * และ & เครื่องหมาย * ให้ค่าของข้อมูลซึ่งเก็บอยู่ในแอดเดรส โดยแอดเดรสนี้เก็บอยู่ในตัวแปรพอยน์เตอร์ซึ่งอยู่หลังเครื่องหมาย * สำหรับเครื่องหมาย & ให้ค่าแอดเดรสของตัวแปรซึ่งอยู่หลังเครื่องหมาย &

2.6 เชลล์สคริปต์ (Shell Script)

ในการติดต่อกับระบบปฏิบัติการลีนุกซ์มักติดต่อผ่านโปรแกรมเล็ก ๆ โปรแกรมหนึ่ง เรียกว่าเชลล์ (Shell) เชลล์เป็นชั้นของแอพพลิเคชันที่ตีความคำสั่งจากผู้ใช้และส่งต่อไปให้กับเคอร์เนลของระบบ จากนั้นแสดงผลลัพธ์ที่ได้จากเคอร์เนลกลับมาให้ผู้ใช้อีกที และจากการที่ส่วนใหญ่ผู้ใช้ต้องติดต่อกับระบบปฏิบัติการลีนุกซ์ผ่านทางเชลล์นี้เอง จึงทำให้คนส่วนใหญ่เข้าใจว่าเชลล์ที่เห็นนั้นคือระบบปฏิบัติการ ซึ่งอันที่จริงแล้วเชลล์เป็นเพียงแอพพลิเคชันบนระบบเท่านั้น

2.6.1 ประวัติของเชลล์

ในระบบยูนิกซ์รวมถึงลีนุกซ์นั้น สามารถเลือกใช้เชลล์ได้หลายแบบ ซึ่งแตกต่างจากระบบดอสที่จำกัดให้ใช้เชลล์ที่ติดตั้งมาแล้วกับดอส คือ command.com ได้เพียงอย่างเดียว สำหรับโปรแกรมเชลล์ตัวแรกที่มีใช้กันนั้นคือ เบอร์นเชลล์ (Bourne Shell) ซึ่งตั้งชื่อตามชื่อผู้คิดค้นโปรแกรมนี้ขึ้น คือนายสตีเฟน บอร์น (Steven Bourne) เบอร์นเชลล์นี้แจกจ่ายไปพร้อมกับยูนิกซ์เวอร์ชัน 7 ซึ่งเป็นเวอร์ชันที่ได้รับความนิยมมารที่สุดตัวหนึ่งในปี ค.ศ.1979 ผู้ใช้ส่วนใหญ่รู้จักเบอร์นเชลล์จากการเรียกคำสั่งในระบบว่า sh ถึงแม้ว่าในปัจจุบันมีโปรแกรมเชลล์ออกมามากมาย เบอร์นเชลล์ก็ยังถือว่าเป็นโปรแกรมสำคัญพื้นฐานที่ยูนิกซ์ทุกระบบต้องมี นอกจากนี้โปรแกรมเชลล์ตัวอื่นๆ ที่ถูกพัฒนาขึ้นมาภายหลังส่วนใหญ่ก็มีความเข้ากันได้กับเบอร์นเชล์ลอยู่ทั้งสิ้น

โปรแกรมเชลล์ที่พัฒนาขึ้นมาภายหลังและได้รับความนิยมอย่างสูงอีกตัวหนึ่งก็คือ ซีเชลล์ (C Shell) หรือ csh ซึ่งพัฒนาโดยนายบิล จอย (Bill Joy) (หนึ่งในผู้ก่อตั้งบริษัทซันไมโครซิสเต็ม (Sun Microsystem) ซึ่งเป็นผู้ผลิตและจำหน่ายระบบคอมพิวเตอร์ที่ใช้ยูนิกซ์รายใหญ่ที่สุดรายหนึ่งในปัจจุบัน) ซึ่งในสมัยนั้นอยู่ที่มหาวิทยาลัยแคลิฟอเนีย (California) โปรแกรมนี้ถือเป็นส่วนหนึ่งของระบบยูนิกซ์รุ่นของเบิร์กเลย์ หรือที่เรียกกันว่าเบิร์กเลย์ซิสเต็มดิสทริบิวชัน (Berkeley System Distribution) หรือ BSD Unix สาเหตุที่เรียกเชลล์นี้ว่าซีเชลล์ (C Shell) เพราะมีรูปแบบของคำสั่งและการใช้งานที่คล้ายกับโปรแกรมภาษาซีทำให้ผู้ใช้ที่เป็นโปรแกรมเมอร์มีความคุ้นเคยกับการใช้งานเชลล์นี้มาก

โปรแกรมเชลล์ตัวอื่นที่ถูกพัฒนาขึ้นภายหลังและได้รับความนิยมสูงอีกโปรแกรมหนึ่งก็คือ คอร์นเชลล์ (Korn Shell) หรือ ksh ถูกพัฒนาขึ้นมาโดยนายเดวิด คอร์น (David Korn) จากเอทีแอนด์ทีเบลแลบ (AT&T Bell Lab) ในราวกลางทศวรรษ 1980 คอร์นเชลล์นี้มีความเข้ากันได้กับเบอร์นเชลล์ได้โดยง่าย นอกจากนี้คอร์นเชลล์ก็ยังมีรูปแบบพิเศษบางอย่างที่ถูกเพิ่มเติมเข้ามาเพื่อให้ผู้ใช้สามารถใช้งานได้สะดวกขึ้นด้วย

ในระบบลีนุกซ์สามารถเลือกติดตั้งเชลล์ได้หลายแบบ ซึ่งสามารถตรวจสอบได้ว่าขณะนี้กำลังใช้เชลล์อะไรอยู่โดยใช้คำสั่งต่อไปนี้

```
$ echo $SHELL
/bin/bash
```

ตัวอย่างนี้แสดงว่ากำลังใช้ bash หรือ Bourne again shell อยู่ หากต้องการเปลี่ยนหรือเรียกใช้งานเชลล์ชนิดอื่นๆ ขึ้นมา ก็สามารถใช้คำสั่งโดยป้อนชื่อของเชลล์ที่ต้องการลงไปได้เช่น

```
$ bash (call Bourne again shell)
$ csh (call c shell)
```

หรือสามารถเปลี่ยนแปลงดีฟอลต์เชลล์ (default shell) ที่ถูกเรียกขึ้นมาทุกครั้งที่มีการล็อกอินเข้าใช้งานระบบได้จากแฟ้มข้อมูล /etc/passwd

2.6.2 การเขียนโปรแกรมเชลล์

การเขียนโปรแกรมเชลล์หรือ เชลล์สคริปต์ (Shell script) เป็นการนำเอาคำสั่งต่างๆ ของระบบมารวมเข้าด้วยกัน โดยมีคำสั่งควบคุมการทำงาน (flow control) คอยเป็นตัวบังคับ และควบคุมรูปแบบของโปรแกรม

การควบคุมลักษณะของการแสดงผล

การควบคุมการแสดงผลบนจอภาพใช้รหัสพิเศษที่ขึ้นต้นด้วยเอสเคป (escape) หรือ \033 เรียกรหัสชุดเหล่านี้ว่าเอสเคปซีเควนต์ (escape sequence) สามารถบังคับให้ระบบแสดงผลด้วยตัวอักษรแบบกระพริบ, แบบกลับขาวเป็นดำ หรือขีดเส้นใต้ได้ ตัวอย่างของรหัสต่างๆ ที่ใช้มีดังนี้

\033[line;colH	ตำแหน่งของเคอร์เซอร์ไปอยู่บนบรรทัด line และคอลัมน์ col
\033[2J	ลบหน้าจอ (คล้ายกับคำสั่ง tput clear บนยูนิกส์หรือคำสั่ง CLS บน DOS)
\033[4m	เข้าสู่โหมดขีดเส้นใต้ ตัวอักษรที่พิมพ์ต่อจากรหัสชุดนี้เป็นตัวอักษรที่มีการขีดเส้นใต้ทั้งหมด
\033[5m	เข้าสู่โหมดกระพริบ (blink)
\033[7m	โหมดกลับขาวดำ (reverse)
\033[m	กลับสู่โหมดปกติ

สามารถใช้คำสั่ง echo เพื่อส่งรหัสเหล่านี้ได้ สำหรับรหัสเอสเคป ให้กดปุ่ม Ctrl-v (กดปุ่ม Ctrl ค้างไว้และตามด้วย v) หลังจากนี้ เมื่อกดปุ่มใดก็เป็นการส่งรหัสประจำปุ่มนั้นไปให้ระบบ ในที่นี้จะกดปุ่ม Esc เมื่อกดแล้วเห็นเป็นตัวอักษร "^[" หากต้องการกลับสู่โหมดตัวอักษรกระพริบ ต้องกดปุ่ม Ctrl-v ตามด้วยปุ่ม Esc ปุ่ม [, 5 และ m ตามลำดับ

```
$ echo ^[[5m
```

เมื่อพิมพ์ตัวอักษรหลังจากนี้ เห็นเป็นตัวอักษรกระพริบทั้งหมด (สำหรับจอที่มีการกำหนดเป็นแอตทริบิวต์สี ก็จะเห็นเป็นสีที่แตกต่างจากปกติ) จากนั้นให้ป้อนคำสั่งต่อไปนี้เพื่อกลับสู่โหมดปกติ

```
$ echo ^[[m
```

ถ้าต้องการลบหน้าจอ ให้ใช้คำสั่ง tput clear คำสั่งต่างๆ สามารถนำมาใส่ไว้ในไฟล์เดียวกันได้ เรียกว่า เชลล์สคริปต์ และเรียกใช้งานโดยใช้คำสั่ง

```
$ bash myscript     (สำหรับ Bourne Again Shell)
```

หรือ

```
$ ksh myscript      (สำหรับ Korn shell)
```

หรือใช้คำสั่ง chmod เพื่อเปลี่ยนโหมดของไฟล์ให้เป็นชนิดที่ทำงานได้ (executable) แล้วเรียกเชลล์สคริปต์นั้นให้ทำงานด้วยการป้อนชื่อเข้าไปโดยตรง ดังนี้

```
$ chmod +x myscript      -> เปลี่ยนโหมดของไฟล์ให้เป็นแบบ executable
$ myscript               -> เรียกให้สคริปต์ทำงานโดยพิมพ์ชื่อลงไปโดยตรง
```

ตัวอย่างซอร์สโค้ดของสคริปต์

```
tput clear
echo "^[[07;20H+--------------------------------------+"
echo "^[[07;38H^[[7m MENU  ^[[m"
echo "^[[08;20H|                                      |"
echo "^[[09;20H|  1. Option 1.                        |"
echo "^[[10;20H|  2. Option 2.                        |"
echo "^[[11;20H|  3. Option 3.                        |"
echo "^[[12;20H|  4. Option 4.                        |"
echo "^[[13;20H|  5. Exit.                            |"
echo "^[[14;20H|                                      |"
echo "^[[15;20H+--------------------------------------+"
echo "^[[16;20H|  Please select your choice :"
read choice
```

การเปลี่ยนแปลงทิศทางของช่องทางการสื่อสาร (I/O redirection)

ในการสื่อสารระหว่างเทอร์มินอลกับตัวโฮสต์ของคอมพิวเตอร์มีช่องทางที่ใช้กันอยู่ 3 ทาง คือ STDIN (ช่องทางรับข้อมูลมาตรฐาน), STDOUT (ช่องทางแสดงผลลัพธ์มาตรฐาน) และ STDERR (ช่องทางแสดงผลลัพธ์ที่เป็นข้อผิดพลาดมาตรฐาน) ทั้งนี้ความสามารถในเรื่องการจัดการเทอร์มินอลหรือเรื่องของการเปลี่ยนทิศทางของช่องทางสื่อสาร (I/O redirection) ไม่ได้เป็นความสามารถจากโปรแกรมแอพพลิเคชันแต่เป็นความสามารถจากตัวระบบปฏิบัติการเอง

ลีนุกซ์ได้รับอิทธิพลมาจากระบบยูนิกซ์ จึงยังคงความสามารถเกี่ยวกับเรื่องของการจัดการเทอร์มินอลและการเปลี่ยนทิศทางของช่องทางสื่อสารไว้อย่างครบถ้วน จึงสามารถสังเปลี่ยนแปลงช่องทางการสื่อสารเหล่านี้จากเชลล์ได้ตลอดเวลา

การเปลี่ยนแปลงช่องทางการแสดงผลลัพธ์ออก (standard output redirection)

ในที่นี้ยกตัวอย่างการใช้งานคำสั่ง "cat" ซึ่งรอรับข้อมูลจากคีย์บอร์ด แล้วแสดงผลลัพธ์กลับมาทางจอภาพ ดังนั้นในที่นี้ STDIN คือคีย์บอร์ด และ STDOUT คือจอภาพหรือมอนิเตอร์

```
$ cat
This is first line       -> พิมพ์ข้อความ "This is first line"
This is first line       -> cat แสดงผลลัพธ์ คือ ข้อความที่พึ่งพิมพ์ลงไป
This is second line      -> พิมพ์ข้อความ "This is second line"
This is second line      -> cat แสดงผลลัพธ์ คือ ข้อความที่เพิ่งพิมพ์ลงไป
```

ต่อไปนี้จะเปลี่ยนช่องทางแสดงผลลัพธ์ของโปรแกรม ซึ่งปกติแสดงออกทางจอภาพให้ไปเก็บลงไฟล์แทน

```
$ cat > myfile
This is first line          -> พิมพ์ข้อความ "This is first line"
This is second line         -> พิมพ์ข้อความ "This is second line"
```

โปรแกรม cat จัดเก็บผลลัพธ์ลงในไฟล์ที่ชื่อ "myfile" แทน นั่นคือ STDOUT ได้ถูกเปลี่ยนเป็นไฟล์ "myfile" แล้ว เครื่องหมาย ">" ใช้ในการเปลี่ยนช่องทางการแสดงผลลัพธ์ เป็นการเขียนทับไฟล์ที่ระบุชื่อ ถ้าต้องการให้เขียนผลลัพธ์ต่อท้ายข้อมูลเดิมที่มีอยู่แล้วในไฟล์ ต้องใช้เครื่องหมาย ">>" ดังตัวอย่าง

```
$ cat >> myfile
```

การเปลี่ยนแปลงช่องทางการนำข้อมูลเข้า (standard input redirection)

ในที่นี้ ขอยกตัวอย่างคำสั่ง sort ซึ่งโดยปกติรับข้อมูลเข้าจากคีย์บอร์ดแล้วแสดงผลลัพธ์กลับออกมาทางจอภาพ ดังนั้น STDIN คือคีย์บอร์ด และ STDOUT คือจอภาพ

การทำงานของคำสั่ง sort

```
$ sort
Peter                -> พิมพ์ Peter
Mary                 -> พิมพ์ Mary
John                 -> พิมพ์ John เสร็จแล้วกด Ctrl-D
John                 -> sort จัดเรียงข้อมูลใหม่
Mary
Peter
```

เตรียมไฟล์ชื่อ "namelist" ไว้ให้กับโปรแกรม sort

```
$ cat > namelist
```

```
Peter                -> พิมพ์ Peter
Mary                 -> พิมพ์ Mary
John                 -> พิมพ์ John เสร็จแล้วกด Ctrl-D
```

การเปลี่ยนทิศทางของช่องทางนำข้อมูลเข้าจากคีย์บอร์ด ไปเป็นการนำข้อมูลเข้าจากไฟล์ที่ชื่อ "namelist" ทำดังต่อไปนี้

```
$ sort < namelist
Peter
Mary
John
```

กล่าวคือ เครื่องหมายที่ใช้ในการเปลี่ยนช่องทางน้ำข้อมูลเข้า คือ "<" จากตัวอย่างข้างต้น STDIN เปลี่ยนไปเป็นไฟล์ที่ชื่อ "namelist"

การเปลี่ยนทั้ง STDIN และ STDOUT พร้อมกัน

ยกตัวอย่าง เช่น

```
$ sort < namelist > sortlist
```

คำสั่งนี้โปรแกรม sort รับข้อมูลจากไฟล์ "namelist" และเขียนผลลัพธ์ไปใส่ในไฟล์ "sortlist" สามารถลองใช้คำสั่ง cat ตรวจดูข้อมูลในไฟล์ "sortlist" ได้

การใช้งานไปป์ (pipe)

การใช้งานไปป์ (pipe) เป็นการนำเอาผลลัพธ์จากคำสั่งหนึ่งไปเป็นข้อมูลเข้าของอีกคำสั่งหนึ่ง ซึ่งโปรแกรมที่มีลักษณะการใช้งานเป็นแบบฟิลเตอร์ ได้แก่ cat, sort, more, grep, tee และ tr

ตัวอย่างการใช้ไปป์

```
$ ls | grep file | sort
```

เป็นการเรียงลำดับข้อมูลที่ได้จากการคัดเลือกรายชื่อไฟล์ที่มีชื่อ "file" เป็นส่วนประกอบ

คำสั่งควบคุมทิศทางแบบวนรอบ (loop flow control)

คำสั่งควบคุมทิศทางแบบวนรอบหรือวนลูป มีอยู่สามแบบคือ for, while และ until

for loop

โครงสร้าง

For *VAR* **in** *Arg-list*
do
    command list
done

คำสั่ง for นี้ทำงานวนรอบจนกระทั่งสมาชิกใน "Arg-list" (หมายถึง argument list) ถูกใช้หมด ชุดคำสั่งในลูป for ต้องอยู่ภายใต้คำบังคับ "do" และ "done" การใช้ for ส่วนใหญ่ใช้กับลูปที่รู้จำนวนรอบที่ต้องทำและ "Arg-list" ที่แน่นอน

ตัวอย่าง

```
SUM=0
for i in 1 2 3 4 5 6 7 8 9
do
        SUM=`expr $SUM + $i`
done
echo  "Sum is $SUM"
```

ให้บันทึกเชลล์สคริปต์ข้างต้นเป็นไฟล์ชื่อ ex8 แล้วสั่งให้โปรแกรมทำงานดังนี้

```
$ ex8
Sum is 45
```

จากตัวอย่างเป็นการรวมตัวเลขจำนวนเต็มตั้งแต่ 1 ถึง 9 เสร็จแล้วพิมพ์ค่าที่ได้

while และ until

โครงสร้างของ while

while [*true condition*]
do
    command list
done

คำสั่ง while ใช้กับเงื่อนไขที่เป็นจริง ตัวอย่างเช่น

```
SUM=0
i=1
while [ $i -lt 10]
do
        SUM=`expr $SUM + $i`
        i=`expr $i + 1`
done
echo  "Sum is $SUM"
```

โปรแกรมตัวอย่างข้างบนทำงานเหมือนกับ ex8 แต่ในที่นี้ใช้รูปแบบของ while loop โดยโปรแกรมบวกตัวเลข "i" ไปเรื่อยๆ เริ่มตั้งแต่ 1 และเพิ่มค่า i ขึ้นทีละหนึ่ง ลูปนี้ทำงานไปเรื่อยๆ ตราบใดที่ค่า i ยังน้อยกว่า 10 และเมื่อค่า i มากกว่าเท่ากับ 10 แล้วออกจากลูปและพิมพ์ผลบวกที่ได้ทั้งหมด

โครงสร้างของ until

until [*false condition*]
do
    command list
done

คำสั่ง until ใช้กับเงื่อนไขที่เป็นเท็จ ตัวอย่างเช่น

```
SUM=0
i=1
until [ $i -eq 10]
do
      SUM='expr $SUM + $i'
      i='expr $i + 1'
done
echo  "Sum is $SUM"
```

ตัวอย่างข้างต้นนี้ก็เป็นโปรแกรมที่ให้ผลลัพธ์แบบเดียวกับในตัวอย่างที่ใช้โครงสร้างของ for และ while นั่นเอง และจะเห็นว่ามีลักษณะคล้ายกับโปรแกรมที่ใช้โครงสร้างแบบ while-loop มาก เพียงแต่เงื่อนไขที่ใช้ตรวจสอบกลับกันเท่านั้น กล่าวคือเงื่อนไขนี้ให้ตรวจสอบว่าค่าของตัวแปร i เท่ากับ 10 หรือยัง ทั้งนี้ until-loop จะทำการวนลูปทำงานไปเรื่อยๆ ตราบใดที่เงื่อนไขที่ตามหลัง until มานั้นยังคงเป็นเท็จ

ในการเขียนโปรแกรมสามารถเลือกใช้เฉพาะ while-loop หรือ until-loop ตัวใดตัวหนึ่งเพียงอย่างเดียวก็ได้ เพราะเงื่อนไขของลูปทั้งสองแบบสามารถดัดแปลงให้ทดแทนกันได้ ดังที่ได้แสดงในโปรแกรมตัวอย่าง

break และ continue

คำสั่ง break และ continue จะหยุดการทำงานของชุดคำสั่งที่ตามหลังสองคำสั่งนี้ สำหรับคำสั่ง break เมื่อหยุดแล้วก็กระโดดออกไปนอกลูป ตรงจุดที่อยู่ถัดจากคำบังคับ "done" ส่วน continue กลับไปทำงานในลูปต่อตรงจุดต้นลูปที่อยู่ถัดจากคำบังคับ "do" ดังตัวอย่างต่อไปนี้

```
SUM=0
While [ true ]
do
      echo -n "Please enter number 1-9 , except 5 , (q-quit)"
      read INPUT
      if [ "$INPUT" = "q" ]
      then
            echo "Break loop now!"
            break
            elif [ $INPUT -eq 5 ]
            then
                  echo "Please not input 5"
                  continue
            elif [$INPUT -gt 9 ]
            then
                  echo "Please do not input data > 9"
                  continue
            fi
            SUM='expr $$SUM + $INPUT'
done
echo  "Sum is $SUM"
```

ในตัวอย่างนี้นอกจากแสดงถึงการใช้ break กับ continue แล้ว ยังแสดงให้เห็นถึงการใช้ while ในแบบที่นิยมใช้กันอีกลักษณะหนึ่ง คือการใช้กับเงื่อนไข "true" วิธีการใช้งานแบบนี้ทำให้โปรแกรมวนลูปแบบไม่รู้จบ ซึ่งมักใช้กับการวนทำงานเพื่อรับข้อมูลเข้าไปตามตัวอย่างข้างต้น โดยมีเงื่อนไขของการรับข้อมูลบางรูปแบบที่ใช้กระโดดออกจากลูปเพื่อจบการทำงาน ซึ่งในที่นี้ก็ใช้คำสั่ง "break" เมื่อผู้ใช้มีการป้อนตัวอักษร "q" เข้ามา ทั้งนี้อาจใช้คำสั่ง "exit" แทน "break" ก็ได้ แต่ทำให้ไม่มีการทำงานตามคำสั่งในส่วนที่พิมพ์ผลลัพธ์ ซึ่งอยู่ท้ายโปรแกรมไปด้วย

สำหรับคำสั่ง "continue" ใช้กับการป้อนข้อมูลที่เป็นตัวเลข 5 ซึ่งในที่นี้เป็นการข้ามการบวกเลข 5 ไป ตัวเลขที่ใส่ได้และโปรแกรมทำการบวกเลขให้มีเฉพาะตัวเลข 1-4 และ 6-9 เท่านั้น ส่วนตัวเลขที่น้อยกว่า 1 และมากกว่า 9 ก็จะถูกคำสั่ง "continue" ข้ามส่วนของโปรแกรมที่เป็นการบวกเลขไปเช่นเดียวกัน

คำสั่ง continue และ break นี้กระโดดออกไปจากลูปชั้นที่โปรแกรมกำลังทำงานอยู่ แต่สามารถระบุให้ continue และ break กระโดดออกไปยังลูปที่ชั้นใดก็ได้ (ในกรณีที่ลูปของมีการครอบซ้อนอยู่หลายชั้น) วิธีการใช้งานก็เพียงแต่เติมตัวเลขตามท้ายคำสั่ง break และ continue เท่านั้น ตัวอย่างเช่น "break 2" ทำการกระโดดออกจากลูปไปสองชั้น (ไปอยู่หลัง done ของลูปที่สอง)

2.6.3 ตัวแปรเชลล์

ตัวแปรเชลล์ (shell variable) มีการเก็บค่าไว้ในลักษณะของข้อความหรือสตริง (string) ในกรณีที่ใช้งานเป็นตัวเลขก้ต้องใช้โปรแกรม expr เข้ามาช่วยแปลงข้อมูลก่อน นอกจากนี้การกำหนดค่าให้กับตัวแปรถือว่าเป็นการประกาศตัวแปรนั้นขึ้นมาด้วย ซึ่งมีรูปแบบดังต่อไปนี้

ชื่อตัวแปร=ค่าตัวแปร

โดยห้ามมีช่องว่าง (space) คั่นระหว่างเครื่องหมายเท่ากับ ตัวอย่างเช่น

```
NAME=KLAUSE
```

หากมีการกำหนดค่าที่เป็นประโยคข้อความยาวๆ ที่มีช่องว่างอยู่ในประโยคนั้น ต้องใช้เครื่องหมายอัญประกาศ (" ") ครอบประโยคนั้น เช่น

```
NAME="Peter Klause"
```

การตั้งชื่อของตัวแปรสามารถใช้ตัวอักษร, ตัวเลข, ตัวขีดเส้นใต้ (underscore) แต่ห้ามใช้ตัวอักษรพิเศษต่างๆ เช่น $, *, #, () และห้ามขึ้นต้นด้วยตัวเลข ตัวอย่างชื่อตัวแปรเช่น VAR1, VAR_NAME, Variable แต่โดยทั่วไปแล้วการตั้งชื่อของตัวแปรมักใช้อักษรตัวใหญ่ในการตั้งชื่อ ส่วนการใช้งานตัวแปรเชลล์หรือการอ้างถึงค่าของตัวแปร ต้องใช้เครื่องหมาย "$" นำหน้าตัวแปรนั้น เช่น ถ้าใช้คำสั่ง

```
$ echo $NAME
```

ในกรณีที่กำหนดค่าของตัวแปรเป็น "Peter Klause" ได้ผลลัพธ์เป็น Peter Klause แต่ถ้าใช้คำสั่ง $ echo NAME เนื่องจากไม่ได้อ้างถึงค่าของตัวแปร ดังนั้นได้ผลลัพธ์เป็น

NAME

ในกรณีต้องการให้เชลล์สามารถแยกชื่อของตัวแปรออกจากค่าอื่นๆ สามารถทำได้โดยครอบชื่อของตัวแปรนั้นด้วยเครื่องหมายปีกกา "{ }" ดังตัวอย่าง

```
$ MYDIR=/home/mary/
$ cat ${MYDIR}myfile
```

คำสั่งข้างต้นเป็นการสั่งให้พิมพ์รายละเอียดของแฟ้มที่ชื่อ /home/mary/myfile (ซึ่งเป็นการเอาค่าของตัวแปร (/home/mary/) มาต่อกับชื่อไฟล์ (myfile)) ออกมาให้ หากไม่ครอบชื่อของตัวแปรด้วยเครื่องหมายปีกกาแล้ว เชลล์จะมองชื่อตัวแปรเป็น $MYDIRmyfile (ซึ่งไม่มีค่าอะไรอยู่)

การยกเลิกตัวแปรสามารถทำได้โดยใช้คำสั่ง

$ unset ชื่อตัวแปร

ขอบเขตของตัวแปรเชลล์

ตัวแปรเชลล์ที่ถูกกำหนดขึ้นมามีลักษณะเป็น ตัวแปรท้องถิ่น (local variable) คือรู้จักเฉพาะในเชลล์ของชั้นตนเองเท่านั้น เชลล์ในชั้นอื่นๆ ที่ถูกเรียกซ้อนกันจะไม่รู้จักตัวแปรดังกล่าวจนกว่าประกาศตัวแปรนี้ให้เชลล์ชั้นอื่นๆ ได้รู้จักก่อน ซึ่งทำได้โดยใช้คำสั่ง "export"

การเรียกเชลล์ซ้อนกันหลายชั้นหรือ "ซับเชลล์" (sub shell) หมายถึงการพิมพ์คำสั่ง "sh" หรือ "ksh" ซึ่งเป็นการเรียกเบอร์นเชลล์หรือคอร์นเชลล์ซ้อนขึ้นมาบนเชลล์ปัจจุบัน เมื่อเรียกขึ้นมาแล้วจะเห็นเป็นเครื่องหมาย prompt ของระบบตามปกติ ซึ่งหมายความว่าซับเชลล์ได้ถูกเรียกขึ้นมาใหม่แล้ว อาจใช้คำสั่ง "ps" เพื่อตรวจดูว่าซับเชลล์ถูกเรียกขึ้นมาหรือไม่ ดังนี้

```
$ ps
PID   TTY    STAT   TIME         COMMAND
962   p2     S      0:00   login -h localhost -p
963   p2     S      0:00   -bash
974   p2     R      0:00   ps
```

ตัวอย่างเชลล์แรกที่ลีนุกซ์กำหนดไว้ให้เมื่อทำการล็อกอินเข้ามาใช้ระบบครั้งแรกคือ bourne again shell (bash) ซึ่งเป็นโพรเซสหมายเลข 963

```
$ sh
$ ps
PID   TTY    STAT   TIME         COMMAND
962   p2     S      0:00   login -h localhost -p
963   p2     S      0:00   -bash
975   p2     S      0:00   sh
976   p2     R      0:00   ps
```

เมื่อเรียกซับเชลล์เห็นโพรเซสหมายเลข 975 ซึ่งเป็นโพรเซสของเบอร์นเชลล์ที่เรียกด้วยคำสั่ง "sh"ขึ้นมา

```
$ exit
exit
$ ps
PID   TTY    STAT   TIME         COMMAND
962   p2     S      0:00   login -h localhost -p
963   p2     S      0:00   -bash
989   p2     R      0:00   ps
```

สามารถออกจากซับเชลล์ได้โดยการพิมพ์คำสั่ง "exit" ตามปกติ เมื่อใช้คำสั่ง exit เพื่อออกจากซับเชลล์ จะเห็นว่าโพรเซสของเบอร์นเชลล์หายไป เนื่องจากถูกฆ่าด้วยคำสั่ง exit ไปแล้ว แสดงว่าตอนนี้ได้กลับมาอยู่ที่เชลล์แรกซึ่งก็คือ bash อีกครั้ง (โพรเซสหมายเลข 963)

ตัวอย่างการเรียกซับเชลล์และการประกาศตัวแปรให้เชลล์ชั้นอื่นๆ ได้รู้จัก

```
$ NAME=KLAUSE       -> ประกาศตัวแปร NAME ในเชลล์ชั้นปัจจุบัน
$ sh                -> เรียกซับเชลล์ซ้อนขึ้นมา
$ echo $NAME        -> ไม่พิมพ์ผลลัพธ์ใดๆ ออกมา เนื่องจากซับเชลล์ไม่รู้จัก
$ exit              -> ออกจากซับเชลล์กลับมาสู่เชลล์ปัจจุบัน
$ export NAME       -> ประกาศตัวแปร NAME ให้กับเชลล์ชั้นอื่นๆ
$ sh                -> เรียกซับเชลล์อีกครั้งหนึ่ง
$ echo $NAME
KLAUSE              -> เมื่อซับเชลล์รับรู้ตัวแปรแล้ว จึงพิมพ์ผลลัพธ์ออกมาให้
$ readonly NAME     -> คำสั่งนี้ทำให้ไม่สามารถเปลี่ยนแปลงค่าตัวแปรได้
```

ตัวแปรสงวนหรือตัวแปรระบบ

ในระบบลีนุกซ์และยูนิกซ์มีตัวแปรสงวน (reserved) ที่ตัวระบบปฏิบัติการหรือโอเอสนำไปใช้ประโยชน์โดยเฉพาะเท่านั้น แต่หากรู้ความหมายของตัวแปรเหล่านั้นก็สามารถแก้ไขเปลี่ยนแปลงค่าของตัวแปรดังกล่าวได้ สามารถตรวจสอบค่าของตัวแปรระบบได้โดยใช้คำสั่งดังนี้

```
$ set
        PPID=373
        PS1=$
        PS2=>
        PS4=+
        PWD=/home/twg
        SHELL=/bin/bash
```

```
SHLVL=1
TERM=xterm
UID=406
USER=twg
WINDOWID=37748749
_=set
i=/etc/profile.d/mh.sh
```

พบว่าระบบแสดงรายละเอียดของชื่อและค่าของตัวแปรสงวนต่างๆ ออกมาให้ ต่อไปเป็นคำอธิบายของตัวแปรสงวนบางตัว

```
$ HOME      -> Home directory
$ PATH      -> search patch เมื่อคำสั่งเรียกโปรแกรมให้ทำงาน ระบบทำการค้นหาตัว
               โปรแกรมนั้นจากรายชื่อของไดเรกทอรีที่ได้ระบุไว้ในตัวแปรนี้
$ MAIL      -> mailbox
$ PS1       -> prompt ตัวที่ 1 ของเชลล์
$ PS2       -> prompt ตัวที่ 2 ของเชลล์
$ LOGNAME   -> ชื่อทะเบียนผู้ใช้ระบบ (login name)
$ TERM      -> ชนิดของจอ
```

นอกจากการใช้คำสั่ง "set" เพื่อแสดงรายชื่อและค่าของตัวแปรต่างๆ ออกมาแล้ว ยังสามารถใช้คำสั่ง "echo <ชื่อตัวแปร>" เพื่อแสดงค่าของตัวแปรดังกล่าวออกมาก็ได้เช่นกัน และการเปลี่ยนหรือกำหนดค่าให้กับตัวแปรก็สามารถทำได้เช่นเดียวกับตัวแปรเชลล์ปกติทั่วไป ดังตัวอย่าง

```
$ echo $PS1          -> ให้แสดงค่าของตัวแปร PS1 (prompt ของระบบ)
$                    -> ระบบพิมพ์ตัวอักษร "$" ซึ่งถูกใช้เป็น prompt ออกมาให้
$ PS1="Linux:> "     -> กำหนดค่าของ prompt ใหม่เป็น "Linux:>"
Linux:>              -> prompt ของระบบเปลี่ยนเป็น "Linux:>"
```

2.6.4 การสร้างเชลล์สคริปต์

การสร้างเชลล์สคริปต์ เป็นการนำเอาคำสั่งต่างๆ ในลีนุกซ์มาประกอบเข้าเป็นไฟล์ จากคำสั่งต่อไปนี้ ลองสร้างไฟล์ชื่อ "ex1"

```
$ sh                    #call subshell
$ echo "Hello world"     #print "Hello world"
$ date                  #print date information
$ pwd                   #print current directory
$ exit                  #exit from subshell
```

เมื่อบันทึกข้อมูลเก็บลงไฟล์ชื่อ "ex1" แล้ว สามารถสั่งให้โปรแกรมทำงานได้สองวิธีคือ

1. สั่งผ่านเชลล์ ให้พิมพ์คำสั่งดังนี้

```
$ sh ex1
```

วิธีการสั่งงานโปรแกรมแบบนี้ "ex1" ต้องถูกตั้งไว้ว่าอนุญาตให้สามารถอ่านไฟล์ได้ (readable) ไม่เช่นนั้นเชลล์จะไม่สามารถทำให้โปรแกรมทำงานได้

2. สั่งโดยตรงจากชื่อโปรแกรม ให้พิมพ์คำสั่ง

```
$ ex1
```

สำหรับวิธีนี้โปรแกรม "ex1" ต้องถูกตั้งว่าอนุญาตให้สามารถสั่งทำงานได้ (executable) ไม่เช่นนั้นเชลล์จะปฏิเสธการสั่งงานโปรแกรม (Permission denied)

สามารถกำหนดคุณสมบัติให้ไฟล์อนุญาตให้มีการสั่งทำงานได้โดยใช้คำสั่งดังนี้

$ chmod u+x ex1	อนุญาตให้สั่งทำงานได้ในระดับผู้ใช้
$ chmod g+x ex1	อนุญาตให้สั่งทำงานได้ในระดับกลุ่ม
$ chmod o+x ex1	อนุญาตให้สั่งทำงานได้ในระดับบุคคลอื่นๆ
$ chmod +x ex1 และ $ chmod a+x ex1	เปิดให้สั่งทำงานได้ในทุกระดับชั้น

หากกำหนดให้ไฟล์อนุญาตให้มีการสั่งทำงานได้แล้ว และพิมพ์ชื่อโปรแกรมเพื่อสั่งให้ทำงาน แต่ปรากฏว่าเชลล์ยังคงปฏิเสธการสั่งงาน (เช่นตอบว่า ex1 : command not found) แสดงว่าเชลล์ไม่ทราบว่าจะไปค้นหาชื่อโปรแกรมได้จากไดเรกทอรีใด วิธีการระบุชื่อที่อ้างอิงกับไดเรกทอรีก็ทำได้สองวิธีคือ

1. Relative Pathname วิธีระบุโดยอ้างอิงจากไดเรกทอรีปัจจุบัน

```
$ ./ex1
```

2. Full Pathname ระบุแบบอ้างชื่อเต็ม เช่นสมมติว่าไฟล์อยู่ในไดเรกทอรี /home/user1

```
$ /home/user1/ex1
```

การเรียกใช้งานโปรแกรมโดยที่ต้องระบุชื่อไดเรกทอรีด้วย ค่อนข้างเป็นเรื่องยุ่งยาก หากเก็บโปรแกรมไว้ในไดเรกทอรีเฉพาะ เช่น /home/user1/shell สำหรับเก็บโปรแกรมเชลล์ที่เรียกใช้บ่อยๆ ก็ทำให้สะดวกขึ้น ในระบบลีนุกซ์มีวิธีการกำหนดให้เชลล์ทำการค้นหาโปรแกรมที่สามารถสั่งให้ทำงานได้จากไดเรกทอรีกลุ่มหนึ่ง ตามที่ระบุไว้ในตัวแปรของระบบชื่อ PATH (หรือที่เรียกว่า search path) สามารถตรวจสอบชื่อไดเรกทอรีที่เชลล์ค้นหาเมื่อมีการเรียกใช้งานโปรแกรมโดยไม่ระบุชื่อไดเรกทอรีได้โดยการใช้คำสั่ง

```
$ echo $PATH
/usr/local/bin:/bin:/usr/bin:/usr/X11R6/bin
```

จากผลลัพธ์ข้างต้น หากมีการสั่งงานโปรแกรมโดยไม่ระบุชื่อไดเรกทอรีดังกรณีข้างต้น เชลล์ค้นหาโปรแกรมจากไดเรกทอรีทั้งสี่ที่ระบุไว้ในตัวแปร PATH คือ /usr/local/bin, /bin, /usr/bin และ /usr/X11R6/bin ซึ่งแต่ละไดเรกทอรีถูกคั่นด้วยเครื่องหมาย ":" และโดยปรกติตัวแปร PATH กำหนดให้บรรจุไดเรกทอรีเหล่านี้ไว้สำหรับค้นหาและเรียกโปรแกรมที่ใช้บ่อยๆ (เช่น "ls" อยู่ในไดเรกทอรี "/bin" และ "ftp" อยู่ในไดเรกทอรี "/usr/bin" เป็นต้น) สำหรับโปรแกรม ex1 ของนั้น เนื่องจากอยู่ในไดเร็คทอรี /home/user1 ดังนั้นเชลล์จึงไม่สามารถค้นหาพบได้ จึงต้องเพิ่มไดเรกทอรีนี้ลงไปในตัวแปร PATH เอง

วิธีการเพิ่มไดเรกทอรีใช้คำสั่งดังต่อไปนี้

```
$ PATH=$PATH:/home/user1
$ export PATH
```

สำหรับคำสั่งแรกเป็นการนำเอาชื่อของกลุ่มไดเรกทอรีเดิม ($PATH) มาต่อเข้ากับไดเรกทอรีใหม่ (/home/user1) หากไม่ระบุชื่อไดเรกทอรีเดิม ($PATH) กลายเป็นการแทนที่ของเดิมทั้งหมดด้วยของใหม่ไป ซึ่งถ้าเป็นเช่นนั้นทำให้ไม่สามารถเรียกใช้โปรแกรมที่เคยใช้ได้อีก (เช่น โปรแกรม ls) นอกจากนี้หากต้องการให้มีการสั่งงานโปรแกรมที่อยู่ในไดเรกทอรีปัจจุบัน (current directory) ได้ด้วย ต้องใช้คำสั่งต่อไปนี้

```
$ PATH=$PATH:.
$ export PATH
```

คำสั่งข้างต้นเป็นการเพิ่มไดเรกทอรีปัจจุบัน (:.) ลงไปในตัวแปร PATH ด้วย คือจุด (dot) "." ซึ่งเป็นการแทนความหมายของไดเรกทอรีปัจจุบัน ให้ใช้คำสั่ง "echo" เพื่อตรวจดูว่าได้เพิ่มไดเรกทอรีเข้าไปในตัวแปร PATH เรียบร้อยหรือไม่ ดังนี้

```
$ echo $PATH
/usr/local/bin:/bin:/usr/bin:/usr/X11R6/bin:/home/usr1:.
```

สำหรับการกำหนดให้เพิ่มไดเรกทอรีปัจจุบัน (:.) ลงไปในตัวแปร PATH นี้ถ้าคำนึงถึงเรื่องของการรักษาความปลอดภัยแล้วเป็นเรื่องที่ไม่ควรทำอย่างยิ่ง ปกติระบบยูนิกซ์ส่วนใหญ่ไม่นิยมให้ผู้ใช้กลุ่มที่อยู่ในระดับของการดูแลรักษาระบบและมีสิทธิ์สูงที่สุดในระบบอย่าง root มีไดเรกทอรีปัจจุบัน (:.) อยู่ในตัวแปร PATH เลย เพราะหากกำหนดให้ไดเรกทอรีปัจจุบันอยู่ในตัวแปร PATH แล้ว อาจมีการอัพโหลดโปรแกรมหรือสคริปต์ที่มีจุดประสงค์ในการบุกรุกระบบไว้ในไดเรกทอรีปัจจุบัน และผู้บุกรุกที่เข้าใช้ระบบก็สามารถเรียกโปรแกรมให้ทำงานได้ทันทีโดยไม่จำเป็นต้องระบุไดเรกทอรีเต็มรูปแบบกำกับไปด้วยวิธีการกำหนดตัวแปร PATH แบบนี้ช่วยป้องกันการบุกรุกระบบได้ในระดับหนึ่ง

โดยปกติแล้วเมื่อเรียกเชลล์สคริปต์ให้เริ่มทำงาน หมายถึงการเริ่มต้นทำคำสั่งในสคริปต์ทีละคำสั่งตามลำดับ โดยก่อนการทำคำสั่งแรกในสคริปต์ก็จะมีการเรียกซับเชลล์ให้ด้วย และหลังจากทำคำสั่งทุกคำสั่งในเชลล์สคริปต์เสร็จสิ้นแล้วก็เสมือนกับการออกจากซับเชลล์นั้นด้วยคำสั่ง "exit"

แม้ว่าไม่ได้ใส่คำสั่ง exit ไว้ในเชลล์สคริปต์ แต่ก็เสมือนว่ามีคำสั่งนั้นอยู่ท้ายเชลล์สคริปต์ด้วย และโดยทั่วไปหากเป็นการจบการทำงานแบบปรกติที่ไม่มีข้อผิดพลาดเกิดขึ้น มีการส่งรหัสค่า "0" (return code 0) ออกมาให้ (คำสั่ง exit ที่ไม่มีตัวเลขตามท้ายนี้ถือว่าส่ง return code 0) แต่ถ้าเป็นการจบการทำงานแบบไม่ปรกติ คืออาจมีข้อผิดพลาดเกิดขึ้น ก็ส่งรหัสในค่าที่ไม่เป็น "0" ออกมาให้ (ปรกติเป็นรหัส "1") สำหรับรหัสค่าศูนย์ดังกล่าวนี้นำไปใช้ทดสอบเงื่อนไขร่วมกันกับคำสั่งอื่น ๆ ในเชลล์ต่อไป

ที่จริงไม่ได้ไม่การใส่ทั้งคำสั่งในการเรียกซับเชลล์และคำสั่งที่ให้ออกจากซับเชลล์นั้นลงไปในเชลล์สคริปต์ด้วย แต่เมื่อสั่งให้เชลล์สคริปต์ทำงานแล้ว โปรแกรมเชลล์จะเติมคำสั่งทั้งสองลงไปให้โดยอัตโนมัติ นั่นหมายความว่าโดยปกติแล้วทุกคำสั่งในเชลล์สคริปต์ต้องทำงานภายใต้ซับเชลล์ซึ่งการทำงานภายใต้ซับเชลล์นี้ก็มีข้อจำกัดดังที่อธิบายแล้ว นั่นคือภายใจซับเชลล์ไม่สามารถเรียกใช้งานตัวแปรจากเชลล์ชั้นบนก่อนหน้านั้นทั้งหมดได้หากไม่ได้ทำการเอ็กซ์พอร์ตตัวแปรนั้นๆ เสียก่อน อย่างไรก็ตามก็มีวิธีที่บังคับให้เชลล์สคริปต์ทำงานในเชลล์ชั้นปัจจุบันได้โดยไม่ไปเรียกซับเชลล์ ดังจะได้อธิบายต่อไป

2.6.5 วิธีการสั่งให้เชลล์สคริปต์ทำงาน

รูปแบบการสั่งงานเชลล์สคริปต์ มีได้ 4 แบบดังต่อไปนี้

แบบที่ 1

$ script-name สคริปต์ต้องการ execute permission และวิ่งที่ subshell

แบบที่ 2

$ sh script-name สคริปต์ต้องการ read permission และวิ่งที่ subshell

แบบที่ 3

$ .script-name สคริปต์วิ่งที่ current shell

แบบที่ 4

$ exec scriptname สคริปต์วิ่งที่ current shell โดยทับเชลล์ชั้นปัจจุบันที่ทำงานอยู่เดิม

สำหรับการสั่งงานเชลล์สคริปต์ในแบบที่ 1 และแบบที่ 2 เมื่อล็อกอินเข้ามา ได้เชลล์ใช้งาน (ซึ่งปกติลีนุกซ์เป็น bash) ในที่นี้เรียกว่าเชลล์เริ่มต้น (Parent Shell) เมื่อมีการสั่งให้เชลล์สคริปต์ทำงานตามรูปแบบที่ 1 หรือ 2 ก็เปรียบเสมือนมีการเรียกซับเชลล์เกิดขึ้น (คำสั่ง sh) เมื่อเรียกซับเชลล์แล้วก็ทำงานตามคำสั่งในเชลล์สคริปต์ทีลำคำสั่งจนจบก็กลับไปสู่เชลล์เริ่มต้น (คำสั่ง exit) อีกทีหนึ่ง ในกรณีนี้หากเชลล์สคริปต์ต้องการใช้งานตัวแปรที่อยู่ในเชลล์เริ่มต้นต้องมีการเอ็กซ์พอร์ตตัวแปรนั้น ๆ ออกมาจากเชลล์เริ่มต้นเสียก่อน

สำหรับการสั่งงานในแบบที่ 3 มีผลให้เชลล์สคริปต์ทำงานในเชลล์ชั้นปัจจุบันทันที (ในที่นี้คือเชลล์เริ่มต้น) และทำงานตามคำสั่งในเชลล์สคริปต์ทีละคำสั่งจนจบโดยไม่มีการเรียกซับเชลล์ขึ้นมา ซึ่งทำให้สามารถใช้งานตัวแปรเชลล์ในเชลล์เริ่มต้นได้ทันทีโดยไม่ต้องเอ็กซ์พอร์ตตัวแปรออกมาก่อน

สำหรับการสั่งงานในแบบที่ 4 เชลล์เริ่มต้นถูกทำลายไป แล้วนำเอาเชลล์สคริปต์มาแทนที่โพรเซสของ Parent Shell ที่ถูกทำลาย ดังนั้นเมื่อเชลล์สคริปต์ทำงานทีละคำสั่งจนจบแล้วก็กลับไปสู่หน้าจอของการล็อกอินโดยอัตโนมัติ

บทที่ 3

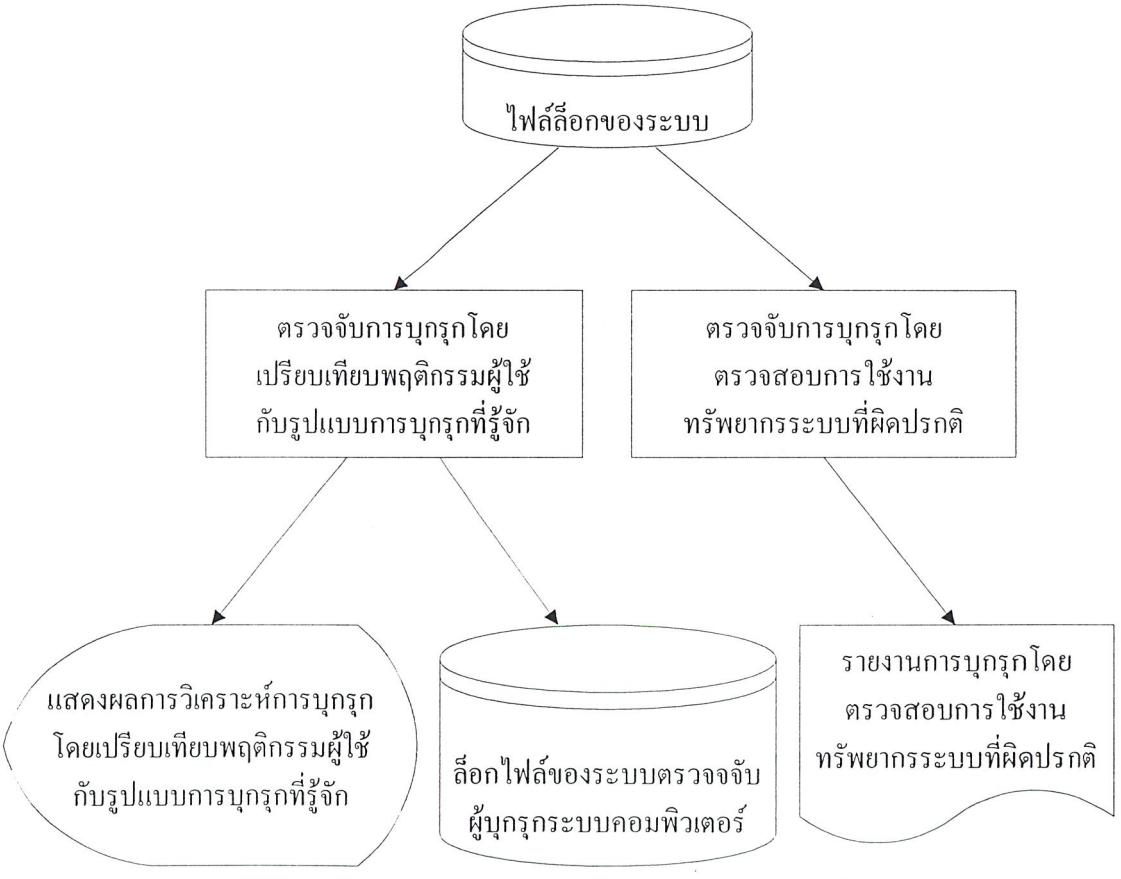
การคำนวณ การสร้างและการออกแบบ

3.1 ระบบโดยรวม

ระบบตรวจจับผู้บุกรุกระบบคอมพิวเตอร์ที่ออกแบบแบ่งออกเป็น 2 ส่วนคือ

1. ส่วนตรวจจับการบุกรุกโดยเปรียบเทียบพฤติกรรมผู้ใช้กับรูปแบบการบุกรุกที่รู้จัก (Misuse Detection)
2. ส่วนตรวจจับการบุกรุกโดยตรวจสอบการใช้งานทรัพยากรระบบที่ผิดปรกติ (Anomaly Detection)

ซึ่งทั้งสองส่วนมีรายงานการบุกรุกระบบแยกกัน และมีไฟล์ล็อกสำหรับส่วนตรวจจับการบุกรุกโดยเปรียบเทียบพฤติกรรมผู้ใช้กับรูปแบบการบุกรุกที่รู้จัก ดังแสดงในรูปที่ 3-1 รายละเอียดของการออกแบบแต่ละส่วนได้อธิบายไว้ในหัวข้อ 3.2 และ 3.3 ตามลำดับ



รูปที่ 3-1 ระบบตรวจจับผู้บุกรุกระบบคอมพิวเตอร์

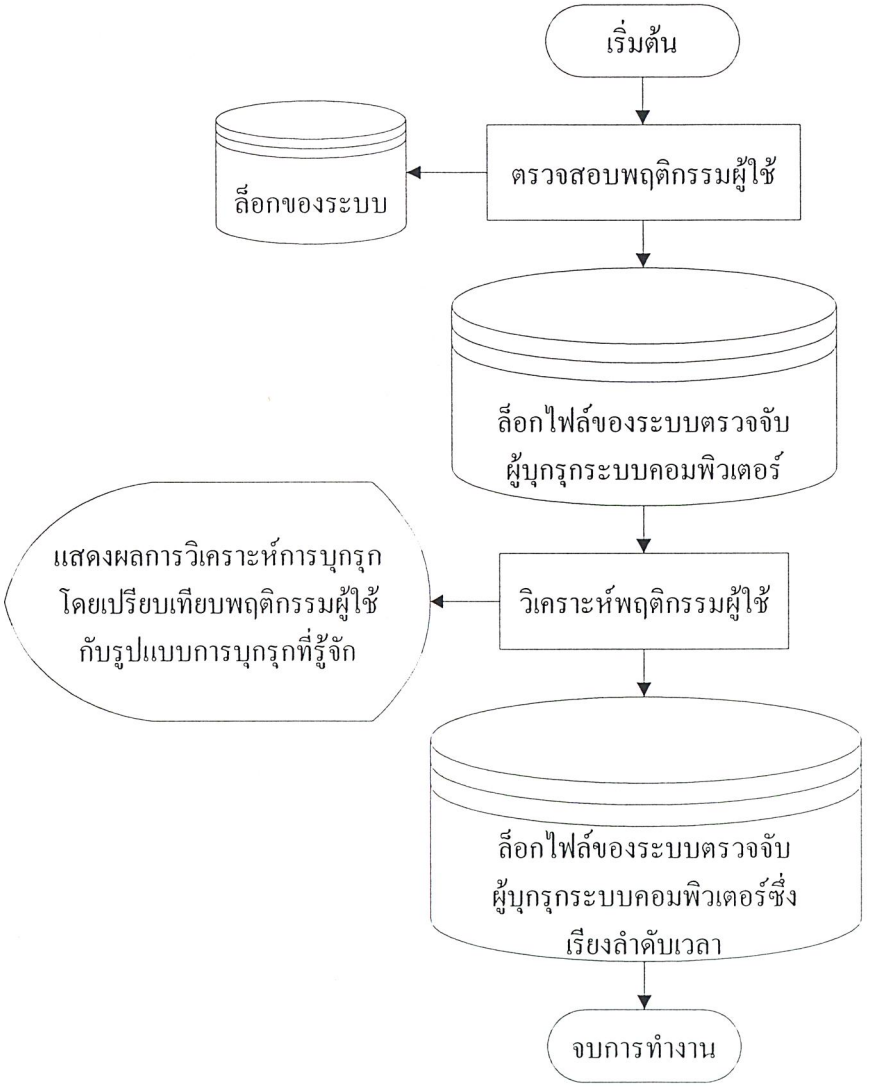
3.2 ส่วนตรวจจับการบุกรุกโดยเปรียบเทียบพฤติกรรมผู้ใช้กับรูปแบบการบุกรุกที่รู้จัก (Misuse Detection)

ส่วนตรวจจับการบุกรุกโดยเปรียบเทียบพฤติกรรมผู้ใช้กับรูปแบบการบุกรุกที่รู้จักแบ่งเป็น 3 ส่วนหลัก คือ

1. *ส่วนตรวจสอบพฤติกรรมผู้ใช้* มีหน้าที่คัดเลือกล็อกของระบบที่เกี่ยวข้องกับพฤติกรรมของผู้บุกรุก เพื่อนำมาวิเคราะห์ต่อไป

2. *ส่วนวิเคราะห์พฤติกรรมผู้ใช้* มีหน้าที่นำล็อกที่ได้คัดเลือกแล้วมาเรียงลำดับตามเวลาที่เกิดขึ้น และเปรียบเทียบรูปแบบพฤติกรรมของผู้ใช้ว่ามีรูปแบบเหมือนกับพฤติกรรมการบุกรุกที่ได้ศึกษามา

3. *ส่วนตอบสนองการวิเคราะห์* มีหน้าที่รายงานผลการวิเคราะห์พฤติกรรมว่าผู้ใดมีพฤติกรรมเป็นผู้บุกรุกระบบบ้าง มีการแสดงผลการวิเคราะห์ทางหน้าจอและบันทึกลงไฟล์



รูปที่ 3-2 โฟลว์ชาร์ตการทำงานส่วนตรวจจับการบุกรุกโดยเปรียบเทียบพฤติกรรมผู้ใช้กับรูปแบบการบุกรุกที่รู้จัก

3.2.1 ส่วนตรวจสอบพฤติกรรมผู้ใช้

การตรวจจับการบุกรุกโดยเปรียบเทียบพฤติกรรมผู้ใช้กับรูปแบบการบุกรุกที่รู้จัก ได้ออกแบบระบบให้ตรวจสอบพฤติกรรมทั้งของระบบเองและของผู้ใช้รวมทั้งหมด 25 พฤติกรรม ซึ่งมีรายละเอียดดังต่อไปนี้

พฤติกรรมที่ 1 พยายามเข้าใช้ระบบโดยใช้ชื่อผู้ใช้ที่ไม่มีอยู่จริง (Bad username)

มีผู้พยายามล็อกอินเข้าสู่ระบบโดยใช้ชื่อผู้ใช้ที่ไม่มีอยู่ในระบบ ผู้บุกรุกที่ไม่มีแอ็คเคาท์ของระบบมักพยายามบุกรุกเข้ามาในระบบโดยการเดาชื่อผู้ใช้และรหัสผ่านในระบบที่เป็นไปได้ โดยสมมติฐานว่าในระบบมีชื่อผู้ใช้ที่มีอยู่ทั่วไป เช่น joe, jack, ann, joy เป็นต้น

พฤติกรรมที่ 2 พยายามเข้าใช้ระบบโดยใส่รหัสผ่านไม่ถูกต้อง (Bad password)

ผู้บุกรุกอาจทราบชื่อผู้ใช้มาโดยวิธีใดวิธีหนึ่ง และพยายามล็อกอินโดยการคาดเดารหัสผ่านที่เป็นไปได้ ลักษณะรหัสผ่านที่ผู้บุกรุกคาดเดา เช่น

- ตัวอักษรง่าย ๆ เรียงกัน เช่น abc123
- รหัสผ่านเหมือนชื่อผู้ใช้ เช่น ผู้ใช้ชื่อ jack ใช้รหัสผ่าน jack เช่นกัน ลักษณะการตั้งรหัสผ่านแบบนี้มีชื่อเรียกเฉพาะว่า joe account
- เรียงอักษรชื่อผู้ใช้ใหม่ เช่น ผู้ใช้ชื่อ adisak ใช้รหัสผ่าน kasida
- เป็นคำที่มีอยู่พจนานุกรม เช่น password table mirror
- หากผู้บุกรุกรู้จักกับผู้ใช้คนนั้นอาจเดารหัสผ่านโดยใช้ชื่อบุคคล สถานที่ สิ่งของ ที่เกี่ยวข้องกับผู้ใช้ เช่น ชื่อบิดา มารดา วันเกิด เป็นต้น

พฤติกรรมที่ 3 พยายามเข้าใช้งานในชื่อของผู้ดูแลระบบที่ไม่สำเร็จ (Failed root login)

ปรกติในระบบต้องมีผู้ใช้ชื่อ root อยู่ ผู้บุกรุกอาจพยายามล็อกอินโดยการเดารหัสผ่าน วิธีการเดารหัสผ่านที่ผู้บุกรุกใช้มีลักษณะเหมือนกับที่เสนอไปแล้วในพฤติกรรมที่ 2

พฤติกรรมที่ 4 พยายามเปลี่ยนสิทธิ์เป็นรูตที่ไม่สำเร็จ (Failed su root)

ผู้ใช้พยายามใช้คำสั่ง su เพื่อให้ตัวเองได้สิทธิ์เป็นรูตแต่ไม่สำเร็จหลาย ๆ ครั้ง เป็นไปได้ว่าผู้ใช้พยายามเปลี่ยนตัวเองให้เป็นรูตโดยไม่มีสิทธิ์โดยการเดารหัสผ่านของรูต

พฤติกรรมที่ 5 มีการเข้าใช้งานของผู้ดูแลระบบ (Root login)

ผู้ใช้ชื่อ root ล็อกอินเข้าสู่ระบบ

พฤติกรรมที่ 6 นโยบายในการตรวจสอบระบบเปลี่ยนแปลง (Audit policy changed)

โดยปรกติแล้วไฟล์ที่เก็บนโยบายในการตรวจสอบระบบคือ syslogd.conf ซึ่งควรมีการเปลี่ยนแปลงโดยผู้ดูแลระบบเท่านั้น ข้อมูลของ syslog สำหรับระบบตรวจจับผู้บุกรุกที่ออกแบบไว้เก็บอยู่ในไฟล์ KeyfileInfo โดยการเรียกสคริปต์ permsetup ดังนั้นหากผู้ดูแลระบบมั่นใจว่าได้ทำตามขั้นตอนของระบบตรวจจับผู้บุกรุกที่กำหนดไว้ กล่าวคือเรียกสคริปต์ permsetup ทุกครั้งหลังจากแก้ไขนโยบายในการตรวจสอบระบบ (syslogd.conf) แล้วระบบตรวจจับผู้บุกรุกพบว่ามีการแก้ไขนโยบายการตรวจสอบระบบแสดงว่าการแก้ไขนั้นเกิดจากผู้บุกรุกที่บุกรุกเข้ามาในระบบและพยายามซ่อนร่องรอยของตนเองไม่ให้ระบบบันทึกการกระทำในการเข้ามาครั้งต่อไป

พฤติกรรมที่ 7 มีการเรียกใช้บริการสำหรับผู้ที่มีสิทธิ์พิเศษในระบบ (Calls to privileged services)

บริการของระบบบางอย่างสามารถเรียกใช้โดยผู้ดูแลระบบเท่านั้น เช่น mount , swap เมื่อผู้บุกรุกสามารถบุกรุกเข้ามาในระบบและได้สิทธิ์เป็นรูตแล้วมักเรียกใช้บริการเหล่านี้

พฤติกรรมที่ 8 มีการเปลี่ยนแปลงกลุ่มผู้ใช้ระบบ (Group changed)

พฤติกรรมที่ 9 มีการสร้างกลุ่มผู้ใช้ระบบกลุ่มใหม่ (Group created)

พฤติกรรมที่ 10 มีการลบกลุ่มผู้ใช้ระบบ (Group deleted)

หลังจากบุกรุกระบบสำเร็จแล้วพบว่าผู้บุกรุกอาจทำการเปลี่ยนแปลงกลุ่มผู้ใช้ระบบ เช่น เปลี่ยนกลุ่มผู้ใช้ของตนเอง สร้างกลุ่มผู้ใช้ใหม่ หรือลบกลุ่มผู้ใช้เพื่อกลบเกลื่อนร่องรอย หรือเพื่อทำความเสียหายให้ระบบ

พฤติกรรมที่ 11 มีการสร้างชื่อผู้ใช้งานที่มีสิทธิ์เป็นผู้ดูแลระบบ (New root user created)

พฤติกรรมของผู้บุกรุกที่ได้สิทธิ์เป็นรูตมักสร้างผู้ใช้ใหม่ในกลุ่มรูตขึ้นในระบบ ไฟล์สำคัญบางไฟล์และบริการบางอย่างสามารถใช้งานได้โดยผู้ใช้ในกลุ่มรูต(ขึ้นกับเพอร์มิสชันที่ตั้งไว้) ดังนั้นผู้บุกรุกไม่จำเป็นต้องล็อกอินเข้ามาในระบบโดยใช้ชื่อ root อีกต่อไป เพื่อลดความเสี่ยงในการถูกตรวจพบ เช่น กรณีที่มีผู้ใช้จำนวนมาก ผู้ดูแลระบบไม่สามารถตรวจสอบได้ทั่วถึง หากผู้บุกรุกพบว่ามีผู้ใช้เดิมชื่อ paul ก็อาจเพิ่มผู้ใช้ใหม่ชื่อ paul อยู่ในกลุ่มรูต หลังจากนี้ผู้บุกรุกสามารถล็อกอินเข้าสู่ระบบโดยใช้ชื่อ paul และสามารถเรียกใช้ไฟล์สำคัญบางไฟล์และบริการบางอย่างที่ยอมให้ผู้ใช้ในกลุ่มรูตเข้าใช้ได้โดยที่ผู้ดูแลระบบไม่ทราบหากไม่สังเกตให้ดี ต่างจากการล็อกอินโดยใช้ชื่อ root โดยตรง ผู้ดูแลระบบสามารถสังเกตได้ง่ายเนื่องจากเมื่อล็อกอินชื่อ root ไฟล์ล็อกเก็บเป็นอักษรพิมพ์ใหญ่ และพบเห็นความผิดปรกติได้ทันทีหากพบการล็อกอินในช่วงที่ผู้เป็นรูตไม่อยู่

พฤติกรรมที่ 12 มีการเปลี่ยนแปลงสิทธิ์ของผู้ใช้ให้มีสิทธิ์เพิ่มขึ้น (User privileges granted)

เป็นการตรวจสอบว่ามีการเปลี่ยนสถานะของผู้ใช้จากกลุ่มใด ๆ ให้เข้ามาอยู่ในกลุ่มรูต บางครั้งการสร้างผู้ใช้ใหม่ในกลุ่มรูตทำให้ผู้ดูแลระบบผิดสังเกตได้ง่าย เนื่องจากในไฟล์ passwd สามารถเห็นความแตกต่างได้ชัดเจนระหว่างผู้ใช้ธรรมดาซึ่งตามปรกติมีเลขกลุ่มเป็น 1000 กับผู้ใช้ในกลุ่มของรูตซึ่งมีเลขกลุ่มเป็น 0 ผู้บุกรุกอาจเพิ่มผู้ใช้ปรกติขึ้นในระบบก่อนแล้วจึงเพิ่มสิทธิ์ของผู้ใช้คนนั้น หรืออีกกรณีหนึ่งคือผู้บุกรุกเห็นว่าการเพิ่มผู้ใช้เข้าไปในระบบสามารถพบเห็นได้ง่าย อาจขโมยแอ็กเคาท์ของผู้ใช้ที่มีอยู่แล้วและเพิ่มสิทธิ์ของผู้ใช้คนนั้นให้อยู่ในกลุ่มรูตต่อไป

พฤติกรรมที่ 13 ล็อกของระบบเปลี่ยนแปลง (Audit log changed)

ตรวจพบว่าข้อมูลของไฟล์ล็อกในระบบมีการเปลี่ยนแปลง

พฤติกรรมที่ 14 ไฟล์สำคัญเปลี่ยนแปลง (Key file changed)

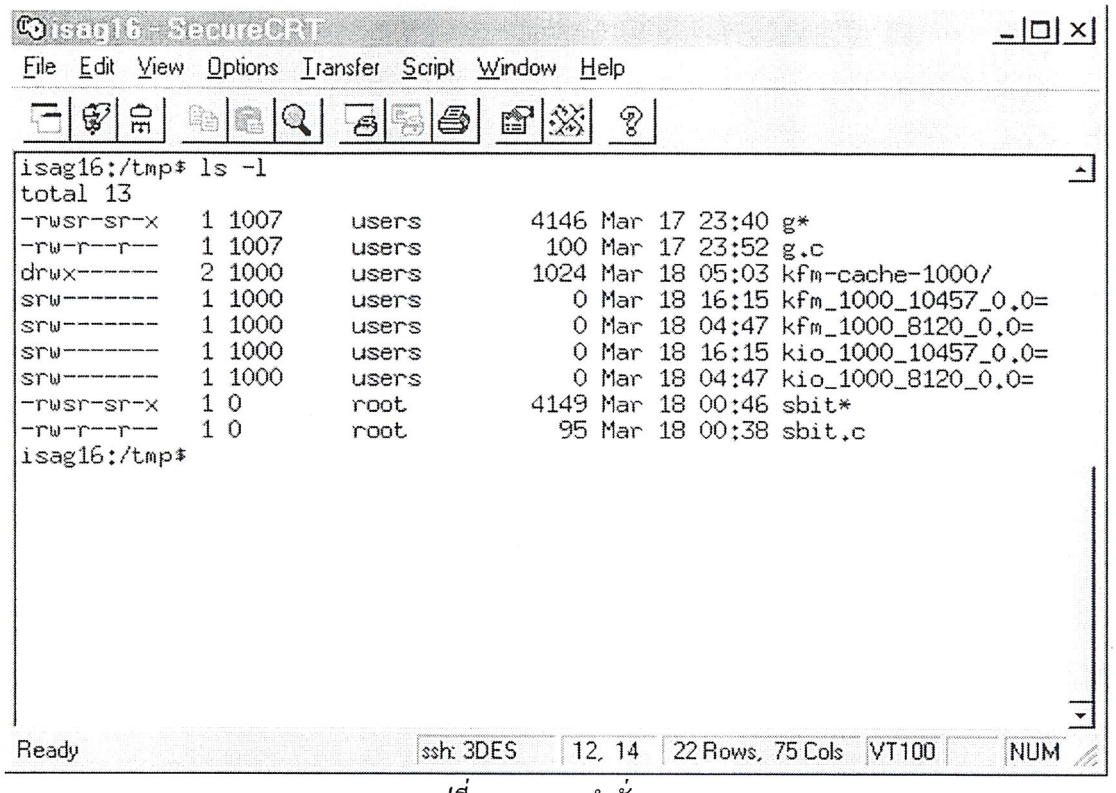
พบว่าไฟล์สำคัญของระบบเปลี่ยนแปลง ซึ่งเป็นไฟล์ที่มีรายชื่ออยู่ในไฟล์ชื่อ keyfiles

พฤติกรรมที่ 15 ไฟล์ SUID เปลี่ยนแปลง (SUID file changed)

พบว่าไฟล์ที่ยอมให้มีการเปลี่ยนหมายเลขผู้ใช้ (SUID file) มีการเปลี่ยนแปลง ไฟล์ที่ยอมให้มีการเปลี่ยนหมายเลขผู้ใช้คือไฟล์ที่มีการตั้งบิตเอส (s-bit) ไว้ ซึ่งดูได้จากการใช้คำสั่ง

```
%ls -l
```

ดังแสดงในรูปที่ 3-3



รูปที่ 3-3 ผลของคำสั่ง ls -l

จากรูปพบว่าไฟล์ชื่อ g และ sbit มีการตั้งค่าเพอร์มิสชันเป็น

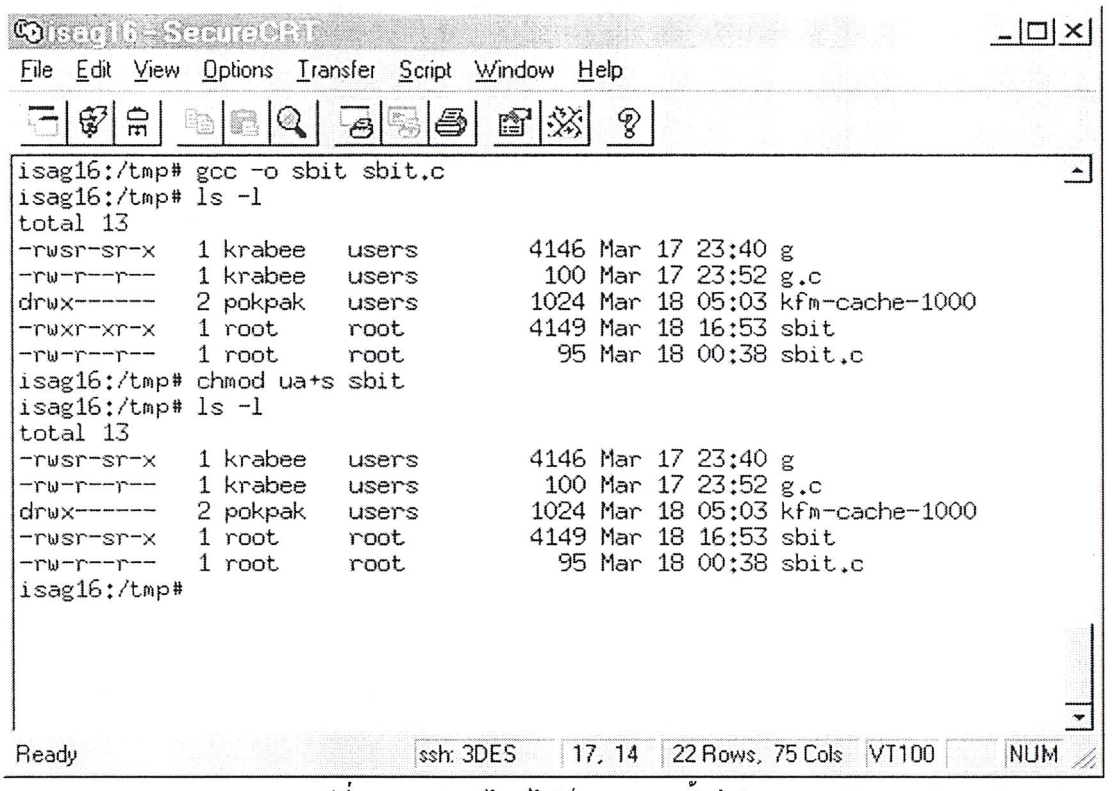
```
-rwsr-sr-x   1 1007      users          4146 Mar 17 23:40 g*
-rwsr-sr-x   1 0         root           4149 Mar 18 00:46 sbit*
```

แสดงว่าสองไฟล์นี้ยอมให้มีการเปลี่ยนหมายเลขผู้ที่เรียกใช้ไฟล์นี้ได้ ตัวอย่างต่อไปนี้แสดงให้เห็นการเปลี่ยนหมายเลขผู้ใช้ซึ่งเป็นทางหนึ่งที่ผู้บุกรุกใช้

เริ่มสร้างไฟล์ sbit.c เป็นดังนี้

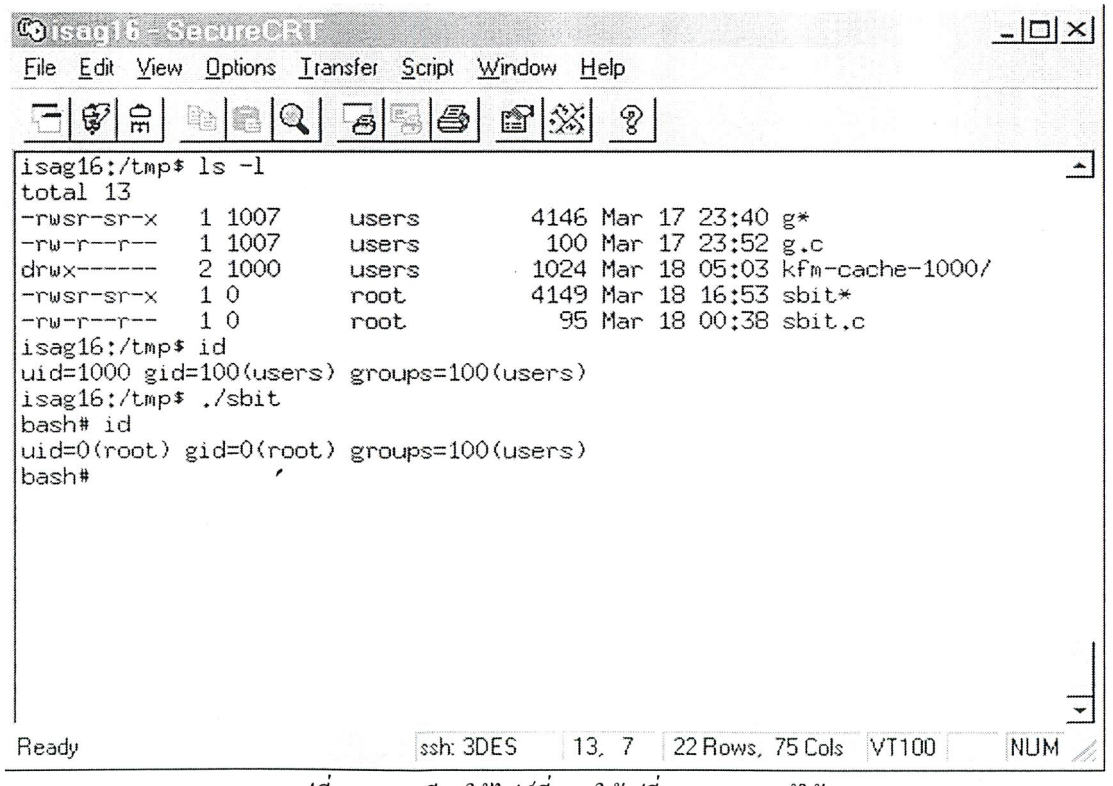
```
#include <unistd.h>
#include <sys/types.h>
main(){
setuid(0);
setgid(0);
system("/bin/sh");
}
```

หลังจากนั้นคอมไพล์ไฟล์นี้และตั้งค่าบิตเอส ดังรูปที่ 3-4



รูปที่ 3-4 การคอมไพล์ไฟล์ sbit.c และตั้งค่าบิตเอส

หากมีผู้ใช้เรียกไฟล์นี้ ผู้ใช้คนนั้นสามารถเปลี่ยนหมายเลขผู้ใช้ของตนให้เป็น 0 และเปลี่ยนหมายเลขกลุ่มเป็น 0 คือกลุ่มรูตได้ ดังแสดงในรูปที่ 3-5



รูปที่ 3-5 การเรียกใช้ไฟล์ที่ยอมให้เปลี่ยนหมายเลขผู้ใช้

ดังนั้นหากมีการเปลี่ยนแปลงไฟล์ที่ตั้งค่าบิตเอสนี้ สามารถแสดงถึงความไม่ปลอดภัยของไฟล์นั้น ๆ รายชื่อไฟล์เหล่านี้ได้จากคำสั่ง

```
#find / -perm -4000
```

ซึ่งได้ออกแบบให้เก็บรายชื่อไฟล์เหล่านี้ไว้ในไฟล์ชื่อ suidfiles

พฤติกรรมที่ 16 ผู้ใช้เข้าใช้งานระบบ (User login)

ผู้ใช้ล็อกอินสำเร็จ

พฤติกรรมที่ 17 ระบบเริ่มการทำงานใหม่ (System restarted)

มีการเริ่มทำงานของระบบ

พฤติกรรมที่ 18 ปิดการทำงานระบบ (System shutdown)

มีการปิดระบบ

พฤติกรรมที่ 19 มีการเปลี่ยนแปลงข้อมูลในล็อกของระบบ (Find evidence may be edited log)

ระบบตรวจจับผู้บุกรุกตรวจสอบข้อมูลใน syslog และ sulog หากพบว่าข้อมูลจากไฟล์ล็อกทั้งสองไม่ตรงกัน ระบบตรวจจับผู้บุกรุกจะระบุว่าผู้ใช้ที่เปลี่ยนสิทธิ์ตนเองหรือพยายามเปลี่ยนสิทธิ์แต่ไม่สำเร็จนั้นเป็นผู้บุกรุกทันที

พฤติกรรมนี้สามารถเกิดได้ 4 กรณีคือ

- กรณีที่ 1 ผู้ใช้สามารถเปลี่ยนสิทธิ์เป็นรูตได้สำเร็จซึ่งมีการบันทึกใน sulog แต่ไม่มีการบันทึกใน syslog ระบบตรวจจับผู้บุกรุกระบุว่าผู้ใช้คนนั้นเป็นผู้บุกรุก
- กรณีที่ 2 ผู้ใช้สามารถเปลี่ยนสิทธิ์เป็นรูตได้สำเร็จซึ่งมีการบันทึกใน syslog แต่ไม่มีการบันทึกใน sulog ระบบตรวจจับผู้บุกรุกระบุว่าผู้ใช้คนนั้นเป็นผู้บุกรุก
- กรณีที่ 3 ผู้ใช้เปลี่ยนสิทธิ์เป็นรูตไม่สำเร็จซึ่งมีการบันทึกใน sulog แต่ไม่มีการบันทึกใน syslog ระบบตรวจจับผู้บุกรุกระบุว่าผู้ใช้คนนั้นเป็นผู้บุกรุก
- กรณีที่ 4 ผู้ใช้เปลี่ยนสิทธิ์เป็นรูตไม่สำเร็จซึ่งมีการบันทึกใน syslog แต่ไม่มีการบันทึกใน sulog ไม่สามารถระบุได้ว่าใครเป็นผู้บุกรุก

พฤติกรรมที่ 20 มีการสร้างชื่อผู้ใช้งานระบบใหม่ (User account created)

เมื่อผู้บุกรุกเข้ามาในระบบได้แล้ว มักสร้างประตูหลัง (Backdoor) ไว้ในระบบ เพื่อเป็นช่องทางในการเข้าระบบต่อไป ดังนั้นหากพบว่ามีผู้ต้องสงสัยที่อยู่ในสถานะรูต และมีการสร้างชื่อผู้ใช้งานใหม่ขึ้น เป็นไปได้ว่าผู้ต้องสงสัยนั้นกำลังสร้างช่องทางดังกล่าวอยู่

พฤติกรรมที่ 21 มีการลบชื่อผู้ใช้งานระบบ (User account deleted)

หากผู้บุกรุกต้องการลบร่องรอยของตน เช่น มีการสร้างผู้ใช้ไว้ในระบบ ผู้บุกรุกมักลบชื่อผู้ใช้ที่ตนไม่ต้องการใช้งานแล้ว เพื่อไม่ให้ผู้ดูแลระบบติดตามได้ นอกจากนี้พฤติกรรมนี้ยังอาจเกิดขึ้นเนื่องจากผู้บุกรุกต้องการสร้างความเสียหายให้แก่ระบบหรือผู้ใช้คนใดคนหนึ่ง

พฤติกรรมที่ 22 มีการเปลี่ยนแปลงรหัสผ่าน (User password changed)

พบการเปลี่ยนแปลงรหัสผ่าน ซึ่งอาจเป็นเพราะผู้ใช้เปลี่ยนแปลงรหัสผ่านตามปรกติหรือมีผู้บุกรุกต้องการขโมยแอ็กเคาท์ของผู้ใช้คนใดคนหนึ่งในระบบเพื่อทำเป็นประตูหลัง ต่อไป

พฤติกรรมที่ 23 มีการติดต่อกับระบบ (Connection start)

โฮสต์ใด ๆ ขอติดต่อ กับระบบ

พฤติกรรมที่ 24 มีการเปลี่ยนสิทธิ์เป็นรูตได้สำเร็จ (User success su command)

ผู้ใช้สามารถเปลี่ยนสิทธิ์ตนเองเป็นรูตได้

พฤติกรรมที่ 25 ผู้ใช้เลิกการใช้งานระบบ (User logout)

ผู้ใช้ล็อกเอาต์ออกจากระบบ

3.2.2 ส่วนวิเคราะห์พฤติกรรมผู้ใช้

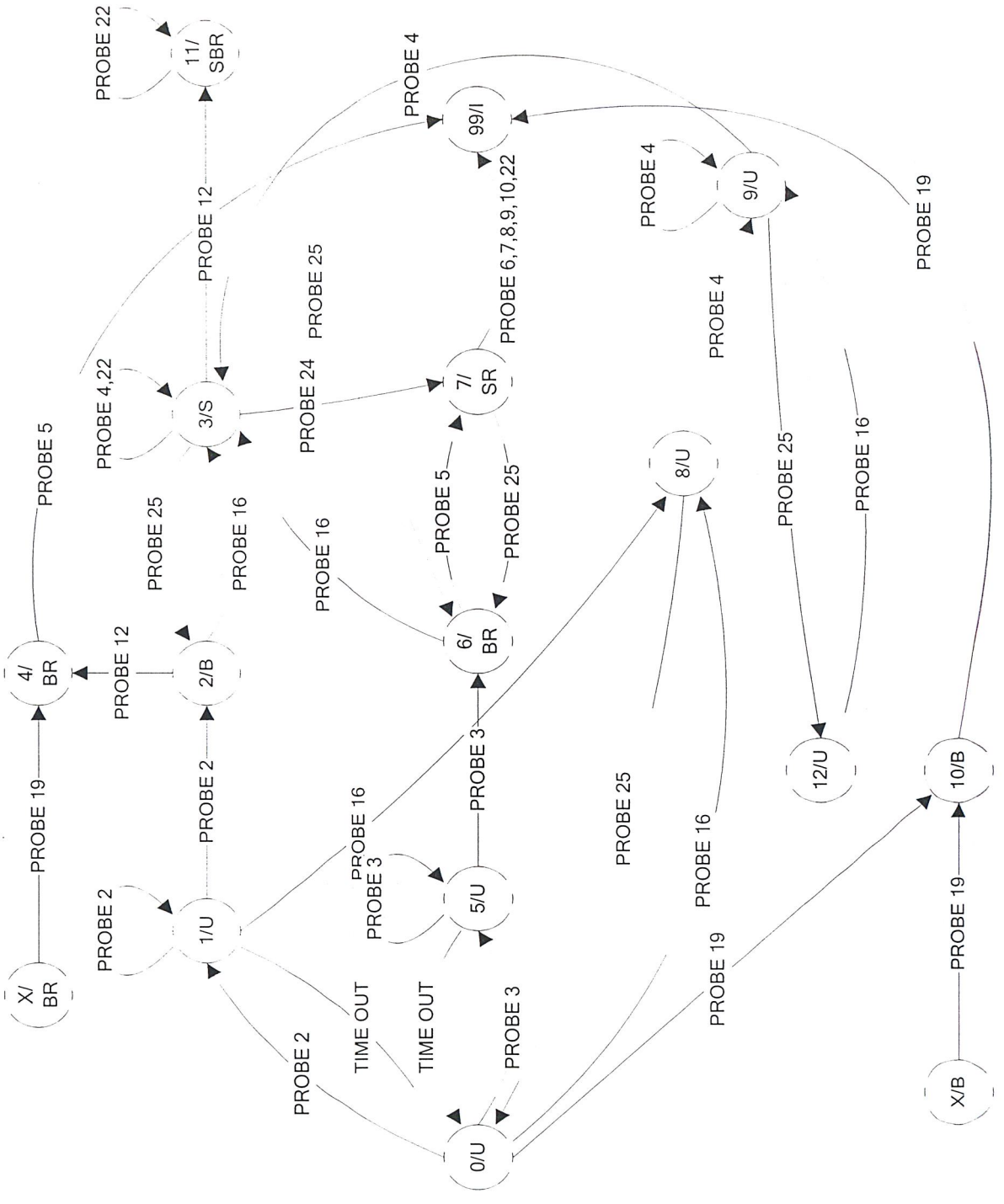
การวิเคราะห์ข้อมูลล็อกสำหรับส่วนตรวจจับการบุกรุกโดยเปรียบเทียบพฤติกรรมผู้ใช้กับรูปแบบการบุกรุกที่รู้จักได้เลือกใช้วิธีการใช้สเตตไดอะแกรม เนื่องจาก

- การวิเคราะห์สามารถมองเห็นภาพเป็นรูปธรรมง่ายกว่าวิธีวิเคราะห์โดยใช้กฎ เพราะเขียนเป็นไดอะแกรมได้
- การแก้ไขส่วนวิเคราะห์ทำได้ง่ายกว่า
- ลีนุกซ์มีเครื่องมือในการพัฒนาแอพพลิเคชันแบบนี้หลากหลายกว่า สามารถพัฒนาบนภาษาคอมพิวเตอร์ที่มีลักษณะเป็นโพรซีเจอร์ซึ่งมีอยู่มากมายได้ เช่น ภาษาปาสคาล (Pascal) หรือภาษาซีได้ ต่างจากแบบใช้กฎที่ต้องพัฒนาบนภาษาที่การทำงานแบบลอจิก (Logic) เช่น ภาษาโปรลอกหรือภาษาลิส์ป

การออกแบบได้แบ่งสถานะต่าง ๆ ของผู้ใช้ไว้ 8 สถานะ รายละเอียดตามตารางที่ 3-5

ตัวย่อ	*ลักษณะของผู้ใช้*
U (User)	ผู้ใช้ปรกติ
B (Black user)	ผู้ใช้ที่บันทึกชื่อลง blackuser แล้วแต่ไม่ได้ล็อกอิน ซึ่งผู้ใช้คนนี้ยังไม่เคยได้สิทธิ์รูตมาก่อน
BR (Black Root)	ผู้ใช้ที่บันทึกชื่อลง blackroot แล้ว แต่ไม่ได้ล็อกอิน เป็นผู้ใช้ที่ได้สิทธิ์ของรูตหรือเคยได้สิทธิ์ของรูตโดยไม่จำเป็นว่าผู้ใช้นั้นอยู่ในกลุ่มรูตหรือไม่
S (Suspect user)	ผู้ใช้ที่บันทึกชื่อลง blackuser แล้วและกำลังล็อกอินอยู่ เป็นผู้ใช้ที่ต้องจับตาดูพฤติกรรม
SR (Suspect root)	ผู้ใช้ที่บันทึกชื่อลง blackroot แล้วและกำลังล็อกอินอยู่ เป็นผู้ใช้ที่ต้องจับตาดูพฤติกรรม
SBR (Suspect Black Root)	ผู้ใช้ที่กำลังล็อกอินอยู่ซึ่งเป็นผู้ใช้ที่ต้องจับตาดูพฤติกรรม และได้สิทธิ์เป็นรูตเมื่อล็อกอินครั้งต่อไป
I (Intruder)	ผู้บุกรุกระบบ

ตารางที่ 3-1 รายละเอียดของสถานะในสเตตไดอะแกรม



รูปที่ 3-6 สเตตไดอะแกรมของการวิเคราะห์ทั้งหมด

ภายในแต่ละสเตตซึ่งแทนด้วยรูปวงกลมมีการระบุชื่อสเตตและสถานะของผู้ใช้ ณ สเตตนั้น ๆ เช่น

1/U หมายถึง สเตตที่ 1 และสถานะของผู้ใช้เป็น U

X/BR หมายถึง สเตตใด ๆ ซึ่งผู้ใช้มีสถานะเป็น BR

การวิเคราะห์ที่ออกแบบสำหรับส่วนนี้แสดงไว้ในรูปที่ 3-6 ซึ่งสามารถนำมาแยกย่อยได้เป็นกรณีสำคัญ ๆ ดังต่อไปนี้

กรณีที่ 1

เมื่อผู้ใช้ที่สเตต 0 พิมพ์รหัสผ่านผิดพลาดครั้งแรก (พฤติกรรม 2) สเตตของผู้ใช้เปลี่ยนเป็น 1 และเพิ่มตัวนับที่ 2 (counter 2) ของผู้ใช้นั้นหนึ่งครั้ง เมื่อมีการพิมพ์รหัสผ่านผิดพลาดครั้งต่อไปตัวนับที่ 2 ถูกเพิ่มขึ้นทุกครั้ง แต่สเตตของผู้ใช้นั้นยังอยู่ที่สเตต 1 เหมือนเดิมจนกว่าตัวนับที่ 2 เท่ากับตัวแปร MAX_BAD_PASSWD สเตตของผู้ใช้จึงเปลี่ยนเป็นสเตต 2 พร้อมกับบันทึกชื่อผู้ใช้นั้นลงใน blackuser และสถานะของผู้ใช้เปลี่ยนจาก U เป็น S ซึ่งหากผู้ใช้คนนั้นล็อกอินเข้าสู่ระบบสำเร็จ (พฤติกรรมที่ 16) สถานะกลายเป็น S

ขณะที่ผู้ใช้อยู่ที่สถานะ 1 แล้วไม่พบว่ามีการใส่รหัสผ่านผิดพลาดเป็นระยะเวลามากกว่าเวลาที่ผู้ดูแลระบบกำหนดไว้ในตัวแปร MAX_DATE, MAX_MONTH, MAX_HOUR และ MAX_MINUTE (จำนวนวัน, เดือน, ชั่วโมง และนาที ตามลำดับ) ตัวนับที่ 2 ของผู้ใช้จะลดค่าลงไปหนึ่งครั้ง เมื่อตัวนับที่ 2 นี้ลดลงจนเป็น 0 สเตตของผู้ใช้เปลี่ยนเป็น 0 เช่นเดิม

กรณีที่ 2

เมื่อผู้ใช้ที่เป็นรูตใส่รหัสผ่านผิดพลาด (พฤติกรรมที่ 3) การวิเคราะห์เหมือนกับในกรณีที่ 1 เพียงแต่เปรียบเทียบจำนวนครั้งที่ใส่รหัสผ่านผิดพลาดกับตัวแปร MAX_BAD_ROOT_PASSWD ซึ่งควรมีค่าน้อยกว่าตัวแปร MAX_BAD_PASSWD เนื่องจากการล็อกอินของรูตต้องละเอียดกว่าผู้ใช้ธรรมดา และตัวนับที่ใช้เป็นตัวนับที่ 3

ถ้าจำนวนครั้งที่ใส่รหัสผิดพลาดมากกว่าตัวแปร MAX_BAD_ROOT_PASSWD สถานะของผู้ใช้เปลี่ยนเป็น BR และสถานะกลายเป็น SR หากสามารถล็อกอินได้สำเร็จ (พฤติกรรมที่ 5)

กรณีที่ 3

หากผู้ใช้ที่ล็อกอินสำเร็จ พยายามใช้คำสั่ง su แต่พิมพ์รหัสผ่านไม่ถูกต้อง การวิเคราะห์เช่นเดียวกับกรณีที่ 1 และ 2

ขณะผู้ใช้อยู่ที่สเตต 9 สเตตเปลี่ยนเป็น 12 เมื่อล็อกเอาต์ออกจากระบบ และกลับมาเป็น 9 เมื่อล็อกอินเข้ามาใหม่

กรณีที่ 4

เมื่อผู้ใช้ที่อยู่ในสถานะ B หรือ BR สามารถล็อกอินเข้าสู่ระบบได้สำเร็จ (พฤติกรรมที่ 12 หรือ 5) สถานะเปลี่ยนจาก B เป็น S หรือเปลี่ยนจาก BR เป็น SR และสเตตเปลี่ยนจาก 2 เป็น 3 หรือจาก 6 เป็น 7 ถ้าผู้ใช้ไม่มีพฤติกรรมอื่นที่ทำให้สเตตเปลี่ยนไป เมื่อล็อกเอาต์ออกจากระบบ สเตตจะเปลี่ยนกลับมาเป็น 2 หรือ 3 และ สถานะเป็น B หรือ BR เช่นเดิม

กรณีที่ 5

ผู้ใช้ที่อยู่ในสถานะ S ได้รับการเพิ่มสิทธิ์ (พฤติกรรมที่ 12) สถานะเปลี่ยนจาก S เป็น SBR และเปลี่ยนสเตตไปเป็น 11

ส่วนผู้ใช้ที่อยู่ในสถานะ B เมื่อได้รับการเพิ่มสิทธิ์ สถานะเปลี่ยนจาก B เป็น BR ซึ่งมีสเตตเป็น 4 ถ้าผู้ใช้ผู้นี้ล็อกอินได้สำเร็จ สถานะกลายเป็น I และสเตตเป็น 99 ทันที

กรณีที่ 6

ผู้ใช้ที่มีสถานะเป็น S ใช้คำสั่ง su เพื่อเปลี่ยนสิทธ์ตนเองเป็นรูตได้สำเร็จ สถานะกลายเป็น SR

กรณีที่ 7

ผู้ใช้ที่มีสถานะเป็น SR และมีกระทำการอย่างใดอย่างหนึ่งต่อไปนี้ สถานะเปลี่ยนจาก SR เป็น I และเปลี่ยนสเตตเป็น 99 ทันที

- มีการเรียกใช้บริการสำหรับผู้ที่มีสิทธิ์พิเศษในระบบ (Calls to privileged services)
- มีการเปลี่ยนแปลงกลุ่มผู้ใช้ระบบ (Group changed)
- มีการสร้างกลุ่มผู้ใช้ระบบกลุ่มใหม่ (Group created)
- มีการลบกลุ่มผู้ใช้ระบบ (Group deleted)
- มีการเปลี่ยนแปลงรหัสผ่าน (User password changed)

กรณีที่ 8

หากพบว่าข้อมูลในไฟล์ล็อกของระบบถูกแก้ไข (พฤติกรรมที่ 19) และสามารถระบุได้ว่าเป็นบันทึกการกระทำของผู้ใช้คนใด ผู้ใช้คนนั้นเปลี่ยนสถานะเป็น I ทันที

3.2.3 ส่วนตอบสนองการวิเคราะห์

เมื่อระบบตรวจจับผู้บุกรุกตรวจพบพฤติกรรมที่น่าสงสัยว่าเป็นการบุกรุกหรือพยายามบุกรุกระบบในรูปแบบที่ได้ศึกษาไว้ ระบบตรวจจับผู้บุกรุกมีการตอบสนองการวิเคราะห์โดยบันทึกลงในไฟล์ดังต่อไปนี้

blackhost

ไฟล์เก็บรายชื่อโฮสต์ที่ผู้ดูแลระบบต้องตรวจสอบเป็นพิเศษ ระบบตรวจจับผู้บุกรุกบันทึกโฮสต์ใด ๆ ลงในไฟล์นี้ก็ต่อเมื่อตรวจพบว่ามีผู้พยายามล็อกอินโดยใช้ชื่อผู้ใช้ที่ไม่มีอยู่ในระบบจากโฮสต์นี้เป็นจำนวนครั้งเกินกว่าที่ผู้ดูแลระบบกำหนดไว้ที่ตัวแปร MAX_BAD_USER ใน midetect.cpp ส่วน Max Value Configuration

blackuser

ไฟล์เก็บรายชื่อผู้ใช้ที่ผู้ดูแลระบบต้องตรวจสอบพฤติกรรมเป็นพิเศษ ระบบตรวจจับผู้บุกรุกบันทึกชื่อผู้ใช้ใด ๆ ลงในไฟล์นี้ก็ต่อเมื่อพบว่าผู้ใช้คนนั้นมีการกระทำดังต่อไปนี้อย่างใดอย่างหนึ่งหรือหลายอย่าง

- ใส่รหัสผ่านผิดเกินจำนวนครั้งที่ผู้ดูแลระบบกำหนดไว้ในตัวแปร MAX_BAD_PASSWD ใน midetect.cpp ส่วน Max Value Configuration
- ล็อกเอาต์ออกจากระบบในขณะที่มีสถานะเป็น S และอยู่ในสเตตที่ 3

blackroot

ไฟล์เก็บรายชื่อผู้ใช้ซึ่งได้สิทธิ์เป็นรูตหรือเคยได้สิทธิ์เป็นรูตและมีพฤติกรรมน่าสงสัย ผู้ใช้ที่ระบบตรวจจับผู้บุกรุกบันทึกชื่อลงในไฟล์นี้ก็ต่อเมื่อผู้ใช้มีพฤติกรรมดังต่อไปนี้อย่างใดอย่างหนึ่งหรือหลายอย่าง

- ผู้ใช้ที่อยู่ในกลุ่มรูตใส่รหัสผ่านผิดพลาดเกินกว่าจำนวนครั้งที่ผู้ดูแลระบบกำหนดไว้ในตัวแปร MAX_BAD_ROOT_PASSWD ใน midetect.cpp ส่วน Max Value Configuration
- ล็อกเอาต์ออกจากระบบในขณะที่มีสถานะเป็น SR และอยู่ในสเตตที่ 7
- ล็อกเอาต์ออกจากระบบในขณะที่มีสถานะเป็น SBR และอยู่ในสเตตที่ 11
- ได้รับการเพิ่มสิทธิ์ (การย้ายไปอยู่ในกลุ่มรูต) ขณะที่มีสถานะเป็น B และอยู่ในสเตตที่ 2

ทั้งนี้ในการบันทึกใด ๆ ลงในไฟล์ ระบบตรวจจับผู้บุกรุกจะแสดงผลทางจอภาพด้วย และมีการบันทึกชื่อผู้ใช้หรือชื่อโฮสต์ใด ๆ เพียงครั้งเดียว เพื่อสะดวกต่อการดูผลในภายหลัง ตัวอย่างแสดงไว้ในบทที่ 4

นอกจากการตอบสนองดังกล่าวแล้ว ระบบตรวจจับผู้บุกรุกยังมีการเตือนหากพบว่ามีไฟล์สำคัญในระบบถูกแก้ไขโดยไม่ทราบชื่อผู้แก้ไข เพื่อเตือนให้ผู้ดูแลระบบทราบว่าไฟล์นั้น ๆ อยู่ในสถานะที่ไม่ปลอดภัย ตัวอย่างแสดงไว้ในบทที่ 4

3.2.4 การเรียกใช้งานส่วนตรวจจับการบุกรุกโดยเปรียบเทียบพฤติกรรมผู้ใช้กับรูปแบบการบุกรุกที่รู้จัก

ก่อนเรียกใช้งานส่วนตรวจจับการบุกรุกโดยเปรียบเทียบพฤติกรรมผู้ใช้กับรูปแบบการบุกรุกที่รู้จัก ต้องทำตามขั้นตอนต่อไปนี้

1. ติดตั้งส่วนบันทึกล็อกเพิ่มเติม โดยติดตั้งแอพพลิเคชันสำหรับเก็บล็อกของการเปลี่ยนสิทธิ์สำเร็จ (setuid) ซึ่งปรกติถ้าผู้ใช้ใช้คำสั่ง su สำเร็จ syslog จะไม่เก็บล็อกของเหตุการณ์นี้ ฉะนั้นกรณีนี้หากผู้บุกรุกเข้าไปแก้ไข sulog ผู้ดูแลระบบก็ไม่สามารถทราบเลยว่ามีการใช้คำสั่ง su สำเร็จ ดังนั้นจึงจำเป็นต้องให้ระบบบันทึกล็อกสำหรับเหตุการณ์นี้เพิ่มเติม โดย

1.1 คอมไพล์ไฟล์ suidshow.c โดยใช้คำสั่งดังนี้

```
#gcc -c suidshow.c -o suidshow.o
```

ผลคือได้ไฟล์ suidshow.o ออกมา นำไฟล์นี้ไปไว้ที่ /ids

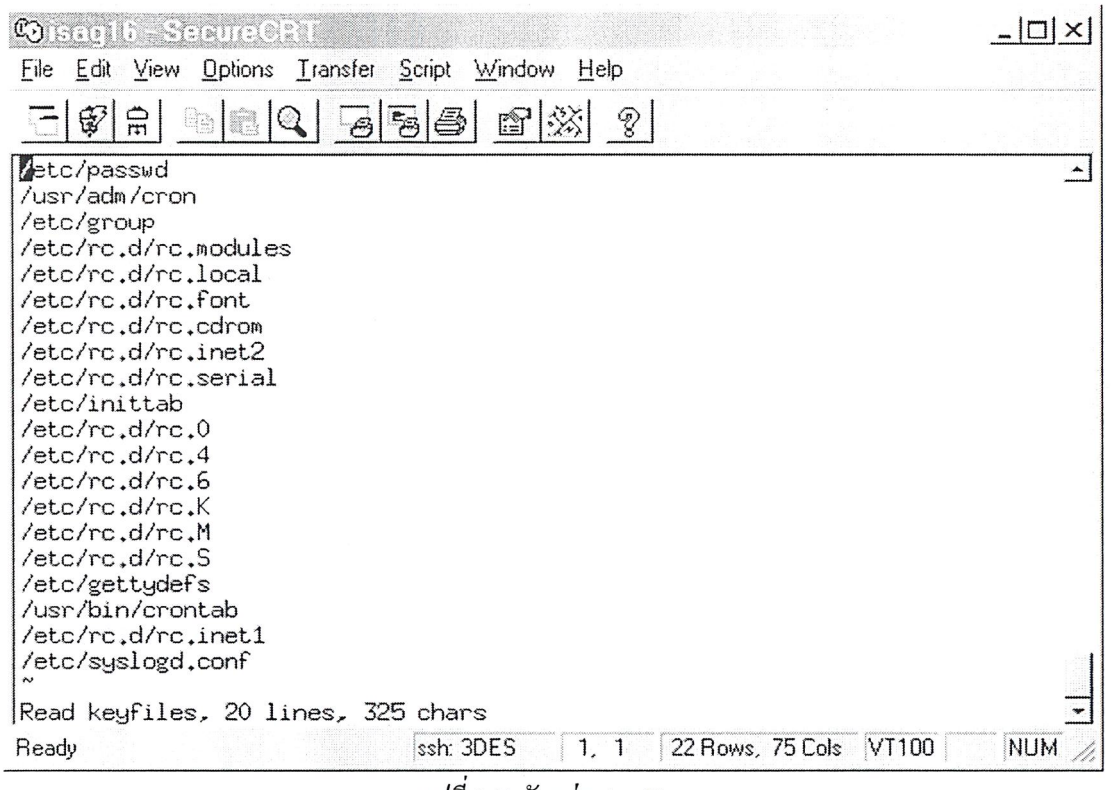
1.2 เพิ่มคำสั่งต่อไปนี้ลงในไฟล์ /etc/rc.d/rc.local เพื่อให้เริ่มทำงานเมื่อเปิดระบบ

```
#insmod /ids/suidshow.o
```

1.3 กำหนดไฟล์สำคัญที่ต้องการตรวจสอบ โดยใส่รายชื่อไฟล์สำคัญลงไปในไฟล์ต่อไปนี้

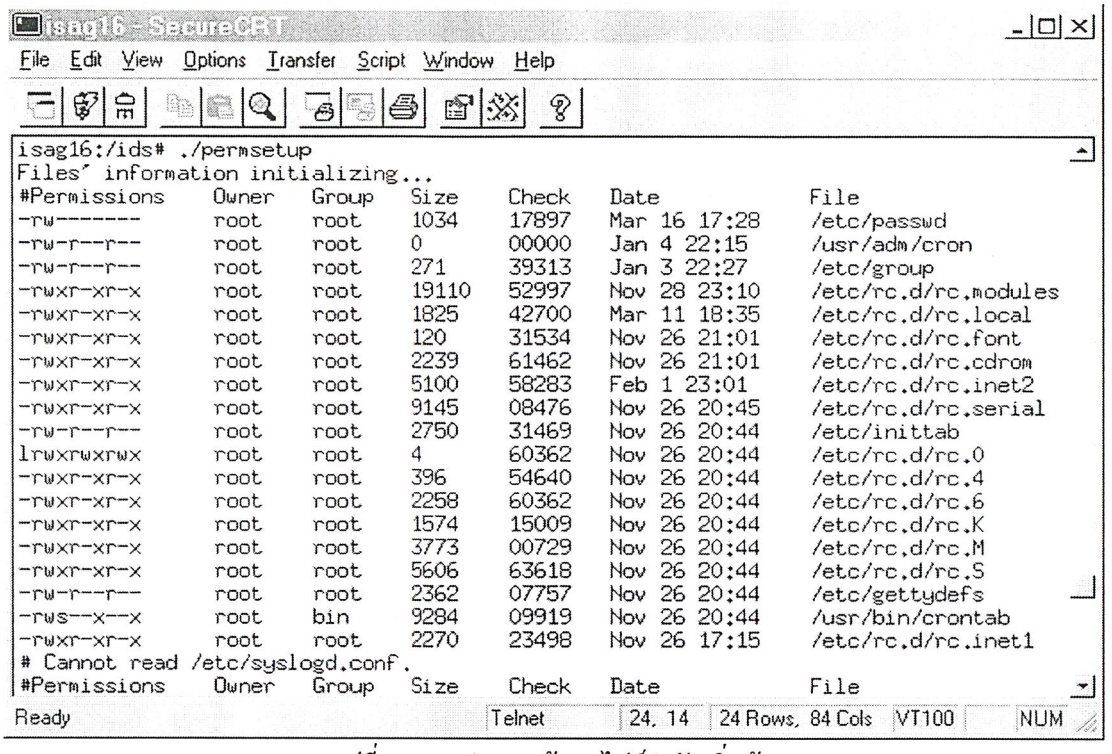
- keyfiles รายชื่อไฟล์สำคัญของระบบ ตัวอย่างเช่น /etc/passwd
- logfiles รายชื่อไฟล์ที่เป็นไฟล์ล็อกของระบบ เช่น sulog หรือ syslog

ตัวอย่างของไฟล์ keyfiles แสดงไว้ในรูปที่ 3-7



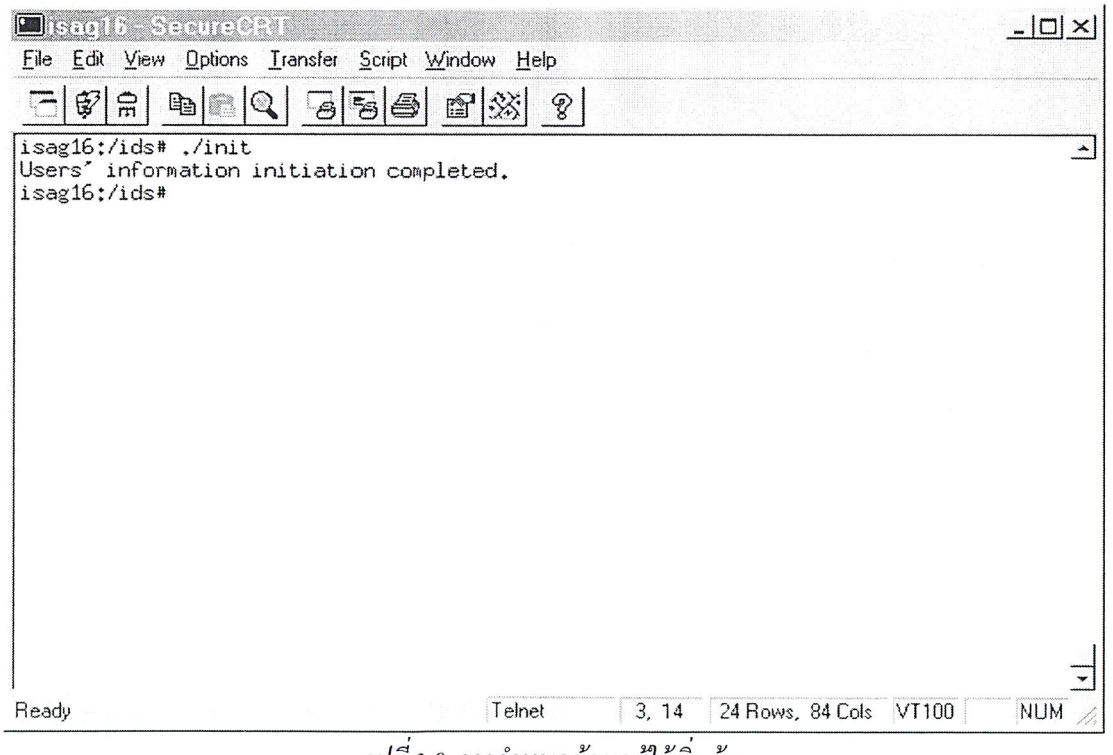
รูปที่ 3-7 ตัวอย่าง keyfiles

2. กำหนดข้อมูลไฟล์เริ่มต้น โดยเรียกใช้สคริปต์ permsetup เพื่อเก็บข้อมูลของไฟล์สำคัญลงไปในไฟล์ข้อมูล SuidInfo, KeyfileInfo, LogInfo ดังรูปที่ 3-8



รูปที่ 3-8 การกำหนดข้อมูลไฟล์สำคัญเริ่มต้น

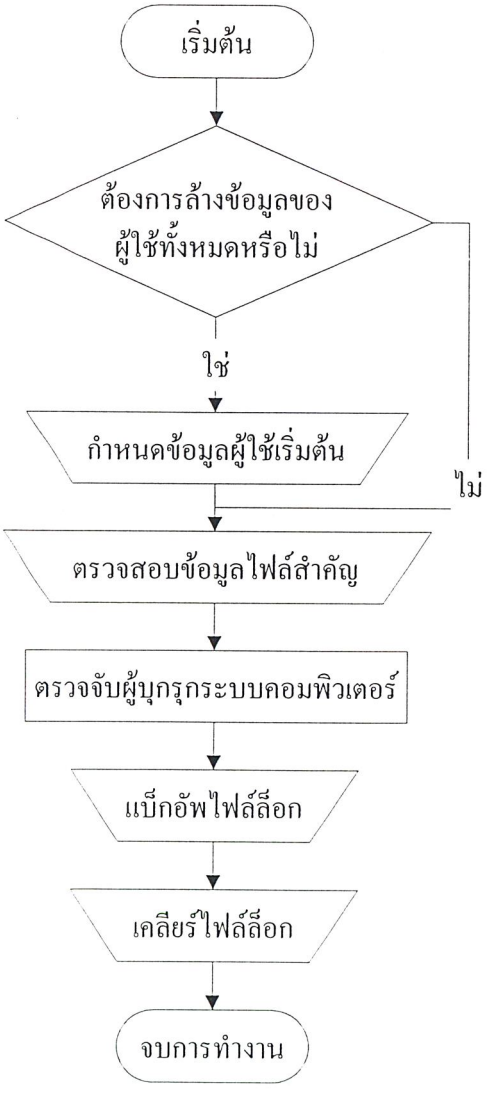
3. กำหนดข้อมูลผู้ใช้เริ่มต้น โดยเรียกใช้แอพพลิเคชัน init ดังรูปที่ 3-9



รูปที่ 3-9 การกำหนดข้อมูลผู้ใช้เริ่มต้น

ส่วนขั้นตอนของการเรียกใช้งานส่วนตรวจจับการบุกรุกโดยเปรียบเทียบพฤติกรรมผู้ใช้กับรูปแบบการบุกรุกที่รู้จักแสดงไว้ในรูป 3-10 ซึ่งมีรายละเอียดดังนี้

1. ถ้าผู้ดูแลระบบต้องการล้างข้อมูลผู้ใช้ทั้งหมด ให้ใช้แอพพลิเคชัน init ถ้าไม่ต้องการ ให้ข้ามขั้นตอนนี้ไป มิฉะนั้นข้อมูลผู้ใช้จากการวิเคราะห์ครั้งก่อนสูญหาย
2. ตรวจสอบข้อมูลไฟล์สำคัญ โดยเรียกใช้สคริปต์ permck
3. เรียกแอพพลิเคชัน midetect เพื่อเริ่มการทำงานของระบบตรวจจับผู้บุกรุก
4. แบ็กอัพไฟล์ล็อก ได้แก่ syslog, sulog, wtmp
5. เคลียร์ไฟล์ล็อกที่ได้แบ็กอัพไว้แล้วทั้งหมด



รูปที่ 3-10 โฟลว์ชาร์ตการเรียกใช้งาน

3.3 ส่วนตรวจจับการบุกรุกโดยตรวจสอบการใช้งานทรัพยากรระบบที่ผิดปรกติ (Anomaly Detection)

ส่วนตรวจจับการบุกรุกโดยตรวจสอบการใช้งานทรัพยากรระบบที่ผิดปรกติแบ่งเป็น 4 ส่วนหลัก ได้แก่

1. ส่วนการติดตั้งระบบ
2. ส่วนการเตรียมประวัติของผู้ใช้งานระบบ
3. ส่วนการตรวจจับพฤติกรรมของผู้ใช้ระบบ
4. ส่วนการแจ้งเตือนผู้ดูแลระบบ

การทำงานโดยรวมแสดงในรูปที่ 3-11

3.3.1 ส่วนการติดตั้งระบบ

ในส่วนนี้ต้องมีการกำหนดค่าบางอย่าง ซึ่งขึ้นอยู่กับนโยบายของบริษัทที่นำระบบไปใช้ เช่น ช่วงเวลาที่อนุญาตให้มีคนเข้าใช้ระบบ ในกรณีที่ระบบต้องออนไลน์ตลอดเวลา แต่อาจมีช่วงเวลาที่ไม่ควรมีผู้เข้าใช้ระบบ นอกจากบางคน เช่น ผู้ดูแลระบบ โดยค่าที่ต้องกำหนดแบ่งเป็น ช่วงเวลาทำงานปรกติ (DayIn-DayOut) ช่วงเวลาการทำงานล่วงเวลา (NightIn-NightOut) และช่วงเวลาวิกาล ที่ไม่ควรมีผู้เข้าใช้ระบบ(CriticalIn-CriticalOut) ปริมาณการใช้ซีพียูโดยเฉลี่ย (หน่วยเป็นเปอร์เซ็นต์ : %CPU) ปริมาณการใช้หน่วยความจำของระบบโดยเฉลี่ย (หน่วยเป็นเปอร์เซ็นต์ : %MEM) ค่าต่างๆ เหล่านี้เก็บไว้ในไฟล์ ids.dft

การเก็บค่าภายในไฟล์ ids.dft ประกอบด้วยข้อมูล ดังนี้

Day	Night	Critical	%CPU	%MEM
เวลาเริ่มงานปรกติ	เวลาเริ่มงานช่วงพิเศษ	ค่าเริ่มต้นเวลาวิกาล	ค่าเฉลี่ยการใช้ซีพียู	ค่าเฉลี่ยการใช้หน่วยความจำ
เวลาเลิกงานปรกติ	เวลาเลิกงานช่วงพิเศษ	ค่าสิ้นสุดเวลาวิกาล	ค่าวิกฤติการใช้ซีพียู	ค่าวิกฤติการใช้หน่วยความจำ

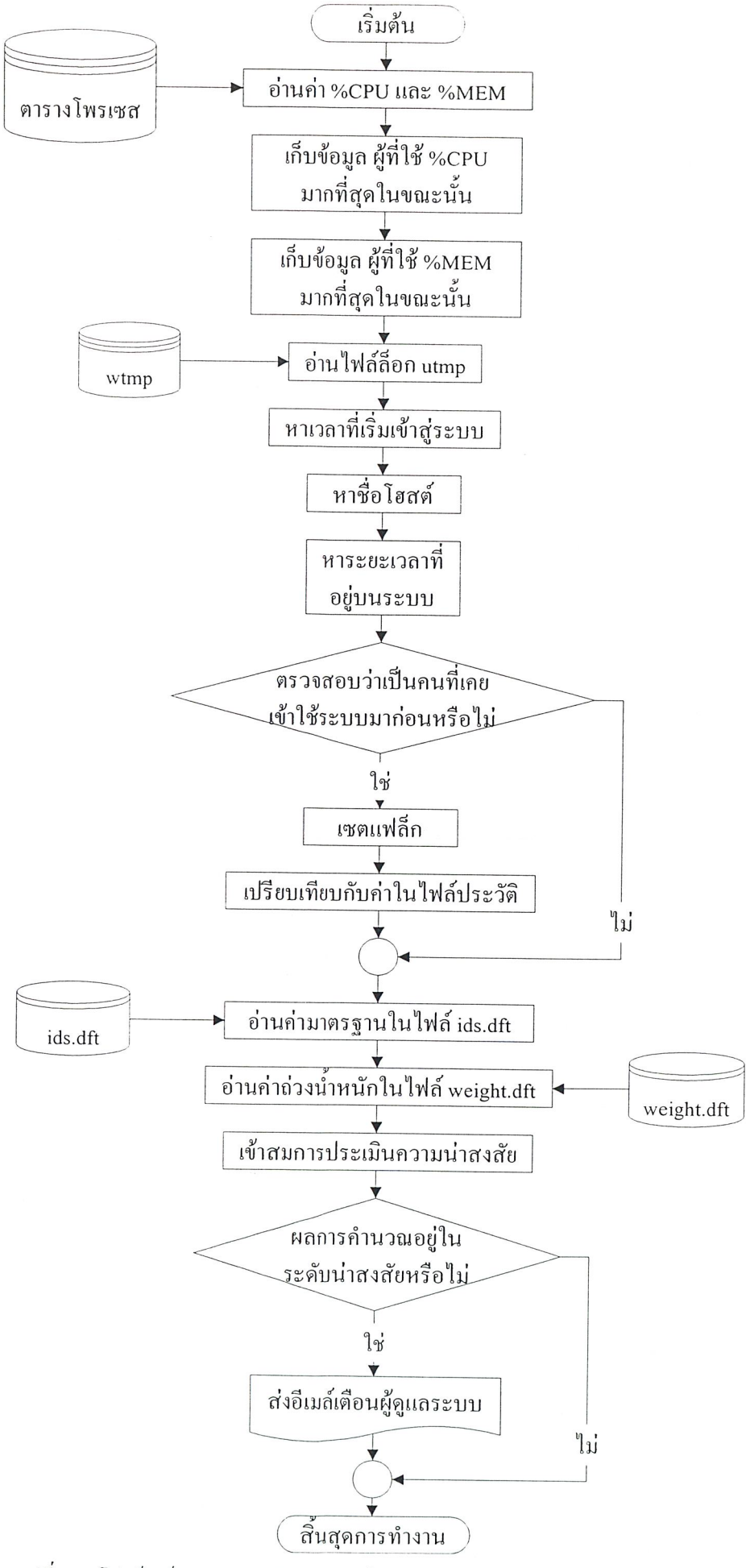
ตัวอย่างข้อมูล

Day	Night	Critical	%CPU	%MEM
08:00	17:00	00:00	1.8	8.9
17:00	00:00	08:00	2.5	2.5

ไฟล์ weight.dft เป็นไฟล์ที่เก็บข้อมูลค่าถ่วงน้ำหนัก สำหรับใช้ในการคำนวณเปอร์เซ็นต์ความน่าสงสัยที่ผู้ใช้อาจกำลังพยายามบุกรุกระบบ ประกอบด้วยข้อมูล ดังนี้

w1	w2	w3	w4	w5	w6
20	10	10	30	20	10

สำหรับตัวเลขข้างต้น สามารถเปลี่ยนแปลงได้ ตามที่ผู้ดูแลระบบเห็นสมควร



รูปที่ 3-11 โฟลว์ชาร์ตแสดงการทำงานของส่วนการตรวจตรวจสอบการใช้งานทรัพยากรระบบที่ผิดปรกติ

3.3.2 ส่วนการเตรียมประวัติของผู้ใช้งานระบบ

ในส่วนนี้มีการตรวจสอบไฟล์ wtmp เพื่อดูว่าช่วงระยะเวลานั้น มีล็อกอินเนมใดบ้างที่มีการเข้าใช้ระบบจริง และทำการเก็บล็อกอินเนมไว้ในไฟล์ login.txt หาค่าเฉลี่ยของช่วงเวลาที่ผู้ใช้คนนั้นอยู่บนระบบ และเก็บไว้ในไฟล์ {ล็อกอินเนม}.cnt บันทึก remote host name ที่ผู้ใช้คนนั้นเคยใช้ล็อกอินเข้ามาเก็บไว้ในไฟล์ {ล็อกอินเนม}.hst การทำงานโดยรวมแสดงไว้ในรูปที่ 3-12

ไฟล์ที่สำคัญในส่วนนี้ได้แก่ ไฟล์ login.txt ไฟล์นามสกุล cnt และไฟล์นามสกุล hst ข้อมูลที่เก็บในไฟล์ logint.txt คือชื่อผู้ใช้ที่มีการเข้าใช้ระบบในช่วงตั้งแต่เริ่มมีการบันทึกล็อก wtmp จนถึงวันที่รันส่วนการเตรียมประวัติของผู้ใช้งานระบบของระบบตรวจจับผู้บุกรุกระบบคอมพิวเตอร์ เนื่องจากภายในระบบหนึ่งหนึ่งอาจมีล็อกอินเนมอยู่เป็นจำนวนมาก แต่มีผู้ที่เข้ามาใช้งานจริงๆ เพียงบางส่วนเท่านั้น และในบางช่วงเวลา ก็อาจมีสมาชิกของระบบบางคน หยุดใช้ระบบไปชั่วคราว ดังนั้นการทำบันทึกชื่อผู้เข้าใช้ระบบช่วยลดความล่าช้าในการเตรียมไฟล์ประวัติ และช่วยประหยัดการใช้หน่วยความจำ

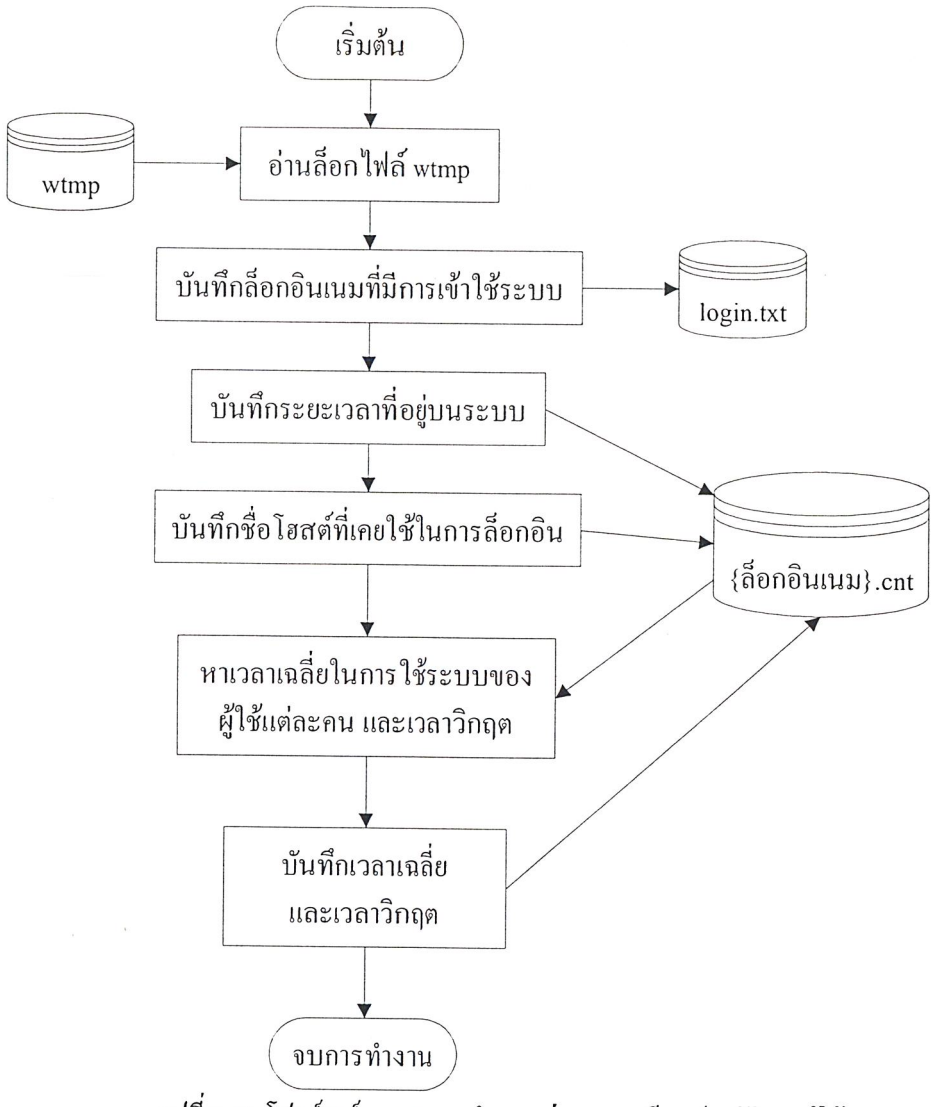
ไฟล์นามสกุล cnt เป็นไฟล์ที่เก็บค่าระยะเวลาโดยเฉลี่ยที่ผู้ใช้คนนั้นอยู่บนระบบ และค่าระยะเวลาวิกฤติ ค่าระยะเวลาเฉลี่ยนำมาจากการอ่านไฟล์ล็อก wtmp นำระยะเวลาที่อยู่บนระบบมารวมกันแล้วหารด้วยจำนวนครั้งที่ล็อกอิน ยกเว้นครั้งที่มีระยะเวลาเป็นศูนย์นาที ค่าระยะเวลาวิกฤติได้จากการหาค่าเฉลี่ยแบบถ่วงน้ำหนัก ยกตัวอย่างเช่น

ระยะเวลาที่อยู่บนระบบ (ชั่วโมง:นาที)	ระยะเวลาที่อยู่บนระบบ (นาที)	ค่าถ่วงน้ำหนัก	ผลคูณของเวลาและน้ำหนัก
00:00	0	1	0
00:06	6	2	12
00:08	8	3	24
00:09	9	4	36
10:00	600	5	3000
01:54	114	5	570
02:39	159	4	636
00:02	2	3	6
00:40	40	2	80
00:20	20	1	20
รวม	958	30	4384
ค่าเฉลี่ย	106.44	-	146.13

การบันทึกข้อมูลในไฟล์ {ล็อกอินเนม}.cnt เป็นดังนี้

106.44 146.13

กล่าวคือค่าแรกเป็นระยะเวลาเฉลี่ย ค่าที่สองเป็นค่าระยะเวลาวิกฤติ



รูปที่ 3-12 โฟลว์ชาร์ตแสดงการทำงาน ส่วนการเตรียมประวัติของผู้ใช้งานระบบ

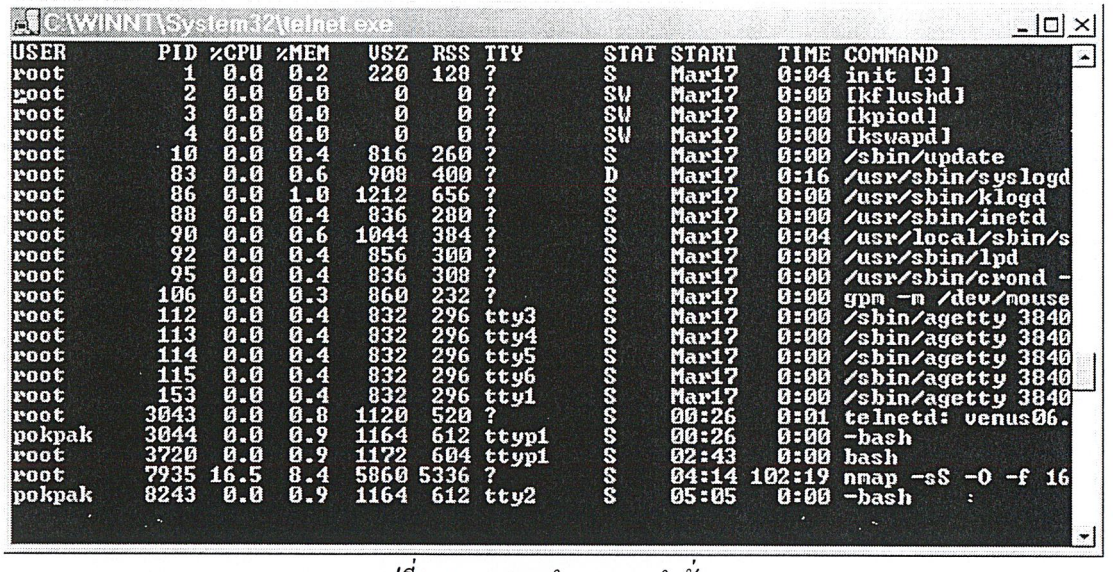
3.3.3 ส่วนการตรวจจับพฤติกรรมของผู้ใช้ระบบ

การทำงานของส่วนนี้แบ่งออกเป็น

1. ติดตามดูพฤติกรรมการใช้ทรัพยากรระบบของผู้เข้าใช้ระบบแต่ละคน คือ ปริมาณการใช้ซีพียู และ ปริมาณการใช้หน่วยความจำ เพื่อค้นหาผู้ที่มีการใช้ปริมาณทรัพยากรมากผิดปรกติ

2. วิเคราะห์ข้อมูลของผู้ต้องสงสัย โดยใช้สมการทางคณิตศาสตร์ ซึ่ง ผลการวิเคราะห์เก็บบันทึกลงไฟล์ result.txt แสดงผลของตัวแปร x1-x6 และเปอร์เซ็นต์ความน่าสงสัย

ในการติดตามดูพฤติกรรม ใช้คำสั่ง ps คอยตรวจดูว่ามีใครที่เข้ามาใช้หน่วยความจำ หรือซีพียูของระบบสูงที่สุด ในทุกช่วงเวลา 1 นาที จากนั้นทำการค้นในไฟล์ login.txt เพื่อดูว่ามีประวัติผู้ใช้คนนั้นหรือไม่ ถ้ามีประวัติอยู่ จะเปรียบเทียบข้อมูลในส่วนของ hostname และช่วงเวลาที่อยู่บนระบบต่อไป



รูปที่ 3-13 ผลการทำงานของคำสั่ง ps

ข้อมูลที่ส่วนการตรวจจับพฤติกรรมของผู้ใช้ระบบต้องทราบ ได้แก่

1. ชื่อผู้ที่ใช้ซีพียูสูงสุดในช่วง 1 นาทีนั้น (เก็บไว้ในตัวแปร cpu_n)
2. เปอร์เซ็นต์ซีพียูที่ใช้ (เก็บไว้ในตัวแปร maxcpu)
3. เปอร์เซ็นต์หน่วยความจำที่ใช้ (เก็บไว้ในตัวแปร MaxCpu_m)
4. เวลาที่เริ่มเข้าสู่ระบบ (เก็บไว้ในตัวแปร CLogTime)
5. ชื่อโฮสต์ของผู้ใช้ในกรณีที่ล็อกอินมาจากข้างนอกระบบ (เก็บไว้ในตัวแปร chost)
6. ช่วงเวลาที่อยู่บนระบบ (connection interval เก็บไว้ในตัวแปร c_connect)
7. เป็นผู้ใช้ที่เคยใช้งานระบบในช่วงเวลาที่มีการทำไฟล์ประวัติหรือไม่
8. ชื่อผู้ที่ใช้หน่วยความจำสูงสุดในช่วง 1 นาทีนั้น (เก็บไว้ในตัวแปร mem_n)
9. เปอร์เซ็นต์ซีพียูที่ใช้ (เก็บไว้ในตัวแปร MaxMem_c)
10. เปอร์เซ็นต์หน่วยความจำที่ใช้ (เก็บไว้ในตัวแปร maxmem)
11. เวลาที่เริ่มเข้าสู่ระบบ (เก็บไว้ในตัวแปร MLogTime)
12. ชื่อโฮสต์ของผู้ใช้ในกรณีที่ล็อกอินมาจากข้างนอกระบบ (เก็บไว้ในตัวแปร mhost)
13. ช่วงเวลาที่อยู่บนระบบ (connection interval เก็บไว้ในตัวแปร m_connect)
14. เป็นผู้ใช้ที่เคยใช้งานระบบในช่วงเวลาที่มีการทำไฟล์ประวัติหรือไม่

ข้อมูลเหล่านี้ใช้ในการวิเคราะห์ ประเมินความน่าสงสัยว่าผู้ใช้คนนั้นอาจกำลังพยายามบุกรุกระบบอยู่หรือไม่

การวิเคราะห์เปอร์เซ็นต์ความน่าสงสัย ใช้สมการที่ 3-1

เปอร์เซ็นต์แนวโน้มการเกิดการบุกรุก = $W_1X_1+ W_2X_2+ W_3X_3+ W_4X_4+ W_5X_5+ W_6X_6$

สมการที่ 3-1 การคำนวณเปอร์เซ็นต์แนวโน้มการเกิดการบุกรุก

ความหมายของตัวแปรในสมการ เป็นดังนี้

1. เวลาที่เริ่มเข้าใช้ระบบ (Time of login) : X_1 เปรียบเทียบเวลาที่ผู้ใช้เริ่มเข้าใช้ระบบกับเวลาที่ผู้ดูแลระบบได้กำหนดไว้ ซึ่งแบ่งเป็น 3 ระดับ ดังนี้

- เวลาทำงาน เป็นช่วงเวลาทำงานปรกติขององค์กร ถือว่าผู้ใช้ที่เข้าใช้ระบบในขณะนั้นเป็นผู้ใช้ปรกติ ค่าตัวแปร X_1 เป็น 0
- เวลาทำงานพิเศษ เป็นช่วงเวลาที่องค์กรมีการทำงานพิเศษ (Overtime) ซึ่งอาจมีผู้เข้ามาใช้ระบบได้ แต่ก็มีโอกาสเกิดการบุกรุกสูงกว่าเวลาทำงานปรกติ เช่น ในกรณีที่ผู้บุกรุกเป็นผู้ที่ทำงานอยู่ภายในองค์กรและเห็นว่ามีผู้ทำงานอยู่น้อยในช่วงเวลาทำงานพิเศษ จึงทำการบุกรุกระบบในเวลานั้น ผู้ที่เข้าใช้ระบบขณะนั้นให้ถือว่าเป็นผู้ต้องสงสัยไว้ก่อน โดยกำหนดให้ X_1 เป็น 0.5
- เวลาวิกาล เป็นช่วงเวลาที่ปรกติไม่มีผู้ใช้งานระบบแล้ว การเข้าใช้งานระบบขณะนั้นถือเป็นสิ่งผิดปรกติ กำหนดให้ค่า X_1 เป็น 1
- เวลาที่ผู้ใช้เข้าสู่ระบบ ถูกนำมาเปรียบเทียบกับเวลาที่กำหนดไว้ในไฟล์ ids.dft เมื่อตอนติดตั้งระบบตรวจจับผู้บุกรุกระบบคอมพิวเตอร์

2. สถานที่ที่ผู้ใช้ติดต่อเข้ามา (Location of login) : X_2 ผู้ใช้ระบบอาจมีการติดต่อเข้ามาจากเครื่องใด ๆ ก็ได้ที่เชื่อมต่อกับอินเทอร์เน็ต ค่าตัววัดนี้แบ่งเป็น

- การติดต่อจากเครื่องที่เชื่อถือได้ (Trusted IP) ถ้าผู้ใช้ทำการติดต่อมาจากเครื่องเหล่านี้ให้ค่า X_2 เป็น 0 คือถือว่าเป็นผู้ใช้ปรกติ
- การติดต่อมาจากเครื่องอื่นนอกเหนือจากนี้ให้เป็นผู้ต้องสงสัยคือให้ X_2 เป็น 1
- การตรวจสอบสถานที่ที่ผู้ใช้ติดต่อเข้ามานี้เป็นการตรวจสอบการบุกรุกจากภายนอก
- ถ้าพบว่าล็อกอินเนมของผู้ที่ใช้หน่วยความจำหรือซีพียูสูงสุดขณะนั้น ตรงกับข้อมูลที่เก็บไว้ในไฟล์ login.txt แสดงว่ามีประวัติของผู้ใช้ซึ่งอาจเป็นผู้ต้องสงสัยอยู่ จะนำชื่อโฮสต์ของผู้ใช้ไปเปรียบเทียบกับประวัติในไฟล์ชื่อ {ล็อกอินเนม}.hst แต่หากไม่พบว่ามีไฟล์ประวัติ กำหนดให้ค่า X_2 เป็น 1 ทันที ถ้าเป็นการล็อกอินจากนอกระบบ และมีค่าเป็น 0 เมื่อล็อกอินจากภายในระบบ

3. ระยะเวลาที่เข้าใช้ระบบ (Connect time) : X_3 เปรียบเทียบระยะเวลาที่ผู้ใช้ติดต่อกับระบบกับระดับระยะเวลาปรกติที่ใช้ติดต่อกับระบบในฐานข้อมูล ระยะเวลาที่ผู้ใช้ติดต่อเข้ามาใช้ระบบนี้แบ่งเป็น 3 ระดับ คือ

- ผู้ใช้เข้าใช้ระบบไม่เกินกว่าระดับปรกติในฐานข้อมูล ให้ค่า X_3 เป็น 0 คืออยู่ในระดับปรกติ
- ผู้ใช้เข้าใช้ระบบเกินกว่าระดับปรกติแต่ไม่เกินค่าวิกฤติให้ค่า X_3 เป็น 0.5
- ผู้ใช้เข้าใช้ระบบเกินกว่าค่าวิกฤติที่ถือว่าอาจเป็นการบุกรุกได้ ให้ค่า X_3 เป็น 1

ถ้าพบว่าล็อกอินเนมของผู้ที่ใช้หน่วยความจำหรือซีพียูสูงสุดขณะนั้น ตรงกับข้อมูลที่เก็บไว้ในไฟล์ login.txt แสดงว่ามีประวัติของผู้ใช้ซึ่งอาจเป็นผู้ต้องสงสัยอยู่ นำระยะเวลาที่เข้าใช้ระบบของผู้ใช้ไปเปรียบเทียบกับประวัติในไฟล์ชื่อ {ล็อกอินเนม}.cnt แต่หากไม่พบว่ามีไฟล์ประวัติ กำหนดให้ค่า X_3 เป็น 1 ทันที

4. ปริมาณการใช้งานหน่วยประมวลผล (CPU time) : X_4 ตัววัดนี้ใช้วัดการใช้งานหน่วยประมวลผลของระบบกับค่าการใช้งานหน่วยประมวลผลระดับปรกติในฐานข้อมูล ปริมาณการใช้งานกำหนดให้แบ่งเป็น 3 ระดับ คือ

- มีการใช้งานหน่วยประมวลผลไม่เกินกว่าระดับปรกติในฐานข้อมูล ให้ค่า X_4 เป็น 0 คือถือว่าเป็นการใช้งานปรกติ
- มีการใช้งานหน่วยประมวลผลเกินกว่าระดับปรกติแต่ไม่เกินกว่าค่าวิกฤติ ให้ค่า X_4 เป็น 0.5
- มีการใช้งานหน่วยประมวลผลเกินกว่าค่าวิกฤติที่ถือว่าอาจเป็นการบุกรุกได้ ให้ค่า X_4 เป็น 1

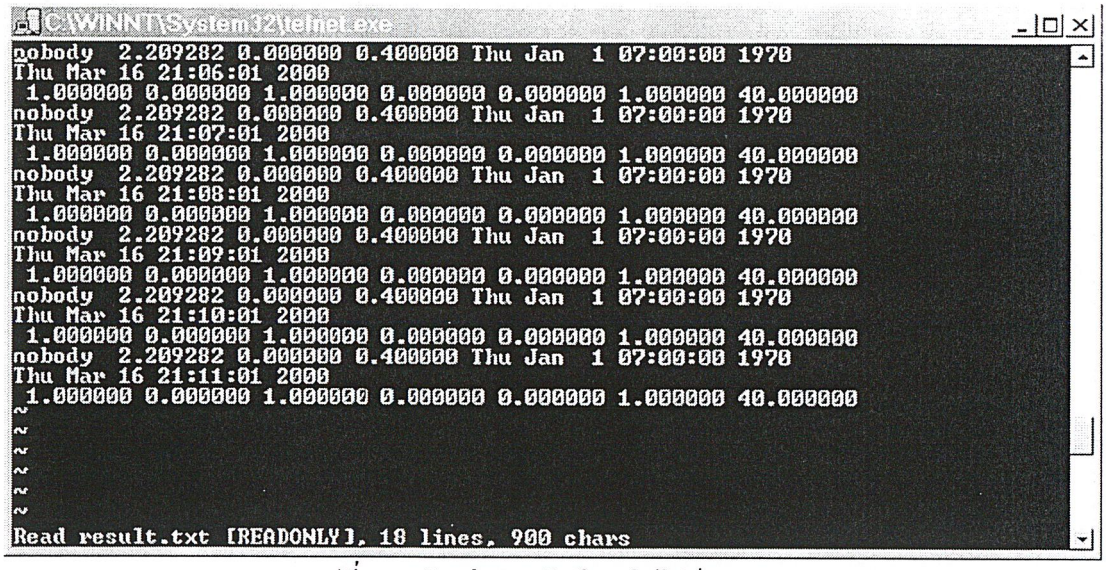
5. ปริมาณการใช้งานหน่วยความจำ (Memory Usage) : X_4 ตัววัดนี้ใช้วัดการใช้งานหน่วยความจำของระบบกับค่าการใช้งานหน่วยความจำระดับปรกติในฐานข้อมูล ปริมาณการใช้งานกำหนดให้แบ่งเป็น 3 ระดับ คือ

- มีการใช้งานหน่วยความจำไม่เกินกว่าระดับปรกติในฐานข้อมูล ให้ค่า X_4 เป็น 0 คือถือว่าเป็นการใช้งานปรกติ
- มีการใช้งานหน่วยความจำเกินกว่าระดับปรกติแต่ไม่เกินกว่าค่าวิกฤติ ให้ค่า X_4 เป็น 0.5
- มีการใช้งานหน่วยความจำเกินกว่าค่าวิกฤติที่ถือว่าอาจเป็นการบุกรุกได้ ให้ค่า X_4 เป็น 1

6. ไม่พบไฟล์ประวัติ (Stranger) : X_6 ตัววัดนี้ใช้ตรวจจับผู้ที่ไม่เคยเข้าใช้ระบบมาก่อน

7. ค่า W_1 - W_6 เป็นค่าถ่วงน้ำหนัก (Weight) ของตัววัดแต่ละตัว ซึ่ง $W_1+W_2+W_3+W_4+W_5+W_6=100$

การกำหนดค่าถ่วงน้ำหนักขึ้นอยู่กับนโยบายทางด้านความปลอดภัยขององค์กร และสิ่งแวดล้อมในองค์กร เช่น ถ้าองค์กรนั้น ๆ ผู้ใช้ส่วนใหญ่เข้าใช้ระบบในเวลาทำงานหรือเวลาทำงานพิเศษ การเข้าใช้ระบบนอกเวลาถือว่าเป็นความผิดปรกติมาก กำหนดค่าถ่วงน้ำหนัก W_1 ให้มีค่าสูง ๆ (W_1 เป็นค่าถ่วงน้ำหนักของตัววัดเวลาที่เริ่มเข้าใช้ระบบหรือ X_1) หรือถ้าลักษณะงานขององค์กรเป็นงานที่ต้องใช้หน่วยประมวลมาก ๆ ก็ให้ค่าถ่วงน้ำหนัก W_4 ซึ่งเป็นค่าถ่วงน้ำหนักของตัววัดปริมาณการใช้งานหน่วยประมวลผลมีค่าน้อย ๆ เนื่องจากมีโอกาสสูงที่ผู้ใช้ปรกติใช้หน่วยประมวลผลมาก



รูปที่ 3-14 ตัวอย่างการบันทึกผลในไฟล์ result.txt

3.3.4 ส่วนแจ้งเตือนผู้ดูแลระบบ

ผู้ดูแลระบบสามารถกำหนดได้ว่าต้องการให้มีการส่งอีเมล์เตือนมา เมื่อพบความน่าสงสัยถึงระดับเปอร์เซ็นต์เท่าไรได้ในตอนติดตั้งระบบ เมื่อส่วนการตรวจจับพฤติกรรมของผู้ใช้ระบบวิเคราะห์พบความน่าสงสัยถึงระดับเปอร์เซ็นต์ที่ผู้ดูแลกำหนดไว้ จะทำการบันทึกผลลงไฟล์ letter แล้วส่งจดหมายไปเตือนตามแอดเดรสที่กำหนดไว้

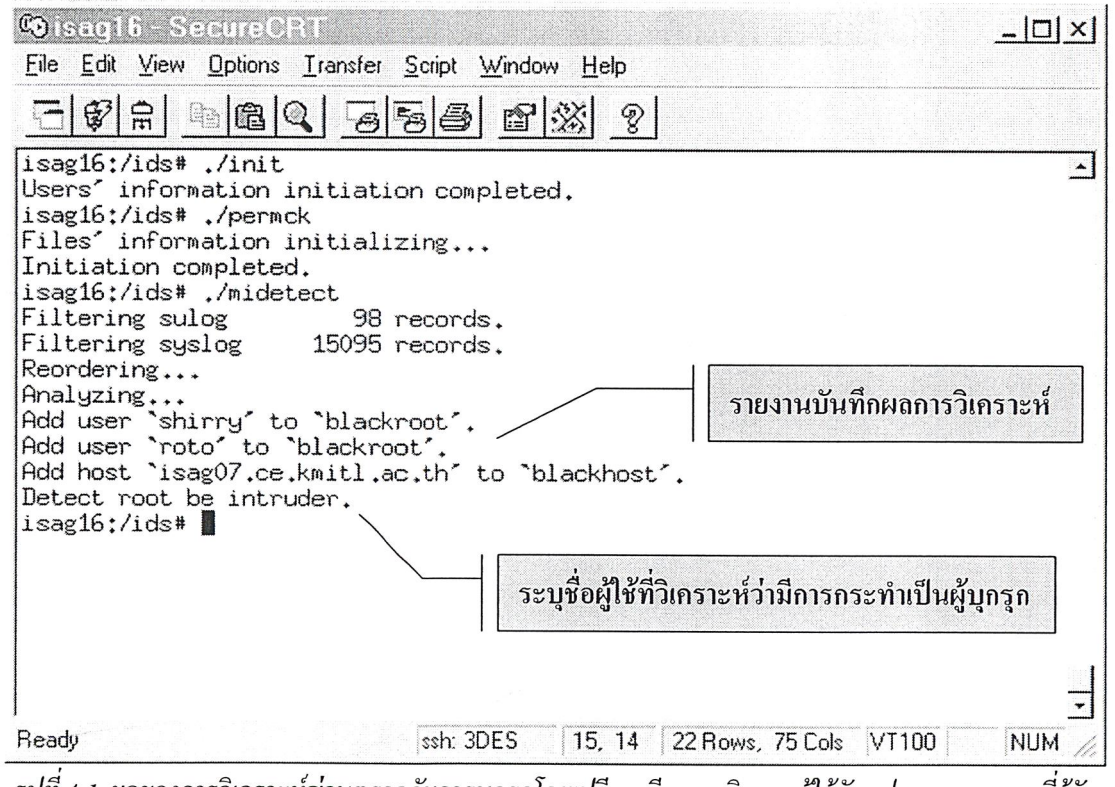
บทที่ 4

ผลการทดลอง

หลังจากที่ผู้จัดทำสร้างระบบตรวจจับผู้บุกรุกระบบคอมพิวเตอร์ตามแนวคิดในบทที่ 3 แล้ว ได้ทดลองการทำงานของ 2 ส่วนหลัก คือส่วนตรวจจับการบุกรุกโดยเปรียบเทียบพฤติกรรมผู้ใช้กับรูปแบบการบุกรุกที่รู้จัก และส่วนตรวจจับการบุกรุกโดยตรวจสอบการใช้งานทรัพยากรระบบที่ผิดปรกติ

4.1 ผลการทดลองส่วนตรวจจับการบุกรุกโดยเปรียบเทียบพฤติกรรมผู้ใช้กับรูปแบบการบุกรุกที่รู้จัก

เมื่อนำระบบตรวจจับผู้บุกรุกส่วนนี้มาติดตั้งเป็นระยะเวลาหนึ่ง และทำการทดสอบการทำงานได้ผลดังนี้

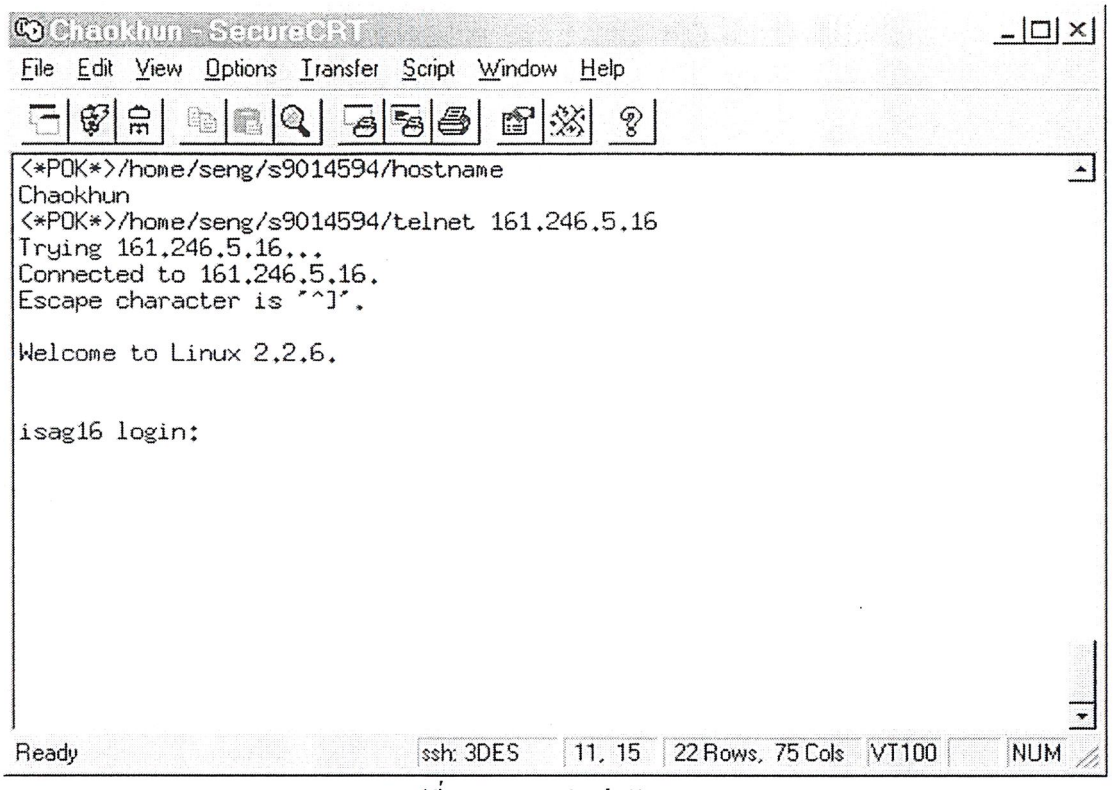


รูปที่ 4-1 ผลของการวิเคราะห์ส่วนตรวจจับการบุกรุกโดยเปรียบเทียบพฤติกรรมผู้ใช้กับรูปแบบการบุกรุกที่รู้จัก

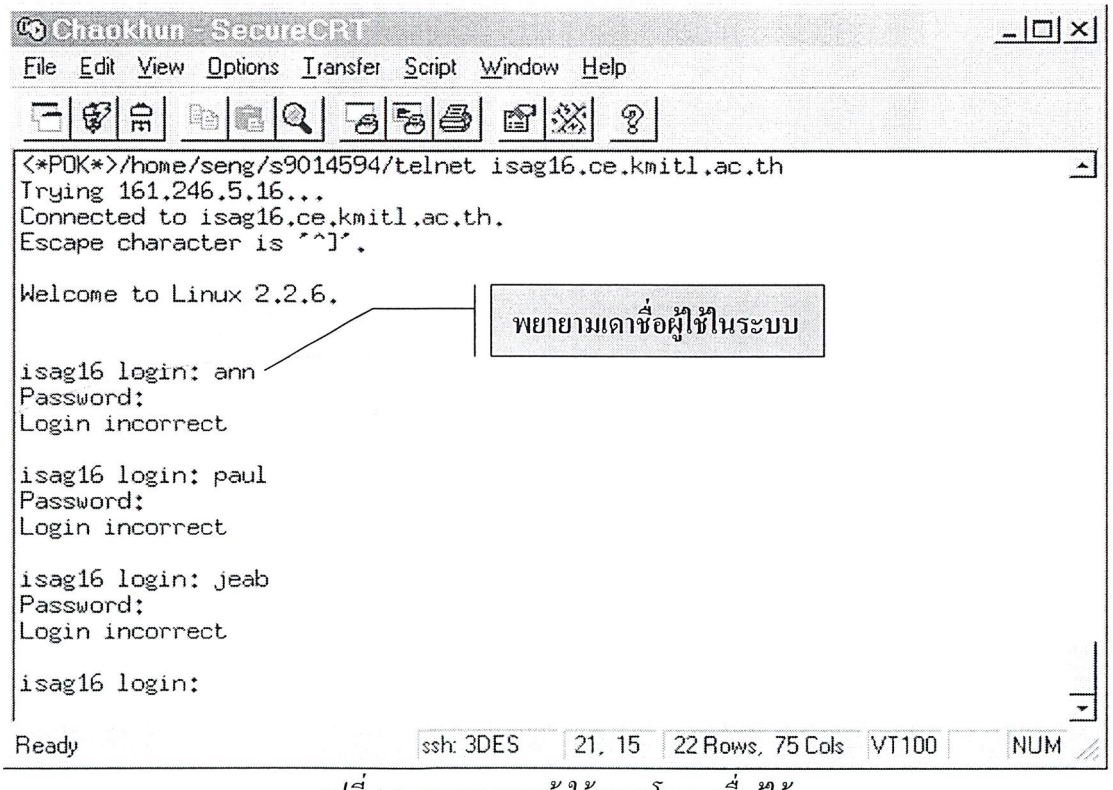
ต่อมาได้นำระบบตรวจจับผู้บุกรุกมาวิเคราะห์กรณีทดสอบดังต่อไปนี้

กรณีทดสอบที่ 1

ทดสอบขอเข้าใช้งานระบบจากโฮสต์อื่น ในที่นี้คือ chaokhun.kmitl.ac.th โดยการเดาชื่อผู้ใช้ตามรูปที่ 4-2 และ 4-3

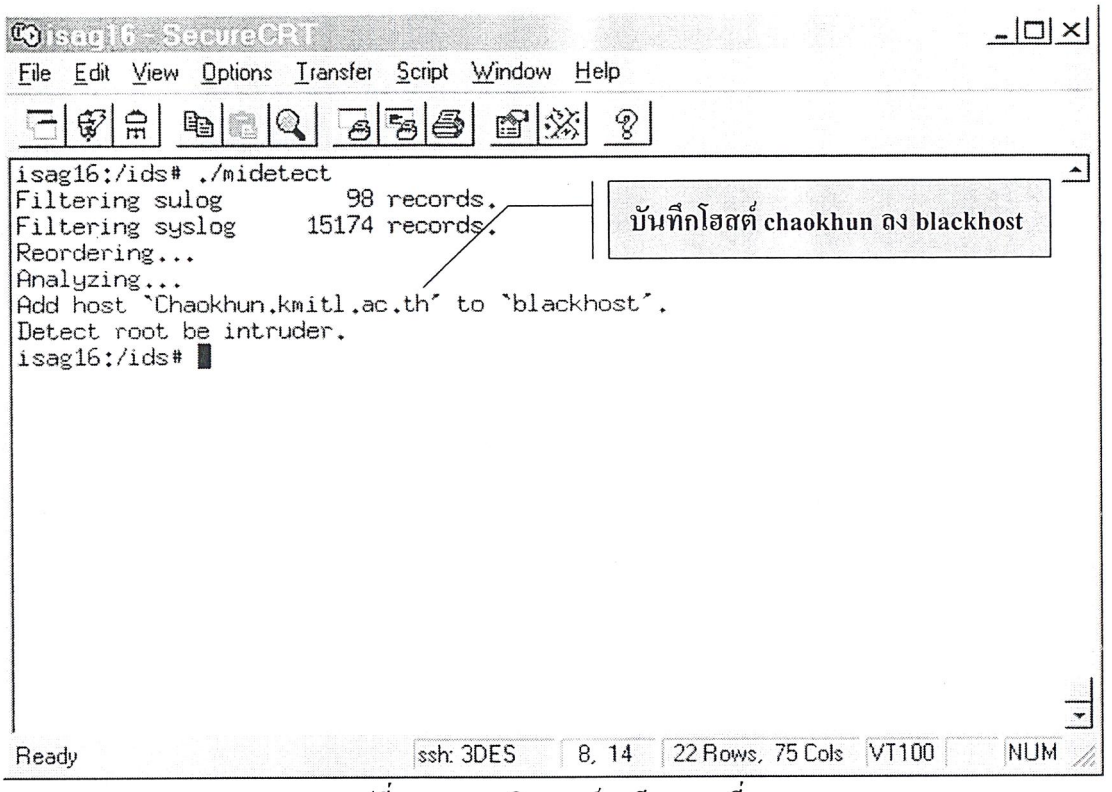


รูปที่ 4-2 การขอติดต่อกับระบบ



รูปที่ 4-3 การพยายามเข้าใช้ระบบโดยเดาชื่อผู้ใช้

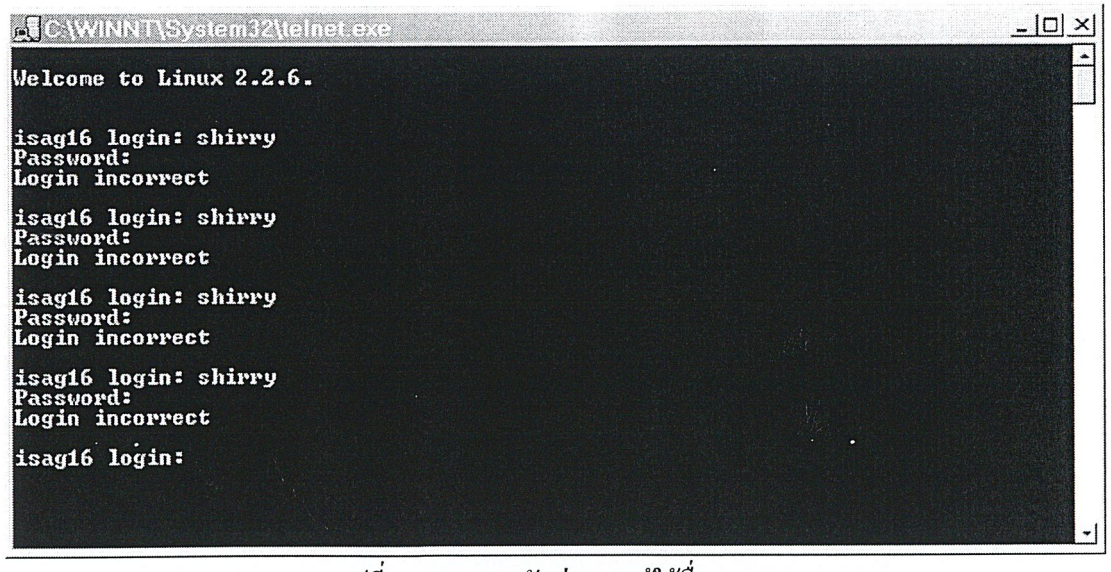
ผลที่ได้คือระบบตรวจจับผู้บุกรุกบันทึกชื่อโฮสต์นั้นลงไฟล์ blackhost ดังรูปที่ 4-4



รูปที่ 4-4 ผลการวิเคราะห์กรณีทดสอบที่ 1

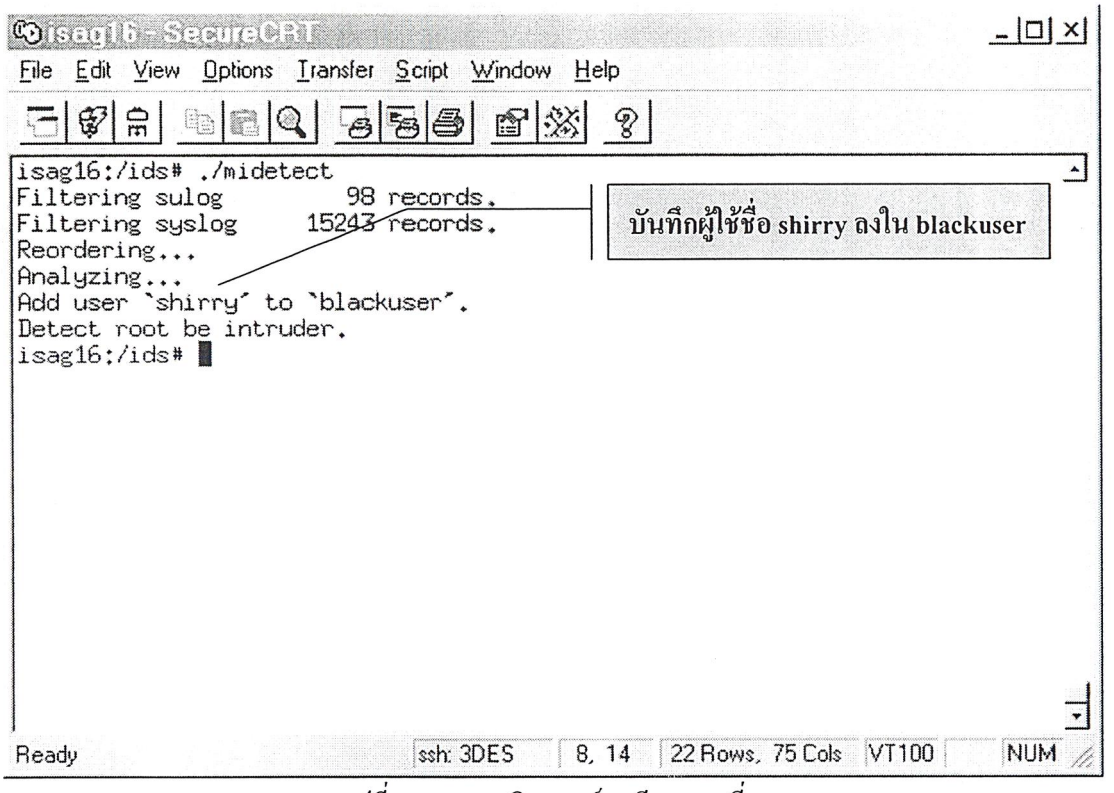
กรณีทดสอบที่ 2

กรณีที่ผู้บุกรุกรู้ชื่อผู้ใช้ที่มีชื่ออยู่จริงในระบบ และพยายามเข้าใช้ระบบโดยการเดารหัสผ่าน ในที่นี้คือผู้ใช้ชื่อ shirry



รูปที่ 4-5 การเดารหัสผ่านของผู้ใช้ชื่อ shirry

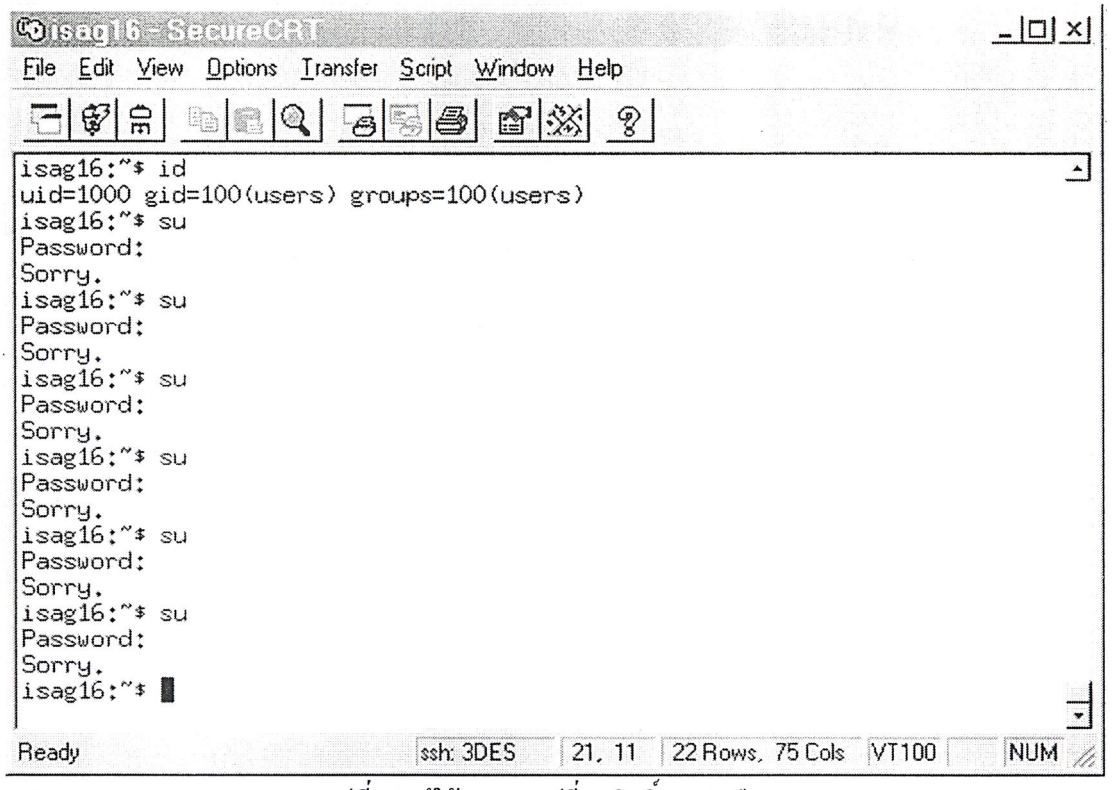
ผลการวิเคราะห์พบว่าระบบตรวจจับผู้บุกรุกระบุผู้ใช้ชื่อ shirry ว่าเป็นผู้บุกรุก ดังรูปที่ 4-6



รูปที่ 4-6 ผลการวิเคราะห์กรณีทดสอบที่ 2

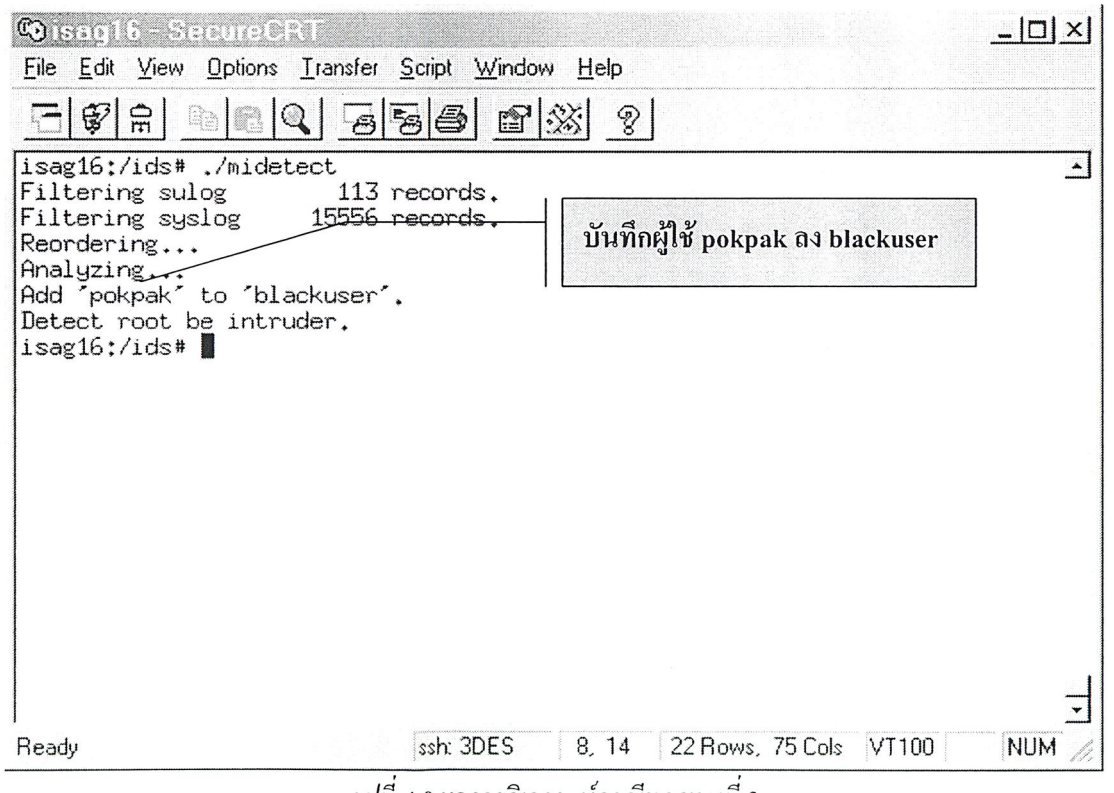
กรณีทดสอบที่ 3

ทดสอบโดยให้ผู้ใช้ที่มีสิทธิ์ถูกต้องในระบบพยายามใช้คำสั่ง su เพื่อเปลี่ยนสิทธิ์ตนเองเป็นรูตหลาย ๆ ครั้ง ตามรูปที่ 4-7



รูปที่ 4-7 ผู้ใช้พยายามเปลี่ยนสิทธิ์ตนเองเป็นรูต

เมื่อเกินจำนวนครั้งที่กำหนด ระบบตรวจจับผู้บุกรุกจะบันทึกชื่อผู้ใช้นั้นลงไฟล์ blackuser ดังรูปที่ 4-8



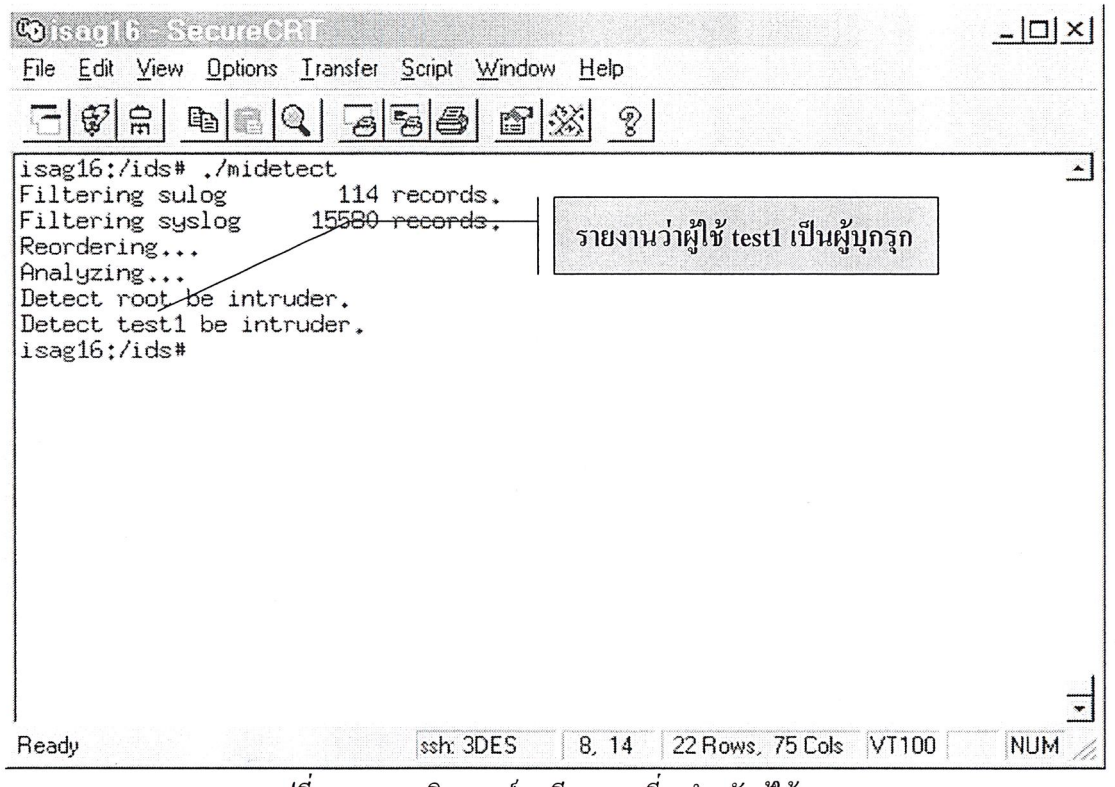
รูปที่ 4-8 ผลการวิเคราะห์กรณีทดสอบที่ 3

กรณีทดสอบที่ 4

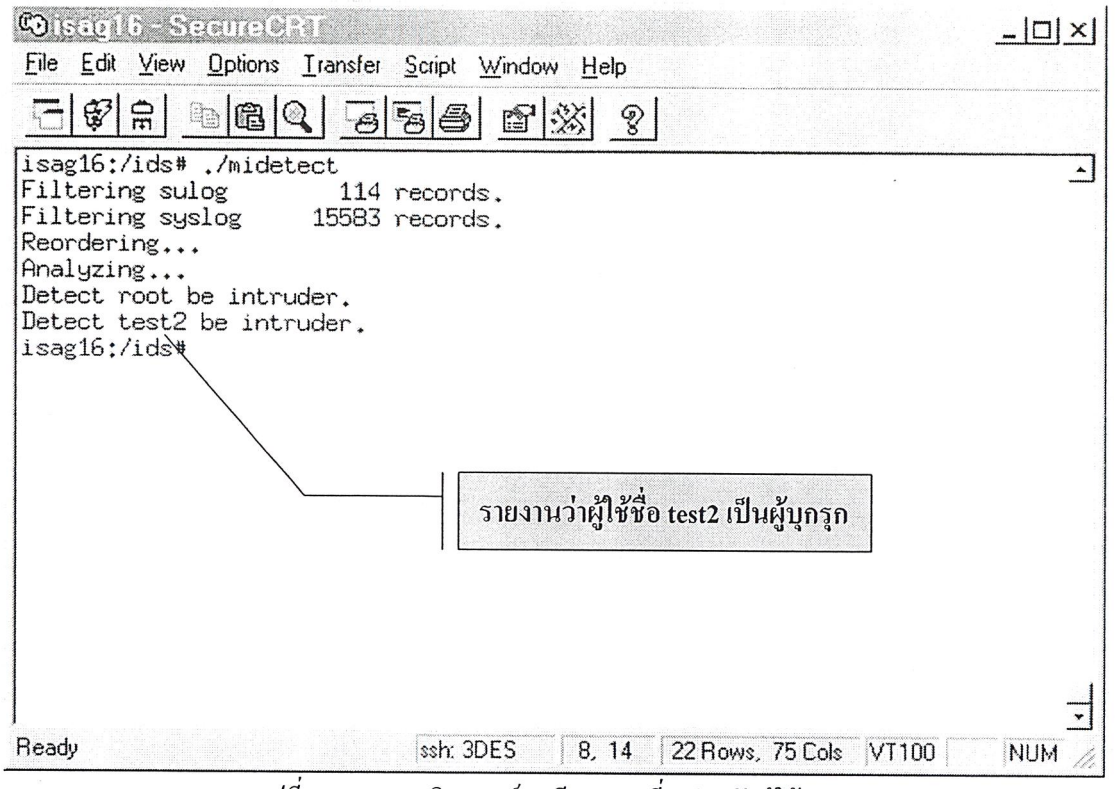
สมมติว่าผู้บุกรุกทำการบุกรุกสำเร็จแล้ว ผู้บุกรุกได้แก้ไขไฟล์ล็อก sulog หรือ syslog ไฟล์ใดไฟล์หนึ่ง เพื่อกลบเกลื่อนร่องรอยการบุกรุก ทำให้ข้อมูลของทั้งสองไฟล์ไม่ตรงกัน

ผู้ใช้ชื่อ test1 ได้แก้ไข syslog และผู้ใช้ชื่อ test2 แก้ไข sulog หลังจากระบบตรวจจับผู้บุกรุกวิเคราะห์พบว่ารายงานผู้ใช้ test1 และ test 2 เป็นผู้บุกรุก

ผลการวิเคราะห์แสดงตามรูปที่ 4-9 และ รูปที่ 4-10 ตามลำดับ



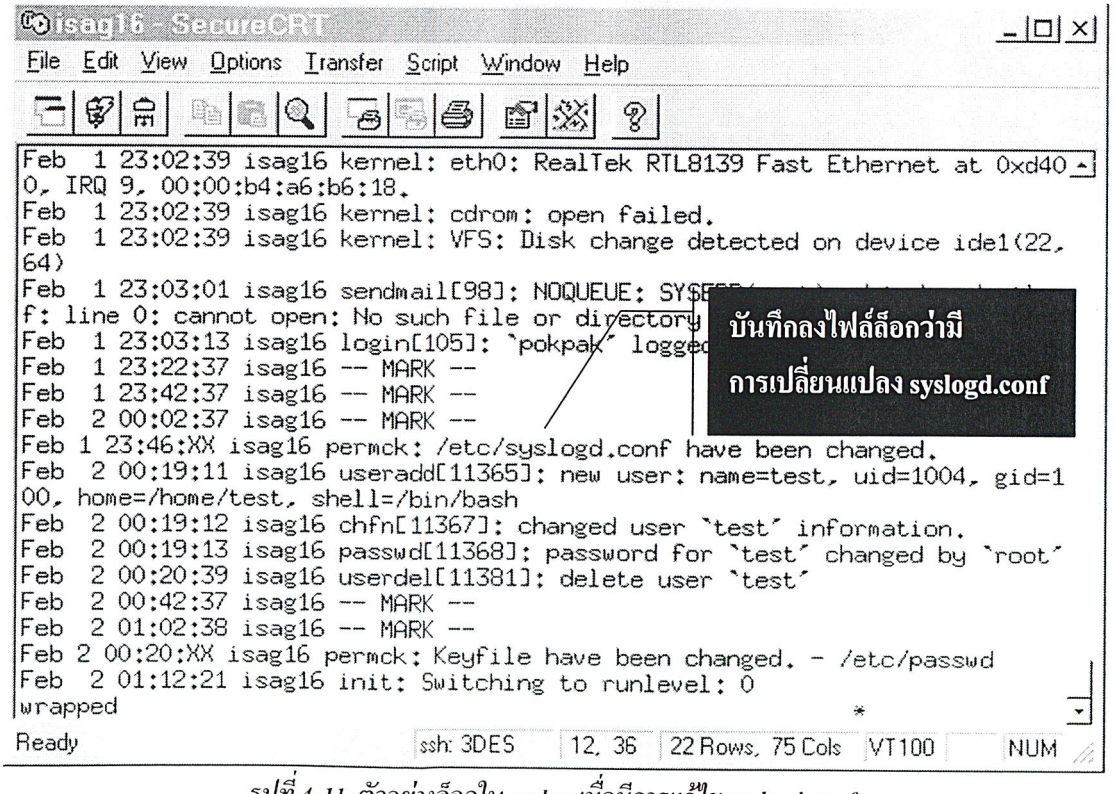
รูปที่ 4-9 ผลการวิเคราะห์กรณีทดสอบที่ 4 สำหรับผู้ใช้ test1



รูปที่ 4-10 ผลการวิเคราะห์กรณีทดสอบที่ 4 สำหรับผู้ใช้ test2

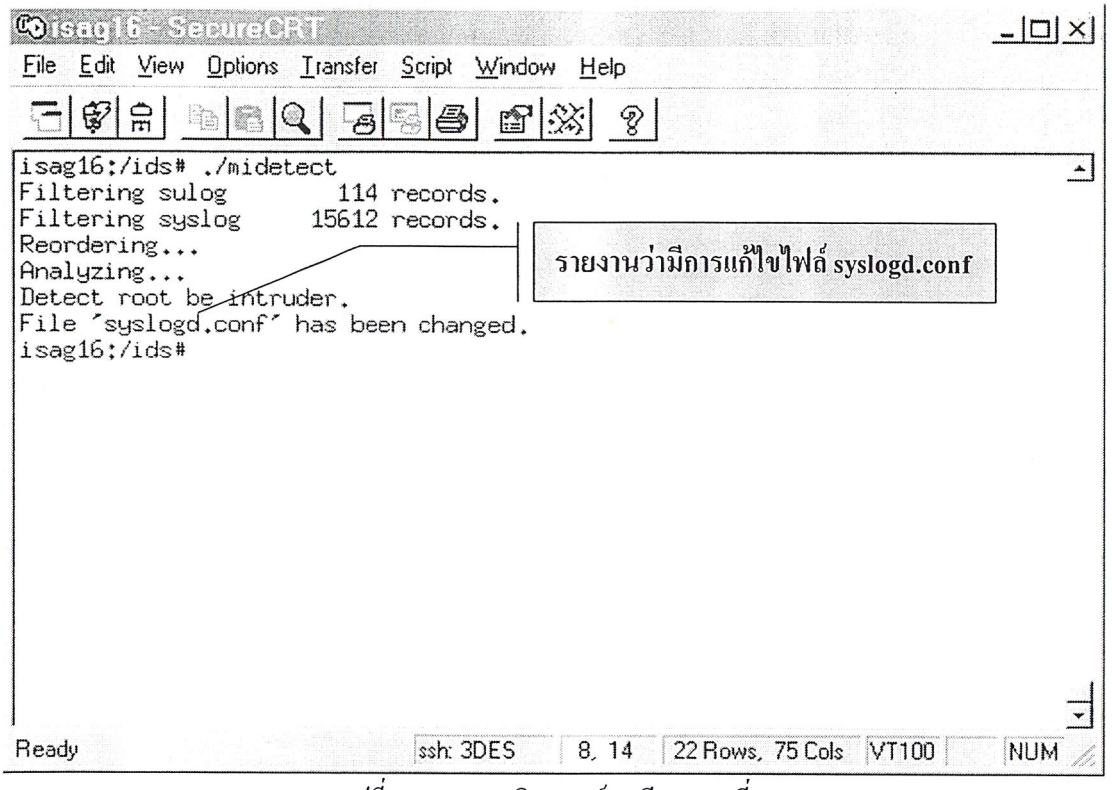
กรณีทดสอบที่ 5

ทดสอบการแก้ไขไฟล์สำคัญ ซึ่งได้ทดสอบการแก้ไขไฟล์ syslogd.conf การกระทำนี้สามารถตรวจจับได้โดยการเรียกสคริปต์ permsetup ซึ่งบันทึกเหตุการณ์นี้ลง syslog ดังรูปที่ 4-11



รูปที่ 4-11 ตัวอย่างล็อกใน syslog เมื่อมีการแก้ไข syslogd.conf

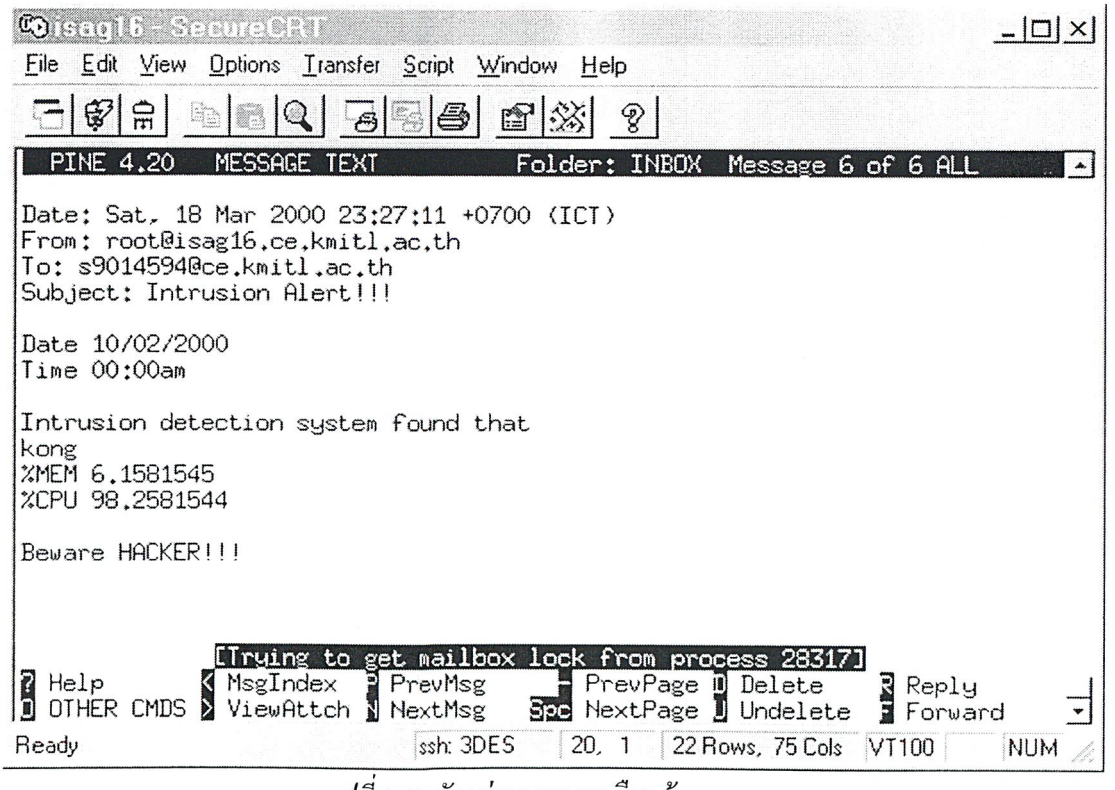
เมื่อพบเหตุการณ์นี้ ระบบตรวจจับผู้บุกรุกแสดงข้อความเตือน ดังรูปที่ 4-12



รูปที่ 4-12 ผลการวิเคราะห์กรณีทดสอบที่ 5

4.2 ผลการทดลองส่วนตรวจจับการบุกรุกโดยตรวจสอบการใช้งานทรัพยากรระบบที่ผิดปรกติ

ในส่วนนี้ผลการตอบสนอง ได้ทดสอบการเตือนผู้ดูแลระบบผ่านทางจดหมายอิเล็กทรอนิกส์เมื่อมีผู้ใช้ในระบบใช้ทรัพยากรอย่างผิดปรกติ ตามรูปที่ 4-13



รูปที่ 4-13 ตัวอย่างจดหมายเตือนผู้ดูแลระบบ

บทที่ 5

วิเคราะห์ผลการทดลองและสรุป

5.1 วิเคราะห์ผลการทดลอง

จากการทดลองพบว่าพฤติกรรมการบุกรุกที่ระบบตรวจจับผู้บุกรุกระบบคอมพิวเตอร์สามารถตรวจจับได้มีดังนี้

- พยายามล็อกอินโดยเดาชื่อผู้ใช้ในระบบ
- พยายามล็อกอินเป็นผู้ใช้โดยเดารหัสผ่าน
- พยายามล็อกอินเป็นผู้ดูแลระบบโดยเดารหัสผ่าน
- พยายามเปลี่ยนสิทธิ์เป็นผู้ดูแลระบบโดยเดารหัสผ่าน
- แก้ไขไฟล์สำคัญในระบบ ซึ่งสามารถแก้ไขรายชื่อไฟล์สำคัญได้
- แก้ไขไฟล์ล็อกของระบบ ได้แก่ syslog และ sulog ไฟล์ใดไฟล์หนึ่ง เพื่อกลบเกลื่อนร่องรอย
- แก้ไขไฟล์กำหนดค่านโยบายการตรวจสอบระบบ (syslogd.conf)
- แก้ไขไฟล์ที่ยอมให้ผู้เรียกใช้ไฟล์นี้เปลี่ยนหมายเลขผู้ใช้ (uid) ซึ่งไฟล์นี้เป็นของรูต
- ล็อกอินจากเครื่องที่ไม่เคยใช้มาก่อน
- ล็อกอินจากสถานที่ที่ผิดปรกติ
- ล็อกอินของผู้ใช้ที่ไม่เคยล็อกอินมาก่อน
- ล็อกอินในเวลาผิดปรกติ
- ใช้งานหน่วยความจำมากผิดปรกติ
- ใช้งานหน่วยประมวลผลมากผิดปรกติ
- ใช้งานระบบเป็นระยะเวลานานผิดปรกติ
- ใช้ทรัพยากรโดยรวมมากผิดปรกติ
- สร้างประตูหลัง (Back door) เพื่อใช้เข้ามาในระบบครั้งต่อไปโดยสร้างผู้ใช้ใหม่ในระบบ

เมื่อตรวจพบพฤติกรรมดังกล่าว ระบบจะแจ้งเตือนแก่ผู้ดูแลระบบโดย

- แจ้งเตือนทางหน้าจอ เมื่อเรียกใช้ระบบตรวจจับผู้บุกรุกระบบคอมพิวเตอร์ส่วนเปรียบเทียบพฤติกรรม
- แจ้งเตือนทางอีเมล์ เมื่อตรวจพบการใช้งานทรัพยากรที่ผิดปรกติ
- บันทึกผลลงไฟล์ล็อก เมื่อตรวจพบพฤติกรรมการบุกรุกทั้งสองประเภท

ทั้งนี้ ระบบตรวจจับผู้บุกรุกระบบคอมพิวเตอร์ที่สร้างขึ้นไม่สามารถตรวจจับการบุกรุกในลักษณะต่อไปนี้

- การบุกรุกโดยทำให้เกิดบัฟเฟอร์โอเวอร์โฟลว์ เนื่องจากระบบไม่บันทึกล็อกของเหตุการณ์นี้ และผลของการบุกรุกโดยใช้บัฟเฟอร์โอเวอร์โฟลว์มีลักษณะไม่แน่นอน
- การบุกรุกจากภายนอกโดยติดต่อขอเข้าใช้ระบบด้วยแอพพลิเคชันใหม่ ๆ เนื่องจากล็อกในไฟล์ล็อกมีรูปแบบที่ระบบตรวจจับผู้บุกรุกไม่สามารถตรวจจับได้
- การบุกรุกผ่านทางเครื่องที่เชื่อถือกัน โดยใช้ .rhost
- การบุกรุกที่ไม่มีบันทึกล็อกในไฟล์ล็อกของระบบ ทั้งนี้อาจเกิดจากระบบไม่บันทึกเอง หรือระบบบันทึกเหตุการณ์นั้น แต่ผู้บุกรุกลบล็อกนั้นออกจากไฟล์ล็อก
- การบุกรุกรูปแบบอื่น ซึ่งเป็นวิธีใหม่ที่ไม่ได้ศึกษามา และไม่ได้ใช้ทรัพยากรระบบมากผิดปรกติ
- การบุกรุกที่ใช้ทรัพยากรระบบมาก แต่ไม่พบความผิดปรกติเนื่องจากโพรเซสไม่ได้ทำงานในช่วงที่ระบบตรวจจับ

สำหรับความถูกต้องแม่นยำในการตรวจจับนั้น ในส่วนตรวจจับโดยเปรียบเทียบพฤติกรรมทำงานผิดพลาดเมื่อวิเคราะห์พฤติกรรมการบุกรุกที่ไม่อยู่ภายในปีเดียวกัน เนื่องจากปรกติระบบจะบันทึกล็อกโดยระบุเวลาที่เกิดแต่เพียงวัน เดือน และเวลา ดังนั้นการตรวจจับด้วยวิธีนี้จึงไม่สามารถวิเคราะห์เหตุการณ์ที่เกิดขึ้นคนละปีได้ และด้วยสาเหตุเดียวกันทำให้เกิดความผิดพลาดของระบบขึ้นเมื่อเหตุการณ์นั้นเป็นวันที่ 29 กุมภาพันธ์

ความถูกต้องในส่วนตรวจจับโดยวิเคราะห์การใช้ทรัพยากรระบบขึ้นอยู่กับนโยบายในการตรวจจับของแต่ละองค์กร ซึ่งก็คือค่าเปอร์เซ็นต์แนวโน้มการบุกรุกต่ำสุดที่ยอมรับได้ และค่าอื่น ๆ ที่กำหนดไว้ในไฟล์ ids.dft (ดูรายละเอียดในหัวข้อ 3.3.1) ถ้าสามารถกำหนดค่าเหล่านี้ได้ตรงกับการใช้งานระบบในองค์กร ผลการวิเคราะห์ก็จะมีความถูกต้องสูง แต่ผลการวิเคราะห์ในส่วนนี้ไม่ถูกต้องทั้งหมด จากการทดลองพบว่ามีบางกรณีที่ผู้ใช้ใช้ทรัพยากรระบบมากผิดปรกติแต่ไม่ได้มีเจตนาบุกรุกระบบ และยังพบกรณีที่ผู้บุกรุกไม่ได้ใช้ทรัพยากรระบบเกินกว่าที่กำหนด

จากการทดลองไม่สามารถเปรียบเทียบประสิทธิภาพกับระบบตรวจจับผู้บุกรุกอื่นในปัจจุบันได้ เนื่องจากระบบตรวจจับผู้บุกรุกที่สร้างขึ้นนี้ได้ออกแบบให้ตรวจจับในส่วนไม่มีในระบบอื่น เพราะต้องการให้ทำงานเสริมกับระบบอื่น ๆ ทำให้ไม่เกิดการทำงานซ้ำซ้อนและประหยัดทรัพยากรระบบ

5.2 สรุปผล

การทำงานของระบบตรวจจับผู้บุกรุกที่สร้างขึ้นมีผลการทำงานเป็นที่น่าพอใจ ทั้งส่วนตรวจจับการบุกรุกโดยเปรียบเทียบพฤติกรรมผู้ใช้กับรูปแบบการบุกรุกที่รู้จัก และส่วนตรวจจับการบุกรุกโดยตรวจสอบการใช้งานทรัพยากรระบบที่ผิดปรกติ โดยให้ผู้เชี่ยวชาญระบบปฏิบัติการลีนุกซ์ทดสอบบุกรุกระบบ ระบบตรวจจับผู้บุกรุกสามารถตรวจจับได้ดีพอที่จะนำไปใช้งานจริงได้ ทั้งนี้การนำระบบตรวจจับผู้บุกรุกนี้ไปใช้งานจริงต้องทำตามขั้นตอนที่กำหนดอย่างเคร่งครัด เพื่อให้ได้ผลการวิเคราะห์ที่เที่ยงตรงที่สุด

ระบบตรวจจับผู้บุกรุกนี้ยังมีข้อจำกัดเนื่องจากการบันทึกล็อกของระบบไม่สามารถจับพฤติกรรมของผู้บุกรุกได้ทั้งหมด ทำให้ไม่สามารถระบุพฤติกรรมการบุกรุกทุกรูปแบบ หากต้องการบันทึกพฤติกรรมผู้ใช้ให้ครอบคลุมพฤติกรรมที่ต้องการนำมาวิเคราะห์ต้องสร้างตัวติดตามพฤติกรรมผู้ใช้ขึ้นเองและนำมาใช้สนับสนุนการทำงานของ syslogd

ปัจจุบันมีระบบตรวจจับผู้บุกรุกอื่นมากมาย ซึ่งระบบตรวจจับผู้บุกรุกที่สร้างขึ้นนี้ ผู้จัดทำสนใจเฉพาะในจุดที่ระบบตรวจจับผู้บุกรุกอื่นไม่ได้วิเคราะห์ ดังนั้นหากต้องการใช้ระบบตรวจจับผู้บุกรุกให้ได้ประสิทธิภาพสูงสุด ควรนำระบบตรวจจับผู้บุกรุกนี้ไปใช้ร่วมกับระบบตรวจจับผู้บุกรุกอื่น ๆ ด้วย

5.3 แนวทางในการพัฒนาระบบสำหรับผู้สนใจในอนาคต

1. การพัฒนาส่วนตรวจจับการบุกรุกโดยเปรียบเทียบพฤติกรรมผู้ใช้กับรูปแบบการบุกรุกที่รู้จัก
   - เพิ่มส่วนติดตามพฤติกรรมผู้ใช้มาสนับสนุนการทำงานของ syslogd เพื่อตรวจสอบพฤติกรรมที่ปรกติไม่ได้บันทึกลงไฟล์ล็อกของระบบ
   - พัฒนาส่วนวิเคราะห์ให้ทันสมัย โดยเพิ่มส่วนวิเคราะห์การบุกรุกแบบใหม่ ๆ
   - พัฒนาให้เป็นเรียลไทม์ (Real-time) เพื่อให้เตือนผู้ดูแลระบบได้ทันท่วงที
   - พัฒนาให้สามารถเรียนรู้รูปแบบการบุกรุกแบบใหม่ ๆ เองได้ (Self-learning)
2. การพัฒนาส่วนตรวจจับการบุกรุกโดยตรวจสอบการใช้งานทรัพยากรระบบที่ผิดปรกติ
   - พัฒนาสมการตามหลักสถิติให้คำนวณผลได้แม่นยำมากขึ้น
   - เพิ่มความถี่ในการวิเคราะห์เพื่อให้ใกล้เคียงเรียลไทม์มากที่สุด
   - พัฒนาให้ระบบตรวจจับผู้บุกรุกวิเคราะห์การใช้ทรัพยากรของผู้ใช้แต่ละคนแยกจากกัน
   - พัฒนาให้สามารถปรับเปลี่ยนเปอร์เซ็นต์แนวโน้มการบุกรุกเองได้ตามสภาพแวดล้อมและสภาพการใช้งานของผู้ใช้แต่ละคน
3. การพัฒนาโดยรวม
   - นำผลการวิเคราะห์ของทั้งสองส่วนมาวิเคราะห์ร่วมกันให้มากขึ้น
   - นำลักษณะการวิเคราะห์ของระบบตรวจจับผู้บุกรุกอื่น ๆ มารวมกับระบบนี้ เพื่อให้เป็นระบบตรวจจับผู้บุกรุกที่สมบูรณ์ และไม่ต้องใช้งานระบบตรวจจับผู้บุกรุกหลาย ๆ ระบบพร้อมกัน

ภาคผนวก ก

รายละเอียดของไฟล์ในการทำงานของระบบตรวจจับผู้บุกรุกระบบคอมพิวเตอร์

syslogd.conf

ไฟล์ที่ใช้ตั้งค่าการทำงานของ syslogd ซึ่งต้องเพิ่มการกำหนดค่าต่อไปนี้ลงไปด้วย

```
auth.*                          /var/adm/syslog/syslog
daemon.info;daemon.notice       /var/adm/syslog/syslog
kern.info                       /var/adm/syslog/syslog
```

โดยที่ /var/adm/syslog/syslog เป็นล็อกไฟล์ของระบบที่บันทึกโดย syslogd

suidshow.o

โปรแกรมที่เกิดจากการคอมไพล์ suidshow.c โดยใช้คำสั่ง gcc -c suidshow.c -o suidshow.o ทำหน้าที่บันทึกล็อกลง syslog เมื่อมีการเรียกใช้ฟังก์ชัน setuid ( ) เป็น 0 ในระบบ เช่น เมื่อมีการใช้คำสั่ง su เป็นรูตได้สำเร็จ

ปรกติ sulog เป็นที่เก็บบันทึกล็อกเมื่อมีการ su เป็น root สำเร็จ แต่ต้องบันทึกลง syslog ด้วยเพื่อให้ระบบตรวจจับผู้บุกรุกตรวจสอบข้อมูลจากล็อกไฟล์ทั้งสองว่าตรงกันหรือไม่ เนื่องจากอาจเกิดกรณีที่ผู้บุกรุกเข้ามาแก้ไขไฟล์ล็อกของระบบไฟล์ใดไฟล์หนึ่ง ทำให้ข้อมูลจากล็อกทั้งสองไม่ตรงกัน หากพบว่าข้อมูลการเรียกฟังก์ชัน setuid ( ) ของผู้ใช้คนใดคนหนึ่งในล็อกทั้งสองไม่ตรงกัน ระบบตรวจจับผู้บุกรุกจะระบุว่าผู้ใช้คนนั้นเป็นผู้บุกรุกทันที

keyfiles

รายชื่อไฟล์สำคัญของระบบ ตัวอย่างเช่น /etc/passwd

suidfiles

รายชื่อไฟล์ที่มีการตั้งค่าเพอร์มิสชันเป็น 4000 เป็นไฟล์ที่ยอมให้ผู้เรียกใช้เปลี่ยนหมายเลขผู้ใช้

logfiles

รายชื่อไฟล์ที่เป็นล็อกไฟล์ของระบบ เช่น sulog หรือ syslog

KeyfileInfo

ไฟล์ข้อมูลสำหรับเก็บค่าเพอร์มิสชัน, ชื่อเจ้าของ, กลุ่ม, ขนาด, เวลาเปลี่ยนสถานะครั้งสุดท้าย, และเช็คซัม ของไฟล์สำคัญที่มีรายชื่ออยู่ในไฟล์ชื่อ keyfiles เกิดจากการเรียกใช้สคริปต์ permsetup

SuidInfo

ไฟล์ข้อมูลสำหรับเก็บค่าเพอร์มิสชัน, ชื่อเจ้าของ, กลุ่ม, ขนาด, เวลาเปลี่ยนสถานะครั้งสุดท้าย, และเช็คซัม ของไฟล์ที่ยอมให้เปลี่ยนหมายเลขผู้ใช้ เกิดจากการเรียกใช้สคริปต์ permsetup ได้จากการใช้คำสั่ง

```
find / -perm -4000 -print > suidfiles
```

LogInfo

ไฟล์ข้อมูลสำหรับเก็บค่าเพอร์มิสชัน, ชื่อเจ้าของ, กลุ่ม, ขนาด, เวลาเปลี่ยนสถานะครั้งสุดท้าย, และเช็คซัม ของล็อกไฟล์ ซึ่งผู้ดูแลระบบต้องกำหนดรายชื่อล็อกไฟล์ลงในไฟล์ logfiles ก่อน เกิดจากการเรียกใช้สคริปต์ permsetup

permsetup

สคริปต์สร้างไฟล์สำหรับเก็บค่าเพอร์มิสชัน (Permissions), ชื่อเจ้าของ (Owner), กลุ่ม (Group), ขนาด (Size), เวลาที่เปลี่ยนสถานะครั้งสุดท้าย (Status change date), และเช็คซัม (Checksum) ของไฟล์สำคัญที่มีรายชื่ออยู่ใน keyfiles, suidfiles, และ logfiles แล้วนำข้อมูลค่าเหล่านี้ไปเก็บไว้ในไฟล์ KeyfileInfo, SuidInfo, และ LogInfo ตามลำดับ ผู้ดูแลระบบต้องเรียกสคริปต์นี้หลังจากมีการแก้ไขไฟล์สำคัญที่มีรายชื่ออยู่ในไฟล์ดังกล่าว เพื่อให้ระบบเก็บค่าที่ถูกต้องของไฟล์สำคัญเหล่านั้น ระบบจะนำค่าต่าง ๆ ที่เก็บไว้นี้มาเปรียบเทียบกับค่าปัจจุบันเมื่อเรียกสคริปต์ permck

permchk

สคริปต์ทำหน้าที่ตรวจสอบค่าเพอร์มิสชัน ชื่อเจ้าของ กลุ่ม ขนาด เวลาที่เปลี่ยนสถานะครั้งสุดท้ายและเช็คซัม ณ เวลาที่เรียกสคริปต์ กับค่าที่เก็บไว้ใน KeyfileInfo, SuidInfo, และ LogInfo ซึ่งผู้ดูแลระบบได้เก็บข้อมูลของไฟล์สำคัญไว้ ค่าที่พบ ณ เวลาปัจจุบันกับข้อมูลในไฟล์ดังกล่าวต้องตรงกัน ถ้าผู้ดูแลระบบเรียกใช้สคริปต์ permsetup ทุกครั้งที่แก้ไขไฟล์สำคัญที่ระบุไว้แล้วพบว่าข้อมูลไม่ตรงกัน อาจเกิดจากมีผู้บุกรุกเข้ามาแก้ไขไฟล์สำคัญเหล่านั้น สคริปต์นี้จะบันทึกล็อกลงไปใน syslog เพื่อเป็นข้อมูลในการวิเคราะห์ต่อไป

init

โปรแกรมสร้างข้อมูลเริ่มต้นของผู้ใช้ระบบทุกคน ข้อมูลเหล่านี้เป็นข้อมูลที่กำหนดขึ้นเพื่อใช้ในการวิเคราะห์หาผู้บุกรุก ได้แก่ ชื่อ, หมายเลข, กลุ่ม, สถานะ, สเตต, ตัวนับของโพรบต่าง ๆ , ไทม์แสตมป์ของโพรบต่าง ๆ , และ จำนวนครั้งในการล็อกอิน ข้อมูลมีรูปแบบเป็นสตรักเจอร์ของผู้ใช้แต่ละคน และไฟล์ที่เก็บได้ออกแบบให้เป็นไบนารีไฟล์ เพื่อให้ยากต่อการอ่านหากไม่ทราบรูปแบบของสตรักเจอร์ ข้อมูลนี้เก็บไว้ในไฟล์ userinfo

ผู้ดูแลระบบต้องเรียกโปรแกรมนี้ก่อนเรียกโปรแกรม midetect หรือเมื่อต้องการสร้างข้อมูลเริ่มต้นของผู้ใช้ทั้งหมดใหม่ หรือมีการเพิ่มผู้ใช้งานในระบบ

userinfo

ไฟล์ไบนารีที่เก็บข้อมูลของผู้ใช้แต่ละคน สร้างขึ้นเมื่อเรียกใช้โปรแกรม init

midetect

ไฟล์กรองและวิเคราะห์การบุกรุก รายละเอียดการทำงานอยู่ในบทที่ 3

blackhost

ไฟล์เก็บรายชื่อโฮสต์ที่ผู้ดูแลระบบต้องตรวจสอบเป็นพิเศษ ระบบตรวจจับผู้บุกรุกบันทึกโฮสต์ใด ๆ ลงในไฟล์นี้ก็ต่อเมื่อตรวจพบว่ามีผู้พยายามล็อกอินโดยใช้ชื่อผู้ใช้ที่ไม่มีอยู่ในระบบจากโฮสต์นี้เป็นจำนวนครั้งเกินกว่าที่ผู้ดูแลระบบกำหนดไว้ที่ตัวแปร MAX_BAD_USER ใน midetect.cpp ส่วน Max Value Configuration

blackuser

ไฟล์เก็บรายชื่อผู้ใช้ที่ผู้ดูแลระบบต้องตรวจสอบพฤติกรรมเป็นพิเศษ ระบบตรวจจับผู้บุกรุกบันทึกชื่อผู้ใช้ใด ๆ ลงในไฟล์นี้ก็ต่อเมื่อพบว่าผู้ใช้คนนั้นมีการกระทำดังต่อไปนี้อย่างใดอย่างหนึ่งหรือหลายอย่าง

- ใส่รหัสผ่านผิดเกินจำนวนครั้งที่ผู้ดูแลระบบกำหนดไว้ในตัวแปร MAX_BAD_PASSWD ใน midetect.cpp ส่วน Max Value Configuration
- ล็อกเอาต์ออกจากระบบในขณะที่มีสถานะเป็น S และอยู่ในสเตตที่ 3

blackroot

ไฟล์เก็บรายชื่อผู้ใช้ซึ่งได้สิทธิ์เป็นรูตหรือเคยได้สิทธิ์เป็นรูตและมีพฤติกรรมน่าสงสัย ผู้ใช้ที่ระบบตรวจจับผู้บุกรุกบันทึกชื่อลงในไฟล์นี้ก็ต่อเมื่อผู้ใช้มีพฤติกรรมดังต่อไปนี้อย่างใดอย่างหนึ่งหรือหลายอย่าง

- ผู้ใช้ที่อยู่ในกลุ่มรูตใส่รหัสผ่านผิดพลาดเกินจำนวนครั้งที่ผู้ดูแลระบบกำหนดไว้ในตัวแปร MAX_BAD_ROOT_PASSWD ใน midetect.cpp ส่วน Max Value Configuration
- ล็อกเอาต์ออกจากระบบในขณะที่มีสถานะเป็น SR และอยู่ในสเตตที่ 7
- ล็อกเอาต์ออกจากระบบในขณะที่มีสถานะเป็น SBR และอยู่ในสเตตที่ 11
- ได้รับการเพิ่มสิทธิ์ (การย้ายไปอยู่ในกลุ่มรูต) ขณะที่มีสถานะเป็น B และอยู่ในสเตตที่ 2

บรรณานุกรม

หนังสืออ้างอิง

[1] Kaare Christian: *"(Second Edition)The UNIX Operating System"*,John Wiley & Sons (sea) PTE LTD,1989

[2] David A Curry : *"C on the UNIX System"*,O'Reilly & Associates,Inc,1991

[3] (Editor) วศิน เพิ่มทรัพย์, สิดารัศมี ธำรงสมบัติสกุล : *"คู่มือติดตั้งและใช้งาน Linux "*,โครงการลีนุกซ์ภาษาไทย บริษัท ไกวัล ซอฟท์แวร์ จำกัด,2542

[4] นุกูล กระจาย : *"การเขียนโปรแกรมในดอสและวินโดวส์ด้วยบอร์แลนด์ C^{++} 5.0"*,บริษัท ซีเอ็ดยูเคชัน จำกัด (มหาชน),2540

[5] Rebecca Thomas PhD, Rik Farrow : *"UNIX Administration Guide for System V"*,Prentice Hall Englewood Cliffs New Jersey 07632,1989

[6] Simson Garfinkel and Gene Spafford : *"Practical UNIX Security"*,O'Reilly & Associates,Inc

[7] (ผู้แปล) ศรัณย์ อินทโกสม : *"ทฤษฎีและตัวอย่างโจทย์การเขียนโปรแกรมด้วยภาษาซี"*,แมคกรอ-ฮิล อินเตอร์เนชันแนล เอ็นเตอร์ไพร์ส, อิงค์,1990

[8] Evi Nemeth Garth Sryder Scott Seebass : *"UNIX System Administration Handbook"*,Prentice Hall Englewood Cliffs New Jersey 07632,1989

[9] Terry Escamila : *"Intrusion Detection Network Security Beyond the Fire Wall"*,John Wiley & Sons,Inc.,1998

[10] Kirk Waingrow : *"UNIX Hints & Hacks"*, Qve Corporation,1999

[11] Anonymous : *"Maximum Linux Security"*,Sams Publishing,2000

[12] สันติ ศรีลาศักดิ์, วรวุฒิ เที่ยงธรรม : *"เจาะประเด็นงานเขียนโปรแกรมบนลีนุกซ์"*,บริษัท ออฟเซ็ท เพรส จำกัด,2542

[13] มนตรี พจนารถลาวัณย์ : *"การเขียนโปรแกรมคอมพิวเตอร์ด้วยเทอร์โบซี"*, บริษัท ซีเอ็ดยูเคชั่น จำกัด (มหาชน),2540

เว็บไซต์อ้างอิง

[14] http://www.iss.net : *(White Paper)"Intrusion Detection in the Enterprise Network: Managing Hacker-Related Risk"*,Compaq Computer Corporation,March 1998

[15] http://www.ticm.com/kb/faq/idsfaq.html : Network Intrusion Detection Systems Frequently Asked Questions

[16] http://seclab.cs.ucdavis.edu/arpa/priv/welcome.html : Specification-based Detection for Unix Privileged Programs

[17] http://www.iss.net : Introduction to RealSecure Version 3.0

[18] http://www.cert.org/research/JHThesis/Chapter6.html : A Taxonomy of Computer and Network Attacks

[19] http://www.cert.org/research/JHThesis/Chapter7.html : Classification of Internet Incidents and Internet Activity

[20] http://www.cert.org/research/JHThesis/Chapter8.html : Methods of Operation and Corrective Actions

[21] http://www.cert.org/research/JHThesis/Chapter9.html : Case Study – Site A

[22] http://www.cert.org/research/JHThesis/Chapter10.html : Severe Incidents

[23] http://www.cert.org/research/JHThesis/Chapter6.html : A Taxonomy of Computer and Network Attacks

[24] Timothy Parker Ph.D., et al. : *"Slackware Linux (Third Edition)"*,SAMS Publishing,1997

[25] http://www.ieee.com : A System for Distributed Intrusion Detection

[26] http://www.ieee.com : A System for Distributed Intrusion Detection

[27] http://www.cs.nps.navy.mil/people/faculty/rowe/barruspap.htm : A Distributed Autonomous-Agent Network-Intrusion Detection and Response

[28] http://www.cert.org/ftp/tech_tips/intruder_detection_checklist : Intruder Detection Checklist

[29] http://www.csl.sri.com/intrusion.html : EMERALD: Event Monitoring Enabling Responses to Anomalous Live Disturbances

[30] http://seclab.cs.ucdavis.edu/cidf/papers/isw.txt : The Common Intrusion Detection Framework

[31] http://www.cert.org/security-improvement/modules/m05.html : Prepareing to Detect Signs of Intrusion

[32] http://www.cert.org/security-improvement/practices/p040.html : Establish a policy and set of procedures that prepare your organization to detect signs of intrusion

[33] http://www.cert.org/security-improvement/practices/p041.html : Identify and enable system and network logging mechanisms

[34] http://www.cert.org/security-improvement/practices/p042.html : Identify and install tools that aid in detecting signs of intrusion

[35] http://www.cert.org/security-improvement/practices/p043.html : Generate information required to verify the integrity of your systems and data

[36] http://www-rnks.informatik.tu-cottbus.de/~subirey/aid.e.html : The Intrusion Detection System AID

[37] http://www.cs.purdue.edu/coast/intrusion-detection/policy.htm : Security Policy