สำนักหอสมุดกลาง พระจอมเกล้าลาดกระบัง

ระบบตรวจจับผู้บุกรุกทางคอมพิวเตอร์

INTRUSION DETECTION SYSTEM



นายศราวุธ ฤทธิ์นันท์

นายศักดิ์ชัย ฉายสุวรรณ

สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

เลขหมู่.....

เลขทะเบียน 46137

วัน, เดือน, ปี 20 ส.ค. 2546

.b.....

.i.....

ปริญญานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิศวกรรมศาสตรบัณฑิต

ภาควิชาวิศวกรรมคอมพิวเตอร์

คณะวิศวกรรมศาสตร์

สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

ปีการศึกษา 2544

b11287780

ระบบตรวจจับผู้บุกรุกทางคอมพิวเตอร์

INTRUSION DETECTION SYSTEM

โดย

นายศราวุธ ฤทธิ์นันท์

นายศักดิ์ชัย ฉายสุวรรณ

อาจารย์ที่ปรึกษา

อาจารย์ธนา หงษ์สุวรรณ

อาจารย์อัครเดช วัชระภูพงษ์



ปริญญานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิศวกรรมศาสตรบัณฑิต

ภาควิชาวิศวกรรมคอมพิวเตอร์

คณะวิศวกรรมศาสตร์

สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

ปีการศึกษา 2544

ปริญญานิพนธ์ปีการศึกษา 2544

ภาควิชา วิศวกรรมคอมพิวเตอร์

คณะวิศวกรรมศาสตร์ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

เรื่อง ระบบตรวจจับผู้บุกรุกระบบคอมพิวเตอร์

INTRUSION DETECTION SYSTEM

ผู้จัดทำ

1. นายศราวุธ ฤทธิ์นันท์ รหัสนักศึกษา 41014410
2. นายศักดิ์ชัย ฉายสุวรรณ รหัสนักศึกษา 41014415



_____ อาจารย์ที่ปรึกษา

(นายธนา หงษ์สุวรรณ)

_____ อาจารย์ที่ปรึกษา

(นายอักรเดช วัชระภูพงษ์)

ระบบตรวจจับผู้บุกรุกระบบคอมพิวเตอร์

นายศราวุธ ฤทธิ์นันท์ 41014410
นายศักดิ์ชัย ฉายสุวรรณ 41014415
อ.ธนา หงษ์สุวรรณ อาจารย์ที่ปรึกษา
อ.อัครเดช วัชระภูพงษ์ อาจารย์ที่ปรึกษา
ปีการศึกษา 2544

บทคัดย่อ

ระบบปฏิบัติการลีนุกซ์เป็นระบบปฏิบัติการที่เป็นที่รู้จักทั่วโลกซึ่งมีอัตราการเจริญเติบโตสูงและมีแนวโน้มที่จะถูกนำมาใช้อย่างแพร่หลายในอนาคต ลีนุกซ์นั้นได้ถูกผ่านกระบวนการพัฒนามาเป็นเวลายาวนานโดยโปรแกรมเมอร์ทั่วโลกผ่านทางเครือข่ายอินเทอร์เน็ต แต่ว่าลีนุกซ์ยังมีปัญหาด้านความน่าเชื่อถือของระบบเนื่องมาจากเป็นซอฟต์แวร์ฟรีและเปิดเผยซอร์สโค้ดที่สามารถดาวน์โหลดได้ทางอินเทอร์เน็ต ดังนั้นลีนุกซ์จึงยังมีความปลอดภัยไม่เพียงพอสำหรับการใช้งานในองค์กรที่ต้องการความปลอดภัยสูง

ปริญญานิพนธ์ฉบับนี้จัดทำขึ้นเพื่อสร้างซอฟต์แวร์ในส่วนตรวจจับการบุกรุกโดยตรวจสอบการใช้งานระบบที่ผิดปกติ วิธีตรวจจับการบุกรุกตั้งอยู่บนสมมติฐานที่ว่าการกระทำใดๆ ที่เป็นการบุกรุกต้องมีการใช้งานระบบผิดไปจากเดิม โดยทำการสังเกตลักษณะการใช้งานคำสั่งของผู้ใช้ในช่วงระยะเวลาปัจจุบันกับฐานข้อมูลการใช้งานในอดีต โดยผู้ใช้แต่ละคนจะมีฐานข้อมูลแยกออกจากกัน หากตรวจพบการใช้งานที่ผิดปรกติก็แจ้งเตือนแก่ผู้ดูแลระบบ

Intrusion Detection System

Mr. Sarawut Ritnant
Mr. Sakchai Chaysuwan
Mr. Thana Hongsuwan Advisor
Mr. Akkradach Watcharapupong Advisor

ABSTRACT

Linux is well-known operating system which has high growth rate and trends to be used more widely in the future. It is developed by world wide programmer for a long time though internet. But the problem is its reliability because it is free opened source code software which can download from internet. Consequently Linux does not have enough security for organization that need high security.

The objective of this thesis is to build software which is able to detect intruder by using anomaly detection method. Anomaly detection is based on assumption that intruder must have abnormal behavior. This software monitors users' behaviors by compare characteristic and sequence of commands for each user with their own command database. If abnormal behavior detected, it will report to system administrator.

กิตติกรรมประกาศ

ปริญญานิพนธ์ฉบับนี้ไม่สามารถเสร็จสมบูรณ์ได้หากไม่ได้รับคำแนะนำ คำเตือน ควบคุม เอาใจใส่ ช่วยเหลือและดูแลการทำปริญญานิพนธ์นี้อย่างใกล้ชิดอย่างสูงจากอาจารย์ธนา หงษ์สุวรรณและอาจารย์อัครเดช วัชระภูพงษ์ นอกจากนี้ท่านอาจารย์ทั้งสองยังเป็นกำลังใจและแรงผลักดันให้ปริญญานิพนธ์ฉบับนี้เสร็จทันตามกำหนด สามารถเผยแพร่สู่สายตาสาธารณชนได้อย่างภาคภูมิใจ คณะผู้จัดทำขอขอบพระคุณยิ่งสำหรับทุกสิ่งทุกอย่างเป็นอย่างสูงไว้ ณ ที่นี้

นอกจากนี้ต้องขอขอบพระคุณอาจารย์ทุกท่านในสถาบันนี้ที่ได้สอนสั่งคณะผู้จัดทำจนมีความรู้ความสามารถจนถึงทุกวันนี้ รวมทั้งคณาจารย์ทุกท่านในภาควิชาวิศวกรรมคอมพิวเตอร์ที่ทำให้คณะผู้จัดทำได้เป็นวิศวกรคอมพิวเตอร์อย่างเต็มภาคภูมิ

ขอขอบคุณภาควิชาวิศวกรรมคอมพิวเตอร์โดยเฉพาะห้องวิจัยและพัฒนาการรักษาความปลอดภัยข้อมูล (Information Security Advisory Group : ISAG) ที่ได้เอื้อเฟื้อสถานที่ให้คณะผู้จัดทำได้ทำการวิจัยและช่วยอำนวยความสะดวกต่าง ๆ

ขอขอบคุณพี่ต่อ, พี่ตั้ม, พี่ก่อ, พี่ต้น, พี่เชาว์, พี่เกียง, พี่เก๋, พี่เดียร์, พี่หนึ่ง, พี่นัน, พี่ตูน, ตั้ม, ท๊อป, เก๋, เดียร์, จิ๊บ, ตั้ม, โอ๋, แอม, เมย์, ตั้มแมน, หมิน, โอ, เอ็มด้วยที่อุตส่าห์นอนเผื่อวันนั้นเราไม่ง่วงเลยนะ , ขอบคุณปอด้วยและบุคคลอื่นๆ ที่ไม่ได้กล่าวถึง ณ ที่นี้ ซึ่งอาจเกิดจากความหลงลืมของทางคณะผู้จัดทำนั้น ขอขอบคุณที่ให้ความช่วยเหลือคณะผู้จัดทำในการทำงานตลอดเวลา ให้คำปรึกษาในยามที่เกิดปัญหาขึ้น ให้ความรู้และสนับสนุนอุปกรณ์ในการทำงานต่าง ๆ ขอบคุณชุมนุมดนตรีสากลคณะวิศวกรรมศาสตร์ที่สร้างให้เติบโตมาขนาดนี้เรารักมากนะ

ประการสุดท้ายที่สำคัญที่สุด คือ ขอขอบพระคุณบิดา มารดาที่ได้ให้กำเนิด อบรม สั่งสอน และสนับสนุนการศึกษาและเป็นกำลังใจให้คณะผู้จัดทำอย่างดีมาโดยตลอด นับเป็นพระคุณที่เปรียบมิได้ ทางคณะผู้จัดทำขอขอบพระคุณและจักรำลึกพระคุณของท่านไว้ตลอดไป

ศราวุธ ฤทธิ์นันท์
ศักดิ์ชัย ฉายสุวรรณ

สารบัญ

	หน้า
บทคัดย่อภาษาไทย	I
บทคัดย่อภาษาอังกฤษ	II
กิตติกรรมประกาศ	III
สารบัญ	IV
สารบัญรูปภาพ	VI
สารบัญตาราง	VII
บทนำ	VIII
บทที่ 1 ทฤษฎีและหลักการเบื้องต้น	
1.1 ระบบตรวจจับผู้บุกรุกระบบคอมพิวเตอร์(Intrusion Detection System)	1
1.2 ข้อดีและข้อเสียของระบบตรวจจับผู้บุกรุก	1
1.3 พฤติกรรมโดยทั่วไปของผู้บุกรุก	5
1.4 รูปแบบของระบบตรวจจับผู้บุกรุกระบบคอมพิวเตอร์	7
1.5 แนวทางการปฏิบัติเมื่อมีการบุกรุกระบบ	8
1.6 ลีนุกซ์ (Linux)	9
บทที่ 2 โพรเซสและข้อมูลเกี่ยวกับโพรเซส	
2.1. ความหมายของโพรเซส	12
2.2. โครงสร้างของโพรเซส	12
2.3. การตรวจสอบข้อมูลของโพรเซส	14
2.4. การจัดเก็บข้อมูลใน /proc	15
2.5. ล็อกไฟล์ pacct	17
บทที่ 3 ลีนุกซ์เคอร์เนลโมดูลและซิสเต็มคอลล์(System call)	
3.1 ลีนุกซ์เคอร์เนลโมดูล	19
3.2 ซิสเต็มคอลล์	25
3.3 การดักการทำงานของ ซิสเต็มคอลล์	28
บทที่ 4 การทำงานของระบบ	
4.1 ภาพรวมของระบบตรวจจับผู้บุกรุก	31
4.2 โครงสร้างระบบตรวจจับผู้บุกรุกโดยวิธีการตรวจสอบการใช้งานระบบที่ผิดปรกติ	32
4.2.1 ส่วนเก็บข้อมูลพฤติกรรมผู้ใช้งาน	32
4.2.2 ส่วนเก็บข้อมูลและวิเคราะห์พฤติกรรมผู้ใช้งาน	34
4.2.3 ส่วนตอบสนองการวิเคราะห์	39
บทที่ 5 การปรับค่าพารามิเตอร์ของระบบและผลการทดลอง	
5.1 การปรับค่าพารามิเตอร์ของระบบ	41

หน้า

5.1.1 การทดลองหาค่าพารามิเตอร์ 3 ค่าแรกที่เหมาะสม 41
5.1.2 การหาค่าต่ำสุดที่ยอมรับได้ของการเปลี่ยนแปลงพฤติกรรม 46
5.2 ทดลองบุกรุกโดยให้มีพฤติกรรมการใช้งานที่ผิดปรกติของผู้ใช้ 47
บรรณานุกรม 51

สารบัญรูป

หน้า

รูปที่ 1.1 การทำงานของการตรวจจับผู้บุกรุกโดยวิธีตรวจสอบการใช้งานระบบที่ผิดปรกติ 8
รูปที่ 2.1 แสดงโครงสร้างอย่างคร่าว ๆ ของโพรเซส 12
รูปที่ 2.2 โครงสร้างของ process descriptor ในไฟล์ /usr/include/linux/sched.h 13
รูปที่ 2.3 ลักษณะของการเชื่อมโยงกันของ process descriptor 14
รูปที่ 2.4 แสดงการผลลัพธ์จากการรันคำสั่ง ps 14
รูปที่ 2.5 แสดงผลลัพธ์ของการรันคำสั่ง top 15
รูปที่ 2.6 แสดงสถิติของผู้ใช้งานโดยคำสั่ง sa 18
รูปที่ 3.1 แสดงความสัมพันธ์ของการเรียก ซิสเต็มคอลล์ จากแอพพลิเคชัน 26
รูปที่ 3.2 แสดงวิธีดักการทำงานของซิสเต็มคอลล์ 30
รูปที่ 4.1 ภาพรวมการทำงานของระบบตรวจจับผู้บุกรุก 32
รูปที่ 4.2 ขั้นตอนการทำงานของส่วนเก็บข้อมูลพฤติกรรม 33
รูปที่ 4.3 ภาพรวมของส่วนเก็บข้อมูลและวิเคราะห์พฤติกรรมผู้ใช้งาน 34
รูปที่ 4.4 ขั้นตอนการทำงานของส่วนเก็บข้อมูลและวิเคราะห์พฤติกรรมผู้ใช้งาน 35
รูปที่ 4.5 แสดงข้อมูลที่อยู่ในไฟล์ blacklist 39
รูปที่ 4.6 แสดงข้อมูลที่อยู่ในไฟล์ eventlist 40
รูปที่ 5.1 การทดลองหาค่าพารามิเตอร์โดยให้ค่า DB_WINDOW_SIZE=100 42
รูปที่ 5.2 การทดลองหาค่าพารามิเตอร์โดยให้ค่า DB_WINDOW_SIZE=200 43
รูปที่ 5.3 การทดลองหาค่าพารามิเตอร์โดยให้ค่า DB_WINDOW_SIZE=400 44
รูปที่ 5.4 การทดลองหาค่าพารามิเตอร์โดยให้ค่า DB_WINDOW_SIZE=700 45
รูปที่ 5.5 แสดงผลของการทำงาน ping sweeps 48
รูปที่ 5.6 แสดงผลการทำงานของ IDS host 48
รูปที่ 5.7 แสดงผลการทำงานของ IDS host เมื่อตรวจพบการบุกรุก 49
รูปที่ 5.8 บันทึกข้อมูลการบุกรุกลงไฟล์ blacklist 49
รูปที่ 5.9 แสดงไฟล์ eventlist 50

สารบัญตาราง

หน้า

ตารางที่ 1.1 เทคโนโลยีและข้อมูลสำคัญที่แฮ็กเกอร์ ต้องการค้นหา 6

ตารางที่ 2.1 แสดงความหมายของข้อมูลในไฟล์ /proc/[number]/stat 16

ตารางที่ 2.2 แสดงความหมายของตัวแปรในโครงสร้าง struct acct 18

ตารางที่ 3.1 แสดงรายชื่อของฟังก์ชันใน Kernel space ที่ใช้บ่อย 24

ตารางที่ 3.2 แสดงรายชื่อ ซิสเต็มคอลล์ ที่ใช้บ่อย 26

ตารางที่ 4.1 แสดงการคำนวนค่าความเหมือนของลำดับ 38

ตารางที่ 5.1 ผลการทดลองการเปรียบเทียบพฤติกรรม 47

บทนำ

ที่มาของโครงงาน

ในปัจจุบันระบบการรักษาความปลอดภัยบนคอมพิวเตอร์มีความสำคัญมากขึ้น เนื่องจากการพัฒนาอินเทอร์เน็ตและระบบเครือข่ายไปอย่างรวดเร็ว ทำให้มีผู้ใช้มากขึ้นซึ่งย่อมมีผู้ใช้แบบปรกติและผู้ใช้ที่มีลักษณะที่ผิดปรกติ เช่น การโจรกรรมข้อมูลในระบบ การทำลายข้อมูลที่มีความสำคัญต่อองค์กร ฯลฯ ได้มีความตระหนักในเรื่องของความปลอดภัยและความเป็นส่วนตัวมานานแล้ว ดังจะเห็นได้จากความพยายามในการสร้างระบบรักษาความปลอดภัยสำหรับงานต่างๆ ขึ้นมา หนึ่งในนั้น คือ ระบบตรวจจับผู้บุกรุกระบบคอมพิวเตอร์ในระบบปฏิบัติการลีนุกซ์ ซึ่งเป็นระบบที่ช่วยเพิ่มความปลอดภัยให้กับคอมพิวเตอร์มากขึ้น

ระบบตรวจจับผู้บุกรุกระบบคอมพิวเตอร์นี้เป็นการป้องกันความไม่ถูกต้อง ความไม่เหมาะสม หรือการทำงานที่ผิดปรกติ โดยเป็นการปฏิบัติการป้องกันการทำงานที่ประสงค์ร้ายต่อฝั่งโฮสต์ เรียกว่า Host-based Intrusion Detection System และหากปฏิบัติการบนการไหลข้อมูลในเครือข่าย เรียกว่า Network-based Intrusion Detection System ในโครงงานนี้เป็นการสร้างระบบตรวจจับผู้บุกรุกในแบบ Host-based ซึ่งเกี่ยวกับการติดตามพฤติกรรมการใช้งานของผู้ใช้และนำพฤติกรรมมาวิเคราะห์หาความเป็นไปได้ในการบุกรุก

วัตถุประสงค์

1. สร้างระบบตรวจจับผู้บุกรุกระบบคอมพิวเตอร์บนระบบปฏิบัติการลีนุกซ์
2. ศึกษาการทำงานของระบบปฏิบัติการลีนุกซ์
3. เน้นการใช้วิธีตรวจจับแบบตรวจสอบการใช้งานระบบที่ผิดปรกติ(Anomaly Intrusion Detection)
4. ศึกษาและเขียนแอพพลิเคชันบนระบบปฏิบัติการลีนุกซ์

เป้าหมายของการพัฒนา

1. เพื่อพัฒนาระบบตรวจจับผู้บุกรุกระบบคอมพิวเตอร์แบบตรวจสอบการใช้งานทรัพยากรระบบที่ผิดปรกติให้มีความปลอดภัย สมบูรณ์ ครบถ้วนมากขึ้น โดยเน้นส่วนของการเก็บข้อมูล
2. เพื่อสนับสนุนการพัฒนาซอฟต์แวร์ประเภทส่วนรักษาความปลอดภัยให้พัฒนามากยิ่งขึ้น
3. เพื่อสนับสนุนการพัฒนาซอฟต์แวร์บนระบบปฏิบัติการลีนุกซ์ในแนวทางของ Open source

ขอบเขตของงานวิจัย

ในโครงงานวิจัยนี้ได้สร้างระบบตรวจจับผู้บุกรุกระบบคอมพิวเตอร์สำหรับระบบปฏิบัติการลีนุกซ์ Mandrake 8.0 และมีการตรวจจับการบุกรุกเป็นการตรวจสอบการใช้งานทรัพยากรระบบที่ผิดปรกติ ซึ่งเป็นพฤติกรรมของการบุกรุกที่อาศัยทฤษฎีว่า การบุกรุกต่างๆ นั้นมีการใช้ทรัพยากรระบบอย่างผิดปรกติโดยมีส่วนของการเก็บข้อมูลผู้ใช้ปรกติไว้แล้วนำมาเปรียบเทียบกับพฤติกรรมที่ตรวจพบว่าผิดปรกติหรือไม่ การตรวจจับแบบนี้มีข้อจำกัดตรงที่ไม่สามารถระบุได้แน่นอนว่าการใช้งานระบบที่ผิดปรกตินั้นเกิดจากการใช้งานของผู้ใช้ธรรมดาหรือเกิดจากการพยายามบุกรุกเข้ามาในระบบ เพียงแต่สามารถรายงานผลว่าเกิดการใช้งานระบบอย่างผิดปรกติขึ้น

นอกจากข้อจำกัดที่ได้กล่าวมาข้างต้น ยังมีข้อจำกัดของโครงงานวิจัยอีกอย่างหนึ่งคือข้อจำกัดเกี่ยวกับการบุกรุก คือ หากไม่มีการใช้ทรัพยากรที่ผิดปรกติระบบตรวจจับผู้บุกรุกก็ไม่สามารถตรวจจับได้

วิธีดำเนินงาน

1. ศึกษาทฤษฎีความรู้พื้นฐานเกี่ยวกับระบบปฏิบัติการลีนุกซ์และระบบตรวจจับผู้บุกรุกระบบคอมพิวเตอร์
2. ศึกษาวิธีการที่จะได้มาถึงข้อมูลลักษณะพฤติกรรมการใช้งานระบบของผู้ใช้
3. ออกแบบระบบตรวจจับผู้บุกรุกและเขียนโปรแกรม
4. ทดสอบการใช้งานและปรับปรุงโปรแกรมระบบตรวจจับผู้บุกรุกที่เขียนขึ้น

สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

บทที่ 1
ทฤษฎีและหลักการเบื้องต้น

1.1 ระบบตรวจจับผู้บุกรุกระบบคอมพิวเตอร์ (Intrusion Detection System: IDS)

ระบบตรวจจับผู้บุกรุกระบบคอมพิวเตอร์เป็นกระบวนการตรวจสอบเครือข่ายและระบบคอมพิวเตอร์เมื่อมีการละเมิดกฎการรักษาความปลอดภัย โดยระบบตรวจจับผู้บุกรุกระบบประกอบด้วยหน้าที่หลัก 3 ประการ คือ รวบรวมข้อมูลเหตุการณ์ที่เกิดขึ้นไว้ในเรคอร์ด มีตัววิเคราะห์ในการตรวจจับผู้บุกรุก และส่วนตอบโต้ผู้บุกรุก

คำว่า ผู้บุกรุก หมายถึง ผู้ที่ชอบหาช่องโหว่ของโปรแกรมและเจาะเข้าไปในระบบคอมพิวเตอร์ รวมทั้งหมายถึง แฮ็กเกอร์ คือ ผู้ที่ชอบเข้าไปศึกษาบางสิ่งบางอย่างในระบบ

1.1.1 ความจำเป็นที่ต้องมีระบบตรวจจับผู้บุกรุกระบบคอมพิวเตอร์

เป็นที่รู้กันดีว่าระบบคอมพิวเตอร์และเครือข่ายมีการออกแบบที่ไม่ค่อยปลอดภัยมากนักทำให้ผู้บุกรุกมีโอกาสที่จะบุกรุกระบบได้ง่ายแม้ว่ามีไฟร์วอลล์ (Firewall) อยู่แล้วก็ตามแต่ก็ไม่สามารถป้องกันการบุกรุก 100 เปอร์เซ็นต์ เนื่องจากไฟร์วอลล์เป็นการป้องกันระหว่างระบบคอมพิวเตอร์ภายในเครือข่าย (Internal networks) กับระบบภายนอก (Internet) หากผู้บุกรุกเป็นคนภายในไฟร์วอลล์เอง ก็ไม่สามารถป้องกันได้ ดังนั้นการมีไฟร์วอลล์เปรียบเสมือนการมีรั้วกั้นรอบบ้าน และระบบตรวจจับผู้บุกรุกระบบคอมพิวเตอร์เปรียบเสมือนการติดตั้งกล้องวีดีโอคอยตรวจสอบสิ่งผิดปกติทำให้การป้องกันระบบทำได้ดียิ่งขึ้น

1.2 ข้อดีและข้อเสียของระบบตรวจจับผู้บุกรุก

1.2.1 ข้อดีของระบบตรวจจับผู้บุกรุก

1. การตอบสนองทันทีทันใด

จริง ๆแล้วการวิเคราะห์การบุกรุกนั้น หากเป็นผู้เชี่ยวชาญที่มีความรู้ความเข้าใจด้านเน็ตเวิร์กและโพรโตคอลเป็นอย่างดีก็จะวิเคราะห์ได้โดยอาศัยเครื่องมือเพียงเล็กน้อยเท่านั้น คือ ใช้เครื่องมือทำการจัดเก็บบันทึกข้อมูลทั้งหมดที่มีการสื่อสารกันบนเน็ตเวิร์ก แล้วนำข้อมูลที่ได้เหล่านั้นมาวิเคราะห์โดยพฤติกรรมและความสัมพันธ์ ก็จะสามารถหาสิ่งผิดปกติที่เกิดขึ้นได้ แต่การวิเคราะห์ในลักษณะดังกล่าวจะกระทำได้ก็ต่อเมื่อได้เกิดเหตุการณ์ไปแล้ว เนื่องจากการวิเคราะห์จะเป็นไปในลักษณะการวิเคราะห์ข้อมูลย้อนหลัง ไม่สามารถจะกระทำได้ในทันที ซึ่ง IDS จะช่วยแก้ไขข้อบกพร่องในส่วนนี้ เพราะ IDS สามารถตรวจจับได้ทันทีที่มีความผิดปกติเกิดขึ้น และช่วยให้ทำการแก้ไขได้ทันท่วงที การทำงานพื้นฐานของ IDS จะเหมือนกับการที่ทำโดยคน เพียงแต่ IDS นั้นทำงานโดยอัตโนมัติและการทำงานอยู่ตลอดเวลาไม่มีหยุด จึงสามารถตอบสนองต่อสิ่งผิดปกติได้รวดเร็วกว่า ซึ่งถ้าให้คนมานั่งตรวจจับก็ไม่สามารถทำได้ตลอดเวลา

2. การมีฐานความรู้ของการวิเคราะห์

จากที่ได้กล่าวมาข้างต้น การที่จะตรวจจับสิ่งผิดปกติและแยกแยะกิจกรรมเหล่านั้นออกจากการสื่อสารข้อมูลตามปกติได้นั้นจะต้องอาศัยความชำนาญ และเข้าใจในรูปแบบของการสื่อสารข้อมูลและการ

บุกรุกเป็นอย่างดี นั่นคือทักษะที่จำเป็นของนักวิเคราะห์การบุกรุก (Intrusion Analyst) ซึ่งผู้เชี่ยวชาญในระดับที่จะทำงานเช่นนี้ได้มีไม่มากนัก ประกอบกับเทคนิคและกลวิธีในการบุกรุกหรือก่อกวนนั้นได้พัฒนาขึ้นทุกวัน วิธีการตรวจจับและวิเคราะห์จำเป็นต้องพัฒนาตามให้สอดคล้องกันจึงจะตรวจจับได้อย่างมีประสิทธิผล ซึ่งในส่วนนี้ผู้เชี่ยวชาญเองก็อาจจะทำได้ไม่ดีเท่า

IDS สามารถช่วยแบ่งเบาภาระของนักวิเคราะห์ลงได้มาก โดยหากรู้รูปแบบพฤติกรรมแน่ชัดว่าเป็นการมุ่งร้ายก็ให้จัดเก็บข้อมูลรูปแบบเหล่านี้ใน IDS เสีย เมื่อมีกิจกรรมดังกล่าวเกิดขึ้นในเน็ตเวิร์ก IDS ก็สามารถตรวจพบได้ทันที และเมื่อค้นพบรูปแบบใหม่ก็จัดเก็บลงใน IDS อีก ทำให้ IDS เสมือนมีฐานความรู้ในการวิเคราะห์การบุกรุกได้ดีในระดับหนึ่ง และขีดความสามารถก็จะเพิ่มขึ้นเรื่อย ๆตามปริมาณของรูปแบบที่เก็บอยู่ในฐานความรู้นั่นเอง หากมีการบำรุงรักษาฐานความรู้ในตัว IDS ได้ดีและนำ IDS ไปใช้ในจุดที่เหมาะสมแล้ว การบุกรุกที่ไม่ใช่เทคนิคใหม่ล่าสุดจริง ๆก็แทบจะไม่สามารถเล็ดรอดสายตา IDS ไปได้ ถึงขั้นนี้แล้วแม้ว่าจะเป็น IDS แบบธรรมดา ๆก็มีความสามารถมากกว่าผู้บริหารระบบทั่วไปเสียอีก

สำหรับนักวิเคราะห์แล้วเมื่อมี IDS จะทำให้ไม่ต้องห่วงหน้าพะวงหลัง เพราะการบุกรุกที่สามารถตรวจจับได้ง่าย ๆก็สามารถตรวจพบได้โดย IDS อย่างน้อย IDS ก็ช่วยกลั่นกรองข้อมูลเบื้องต้นได้ในระดับหนึ่งและแบ่งเบาภาระได้พอสมควร

3. การช่วยตรวจสอบข้อบกพร่องของระบบป้องกันอื่น ๆ

เน็ตเวิร์กของผู้ใช้อาจมีการป้องกันการบุกรุกอยู่แล้วโดยใช้ไฟร์วอลล์ (Firewall) อย่างไรก็ตามไฟร์วอลล์มิใช่เครื่องมือที่จะป้องกันการบุกรุกได้โดยอัตโนมัติ จะต้องอาศัยผู้ที่บริหารระบบกำหนดกฎให้เหมาะสมกับการใช้งาน อีกประการหนึ่ง ถึงแม้จะมีการตั้งกฎที่เหมาะสมแล้วก็ตาม แต่กฎเหล่านั้นอาจไม่สามารถป้องกันการบุกรุกได้ การบริหารไฟร์วอลล์ที่ดีก็ควรจะมีการตรวจสอบย้อนหลัง (Audit) และการทดสอบการเจาะระบบ (Penetration Test) เพื่อเป็นการสอบทานระบบอีกครั้งหนึ่ง

IDS สามารถช่วยได้มาก โดยติดตั้ง IDS ไว้หลังไฟร์วอลล์ และทำการทดสอบเจาะระบบด้วยวิธีต่าง ๆเพื่อดูว่าจะมีเทคนิคใดที่สามารถเจาะผ่านไฟร์วอลล์ได้บ้าง และหากมีแพ็กเกจใดผ่านเข้าไปได้ IDS ก็จะตรวจพบ ทำให้ผู้บริหารระบบสามารถปรับปรุงกฎให้รัดกุมมาก

1.2.2 ข้อเสียของระบบตรวจจับผู้บุกรุก

ถึงแม้ IDS จะมีประโยชน์ค่อนข้างมากในการช่วยรักษาความปลอดภัยและการเตือนภัยล่วงหน้า แต่ก็มีข้อเสียอยู่หลายประการซึ่งผู้ที่จะนำไปใช้จะต้องตระหนักไว้

1. การละเมิดความเป็นส่วนบุคคล

เนื่องจาก IDS มีพื้นฐานจากการนำข้อมูลทั้งหมดที่สื่อสารกันมาทำการวิเคราะห์ ซึ่งข้อมูลเหล่านั้นจะต้องครอบคลุมถึงข้อมูลทั่วไปที่มีการสื่อสารกันตามปกติ และการที่จะทราบว่ามีความผิดปกติหรือไม่นั้นก็จะต้องอ่านข้อมูลทั้งหมดด้วย ดังนั้นไม่ว่าจะมีกิจกรรมใด ๆที่เกิดขึ้นในเน็ตเวิร์กไม่ว่าจะเป็นการท่องเว็บ การดาวน์โหลดข้อมูล การแชทคุยกัน ไอซีคิว อี-เมล์ และกิจกรรมอื่น ๆที่สื่อสารข้อมูลผ่านเน็ตเวิร์ก ก็จะสามารถถูกเปิดอ่านได้จาก IDS นั่นหมายความว่า IDS สามารถนำไปใช้ในทางที่ผิดเพื่อละเมิด

สิทธิส่วนบุคคลได้ การทำงานของ IDS เปรียบเสมือนการที่ตำรวจต้องการตรวจสอบและดักจับผู้ไม่หวังดีที่คอยโทรศัพท์ก่อกวนชาวบ้านในหมู่บ้าน และเพื่อการนี้ตำรวจจึงต้องทำการดักฟังโทรศัพท์ของทุกคนที่อยู่ในหมู่บ้านนั้น ซึ่งอาจจะมีเพียงหนึ่งในพันที่เป็นผู้ร้าย แต่ตำรวจผู้ทำหน้าที่ดักฟังก็จะรู้ความลับของคนทุกคน บางทีการที่มีคนดักฟังความลับของทุกคนอาจเป็นอันตรายกว่าการโดนผู้ร้ายก่อกวนก็เป็นได้

ดังนั้นการนำ IDS มาติดตั้งในเน็ตเวิร์กจะต้องได้รับการอนุมัติจากหน่วยงานอย่างถูกต้องแล้วเท่านั้น และผู้ทำหน้าที่ในด้านนี้จะต้องเป็นผู้ที่ได้รับความไว้วางใจและมีความรับผิดชอบสูงในอันที่จะไม่ละเมิดสิทธิส่วนบุคคลของคนอื่น และหากเห็นข้อมูลใด ๆก็จะต้องไม่เปิดเผยข้อมูลเหล่านั้นแก่บุคคลนั้น โดยทั่วไปแล้วการติดตั้งอุปกรณ์ที่สามารถอ่านข้อมูลของผู้อื่นบนเน็ตเวิร์กได้นั้นจะเป็นข้อห้ามอันดับต้น ๆในนโยบายรักษาความปลอดภัยเลยทีเดียว สิ่งที่เป็นข้อสังเกต คือ การกระทำในลักษณะนี้ยากต่อการป้องกันในทางเทคนิค ดังนั้นหน่วยงานโดยทั่วไปจึงต้องกำหนดเป็นข้อห้ามในนโยบายความปลอดภัย และมีบทลงโทษสำหรับผู้ที่ละเมิดในขั้นรุนแรง

2. การตอบโต้อัตโนมัติ

IDS ที่มีจำหน่ายอยู่ในท้องตลาดจะมีส่วนหนึ่งที่ให้ผู้ใช้สามารถกำหนดการดำเนินการอย่างหนึ่งอย่างใดเมื่อตรวจพบการบุกรุกเกิดขึ้น เช่น ส่งจดหมายเตือนผู้ดูแลระบบ เรียกวิทยุติดตามตัว ส่งคำสั่งไปยังไฟร์วอลล์เพื่อจำกัดการเข้าออกของข้อมูล และสิ่งที่สำคัญที่สุดซึ่งอาจจะส่งผลเสียหายใหญ่หลวงต่อเจ้าของได้ก็คือ การโจมตีกลับไปยังต้นกำเนิดของการบุกรุก (counter attack) โดยที่ IDS เองก็จะรู้จักวิธีการโจมตีแบบต่าง ๆดีอยู่แล้ว จึงมิใช่เรื่องยากเย็นแต่อย่างใดที่จะทำการโจมตีผู้อื่น ผู้ผลิตจึงมักเพิ่มเติมส่วนนี้ให้แก่ IDS เสมือนหนึ่งการติดอาวุธ ๆไว้ให้ต่อกรกับแฮ็กเกอร์ เลยทีเดียว

ผู้ดูแลระบบบางส่วนอาจรู้สึกสะใจและคิดว่าเหมาะสมแล้วกับการโจมตีกลับไปยังแฮ็กเกอร์เหล่านั้นให้หลาบจำจะได้ไม่พยายามมาข้องแวะอีก เป็นนโยบายการรักษาความปลอดภัยแบบตาต่อตาฟันต่อฟัน และเชื่อว่าหากกำหนดให้การโจมตีกลับเป็นไปอย่างอัตโนมัติแล้วน่าจะทำให้ปลอดภัยมากขึ้น ในค่ำคืนที่เงียบสงบใครจะไปรู้ว่า IDS อาจจะกำลังต่อกรอยู่กับแฮ็กเกอร์ ที่พยายามแอบเข้ามาในระบบอย่างสุดกำลัง และสู้ยิบตาเพื่อรักษามิให้แฮ็กเกอร์ บุกรุกเข้ามาในเน็ตเวิร์กได้ ในโลกแห่งความเป็นจริงแล้วการตัดสินใจว่าผู้ใดเป็นแฮ็กเกอร์ อย่างชัดเจนมิได้ทำได้โดยง่ายและในเวลาอันรวดเร็ว การที่กำหนดให้ IDS ทำการตอบโต้กลับไปในทันทีโดยมีข้อมูลเพียงผิวเผินนั้น นอกจากจะไม่ช่วยให้เน็ตเวิร์กของเราปลอดภัยแล้ว ยังจะทำให้เรากลายไปเป็นแฮ็กเกอร์ ที่คอยโจมตีผู้อื่นเสียเอง ยกตัวอย่างความเสียหาย เช่น

- การวิเคราะห์ผิดพลาดเข้าใจว่ากิจกรรมที่เกิดขึ้นเป็นการบุกรุกและ IDS ก็ดำเนินการโจมตีกลับไปในทันที กรณีนี้ผู้บริสุทธิ์ก็จะถูกโจมตีจาก IDS ของเราโดยที่ไม่รู้เรื่องใด ๆ
- การวิเคราะห์ถูกต้องแต่แอดเดรสของต้นทางเป็นแอดเดรสปลอม กรณีนี้หาก IDS ไม่มีกลไกในการสอบทานแอดเดรสที่มีประสิทธิภาพ อาจไม่สามารถแยกแยะได้ว่าต้นทางของการโจมตีแท้จริงนั้นเป็นที่ไหน และเมื่อทำการโจมตีกลับไปก็อาจจะมิใช่ตัวการที่แท้จริง และเหตุการณ์จะเลวร้ายยิ่งขึ้นหากแอดเดรสที่ปลอมมานั้นเป็นของหน่วยงานทางความมั่นคงหรือ

หน่วยงานทางทหาร และเมื่อนั้นผู้ดูแลระบบอาจจะตระหนักได้ว่า IDS ตัวเดียวอาจจะทำให้เขาต้องเข้าไปนอนในคุกหลายคืน เทคนิคการปลอมแอดเดรสลักษณะนี้อาจเป็นการยืมมือ IDS ของเราไปโจมตีผู้อื่นอีกทอดหนึ่งได้เป็นอย่างดี

- การวิเคราะห์แอดเดรสที่ถูกต้อง และการโจมตีกลับไปก็ตรงไปยังแฮ็กเกอร์ อย่างถูกต้องตามที่ต้องการ แต่ผลที่ได้ก็เพียงอาจจะทำให้แฮ็กเกอร์ หยุดความพยายามไปชั่วขณะเท่านั้น อีกไม่นานก็จะหาวิธีกลับมาใหม่ และไม่เกิดผลใด ๆเลยนอกจากจะเป็นการยั่วยุให้มีความรุนแรงมากขึ้นเท่านั้น

สิ่งสำคัญที่ผู้ทำหน้าที่ด้านความปลอดภัยและผู้บริหารระบบควรจะตระหนักไว้ให้จงหนักคือท่านไม่มีสิทธิพิเศษที่จะไปตอบโต้ผู้บุกรุกโดยการโจมตีกลับไม่ว่าในกรณีใด สิ่งที่ท่านจะทำได้ดีที่สุด คือ ทำระบบให้แข็งแรงมั่นคงและปลอดภัยที่สุดเท่านั้น นั่นคือ ปิดประตูบ้านให้แน่น ตรวจตราอย่างรัดกุมและใช้งานเท่าที่จำเป็น ส่วนผู้ที่กระทำผิดเหล่านั้นควรจะปล่อยให้เป็นไปตามกฎหมายและกระบวนการยุติธรรมจะดีที่สุด เพราะการตอบโต้การกระทำที่ผิดกฎหมายด้วยวิธีที่ผิดกฎหมายจะทำให้เรากลายเป็นจำเลยไปด้วยในที่สุด

3. การเตือนภัยที่ผิดพลาด

ข้อนี้อาจไม่ใช่ข้อเสียที่สำคัญของการใช้ IDS หากผู้ใช้มีความรู้ในการใช้งานที่ดีพอและเข้าใจหลักการวิเคราะห์การบุกรุกของ IDS ได้ดี อย่างที่ได้กล่าวแล้วข้างต้น ก็คือ อาจจะมีกิจกรรมปกติหลายอย่างที่มีลักษณะใกล้เคียงหรือบางครั้งเหมือนกับการพยายามบุกรุก ซึ่งแน่นอนว่าหาก IDS ได้ถูกกำหนดให้ตรวจจับกิจกรรมประเภทดังกล่าวแล้วก็จะมีการเตือนในทันทีที่ตรวจพบและเป็นหน้าที่ของนักวิเคราะห์ที่จะทำการสืบค้นข้อมูลด้านอื่นๆ มาประกอบการวินิจฉัยอีกครั้งหนึ่งว่าพฤติกรรมดังกล่าวที่ตรวจพบนั้นเป็นการบุกรุกหรือไม่ อย่างไร IDS ที่ถูกกำหนดให้มีความไวเป็นพิเศษมักจะสามารถตรวจจับพฤติกรรมที่ก้ำกึ่งนั้นได้มากเป็นพิเศษ ตัวอย่างเช่น IDS ได้ถูกกำหนดไว้ว่า เมื่อได้รับ Ping Packet จากแอดเดรสเดิมติดต่อกัน 10 แพ็กเกจภายใน 30 วินาที ให้เตือนว่าเป็นการพยายามโจมตีโดยเทคนิค Ping Flood เป็นต้น หากเน็ตเวิร์กดังกล่าวเป็นเน็ตเวิร์กที่ใช้งานโดยวิศวกรระบบ และมีการทดสอบการ Ping บ่อย ๆก็อาจจะทำให้ IDS เตือนอยู่แทบตลอดเวลาโดยไม่ได้มีการบุกรุกที่แท้จริง

การเตือนโดยมิได้มีการบุกรุกจริงนั้น อาจจะดูเหมือนว่าไม่ส่งผลเสียหายประการใดและน่าจะเกิดประโยชน์เสียด้วยซ้ำ เพราะจะทำให้ผู้ดูแลระบบมีความตื่นตัวตลอดเวลา แต่ในความเป็นจริงแล้วธรรมชาติของมนุษย์มีแนวโน้มจะละเลยต่อสิ่งเหล่านี้ หากมีการเตือนแล้วไม่มีการบุกรุกจริงบ่อยครั้งเข้าความน่าเชื่อถือของ IDS ก็จะลดลงตามลำดับ และเมื่อมีความพยายามบุกรุกจริงก็จะไม่ได้ให้ความสนใจเท่าที่ควรและไม่ได้หาทางป้องกันอย่างเหมาะสม นั่นคือ IDS จะกลายเป็นเด็กเลี้ยงแกะที่เวลาหมาป่าเข้ามาจริงก็ไม่มีผู้ใดรับฟัง ดูเผิน ๆอาจจะเหมือนว่ายังดีกว่าการไม่มี IDS เสียเลย แต่การมี IDS อยู่ในระบบโดยไม่ได้นำมาปรับแต่งอย่างเหมาะสม และเชื่อมั่นว่า IDS สามารถจะคอยระแวดระวังและเก็บหลักฐานต่าง ๆไว้ให้นั้นจะทำให้ผู้บริหารระบบนิ่งนอนใจและคลายความเคร่งครัดในการปฏิบัติงานลง อาจจะถึงขั้นหย่อนยานกว่าการป้องกันในระดับปกติที่ไม่มี IDSได้ นอกจากนี้การปล่อยให้ IDS มีการเตือนอย่างไม่

เหมาะสมจะทำให้เกิดข้อมูลในลักษณะที่เป็นการบุกรุกจริงและการเตือนผิดพลาดผสมกันอยู่ อาจทำให้การเตือนที่เป็นของจริงถูกกลบไปและยากต่อการสังเกต อย่าลืมว่าแฮ็กเกอร์ ที่มีความสามารถจะทิ้งร่องรอยของการบุกรุกไว้เพียงเล็กน้อยอาจจะมีเพียง 2-3 ร่องรอยเท่านั้นที่ IDS สามารถตรวจพบได้ หากร่องรอยเหล่านี้ถูกนำไปผสมปนเปกับการตรวจจับอื่น ๆอีกนับพัน ย่อมมีโอกาสสูงที่จะถูกมองเลยไปโดยไม่มีผู้ใดให้ความสนใจ

1.3 พฤติกรรมโดยทั่วไปของผู้บุกรุก

โดยประกอบด้วยขั้นตอนสำคัญ 3 ขั้นซึ่งจำเป็นและเป็นพื้นฐานทั่วไปของการบุกรุก

1.3.1 การแกะรอย (Footprinting) เป็นการรวบรวมข้อมูลของเครื่องเป้าหมายที่ต้องการให้ได้มากที่สุด โดยเฉพาะช่องโหว่ที่มีอยู่บนเน็ตเวิร์ก ทำให้ทราบโพรไฟล์ (Profile) ของเครื่องเป้าหมายที่เชื่อมต่ออยู่กับอินเตอร์เน็ต (Internet) ทั้งในส่วนของเน็ตเวิร์กภายในหรืออินทราเน็ต (Intranet) และเอ็กซ์ทราเน็ต (Extranet) รวมทั้งการให้บริการการเชื่อมต่อจากระยะไกล (remote access) ประกอบด้วย 3 ขั้นตอนย่อย ดังนี้

1. กำหนดขอบเขตของการแกะรอย เป็นการพิจารณาว่าต้องการแกะรอยเน็ตเวิร์กทั้งองค์กรหรือสนใจเฉพาะบางส่วนโดยค้นหาข้อมูลจากแหล่งข้อมูลที่เปิดเผยได้ (Open Source Search) ทำให้ทราบข้อมูลบางอย่างที่น่าสนใจได้ เช่น นโยบายด้านการรักษาความปลอดภัยซึ่งบ่งบอกให้ทราบถึงกลไกการรักษาความปลอดภัยที่ใช้งานอยู่ปัจจุบัน ชื่อผู้ติดต่อและอี-เมล์แอดเดรส หรือตำแหน่งที่ตั้ง
2. การรวบรวมรายละเอียดต่างๆ ของเน็ตเวิร์กเป้าหมาย โดยหาชื่อโดเมนและเน็ตเวิร์กที่เกี่ยวโยงกับเครื่องเป้าหมาย แล้วเรียกใช้โปรแกรม เช่น whois (เป็นโปรแกรมที่ใช้หาข้อมูลว่ามีใครใช้งานอยู่ในระบบบ้าง) ดูโดเมนเนมของระบบ และค้นหาค่ากำหนดของเครื่องที่ทำงานอยู่ในเน็ตเวิร์ก เช่น ชื่อเครื่อง รุ่นของระบบปฏิบัติการ ชื่อผู้ใช้ทั้งหมดในระบบ
3. การสำรวจเน็ตเวิร์ก โดยพยายามสำรวจเส้นทางที่แพ็กเกจไอพี เริ่มส่งจากเครื่องต้นทางไปถึงไปทาง

1.3.2 การสแกนเพื่อตรวจสอบ เปรียบเสมือนการเคาะกำแพงเพื่อสำรวจหาประตูบ้านและหน้าต่าง (ช่องโหว่) โดยการแกะรอยจะทำให้ได้ไอพีแอดเดรส และข้อมูลเกี่ยวกับเน็ตเวิร์กของเครื่องเป้าหมายผ่านการทางสอบถามจากฐานข้อมูล whois ส่วนต่อมาคือ ทำการตรวจสอบว่าเครื่องคอมพิวเตอร์ปลายทางใดบ้างที่เปิดอยู่และสามารถเข้าถึงได้โดยตรงผ่านทางอินเตอร์เน็ตและถ้าเป็นไปได้ควรทราบด้วยว่า มีหมายเลขพอร์ตใดเปิดอยู่บ้าง โดยการใช้เครื่องมือและเทคนิคต่างๆ เช่น ping sweeps , port scans และ automated discovery tools

เทคโนโลยี	ข้อมูลสำคัญเกี่ยวกับเทคโนโลยีนั้นๆ
อินเทอร์เน็ต (Internet)	เน็ตเวิร์กเซอร์วิซที่ทำงานบนโพรโตคอล TCP และ UDP ประเภทของโพรเซสเซอร์(เช่น SPARC, X86)
	Access Control List (ACL) เป็นกลไกควบคุมการเข้าถึงทรัพยากรต่างๆ บนเครื่องระบบป้องกันผู้บุกรุก (Intrusion Detection Systems, IDSes)
	การระบุรายละเอียดเกี่ยวกับระบบในแง่ต่างๆ (บัญชีรายชื่อผู้ใช้และกลุ่มต่างๆ, ตารางเราท์ติงของเราเตอร์, ข้อมูลของ SNMP)
อินทราเน็ต (Intranet)	เน็ตเวิร์กโพรโตคอลที่ใช้งานอยู่ภายใน (เช่น โพรโตคอล IP, IPX และอื่นๆ)
	ชื่อโดเมนภายใน
	เน็ตเวิร์กบล็อก
	ไอพีแอดเดรสของเครื่องคอมพิวเตอร์ที่เชื่อมต่อโดยตรงกับอินเทอร์เน็ต
	เน็ตเวิร์กเซอร์วิซที่ทำงานบนโพรโตคอล TCP และ UDP ประเภทของโพรเซสเซอร์(เช่น SPARC, X86)
	Access Control List (ACL) เป็นกลไกควบคุมการเข้าถึงทรัพยากรต่างๆ บนเครื่อง
	ระบบป้องกันผู้บุกรุก (Intrusion Detection Systems, IDSes)
	การระบุรายละเอียดเกี่ยวกับระบบในแง่ต่างๆ (บัญชีรายชื่อผู้ใช้และกลุ่มต่างๆ, ตารางเราท์ติงของเราเตอร์, ข้อมูลของ SNMP)
การเชื่อมต่อจากระยะไกล (Remote Access)	เบอร์โทรศัพท์ในระบบอะนาล็อก/ดิจิตอล
	ประเภทของเซิร์ฟเวอร์ที่ให้บริการ
	กลไกการตรวจสอบผู้ใช้ (Authentication Mechanism)
เอ็กซ์ทราเน็ต(Extranet)	คอนเน็กชันต้นทางและปลายทาง
	ประเภทของคอนเน็กชัน
	กลไกควบคุมการเข้าถึงทรัพยากร (Access Control Mechanism)

ตารางที่ 1.1 เทคโนโลยีและข้อมูลสำคัญที่แฮ็กเกอร์ ต้องการค้นหา

1.3.3 การค้นหาและรวบรวมรายละเอียด (Enumeration) เป็นการค้นหาบัญชีผู้ใช้หรือค้นหาทรัพยากรที่แชร์ไว้ ซึ่งข้อแตกต่างสำคัญระหว่างเทคนิคการรวบรวมข้อมูลและเบาะแสกับการค้นหาและรวบรวมรายละเอียดต่างๆ คือ ระดับหรือความร้ายแรงของการบุกรุก หมายความว่า มีการเปิดคอนเน็กชันไปยังเครื่องปลายทางโดยตรงและมีการส่งคำถามถามไปด้วย เมื่อแฮ็กเกอร์ ทราบชื่อแอคเคานต์ของผู้ใช้ที่ถูกต้องหรือทราบชื่อแชร์โฟลเดอร์ ผู้บุกรุกจะพยายามทำการคาดเดารหัส

ผ่านหรือค้นหาจุดบกพร่องของสิทธิที่เซตไว้ที่แชร์โฟลเดอร์นั้นๆ เมื่อผู้บุกรุกสามารถเข้าไปเป็นผู้ใช้ทั่วไปแล้วก็สามารถหาช่องทางขยายสิทธิ์ให้ได้เป็นผู้ดูแลระบบ

ประเภทของข้อมูลที่แฮ็กเกอร์ ต้องการรวบรวม สามารถแบ่งออกเป็น 3 กลุ่มใหญ่ คือ

- รายชื่อทรัพยากรในเน็ตเวิร์ก เช่น ชื่อแชร์โฟลเดอร์ เป็นต้น
- รายชื่อแอคเคานต์ของผู้ใช้และรายชื่อกลุ่ม
- รายชื่อแอพพลิเคชัน

1.4 รูปแบบของระบบตรวจจับผู้บุกรุกระบบคอมพิวเตอร์ มีวิธีหลักๆ ในการป้องกันผู้บุกรุก 2 วิธี คือ

1.4.1 วิธีเปรียบเทียบพฤติกรรมผู้ใช้กับรูปแบบการบุกรุกที่รู้จัก (Misuse Intrusion Detection)

วิธีเปรียบเทียบพฤติกรรมผู้ใช้กับรูปแบบการบุกรุกที่รู้จักประกอบด้วยการเก็บบันทึกและการระบุรูปแบบของการบุกรุกซึ่งอาจบุกรุกจากจุดอ่อนของระบบหรือการละเมิดกฎรักษาความปลอดภัย โดยมีตัวตรวจจับคอยดูแลกิจกรรมต่างๆ ที่กระทำในปัจจุบันว่าเหมือนกับพฤติกรรมการบุกรุกที่เคยเกิดขึ้นหรือได้รับรายงานว่าเป็นการบุกรุกหรือไม่

ในบางระบบมีการใช้กฎ (Rule-based expert system) โดยตั้งกฎขึ้นจากพฤติกรรมที่น่าสงสัย เช่น การ login ล้มเหลวเกินกว่า 3 ครั้งต่อเนื่องกัน ในเวลา 5 นาที ถือว่าพยายามบุกรุก และข้อมูลที่ใช้ตรวจสอบจะถูกนำมาเปรียบเทียบกับกฎ (Rules) ที่มีอยู่สังเกตว่ามีการนำข้อมูลทางเวลาเข้ามาพิจารณาด้วย การตรวจจับการบุกรุกโดยวิธีนี้สามารถมีการแก้ไขกฎหรือเพิ่มกฎได้ ในระบบที่เป็นปัญญาประดิษฐ์ (Artificial Intelligence) ระบบอาจทำการแก้ไขกฎหรือเพิ่มกฎได้ด้วยตนเอง

ข้อเสียของวิธีเปรียบเทียบพฤติกรรมผู้ใช้กับรูปแบบการบุกรุกที่รู้จัก คือ

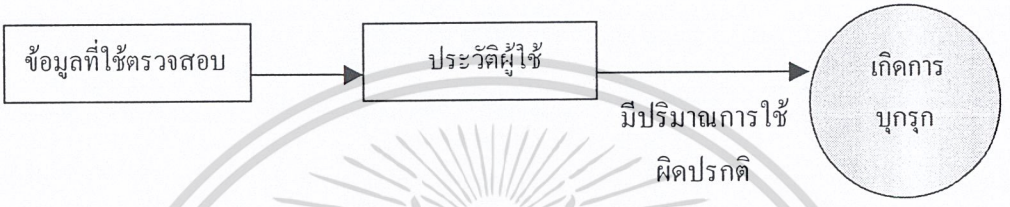
1. ประสิทธิภาพของระบบการตรวจจับชนิดนี้ขึ้นอยู่กับรูปแบบการบุกรุกที่ระบบรู้จัก
2. มีข้อจำกัดในเรื่องจำนวนของรูปแบบในการบุกรุก ซึ่งหากเป็นการบุกรุกที่ระบบไม่รู้จักมาก่อนจะทำให้ไม่สามารถตรวจจับการบุกรุกได้

1.4.2 วิธีตรวจสอบการใช้งานระบบที่ผิดปรกติ (Anomaly Intrusion Detection)

วิธีตรวจสอบการใช้งานทรัพยากรระบบที่ผิดปรกติตั้งอยู่บนสมมติฐานว่าการกระทำใดๆ ที่เป็นการบุกรุกจะต้องมีการใช้งานระบบอย่างผิดปรกติ โดยมีกระบวนการเก็บประวัติพฤติกรรมการใช้งานของผู้ใช้ และสังเกตการทำงานเกิดขึ้นของผู้ใช้ว่าเมื่อเข้ามาในระบบได้กระทำสิ่งใดบ้างแล้วรายงานเก็บเป็นประวัติซึ่งสามารถใช้เป็นข้อมูลตรวจสอบในการนำมาเปรียบเทียบว่าพฤติกรรมในปัจจุบันมีการใช้งานระบบที่ผิดปรกติกว่าประวัติพฤติกรรมของผู้ใช้เดิมมากน้อยเพียงใด หากมีปริมาณการใช้ในปริมาณมากผิดปรกติจึงถือว่าเกิดการบุกรุก

ปัญหาของการตรวจจับโดยวิธีตรวจสอบการใช้งานระบบที่ผิดปรกติ คือ

1. อาจมีพฤติกรรมการใช้งานทรัพยากรระบบของผู้ใช้ที่ผิดปรกติเกิดขึ้นแต่ไม่ได้เป็นการบุกรุกระบบ ทำให้ระบุสถานะผิดพลาด
2. ตรวจไม่พบการบุกรุกระบบเนื่องจากการบุกรุกนั้นไม่ได้ใช้ทรัพยากรระบบอย่างผิดปรกติ
3. หากผู้บุกรุกค่อยๆเปลี่ยนพฤติกรรมการใช้งานไปทีละเล็กละน้อยระบบจะไม่สามารถตรวจจับความผิดปรกติได้



รูปที่ 1.1 การทำงานของการตรวจจับผู้บุกรุกโดยวิธีตรวจสอบการใช้งานระบบที่ผิดปรกติ

1.5 แนวทางการปฏิบัติเมื่อมีการบุกรุกระบบ

1. จัดตั้งกลุ่มที่รับผิดชอบในการจัดการได้ทันทีที่ทราบว่าถูกบุกรุก
2. กำหนดแนวทางในการรับมือ เช่น ควรให้ความสำคัญกับการจัดการให้เน็ตเวิร์กทำงานได้ปกติในสภาพเดิมเร็วที่สุด หรือให้ความสำคัญในการหาตัว ผู้บุกรุกผู้ดูแลระบบสามารถถอดสายเน็ตเวิร์กโดยทันทีหรือไม่หากพบว่ามีผู้บุกรุกกำลังทำความเสียหายกับระบบอยู่เมื่อพบผู้บุกรุกแล้วจะคอยดูพฤติกรรมต่อไปหรือจัดการทันที ให้ลองคิดปัญหาที่เกิดขึ้นแล้วหาแนวทางการดำเนินการก่อนที่จะเกิดขึ้นจริง เพราะเมื่อเกิดเหตุการณ์ขึ้นแล้วจะไม่มีเวลาคิด
3. ให้กำหนดแนวทางในการแจ้งปัญหา เช่น หากเกิดปัญหาแล้วจะแจ้งผู้บังคับบัญชาในลำดับสูงขึ้นไปก่อน หรือแจ้งหน่วยงานที่เกี่ยวข้องเลย ต้องแจ้งเหตุการณ์การบุกรุกที่เกิดขึ้นกับหน่วยงานที่คอยให้คำปรึกษาและช่วยเหลือใด (ในอเมริกามีหลายหน่วยงาน เช่น FIRST หรือ CERT) ต้องแจ้งตำรวจหรือไม่ จะแจ้งเหตุการณ์นี้กับหุ้นส่วนทางธุรกิจหรือไม่ จะปิดข่าวนี้กับหนังสือพิมพ์หรือไม่
4. ให้จัดระบบและขั้นตอนในการดำเนินงานในการเก็บล็อก วิเคราะห์ และเฝ้าติดตามข้อมูลทันทีประเด็นสำคัญในการหาร่องรอยของผู้บุกรุก คือ ต้องมีการเก็บล็อกอย่างเพียงพอสำหรับการวิเคราะห์
5. ให้คำแนะนำกับผู้ใช้ทุกคนให้ทราบเกี่ยวกับการป้องกันการบุกรุก

1.6 ลีนุกซ์ (Linux)

1.6.1 ประวัติความเป็นมา

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

ลักษณะการพัฒนาลีนุกซ์ได้ใช้อินเทอร์เน็ตช่วยโดยอาศัยความร่วมมือของนักพัฒนาจากสถานที่ต่างๆ จึงสังเกตได้ว่าการพัฒนาของระบบปฏิบัติการลีนุกซ์เป็นไปอย่างรวดเร็ว

1.6.2 ลีนุกซ์คืออะไร

ลีนุกซ์เป็นระบบปฏิบัติการแบบ 32 บิตอย่างแท้จริงที่สนับสนุนการใช้งานแบบหลายงาน (Multitasking) และแบบหลายผู้ใช้ (Multiuser) โดยมีระบบการทำงานคล้ายระบบยูนิกซ์ (UNIX Clone) มีระบบเอ็กซ์วินโดว์ (X-windows) ซึ่งเป็นระบบการติดต่อกับผู้ใช้แบบกราฟฟิกที่ไม่ขึ้นระบบปฏิบัติการ (Operating system) หรือฮาร์ดแวร์ใดๆ และมีโพรโตคอล TCP/IP ซึ่งเริ่มแรกได้ถูกพัฒนาขึ้นมาเพื่อใช้งานกับระบบเน็ตเวิร์กของยูนิกซ์ที่ใช้งานกันอย่างกว้างขวางในการเชื่อมกันของเครือข่ายอินเตอร์เน็ต ระบบเน็ตเวิร์กบนลีนุกซ์ถือได้ว่ามีความสมบูรณ์ต่อการใช้งานจริง เช่น สามารถที่จะเชื่อมต่อเป็นเครือข่ายผ่านระบบแลนทั่วไป หรือการเชื่อมต่อในระบบรีโมตเทอร์มินัลผ่านโมเด็ม

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

โดยเทคนิคแล้วลีนุกซ์เป็นเพียงเคอร์เนล (kernel) ของระบบปฏิบัติการ มีหน้าที่สำคัญ คือ การจัดการกับโพรเซสและการจัดการกับไฟล์ รวมทั้งการจัดการกับอุปกรณ์ต่างๆ เป็นส่วนที่จะทำหน้าที่ในการบริหารรวมทั้งควบคุมระบบทั้งหมด นอกจากนี้แล้วลีนุกซ์ยังมีลักษณะพิเศษในการจัดการกับหน่วยความจำและไฟล์ คือ

- ไฟล์เอ็กซิคิวต์เทเบิล (executable file) สามารถแชร์พื้นที่ในหน่วยความจำได้ ลักษณะในการทำงานแบบนี้เรียกว่า copy on write page ซึ่งเป็นการเพิ่มประสิทธิภาพในการใช้งานหน่วยความจำ
- สามารถทำงานในระบบดีมานเพจจิง (demand paging) ซึ่งเป็นการนำเอาโปรแกรมที่ใช้งานบ่อยมาเก็บไว้ในหน่วยความจำ และถ้ามีการใช้งานระบบมากขึ้น ความจุดิสก์แคชจะถูกปรับอัตโนมัติ
- เมื่อต้องการที่จะใช้หน่วยความจำเพิ่มขึ้น ลีนุกซ์สามารถที่จะนำพื้นที่บนดิสก์มาทำหน้าที่เสมือนหน่วยความจำได้ (swap space) แต่ก็มีข้อเสียอันเนื่องมาจากการแอ็กเซสข้อมูลที่ช้าของดิสก์เมื่อเทียบกับหน่วยความจำจริง

- นอกจากนี้แล้ว ถ้ามีแอพพลิเคชันใดใช้งานไลบรารีเหมือนกัน ลีนุกซ์สามารถที่จะจัดการเพื่อให้แอพพลิเคชันเหล่านั้นใช้งานไลบรารีร่วมกัน เป็นการลดขนาดแอพพลิเคชันลง (dynamic shared libraries)

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

สาเหตุที่เลือกใช้ลีนุกซ์ในโครงงาน

สาเหตุที่เลือกให้ลีนุกซ์ในโครงงานนี้ คือ ลีนุกซ์เป็นระบบปฏิบัติการที่แจกฟรี สามารถดาวน์โหลดจากอินเทอร์เน็ตโดยไม่ผิดกฎหมายและมีผู้นิยมใช้มาก มีผู้นำลีนุกซ์ไปแก้ไขให้สามารถใช้งานได้บนตัวประมวลผลหลากหลาย นอกจากนี้ยังมีผู้พัฒนาแอพพลิเคชันสำหรับลีนุกซ์ออกมาเรื่อยๆ

เนื่องจากลีนุกซ์เป็นระบบปฏิบัติการ 32 บิตแบบแท้จริง ทำให้ดึงความสามารถดึงการทำงานของคอมพิวเตอร์ออกมาได้เต็มกำลัง การทำงานของลีนุกซ์จึงมีประสิทธิภาพและลีนุกซ์มีคุณลักษณะของยูนิกซ์เต็มรูปแบบ ทำให้เป็นระบบหลายผู้ใช้และหลายงาน ซึ่งสนับสนุนโพรโตคอลแบบ TCP/IP, SLIP, PPP, UUCP และอื่นๆ โดยเวอร์ชันที่เลือกนำมาใช้เป็น Linux mandrake 8.0 และใช้เคอร์เนล 2.4.3

สำหรับการใช้ประโยชน์ของลีนุกซ์ในด้านอื่นๆ สามารถใช้ลีนุกซ์ในการศึกษาระบบปฏิบัติการยูนิกซ์ และแพ็กเกจของลีนุกซ์ได้รวบรวมโปรแกรมที่ใช้จำลองสภาพการทำงานของดอส เพื่อให้โปรแกรมที่ทำงานบนดอส สามารถทำงานบนลีนุกซ์ สำหรับวินโดวส์มีซอฟต์แวร์ที่สามารถทำให้ลีนุกซ์รันแอพพลิเคชันของวินโดวส์บนระบบ X ชื่อ WABI (Windows Applications Binary Interface)

ลีนุกซ์ได้เตรียมเครื่องมือพัฒนาโปรแกรมไว้อย่างครบครัน มีตั้งแต่แอพพลิเคชันมาตรฐาน คือ คอมไพเลอร์ภาษาซีและซีพลัสพลัส คอมไพเลอร์ของ GNU

คอมไพเลอร์ภาษาอื่นๆ ที่ลีนุกซ์มีให้ เช่น เพิร์ล (Perl), สมอลล์ทอล์ก (Smalltalk), พาสคาล (Pascal), ลิส์ป (Lisp) เป็นต้น หรือถ้าต้องการเขียนโปรแกรมที่เกี่ยวข้องกับดาต้าเบส ลีนุกซ์มีโปรแกรมแบบเอ็กซ์เบส (X-Base) หรือฟ็อกซ์โปร (FoxPro) ไว้ให้เช่นกัน และล่าสุดลีนุกซ์ได้เพิ่มจาวาคอมไพเลอร์ให้สำหรับผู้ที่ต้องการเขียนแอพเพลตจาวาสำหรับการทำงานบนอินเทอร์เน็ตด้วย

ไดเรกทอรี

ไดเรกทอรีเป็นกลุ่มของไฟล์ ข้อมูลภายในไดเรกทอรีจะประกอบด้วยไฟล์ หรือซับไดเรกทอรี ถ้ามีไดเรกทอรี A แล้วภายในไดเรกทอรี A มีไดเรกทอรีอื่นอยู่ด้วยชื่อไดเรกทอรี B จะเรียกไดเรกทอรี A ว่าเป็นแพเรนท์ไดเรกทอรี (Parent Directory) ส่วนไดเรกทอรี B จะเรียกว่าเป็นซับไดเรกทอรี

จำนวนไฟล์หรือซับไดเรกทอรีสำหรับลีนุกซ์นั้นมีได้ไม่จำกัดขึ้นอยู่กับฮาร์ดดิสก์ว่ามีพื้นที่ในการเก็บข้อมูลมากเท่าใด แต่โดยทั่วไปแล้วโครงสร้างของไฟล์จะประกอบไปด้วย root (Root Directory) ซึ่งเป็นแพเรนท์ไดเรกทอรีทั้งหมดบนระบบ ซึ่งจะแสดงด้วยตัวอักษร "/" หมายถึงว่า ตอนนี้อยู่ที่ไดเรกทอรีสูงสุดแล้ว ภายใต้แพเรนท์ไดเรกทอรีจะประกอบไปด้วยไดเรกทอรีที่สำคัญอยู่จำนวนหนึ่งเป็นไดเรกทอรีหลักของระบบ

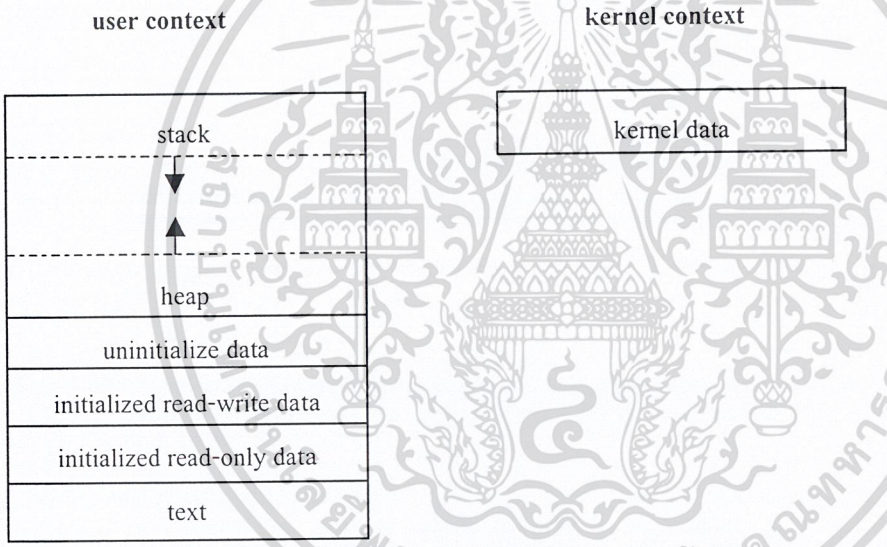
- "/bin" เก็บโปรแกรมที่สำคัญของระบบเช่น “ls” (ใช้แสดงรายชื่อไฟล์), “cp” (ใช้คัดลอกไฟล์ข้อมูล) โดยเก็บเป็นไบนารีไฟล์ (binary file) ซึ่งเป็นภาษาเครื่อง
- "/boot" เก็บไฟล์ที่เก็บส่วนสำคัญสำหรับการบูตระบบ
- "/dev" เป็นไฟล์ที่ใช้แทนอุปกรณ์ต่าง ๆ ของระบบ เช่น “/dev/hda” จะแทนฮาร์ดดิสก์, “/dev/tty” จะแทนเทอร์มินอล (terminal)
- "/etc" ค่าที่ใช้เซตการทำงานของโปรแกรมต่าง ๆ จะถูกเก็บลงในไดเรกทอรีนี้ เช่น lilo.conf , syslog.conf เป็นต้น
- "/home" เก็บข้อมูลต่าง ๆ ของ ผู้ใช้งานในระบบ เช่น ถ้าผู้ใช้ชื่อ “mint” ค่าปรกติของไดเรกทอรีที่เก็บไฟล์จะเป็น “/home/mint” เป็นต้น
- "/lib" เก็บไฟล์ไลบรารี ของระบบ
- "/proc" เก็บข้อมูลที่เกี่ยวกับโพรเซสและเคอร์เนล
- "/root" เก็บข้อมูลของผู้ดูแลระบบ
- "/sbin" เก็บโปรแกรมที่ผู้ดูแลระบบใช้ในการดูแลรักษาระบบ เช่น “shutdown”, “fsck” เป็นต้น
- "/usr" เก็บโปรแกรมที่ติดตั้งโดยผู้ใช้งาน และนอกจากนั้นหากได้ติดตั้งส่วนของซอร์สโค้ด (source code) ของเคอร์เนลไว้ ก็จะสามารถดูได้ที่ “/usr/src”
- "/var" – เก็บไฟล์ที่ใช้ดูแลระบบเช่น ล็อกไฟล์ (log file)ของระบบ และข้อมูลที่มีการเปลี่ยนแปลงอยู่บ่อยครั้ง เช่น อี-เมล์ เป็นต้น

บทที่ 2
โพรเซสและข้อมูลเกี่ยวกับโพรเซส

2.1. ความหมายของโพรเซส

โพรเซส หมายถึง ชุดคำสั่งในหน่วยความจำที่กำลังทำงานอยู่ในขณะนั้น โดยมีชุดควบคุมการทำงาน 1 ชุดเป็นของตัวเองซึ่งได้ใช้ทรัพยากรของระบบไปส่วนหนึ่ง ทั้งนี้ต้องทำความเข้าใจไว้ก่อนว่า โพรเซสกับโปรแกรมไม่ใช่สิ่งเดียวกันยกตัวอย่างเช่น ในระบบมัลติยูสเซอร์สมมติว่ามีผู้ใช้งาน 50 คน กำลังใช้โปรแกรม pine เพื่อทำการอ่านอีเมล์พร้อมๆ กันก็จะมีโพรเซสเกิดขึ้น 50 โพรเซสที่กำลังใช้โปรแกรม pine เพียงโปรแกรมเดียว

2.2. โครงสร้างของโพรเซส



รูปที่ 2.1 แสดงโครงสร้างอย่างคร่าว ๆ ของโพรเซส

จากรูปเราแบ่งโครงสร้างโพรเซส 2 ส่วน คือ

ส่วน user context

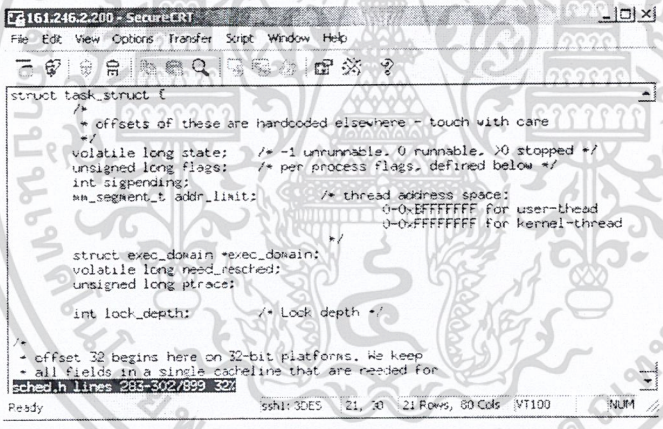
ประกอบไปด้วยส่วนต่าง ๆ ดังนี้

- **text** เก็บภาษาเครื่อง (machine instruction) ที่ใช้รัน โดยปรกติถูกกำหนดให้มีเพอร์มิสชันเป็น read only ดังนั้นโพรเซสจึงไม่สามารถแก้ไขข้อมูลส่วนนี้ได้
- **data** เก็บข้อมูลที่ใช้ในโปรแกรม แบ่งเป็น 3 ส่วนย่อยคือ uninitialized data, initialized read-write data และ initialized read only data

- **heap** ใช้งานในขณะรันโพรเซส เพื่อกำหนดพื้นที่หน่วยความจำให้เพียงพอต่อความต้องการในการใช้งาน
- **stack** สำหรับควบคุมการใช้งานตัวแปร การเรียกใช้ฟังก์ชันและการส่งค่ากลับ

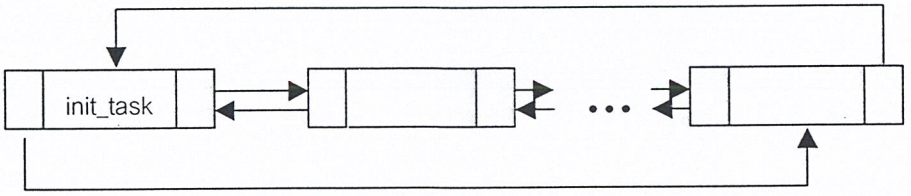
ส่วน kernel context

ถูกควบคุมและใช้งานโดยเคอร์เนลเท่านั้น เพื่อเก็บข้อมูลที่จำเป็นสำหรับควบคุมและติดตามการทำงานของโพรเซสซึ่งการที่เคอร์เนลจะจัดการกับโพรเซสได้นั้น ตัวเคอร์เนลเองต้องทราบข้อมูลต่างๆ ที่เกี่ยวกับโพรเซสแต่ละตัวเก็บไว้เช่น มีหมายเลขประจำโพรเซส (process ID) มีค่าเท่าไร สถานะการทำงานของโพรเซสกำลังประมวลผลอยู่หรือถูกบล็อก (block) อยู่หรือไม่ มีแอดเดรสสเปซ (address space) อยู่ที่ตำแหน่งใด มีไพรออริตี้เท่าไร เป็นต้น ซึ่งข้อมูลต่าง ๆ เหล่านี้ แต่ละโพรเซสมีเก็บไว้ในแต่ละโพรเซสในส่วนที่เรียกว่า process descriptor โดยรายละเอียดโครงสร้างที่เป็นส่วนประกาศในภาษาซี สามารถดูได้ที่ /usr/include/linux/sched.h ซึ่งอาจจะแตกต่างกันไปตามเวอร์ชันของเคอร์เนล



รูปที่ 2.2 โครงสร้างของ process descriptor ในไฟล์ /usr/include/linux/sched.h

ในโครงสร้างข้อมูลนี้ มีโครงสร้างข้อมูลที่ใช้ชี้ไปยัง process descriptor ถัดไปและก่อนหน้านี้ ซึ่งหากลองพิจารณาดูจะเห็นว่ามีโครงสร้างเป็น circular doubly linked list ดังรูปที่ 2.3 ซึ่งส่วนหัวของลิงค์ลิสนี้เป็น init_task descriptor ซึ่งเป็นโพรเซสแรกของลีนุกซ์ที่เกิดขึ้นในช่วงการบูตระบบซึ่งมักถูกเรียกว่า process 0 หรือ swapper จะมีหน้าที่ในการบริหารงานโพรเซส



รูปที่ 2.3 ลักษณะของการเชื่อมโยงกันของ process descriptor

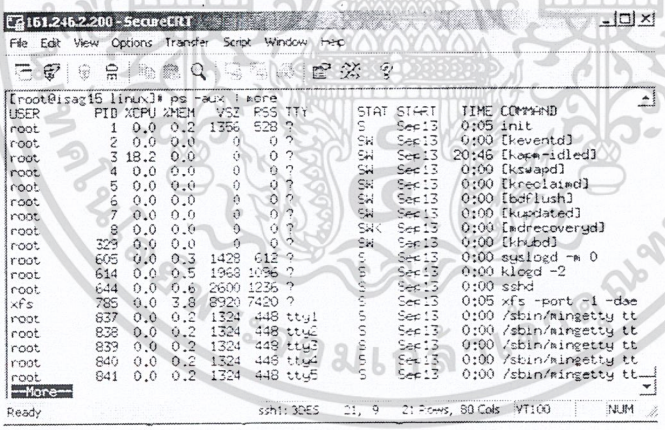
ในการจัดการกับโพรเซสทั้งหมดนั้น เคอร์เนลมีอาร์เรย์ ที่เรียกว่า task array โดยสมาชิกใน array นี้ประกอบด้วย pointer ชี้ไปยัง process descriptor ของโพรเซสต่างๆ มีไว้เพื่อความสะดวกของการจัดการโพรเซสของเคอร์เนลเอง

2.3. การตรวจสอบข้อมูลของโพรเซส

มีคำสั่งอยู่หลายคำสั่งในลีนุกซ์ที่ใช้ในการตรวจสอบข้อมูลของโพรเซส เช่น

คำสั่ง ps

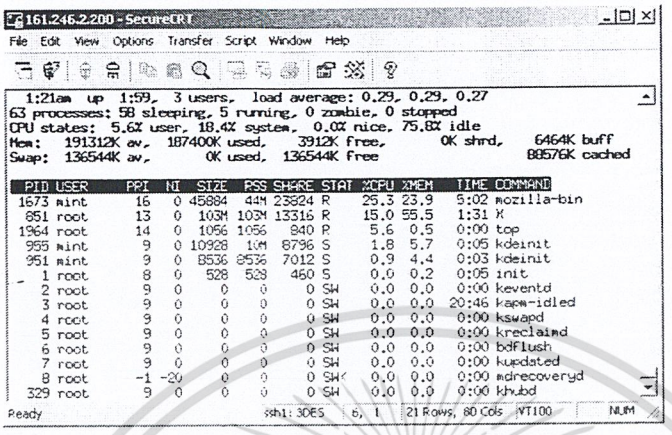
แสดงข้อมูลของโพรเซสที่อยู่ในไดเรกทอรี /proc (ซึ่งจะขออธิบายส่วนนี้ในหัวข้อถัดไป)



รูปที่ 2.4 แสดงการผลลัพธ์จากการรันคำสั่ง ps

คำสั่ง top

แสดงข้อมูลของโพรเซสเช่นกัน แต่มีการสแกนข้อมูลอยู่ตลอดเวลา



รูปที่ 2.5 แสดงผลลัพธ์ของการรันคำสั่ง top

2.4. การจัดเก็บข้อมูลใน /proc

ภายในไดเรกทอรี /proc เป็นที่เก็บข้อมูลรายละเอียดเกี่ยวกับเคอร์เนลและโพรเซสซึ่งในที่นี้ขอกล่าวถึงเฉพาะในส่วนที่เกี่ยวกับโพรเซส

/proc/[number]

เก็บข้อมูลเกี่ยวกับโพรเซสตามหมายเลขที่กำกับไว้ เช่น /proc/1243 ภายในไดเรกเทอรีนี้เก็บข้อมูลเกี่ยวกับโพรเซสที่มีหมายเลขโพรเซสเป็น 1243 ไว้ ซึ่งภายในมีรายละเอียดดังนี้

1. ไฟล์ cmdline

เก็บคำสั่งที่โพรเซสนั้นประมวลผล

2. ไฟล์ environ

เก็บค่าตัวแปรสภาพแวดล้อมของโพรเซสนั้น ๆ

3. ไดเรกทอรี fd

เก็บ symbolic link ของ file descriptor ของไฟล์ที่โพรเซสนั้นได้เรียกใช้งาน

4. cwd Symbolic link

เป็นลิงค์ที่ชี้ไปยังไดเรกทอรีที่โพรเซสทำงานอยู่ (current working directory)

5. exe Symbolic link

เป็นลิงค์ที่ชี้ไปยังไฟล์ที่โพรเซสกำลังประมวลผลอยู่

6. ไฟล์ maps

เก็บส่วน memory map ของโพรเซส ซึ่งประกอบด้วยช่วงตำแหน่ง , permission, offset เป็นต้น

7. root Symbolic link

เป็นลิงค์ไปยังรูตไดเรกทอรี (root directory)

8. ไฟล์ statm

เก็บข้อมูลการใช้งานหน่วยความจำของโพรเซส

9. ไฟล์ stat

เก็บข้อมูลรายละเอียดเกี่ยวกับโพรเซส ซึ่งไฟล์นี้เป็นไฟล์ที่มีรายละเอียดมากที่สุดโดยมีการเก็บข้อมูลเรียง ๆ กันซึ่งลำดับการเรียงข้อมูลนั้นอาจแตกต่างกันไปตามเวอร์ชันของเคอร์เนล (ซึ่งในระบบทีใช้เป็นเป็นเคอร์เนล 2.4.3 ซึ่งสามารถตรวจดูลำดับของข้อมูลได้ใน source code ของเคอร์เนลที่ /usr/src/linux/fs/proc/array.c) ซึ่งความหมายของแต่ละตัวแปรดูได้ตามตารางที่ 2.1

ตัวแปร	ความหมาย
pid	หมายเลขประจำโพรเซส
cmd	คำสั่งที่โพรเซสทำการประมวลผล
state	สถานะของโพรเซส
ppid	หมายเลขประจำโพรเซสของโพรเซสที่เป็นผู้สร้างโพรเซสนี้
pgrp	หมายเลขกลุ่มของโพรเซส
session	หมายเลข session ของโพรเซส
tty	tty ที่โพรเซสใช้
tpgid	หมายเลขประจำโพรเซสของ tty ที่ใช้
flags	flag ของโพรเซส
min_flt	ค่า minor page fault
cmin_flt	ค่า minor page falut ที่โดยรวมค่าที่โพรเซสลูกของโพรเซสนี้ด้วย
maj_flt	ค่า major page fault
cmaj_flt	ค่า major page falut ที่โดยรวมค่าที่โพรเซสลูกของโพรเซสนี้ด้วย
utime	User time ในหน่วย jiffies (1/100 วินาที)
stime	System time ในหน่วย jiffies
cutime	User time โดยรวมเวลาของโพรเซสลูกด้วย ในหน่วย jiffies

ตารางที่ 2.1 แสดงความหมายของข้อมูลในไฟล์ /proc/[number]/stat

สำนักหอสมุดกลาง พระจอมเกล้าลาดกระบัง

ตัวแปร	ความหมาย
cstime	System time โดยรวมเวลาของโพรเซสลูกด้วย ในหน่วย jiffies
priority	ค่า static priority ของโพรเซส
nice	เป็นค่าไพรออริตี้ที่สามารถเปลี่ยนแปลงได้ (Dynamic priority)
start_time	เวลาที่โพรเซสเริ่มทำงาน
vsize	ขนาดของ Visual Memory ในหน่วย byte
rss	Resident set size
rlim	ค่าจำกัดของ RSS
start_code	จุดเริ่มต้นของโค้ดเซ็กเมนต์ (code segment)
end_cod	จุดสิ้นสุดต้นของโค้ดเซ็กเมนต์
start_stack	จุดเริ่มต้นของสแต็กเซ็กเมนต์ (stack segment)
esp	ตำแหน่งของสแต็กเฟรมปัจจุบัน (current stack frame)
eip	ตำแหน่งของสแต็กเฟรมปัจจุบัน (current stack frame)
pending.signal	เก็บสัญญาณที่โพรเซสต้องการตรวจสอบว่ายังค้างอยู่ในระบบหรือไม่
blocked.signal	เก็บสัญญาณที่โพรเซสต้องการบล็อก
Sigign	เก็บสัญญาณที่โพรเซสไม่สนใจ
Sigcatch	เก็บสัญญาณที่โพรเซสกำลังรอ

ตารางที่ 2.1 แสดงความหมายของข้อมูลในไฟล์ /proc/[number]/stat (ต่อ)

10. ไฟล์ status

เก็บข้อมูลของโพรเซสเช่นเดียวกันโดยแสดงให้อยู่ในรูปที่เข้าใจได้ง่าย แต่ข้อมูลน้อยกว่าในไฟล์ stat

2.5. ล็อกไฟล์ pacct

ไฟล์นี้มีหน้าที่เก็บข้อมูลของโพรเซสที่ผู้ใช้ได้เรียกใช้งานซึ่งถูกบันทึกเมื่อโพรเซสจบการทำงาน ลักษณะโครงสร้างจะสามารถเข้าไปดูได้ที่ /usr/include/linux/acct.h ซึ่งมีลักษณะดังนี้

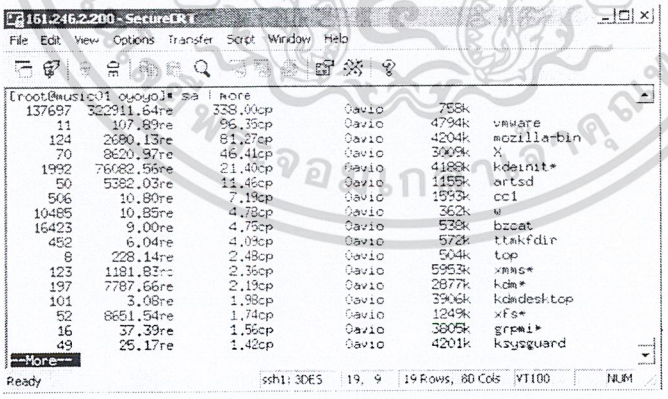
ตัวแปร	คำอธิบาย
ac_flag	Flags ของโพรเซส
ac_uid	หมายเลขประจำตัวของผู้ที่เป็นเจ้าของโพรเซส
ac_gid	หมายเลขประจำกลุ่มของผู้ที่เป็นเจ้าของโพรเซส

46137

ตัวแปร	คำอธิบาย
ac_tty	เทอร์มินอลที่ใช้ในการรันโพรเซส
ac_btime	เวลาที่โพรเซสเกิดขึ้น
ac_utime	User time
ac_stime	System Time
ac_etime	Elapsed Time
ac_mem	ปริมาณการใช้หน่วยความจำโดยเฉลี่ย
ac_io	ปริมาณการใช้ i/o
ac_rw	จำนวนบล็อกที่เขียนหรืออ่าน
ac_minflt	ค่า Minor Pagefaults
ac_majflt	ค่า Major Pagefaults
ac_swaps	จำนวน swap ที่ใช้
ac_exitcode	Exit code ของโพรเซส
ac_comm[ACCT_COMM + 1]	คำสั่งที่ใช้ในโพรเซสนี้

ตารางที่ 2.2 แสดงความหมายของตัวแปรในโครงสร้าง struct acct

คำสั่งที่ใช้ตรวจสอบดูข้อมูลในไฟล์นี้ก็มีเช่น sa เป็นต้น



รูปที่ 2.6 แสดงสถิติของผู้ใช้งานโดยคำสั่ง sa

บทที่ 3

ลีนุกซ์เคอร์เนลโมดูลและซิสเต็มคอลล์(System call)

3.1 ลีนุกซ์เคอร์เนลโมดูล

3.1.1 ทำความรู้จักกับลีนุกซ์เคอร์เนลโมดูล

ลีนุกซ์เคอร์เนลโมดูลใช้ในการขยายฟังก์ชันให้กับเคอร์เนลของลีนุกซ์โดยมีข้อดี คือ เป็นการโหลดข้อมูลแบบไดนามิกซึ่งทำให้ไม่ต้องทำการโหลดข้อมูลทั้งหมดโดยโหลดเพียงส่วนที่มีการใช้งานเท่านั้นและสามารถใช้งานได้เลยโดยไม่ต้องคอมไพล์เคอร์เนลใหม่

ลีนุกซ์โมดูลมีฟังก์ชันพื้นฐาน 2 ฟังก์ชัน คือ

1. int init_module(void) /* เป็นฟังก์ชันที่ใช้ในการกำหนดค่าเริ่มต้น */
2. void cleanup_module(void) /* เป็นฟังก์ชันที่ใช้ในการ shutdown */

โดยทั่วไปแล้วการโหลดโมดูลต้องมีสิทธิ์เป็นรูตเท่านั้น โดยใช้คำสั่งดังนี้ :

```
#insmod module.o
```

คำสั่งนี้ส่งผลต่อระบบ คือ

- โหลดออบเจ็กต์ไฟล์ (object file) ขึ้นมา ในที่นี้ คือ module.o
- เรียกซิสเต็มคอลล์ (systemcall) ชื่อ create_module เพื่อทำการจองเนื้อที่ในหน่วยความจำ
- ซิสเต็มคอลล์ชื่อ get_kernel_syms จะอ้างอิงในส่วนของ Kernel-symbols
- หลังจากกำหนดค่าเริ่มต้นโดยใช้ซิสเต็มคอลล์ init_module แล้วก็จะทำการเอ็กซิคิวต์ (execute) ฟังก์ชัน int init_module(void) ต่อไป

ซึ่งรายละเอียดของ Kernel-Symbols จะขออธิบายไว้ในหัวข้อต่อๆ ไป

ตัวอย่าง ข้างล่างนี้จะเป็นตัวอย่างของการสร้างโมดูลอย่างง่ายๆ

```
#define __KERNEL__
#define MODULE
#include <Linux/kernel.h>
#include <Linux/module.h>
int init_module(void) {
        printk("<1>Hello World\n");
        return 0;
}
void cleanup_module(void) {
        printk("<1Bye, Bye");
}
```

สังเกตว่าในตัวอย่างข้างต้นจะใช้คำสั่ง printk ไม่ใช้ printf เนื่องจากการเขียนโปรแกรมในเคอร์เนล จะแตกต่างกับการเขียนโปรแกรมในยูสเซอร์สเปซมาก ดังนั้นเวลาการเขียนโปรแกรมในเคอร์เนลจำเป็นที่จะต้องระมัดระวังคำสั่งการใช้งานมากเนื่องจากคำสั่งเหล่านี้อาจมีข้อจำกัดในการใช้งาน ดังนั้นจึงควรเรียนรู้ฟังก์ชันต่าง ๆ ในแอพพลิเคชันของ userspace ไว้เพื่อช่วยในการเจาะระบบเคอร์เนลได้

จากตัวอย่างข้างต้นสามารถคอมไพล์โดยใช้คำสั่ง

gcc -c -03 helloworld.c

โหลดโมดูลโดยใช้คำสั่ง

insmod helloworld.o

เมื่อโมดูลถูกโหลดขึ้นมาและแสดงตัวอักษรขึ้นมา เราใช้คำสั่งตรวจสอบเพื่อให้เห็นว่าลีนุกซ์โมดูลยังอยู่ในเคอร์เนลสเปซ ดังนี้

lsmod

Module	Pages	Used by
Helloworld	1	0

คำสั่งข้างต้นจะอ่านข้อมูลใน /proc/modules เพื่อโหลดโมดูลที่ชื่อ helloworld โดยใช้เวลาชั่วขณะหนึ่ง ส่วน Page เป็นข้อมูลของหน่วยความจำ (จำนวนของ page ที่อยู่ในโมดูล) และ Used by จะบอกความถี่ที่ระบบเรียกใช้โมดูล helloworld (อ้างอิงการนับ) เราสามารถถอดโมดูลออกจากระบบได้ หากค่าของตัวแปร counter มีค่าเป็น 0 โดยคำสั่งที่ใช้ย้าย คือ

rmmod helloworld

ในลักษณะของลีนุกซ์โมดูลมีลักษณะคล้ายกับโปรแกรม DOS TSR รุ่นเก่า คือ สามารถจับสัญญาณอินเทอร์รัปต์ทุกสัญญาณที่เราต้องการได้ ดังเช่น WIN9X ของบริษัท Microsoft ที่มีการเรียก (call) VxD ซึ่งเหมือนกับ ลีนุกซ์โมดูลที่มีการเรียกฟังก์ชันของระบบ เรียกว่า ซิสเต็มคอลล์ (systemcalls)

3.1.2 Kernel Symbol Table

เมื่อเข้าใจพื้นฐานของซิสเต็มคอลล์และโมดูลแล้ว ยังมีจุดที่น่าสนใจและเป็นส่วนที่สำคัญมากอีกแห่งหนึ่ง คือ ตาราง Kernel-Symbol ซึ่งอยู่ในไฟล์ /proc/ksyms การเข้ามายังไฟล์นี้จะถือเป็นการเอ็กซ์พอร์ต (export)หรือพลับบิค (public) Kernel Symbol ซึ่งสามารถแอ็กเซส (access) ลีนุกซ์โมดูลที่เขียนขึ้นได้

นอกจากนี้ไฟล์ /proc/ksms ยังสามารถช่วยให้สามารถเห็นสิ่งที่ลีนุกซ์โมดูลทำการโหลดมาได้แต่ยังคงมีปัญหาอยู่ข้อหนึ่ง คือ ทุก Symbol ที่ใช้ใน ลีนุกซ์โมดูล (ซึ่งเหมือนกับฟังก์ชัน) จะเป็นการ export ไปยัง public ทำให้เกิดเป็นลิสต์ขึ้นในไฟล์ซึ่งอาจเป็นการบุกรุกได้ในทางหนึ่งโดยผู้บุกรุกอาจทำการซ่อนลีนุกซ์โมดูล นั้นไว้

ดังนั้นผู้ดูแลระบบที่มีประสบการณ์มากพอจะค้นพบ ลีนุกซ์โมดูล นี้และทำการลบทิ้งเพื่อป้องกันการบุกรุก

นักพัฒนาส่วนของ ลีนุกซ์โมดูล สามารถใช้โค้ดข้างล่างนี้จำกัดการเอ็กซ์พอร์ตซิมโบล (export symbol)ในโมดูลได้

```
static struct symbol_table module_syms= { /*ทำการระบุตารางซิมโบล*/
 #include <linux/symtab_begin.h>          /*ซิมโบลจะถูกจำกัดการเอ็กซ์พอร์ต*/
...
};
register_symtab(&module_syms); /*ทำการรีจีสเตอร์*/
```

หากไม่ต้องการให้ซิมโบลถูกเอ็กซ์พอร์ตไปยังพลับบิค (public) ควรใช้คำสั่งดังนี้ :

```
register_symtab(NULL)
```

ซึ่งคำสั่งนี้ต้องใส่ในส่วนของฟังก์ชัน init_module()

3.1.3 การเปลี่ยนจากส่วนของ Kernel ไปหน่วยความจำใน ยูสเซอร์สเปซ

จากที่หัวข้อที่ผ่านทำให้ทราบข้อดีในการโค้ดดิ้ง (coding) ในส่วนของเคอร์เนลสเปซแล้วแต่ก็ยังคงมีข้อเสียอยู่เหมือนกัน กล่าวคือ เมื่อซิสเต็มคอลล์รับอาร์กิวเมนต์มาจากยูสเซอร์สเปซ (ซิสเต็มคอลล์จะถูก อิมพลีเมนต์ใน wrapper เหมือนกับไลบรารี libc) แต่ลีนุกซ์โมดูลนั้นรันในส่วนของเคอร์เนลสเปซ

ในหัวข้อนี้จะยกตัวอย่างของการเข้าถึงอาร์กิวเมนต์ที่ถูกจองไว้ในส่วนของยูสเซอร์สเปซจากโมดูลในเคอร์เนลสเปซได้ ดูจากตัวอย่างซิสเต็มคอลล์นี้ :

```
int sys_chdir (const char *path)      /* เป็นฟังก์ชันการเปลี่ยนไดเรกทอรี */
```

เมื่อระบบทำการเรียกซิสเต็มคอลล์นี้ก็จะหยุดการคอลล์ (call) อื่นไว้แล้วทำการตรวจสอบพาธ (path) ที่ผู้ใช้ต้องการตั้งค่า ดังนั้นจึงต้องมีการใช้ค่า const char *path ซึ่งหากพยายามใช้ค่าตัวแปร path โดยตรงอาจเกิดปัญหา real problem ได้เช่นเดียวกับการ printk ("<1%s\n", path);

ในเคอร์เนลสเปซไม่สามารถอ่านหน่วยความจำในยูสเซอร์สเปซได้ง่ายนัก แต่ก็สามารถทำได้ โดยจะมีมาโคร(macro) ที่ชื่อ get_user (pointer) ให้ทำหน้าที่ให้ฟังก์ชันพอยน์เตอร์ชี้ไปยัง *path ซึ่งช่วยให้อ่านค่า (เป็นไบต์) จากหน่วยความจำในยูสเซอร์สเปซไปยังเคอร์เนลสเปซ

ต่อไปนี้เป็นการทำงานของ plaguez ในการย้ายสตริงก์จากยูสเซอร์สเปซไปยังเคอร์เนลสเปซ :

```
char *strncpy_fromfs(char *dest, const char *src, int n)    {
        char *tmp = src;
        int compt = 0;
        do {
                dest[compt++] = __get_user(tmp++, 1);
        }
```

```
        while ((dest[compt - 1] != '\0') && (compt != n));
        return dest;
}
```

ถ้าต้องการกลับค่าตัวแปร *path ก็สามารถใช้ส่วนของโค้ดในเคอร์เนลได้ดังนี้

```
char *kernel_space_path;
/* จองหน่วยความจำในเคอร์เนลสเปซ*/
 kernel_space_path = (char *) kmalloc(100, GFP_KERNEL);
(void) strncpy_fromfs(test, path, 20);          /* เรียกฟังก์ชันของ plaguez */
 printk("<1%s\n", kernel_space_path);           /* ใช้ข้อมูลใดก็ได้ที่ต้องการ */
 kfree(test);                                   /* ทำการคืนหน่วยความจำ */
```

จากโค้ดข้างต้นสามารถทำงานได้ดี สำหรับทรานซิชันทั่วไปค่อนข้างมีความซับซ้อนมากซึ่งอธิบายรายละเอียดในหัวข้อถัดไป

3.1.4 วิธีการใช้ยูสเซอร์สเปซให้เหมือนกับฟังก์ชัน

จากหัวข้อที่แล้วได้ใช้ syscall macro ในการสร้าง brk call ของตนเองขึ้นมา และฟังก์ชันไลบรารีในยูสเซอร์นั้นถูกอิมพลีเมนต์ (implement) ผ่าน syscall macro ดังซอร์สโค้ดที่แสดงใน _syscall1(...) macro ซึ่งใช้ในหัวข้อที่ 4 ในการสร้างฟังก์ชัน brk(...) (มาจาก /asm/unistd.h)

```
#define _syscall1(type,name,type1,arg1) \
 type name(type1 arg1) \
 { \
long __res; \
 __asm__ volatile ("int $0x80" \
                   : "=a" (__res) \
                   : "0" (__NR_##name),"b" ((long)(arg1))); \
 if (__res = 0) \
                return (type) __res; \
errno = -__res; \
 return -1; \
 }
```

ในส่วนนี้มันจะทำการเรียกอินเทอร์รัปต์ (interrupt) 0x80 กับอาร์กิวเมนต์ (argument) ที่ถูกส่งค่ามาจาก _syscall1 เพื่อใช้เรียกซิสเต็มคอลล์ที่ต้องการ(ชื่ออาจถูกขยายเป็น _NR_name ซึ่งระบุไว้ใน /asm/unistd.h) วิธีนี้จะใช้อิมพลีเมนต์กับฟังก์ชัน brk เท่านั้น ส่วนฟังก์ชันกับอาร์กิวเมนต์อื่น ๆ ก็จะอิมพลีเมนต์ผ่าน macros อื่นเช่นกัน (_syscallX โดยที่ X เป็นจำนวนอาร์กิวเมนต์)

โดยทั่ว ๆ ไปแล้วอาจใช้วิธีอื่นในการอิมพลีเมนต์ฟังก์ชันได้ ดังจากตัวอย่างต่อไปนี้ :

```
int (*open)(char *, int, int);                    /* ประกาศโพรโตไทป์ */
open = sys_call_table[SYS_open];                  /* สามารถ __NR_open */
```

วิธีนี้อาจไม่ต้องใช้ syscall macro อื่นเลยเพียงแต่ต้องใช้ฟังก์ชันพอยน์เตอร์จาก sys_call_table เท่านั้นและพบว่าการสร้างยูสเซอร์ให้เหมือนฟังก์ชันโดยวิธีนี้จะมีการใช้ใน famous LKM infector โดย SVAT ซึ่งในความเป็นจริงแล้วถือว่าเป็นการแก้ปัญหาที่ดีกว่า แต่ย่อมขึ้นอยู่กับตัวผู้ใช้เอง

ข้อควรระวังเมื่อทำการส่งอาร์กิวเมนต์ไปยังซิสเต็มคอลล์เหล่านี้ เพราะซิสเต็มคอลล์เหล่านี้อาจต้องการอาร์กิวเมนต์ใน ยูสเซอร์สเปซ ไม่ใช่จากใน เคอร์เนลสเปซ สำหรับวิธีการนำข้อมูลจาก เคอร์เนลสเปซ ไปยังหน่วยความจำ ยูสเซอร์สเปซ นั้นให้ศึกษาจากหัวข้อข้างต้นที่ได้กล่าวไว้แล้ว

ในระบบปฏิบัติการลีนุกซ์นั้นจะใช้ segment selector ในการเปลี่ยนระหว่างเคอร์เนลสเปซและยูสเซอร์สเปซ ซึ่งอาร์กิวเมนต์ที่ใช้กับซิสเต็มคอลล์จะถูกส่งมาจากยูสเซอร์สเปซซึ่งเป็นส่วนหนึ่งใน data segment selector (DS) range

Data segment นี้สามารถทำการคืนค่าได้โดยใช้ฟังก์ชัน get_ds() จากไลบรารี asm/segment.h ดังนั้นข้อมูลที่ใช้จะเป็นพารามิเตอร์ (parameter)โดยซิสเต็มคอลล์สามารถแอ็กเซสพารามิเตอร์นี้ได้จากเคอร์เนลสเปซ หากมีการตั้งค่า segment selector ให้ใช้ได้กับ user segment โดยเคอร์เนลไปยังค่า DS ที่ต้องการโดยสามารถใช้ฟังก์ชัน set_fs(…) ข้อควรระวัง คือ ต้องเก็บค่า FS หลังจากทำการแอ็กเซสอาร์กิวเมนต์ของซิสเต็มคอลล์แล้ว ดังตัวอย่างโค้ดบางส่วนข้างล่างนี้

```
-filename is in our kernel space; a string we just created, for example
unsigned long old_fs_value=get_fs();
 set_fs(get_ds);              /*หลังจากส่วนนี้จะสามารถแอ็กเซสข้อมูลในยูสเซอร์สเปซได้*/
 open(filename, O_CREAT|O_RDWR|O_EXCL, 0640);
set_fs(old_fs_value); /*restore fs...*/
```

ฟังก์ชันที่แสดงในตัวอย่างนี้ (brk, open) ถูกอิมพลีเมนต์ผ่านซิสเต็มคอลล์ตัวเดียวเท่านั้นแต่ยังมีกลุ่มของฟังก์ชันยูสเซอร์สเปซที่รวบรวมไว้ในซิสเต็มคอลล์หนึ่ง สามารถดูรายละเอียดได้ในหัวข้อที่ 2 เช่น sys_socketcall จะอิมพลีเมนต์ทุก ๆ ฟังก์ชันที่เกี่ยวกับ socket (การสร้าง, ปิด, ส่ง, รับ) ดังนั้นจึงควรระมัดระวังเมื่อสร้างฟังก์ชันขึ้นมาแล้วจะต้องคอยดูในเคอร์เนลซอร์ส (kernel source) เสมอ

3.1.5 รายชื่อของฟังก์ชันในเคอร์เนลสเปซที่ใช้บ่อย

ฟังก์ชัน printk(…) ที่ได้กล่าวถึงในขั้นต้นนั้นเป็นฟังก์ชันที่ทุกคนเคยใช้ในเคอร์เนลสเปซ ดังนั้นจึงเรียกว่า ฟังก์ชันเคอร์เนลฟังก์ชันเหล่านี้ถูกสร้างขึ้นเพื่อให้ผู้ที่ต้องการฟังก์ชันที่ซับซ้อนได้ใช้งาน ซึ่งโดยปรกติแล้วผู้ใช้งานจะใช้ฟังก์ชันธรรมดาผ่านไลบรารีฟังก์ชันเท่านั้น

ตารางข้างล่างนี้แสดงรายชื่อฟังก์ชันเคอร์เนลที่สำคัญ ๆ ไว้

Function/macro	description
int sprintf (char *buf, const char *fmt,...); int vsprintf (char *buf,const char *fmt, va_list args);	ฟังก์ชันในการเก็บข้อมูลเป็น string
printk(..)	เหมือนกับฟังก์ชัน printf ในยูสเซอร์สเปซ
void *memset (void *s, char c, size_t count); void *memcpy (void *dest, const void *src, size_t count char *bcopy (const char *src,char *dest, int count); void *memmove (void *dest, const void *src,seze_t count); int memcmp (const void *cs, const void *ct, size_t count); void *memscan (void *addr, unsigned char c,size_t size);	ฟังก์ชันหน่วยความจำ
int register_symtab (struct symbol_table *intab);	ดูในตาราง 2.1
char *strcpy (char *dest, const char *src); char *strncpy (char *dest, const char *src,size_t count); char *strcat (char *dest, const char *src); char *strncat (char *dest, const char *src, size_t count); int strcmp (const char *cs, const char *ct); int strncmp (const char *cs, const char *ct,size_t count); char *strchr (const char *s, char c); size_t strlen (const char *s); size_t strnlen (const char *s, size_t count); size_t strspn (const char *s, const char *accept); char *strpbrk (const char *cs, const char *ct); char *strtok (char *s,const char *ct);	ฟังก์ชันการเปรียบเทียบ string ฯลฯ
unsigned long simple_strtoul (const char *cp, char **endp,unsigned int base);	กลับค่าจาก string เป็นตัวเลข

Function/macro	description
Get_user_byte (addr);	ฟังก์ชันในการแอ็กเซสของผู้ใช้
Put_user_byte (x,addr); Get_user_word (addr); Put_user_word (x,addr); Get_user_long (addr); Put_user_long (x,addr);	หน่วยความจำ
Suser(); Fsuser();	ตรวจสอบสิทธิ์ของ Super user
Int register_chrdev (unsigned int major,const char *name, struct file_operations *fops); Int unregister_chrdev (unsigned int major, const char *name); Int register_blkdev (unsigned int major, const char *name, struct file_operations *fops); Int unregister_blkdev (unsigned int major, const char *name);	ฟังก์ชันการรีจิสเตอร์อุปกรณ์ต่าง ๆ …_chrdev -> character devices …_blkdev -> block devices

ตารางที่ 3.1 แสดงรายชื่อของฟังก์ชันใน เคอร์เนลสเปซที่ใช้บ่อย

ฟังก์ชันเหล่านี้บางส่วนอาจเป็นฟังก์ชันที่สร้างขึ้นมาเพื่อใช้ประโยชน์โดยใช้วิธีการที่ได้กล่าวถึงในส่วนที่ 5 (ฟังก์ชันที่มีความสำคัญมากในตาราง คือ ฟังก์ชันการเปรียบเทียบ string)

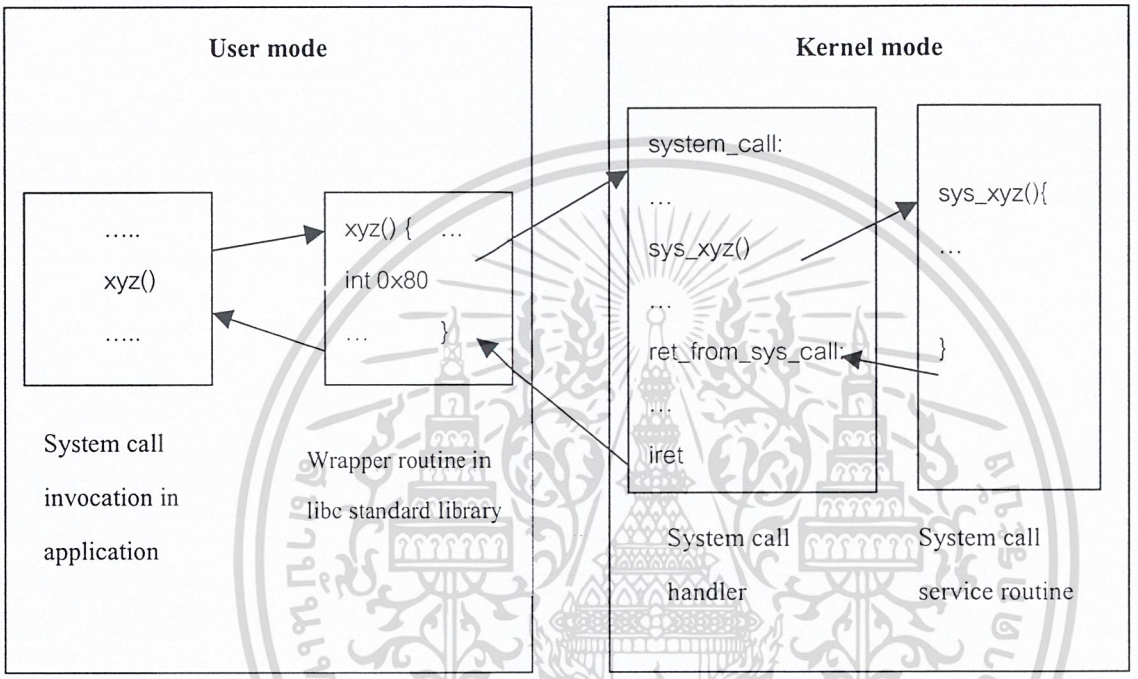
3.2 ซิสเต็มคอลล์

ระบบปฏิบัติการส่วนมากในทุกวันนี้อนุญาตให้โพรเซสต่างๆ สามารถติดต่อกับอุปกรณ์ เช่น ดิสก์ เครื่องพิมพ์ ได้จาก user mode เลยโดยมีการแบ่งเลเยอร์ระหว่างในส่วนฮาร์ดแวร์เลเยอร์ กับ แอพพลิเคชันเลเยอร์ ซึ่งในการจัดการเช่นนี้มีข้อดีอยู่หลายข้อ คือ

- ทำให้การเขียนโปรแกรมทำได้ง่ายขึ้นเพราะคนเขียนโปรแกรมไม่จำเป็นต้องเขียนโปรแกรมในระดับต่ำ
- ในแง่ของทางด้านการรักษาความปลอดภัยเพราะเคอร์เนลสามารถตรวจสอบการร้องขอได้เพราะจำเป็นต้องร้องขอผ่านเคอร์เนล

การติดต่อกันระหว่าง user mode process กับ hardware device นั้นทำโดยการเรียกใช้ซิสเต็มคอลล์ผ่านทางเคอร์เนล สำหรับในลีนุกซ์นั้นการเรียก ซิสเต็มคอลล์ นั้นทำได้โดยการอินเทอร์รัปต์หมายเลข 0x80 และส่งค่าหมายเลข ซิสเต็มคอลล์ ไว้ที่รีจิสเตอร์ eax

จากรูปเป็นการแสดงความสัมพันธ์ของการเรียก ซิสเต็มคอลล์ จากแอพพลิเคชัน เมื่อมีการร้องขอซิสเต็มคอลล์ผ่านจากแอพพลิเคชัน ก็จะส่งต่อไปยัง wapper routine ที่ไลบารี libc เพื่ออินเทอร์รัปต์และเรียกให้ system call handler ทำงานส่วนนี้ใน kernel mode ซึ่งหลังจากนั้นก็รันโค้ดซิสเต็มคอลล์ การเรียก system call routine นั้นเคอร์เนลมีตาราง system call dispatch table เก็บอยู่ในอาร์เรย์ชื่อ sys_call_table โดยสมาชิกแต่ละตัวมีตำแหน่งของ system call service routine อยู่



รูปที่ 3.1 แสดงความสัมพันธ์ของการเรียก ซิสเต็มคอลล์ จากแอพพลิเคชัน

หมายเลขซิสเต็มคอลล์เป็นตัวชี้ไปยังอาร์เรย์ชื่อ sys_call_table[] โดยหมายเลขซิสเต็มคอลล์นี้จะตรงกับ (map) ฟังก์ชันการให้บริการ

ตารางข้างล่างนี้เป็นตารางรายชื่อการใช้งานซิสเต็มคอลล์ที่น่าสนใจ

ซิสเต็มคอลล์	หน้าที่ของซิสเต็มคอลล์
int sys_fork(struct pt_regs regs);	ถูกเรียกเมื่อใช้คำสั่ง fork() ในยูสเซอร์สเปซ
int sys_getuid () int sys_setuid (uid_t uid)	ใช้สำหรับจัดการหมายเลข UID
int sys_get_kernel_sysms (struct kernel_sym *table)	ใช้สำหรับการเข้าถึง Kernel symbol table

ตารางที่ 3.2 แสดงรายชื่อ ซิสเต็มคอลล์ ที่ใช้บ่อย

ซิสเต็มคอลล์	หน้าที่ของซิสเต็มคอลล์
int sys_sethostname (char *name,int len) int sys_gethostname(char *name,int len)	ทำการตรวจสอบค่าและเซ็ตชื่อโฮสต์
int sys_chdir (const char *path); int sys_fchdir (unsigned int fd);	ทำการเปลี่ยนไดเรกทอรีที่ใช้งานอยู่ (คำสั่ง cd)
int sys_chmod (const char *filename, mode_t mode); int sys_chown (const char *filename, mode_t mode);	ทำการจัดการเกี่ยวกับเพอร์มิสชันของผู้ใช้
int sys_chroot (const char *filename);	เซตรูทไดเรกทอรีสำหรับโพรเซสที่กำลังทำงาน
int sys_execve (struct pt_regs regs);	รับผิดชอบในการรันไฟล์โปรแกรมต่างๆ
int sys_link (const char *oldname, const char *newname); int sym_link (const char *oldname, const char *newname); int sys_unlink (const char *name);	ทำหน้าที่ในการสร้างและลบฮาร์ดลิงค์(hardlink) และซอฟต์ลิงค์(softlink)
int sys_rename (const char *oldname, const char *newname);	ทำการเปลี่ยนชื่อไฟล์
int sys_rmdir (const char *name); int sys_mkdir (const *char filename, int mode);	ทำหน้าที่สร้างและลบไดเรกทอรี
int sys_open (const char *filename,int mode); int sys_close (unsigned int fd);	ทำหน้าที่เปิดไฟล์(รวมทั้งสร้างไฟล์ด้วย), ปิดไฟล์
int sys_read (unsigned int fd, char *buf, unsigned int count); int sys_write (unsigned int fd, char *buf, unsigned int count);	ทำหน้าที่อ่านและเขียนไฟล์
int sys_selectt (int n, fd_set *inp, fd_set *outp, fd_set *exp, struct timeval *tvp);	Multiplexing of I/O operations
sys_socketcall (int call, unsigned long args);	ทำหน้าที่เกี่ยวกับซอกเก็ตเช่น สร้างซอกเก็ต, binding, connect เป็นต้น

ตารางที่ 3.2 แสดงรายชื่อ ซิสเต็มคอลล์ ที่ใช้บ่อย(ต่อ)

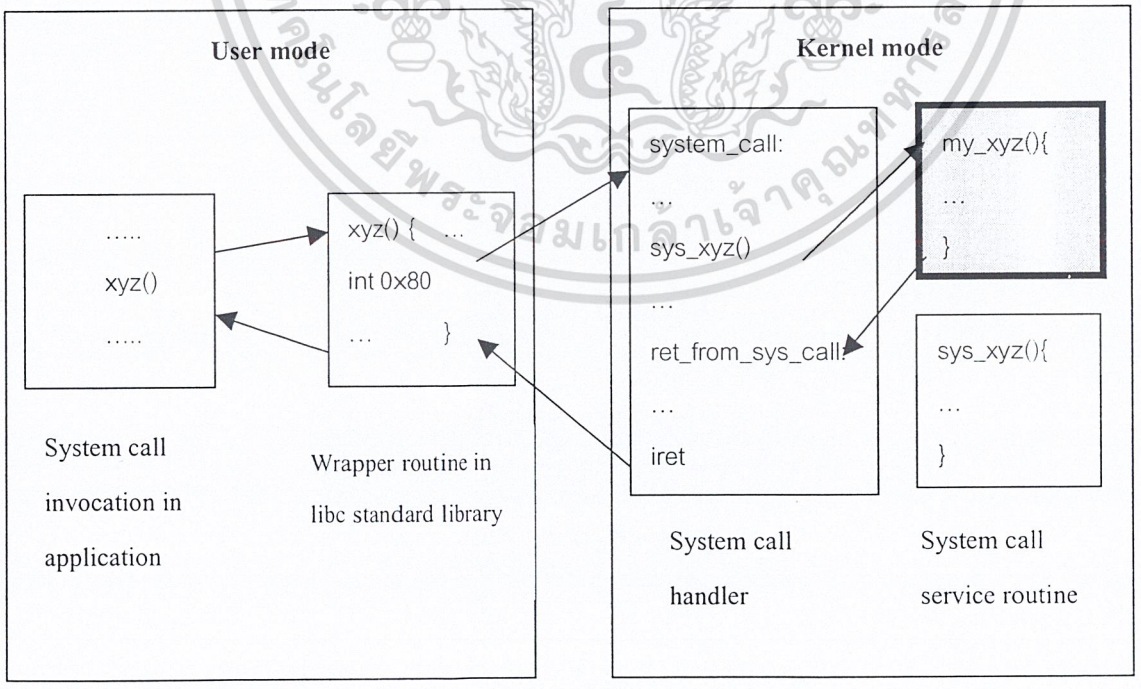
ซิสเต็มคอลล์	หน้าที่ของซิสเต็มคอลล์
unsigned long sys_create_module (char *name, unsigned long size); int sys_delete_module (char *name); int sys_query_module (const char *name, int which, void *buf, size_t bufsize, size_t *ret);	ทำหน้าที่จัดการเกี่ยวกับโมดูลเช่น โหลดโมดูล, ย้ายโมดูลออก, แสดงรายชื่อโมดูล

ตารางที่ 3.2 แสดงรายชื่อ ซิสเต็มคอลล์ ที่ใช้บ่อย

ในตารางนี้เป็นซิสเต็มคอลล์ที่ผู้บุกรุกส่วนใหญ่มักใช้ในการบุกรุก ดังนั้นจึงควรเรียนรู้การใช้ซิสเต็มคอลล์ในระบบให้ชำนาญและมีประสิทธิภาพ

3.3 การดักการทำงานของ ซิสเต็มคอลล์

จากรูปที่ 3.1 สังเกตว่าในส่วนของ kernel mode ก่อนมีการรัน system call routine นั้นต้องผ่านส่วน system call handler โดยดูตำแหน่งโปรแกรมของ system call routine ที่ตาราง sys_call_table ซึ่ง ณ จุดนี้ หากเราเปลี่ยนค่าตำแหน่งของโปรแกรมที่เก็บไว้ให้เป็นตำแหน่งของโปรแกรมที่เป็นโค้ดของเราเอง เมื่อ system call handler ทำการเรียกดูตำแหน่ง system call routine ค่าที่ได้กลายเป็นตำแหน่งของโปรแกรมของเราแทนการรัน system call routine ตามปรกติก็จะมารันในโปรแกรมของเรา จุดนี้เราสามารถเขียนโปรแกรมควบคุมการทำงานได้ตามความต้องการ โดยวิธีการคือสร้างโมดูลขึ้นมาแล้วไปเปลี่ยนค่าตำแหน่งโค้ดในตาราง sys_call_table และโหลดโค้ดเข้าไปในหน่วยความจำ



รูปที่ 3.2 แสดงวิธีดักการทำงานของซิสเต็มคอลล์

จากรูปกล่องสีทึบมุมบนซ้ายมือเป็นโปรแกรมที่เราเขียนขึ้นมาให้ทำงานแทน ซิสเต็มคอลล์ เดิม โดยการเปลี่ยนค่าตำแหน่งโปรแกรมในตาราง sys_call_table ไปชี้ยังที่โปรแกรมเราแทน

อย่างไรก็ตามการดักการทำงานของ ซิสเต็มคอลล์ นั้นอาจดูเหมือนเป็นการประสงค์ร้ายต่อระบบ ซึ่งก็เป็นความจริงเพราะถ้าหากผู้บุกรุกสามารถดักการทำงานของซิสเต็มคอลล์ได้ก็จะสามารถควบคุมการทำงานของเครื่องได้อย่างมากเช่น ป้องกันการลบข้อมูลของตนโดยอาจใช้การดักการทำงานชอง ซิสเต็มคอลล์ sys_unlink ขยายสิทธิ์การใช้งานของตนโดยดักการทำงานของ ซิสเต็มคอลล์ sys_setuid เป็นต้น แต่ในความจริงแล้วเราสามารถนำวิธีการดักการทำงานมาช่วยในการดูแลระบบได้เช่นกันโดยเราต้องเลือกดักซิสเต็มคอลล์ให้ตรงกับความต้องการของสิ่งที่เราต้องการทำ

ตัวอย่างข้างล่างนี้เป็นการใช้ประโยชน์จากการดักการทำงานของซิสเต็มคอลล์โดยหากผู้ดูแลระบบต้องการป้องกันไม่ให้มีการโหลดโมดูลเข้าสู่เคอร์เนลอีก ก็สามารถทำได้โดยการดักการทำงานของซิสเต็มคอลล์ create_module ซึ่งทำหน้าที่ในการโหลดโมดูลเข้าสู่เคอร์เนล (ในคำสั่ง insmod)

```
#define __KERNEL__
#define MODULE
#include <linux/module.h>
#include <linux/kernel.h>

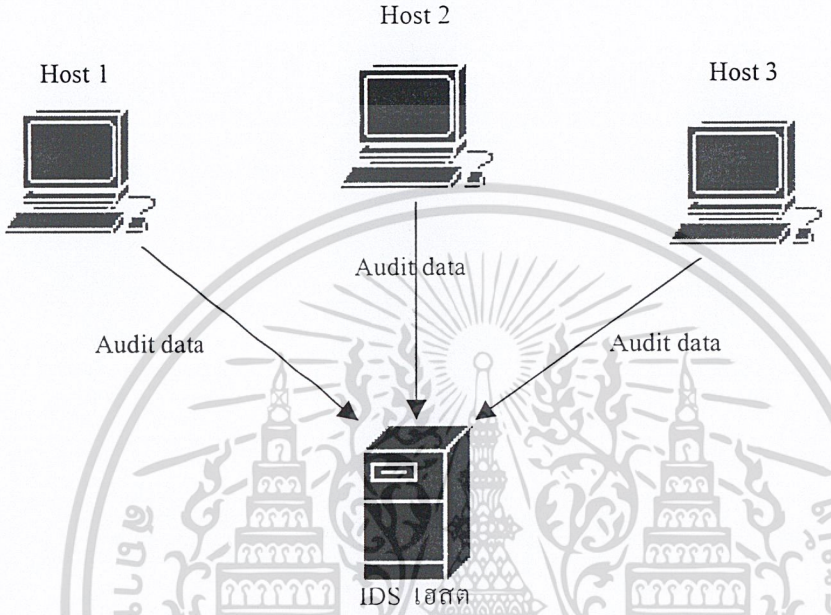
// ตาราง sys_call_table
extern void* sys_call_table[];
// ไว้สำหรับเก็บตำแหน่งปรกติของ ซิสเต็มคอลล์
asmlinkage int (*orig_create_module) (char *name, unsigned long size);
int my_ create_module (char *name, unsigned long size){
        return 0;                       // ที่ส่วนนี้โปรแกรมจะส่งค่า 0 กลับ เหมือนกับว่าไม่มี error เกิด
                                        // ขึ้นเลย แต่ความจริงแล้วเรายังไม่ได้ทำการโหลดโมดูลเลย
}

int init_module(void) {
// เมื่อมีการโหลดโมดูลนี้เข้าสู่เคอร์เนลจะทำการเรียกฟังก์ชันนี้ก่อน
        //  ทำการเก็บตำแหน่งของ ซิสเต็มคอลล์ ต้นฉบับ
        orig_create_module=sys_call_table[__NR_create_module];
        // แทนที่ตำแหน่งโปรแกรมด้วยตำแหน่งของโปรแกรมของเรา
        sys_call_table[__NR_create_module]=(void *)my_create_module;
        return 0;
```

```
}
void cleanup_module(void){
// คืนตำแหน่งปรกติของ ซิสเต็มคอลล์ เมื่อทำการลบโมดูลออกจากหน่วยความจำ
        sys_call_table[__NR_create_module]=orig_rmdir;
}
```

บทที่ 4
การทำงานของระบบ

4.1 ภาพรวมของระบบตรวจจับผู้บุกรุก



รูปที่ 4.1 ภาพรวมการทำงานของระบบตรวจจับผู้บุกรุก

จากรูปที่ 4.1 จะทราบว่าระบบตรวจจับผู้บุกรุกนั้นจะมีศูนย์กลางการทำงานอยู่ที่ IDS โฮสต์โดยแยกออกมาจากโฮสต์อื่นๆ ซึ่งทำหน้าที่วิเคราะห์ข้อมูลที่ส่งมาจากโฮสต์ที่วางตัวดักจับข้อมูลพฤติกรรมผู้ใช้ไว้

ข้อดีของการแยก IDS โฮสต์ออกมาจากโฮสต์ตัวอื่น

- เมื่อแยกส่วนที่เป็นการวิเคราะห์ออกมาทำให้โฮสต์ที่วางระบบตรวจจับผู้บุกรุกไว้ทำหน้าที่แค่ส่งข้อมูลพฤติกรรมผู้ใช้งานมาซึ่งจะทำให้โฮสต์ที่วางระบบตรวจจับผู้บุกรุกไว้ถูกใช้ทรัพยากรระบบไปไม่มาก
- สามารถวิเคราะห์พฤติกรรมของผู้ใช้ในหลายๆ โฮสต์ได้พร้อมกัน
- ถ้าเราต้องการวางระบบตรวจจับผู้บุกรุกไว้ในเครื่องที่มีระบบปฏิบัติการอื่นๆ (ในโครงงานนี้ได้พัฒนาเฉพาะระบบปฏิบัติการลีนุกซ์เท่านั้น) ก็สามารถทำได้ง่ายโดยทำการพัฒนาโปรแกรมเฉพาะส่วนเก็บข้อมูลผู้ใช้ให้ส่งข้อมูลพฤติกรรมผู้ใช้มาให้ IDS โฮสต์ในรูปแบบที่กำหนดไว้เท่านั้น

- ช่วยให้ผู้ดูแลระบบทำงานได้ง่ายขึ้นเพราะทำการตรวจสอบพฤติกรรมผู้ใช้งานหรือความผิดปรกติของระบบสามารถทำได้โดยเริ่มทำการตรวจสอบที่เครื่อง IDS โฮสต์ก่อน และเมื่อเจอปัญหาจึงไปจัดการที่โฮสต์นั้นๆ

4.2 โครงสร้างระบบตรวจจับผู้บุกรุกโดยวิธีการตรวจสอบการใช้งานระบบที่ผิดปรกติ

ระบบตรวจจับผู้บุกรุกจะแบ่งออกเป็น 3 ส่วนหลักคือ

1. ส่วนเก็บข้อมูลพฤติกรรมผู้ใช้งาน
2. ส่วนเก็บข้อมูลและวิเคราะห์พฤติกรรมผู้ใช้งาน
3. ส่วนตอบสนองการวิเคราะห์ ทำการรายงานผลการวิเคราะห์เมื่อพบพฤติกรรมการใช้งานที่ผิดปรกติไป

4.2.1 ส่วนเก็บข้อมูลพฤติกรรมผู้ใช้งาน

ส่วนเก็บข้อมูลพฤติกรรมผู้ใช้งานจะถูกติดตั้งอยู่ที่โฮสต์ที่ต้องการวางระบบตรวจจับผู้บุกรุกไว้ โดยการทำงานคือจะทำการติดตามการใช้งานผ่านทางการเรียกใช้ซิสเต็มคอลล์ execve โดยปรกติซิสเต็มคอลล์ตัวนี้จะถูกเรียกใช้งานเมื่อทำการรันไฟล์โปรแกรมต่างๆ ซึ่งจะมีการส่งทั้งคำสั่งและอาร์กิวเมนต์ (argument)ของคำสั่งนั้นๆ เข้ามาให้ซิสเต็มคอลล์จากหลักการทำงานดังกล่าวเราสามารถดักจับข้อมูลการใช้งานของผู้ใช้โดยดูที่ลักษณะของคำสั่งที่ใช้และลำดับการใช้งานของคำสั่งได้ โดยเราจะโหลดโมดูลเข้าไปเพื่อทำหน้าที่ดังกล่าวโดยลักษณะการทำงานของโมดูลจะเป็นไปตาม Flowchart ดังรูปที่ 4.2

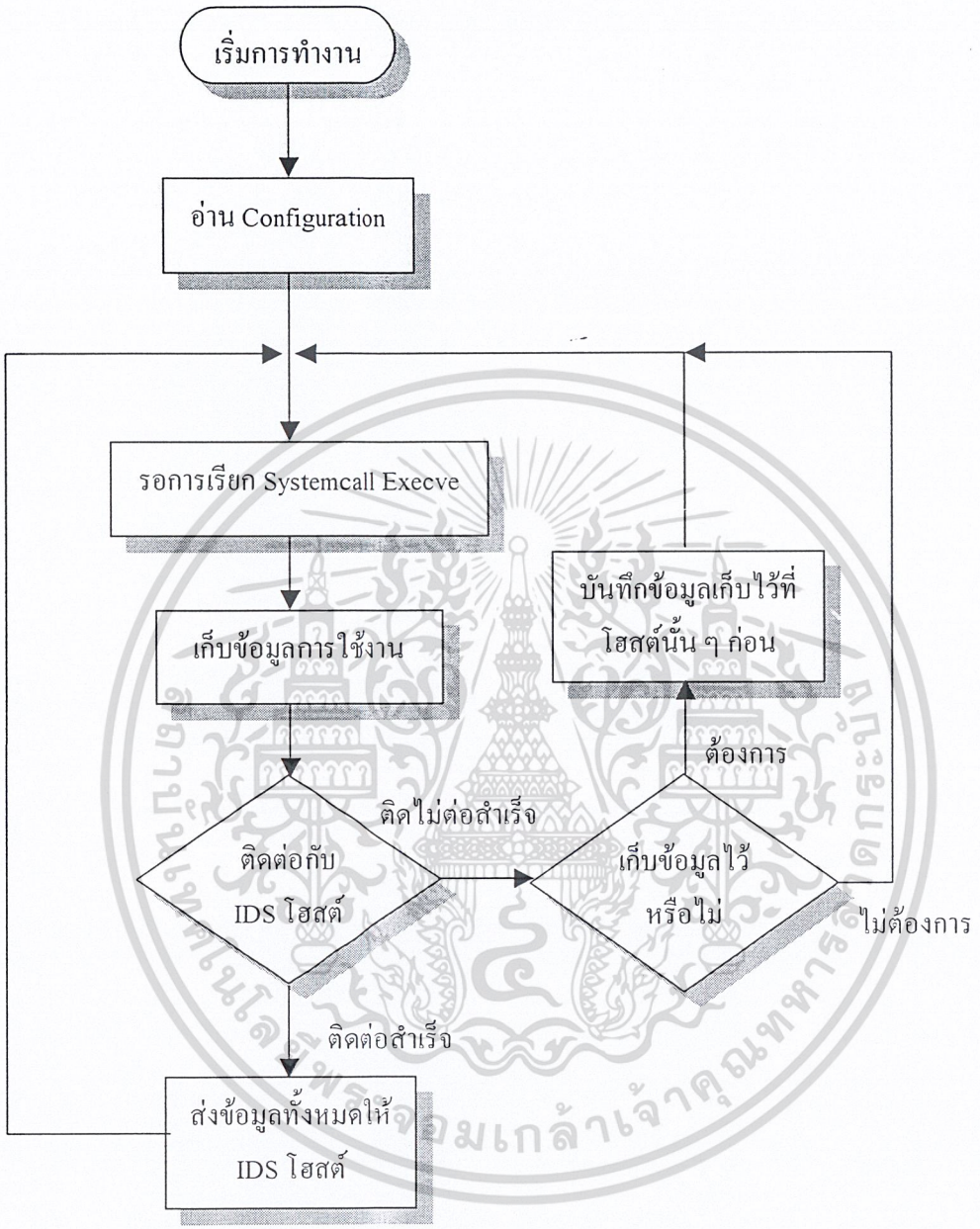
ลักษณะข้อมูลที่ส่งไปให้ IDS โฮสต์จะเก็บเป็นโครงสร้างดังนี้

```
struct execInfor {
  int uid;                          // หมายเลขประจำตัวผู้ใช้งาน
  char pathname[PATH_LEN];          // ตำแหน่งไฟล์ที่เรียกใช้งาน
  char command[MAX_COMM_LEN];       // คำสั่งและอาร์กิวเมนต์
  unsigned long startTime;          // เวลาที่ทำการเรียกคำสั่ง
};
```

หากสังเกตจะพบว่าทั้งๆที่ส่วนของข้อมูลเกี่ยวกับการใช้งานของผู้ใช้นั้นมีมากมายดังได้กล่าวไว้บ้างแล้วในบทที่ 2 แต่เรานำมาใช้เพียงเท่านี้เพราะส่วนวิเคราะห์ของระบบเราต้องการข้อมูลเพียงแค่นี้เท่านั้นถ้าเราเก็บข้อมูลมากไปก็จะทำให้สิ้นเปลืองทรัพยากรระบบและระบบจะทำงานช้าลงเพราะต้องเก็บข้อมูลทุกๆ คำสั่งที่ใช้งานของผู้ใช้ทุกคนที่กำลังใช้งานอยู่และหากสังเกตในอีกแง่หนึ่งก็จะพบว่าสิ่งที่เราทำการเก็บนั้นคล้ายกับข้อมูลที่อยู่ในล็อกไฟล์ pacct แต่ความจริงแล้วต่างกันเพราะ

1. การเก็บข้อมูลคำสั่งที่ใช้งานของล็อกไฟล์ pacct เก็บเฉพาะคำสั่งที่เรียกใช้งานเท่านั้น ข้อมูลที่เหลือเช่น ออปชัน และอาร์กิวเมนต์ต่าง ๆ จะไม่ถูกเก็บไว้ทำให้ได้ข้อมูลพฤติกรรมการใช้งานที่ไม่ครบถ้วน

2. การเก็บข้อมูลใน pacct จะทำขึ้นเมื่อโพรเซสนั้นๆ จบการทำงานแล้วเท่านั้น



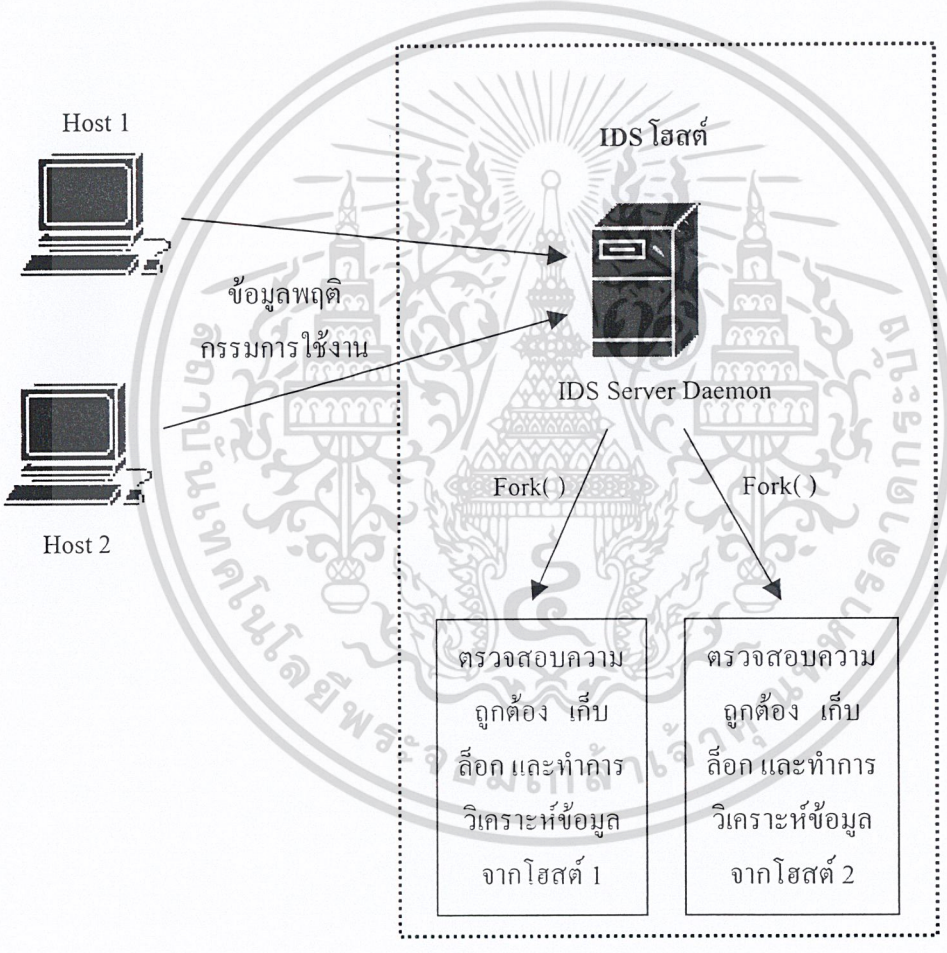
รูปที่ 4.2 ขั้นตอนการทำงานของส่วนเก็บข้อมูลพฤติกรรม

ดังนั้นเพื่อให้ได้ข้อมูลที่ใกล้เคียงกับพฤติกรรมผู้ใช้และมีความถูกต้องมากที่สุดและข้อมูลที่ได้มาเป็นข้อมูลที่สามารถใช้ในการวิเคราะห์ได้จริงๆ จึงใช้วิธีการเก็บข้อมูลด้วยการติดตามการทำงานของซิสเต็มคอลล์ดังที่กล่าวมา

4.2.2 ส่วนเก็บข้อมูลและวิเคราะห์พฤติกรรมผู้ใช้งาน

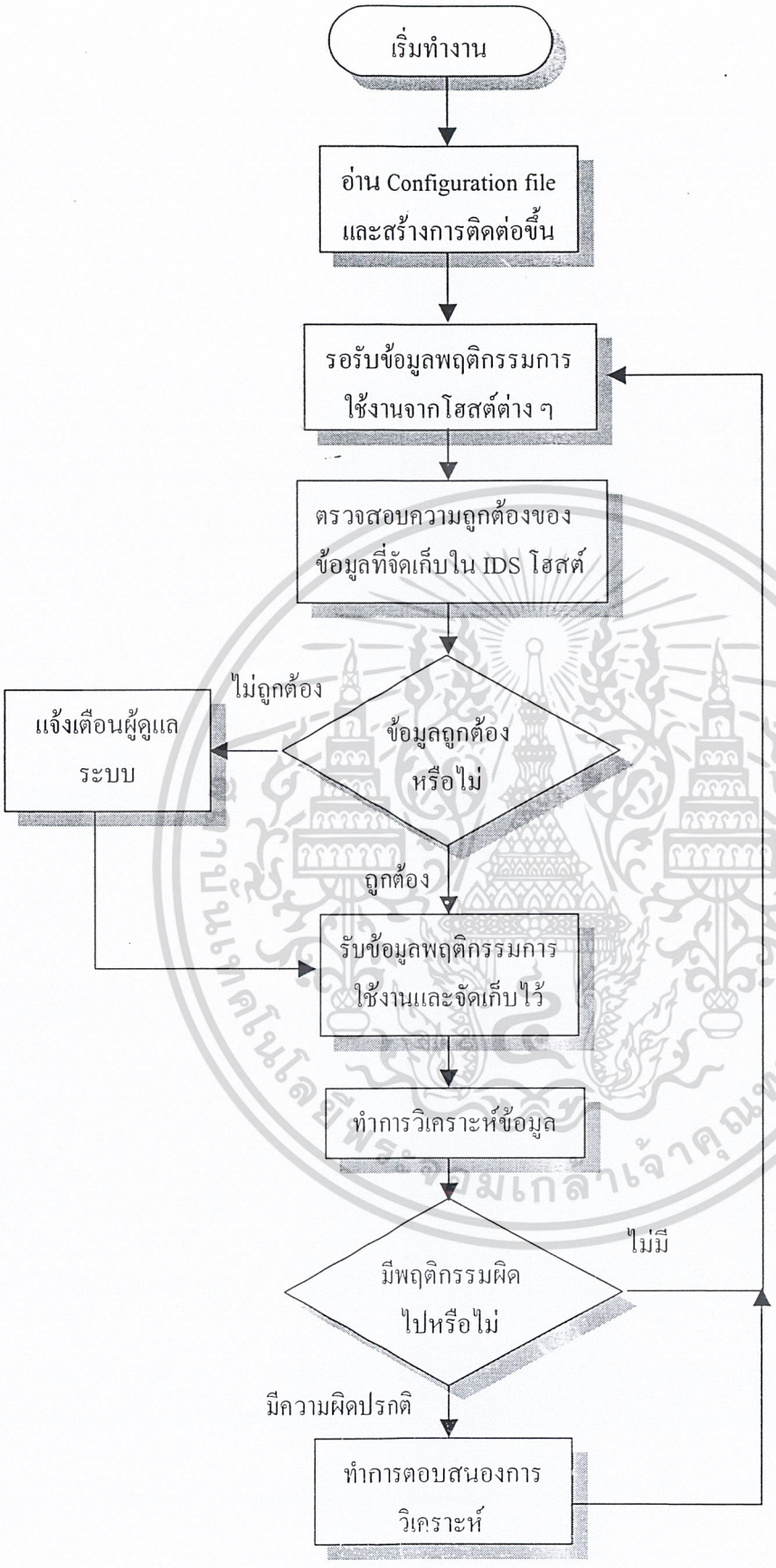
ส่วนเก็บข้อมูลและวิเคราะห์พฤติกรรมผู้ใช้งานจะถูกติดตั้งอยู่ที่ฝั่ง IDS โฮสต์ในส่วนนี้เป็นหน้าที่ของ IDS เดมอนโดยมีหน้าที่ 3 ส่วน คือ

1. ส่วนตรวจสอบความถูกต้องของพฤติกรรมผู้ใช้งาน ทำการตรวจสอบความถูกต้องของข้อมูลที่เก็บใน IDS โฮสต์
2. ส่วนเก็บข้อมูลพฤติกรรมผู้ใช้งาน ทำการรับข้อมูลพฤติกรรมการใช้งานที่ส่งมาจากโฮสต์ต่างๆ และเก็บไว้ใน IDS โฮสต์
3. ส่วนการวิเคราะห์พฤติกรรมผู้ใช้งาน นำข้อมูลในฐานข้อมูลมาผ่านกระบวนการวิเคราะห์และทำการตอบสนองการวิเคราะห์ โดยส่วนการตอบสนองการวิเคราะห์จะขอแยกหัวข้อออกไป



รูปที่ 4.3 ภาพรวมของส่วนเก็บข้อมูลและวิเคราะห์พฤติกรรมผู้ใช้งาน

การทำงานของ IDS เซิร์ฟเวอร์เดมอนมีลำดับการทำงานดังนี้



รูปที่ 4.4 ขั้นตอนการทำงานของส่วนเก็บข้อมูลและวิเคราะห์พฤติกรรมผู้ใช้งาน

ส่วนตรวจสอบความถูกต้องของข้อมูลพฤติกรรมผู้ใช้งาน

ในส่วนนี้จะทำการสร้าง MD (Message digest) ของฐานข้อมูลแต่ละคนแยกออกเป็นไฟล์ออกมาอีกส่วนหนึ่งโดยแต่ละคนก็จะมี MD คนละ 1 ไฟล์ เหตุผลที่ต้องทำ MD ของแต่ละคนแยกกันคนละไฟล์ คือ ในส่วนของ MD นี้จะมีการแก้ไขในทุกครั้งที่มีการเก็บข้อมูลใหม่ซึ่งแน่นอนว่าในช่วงเวลาเดียวกันนั้นต้องมีผู้ใช้งานที่ใช้งานพร้อมๆ กันหลายคนดังนั้นถ้าเราทำการรวม MD ของแต่ละคนเข้าเป็นไฟล์เดียวก็จะเกิดปัญหาในการอัปเดทส่วน MD ที่จะใช้งานซ้อนกัน

โดยอัลกอริทึมที่ใช้ในการสร้าง MD คืออัลกอริทึม MD5 มีความยาว 128 บิต

ส่วนเก็บข้อมูลพฤติกรรมผู้ใช้

โครงสร้างข้อมูลที่ใช้เก็บข้อมูลพฤติกรรมที่ส่วน IDS โฮสต์มีลักษณะดังนี้

```
struct userLog {
        char pathname[PATH_LEN];
        char command[MAX_COMM_LEN];
        unsigned long startTime;
};
```

ลักษณะการแยกเก็บข้อมูลนั้นผู้ใช้แต่ละคนที่มาจากแต่ละโฮสต์นั้นจะแยกตามหมายเลขไอพีแอดเดรสและหมายเลขประจำตัวผู้ใช้ที่ส่งมาพร้อมกับข้อมูล ยกตัวอย่างเช่น ข้อมูลพฤติกรรมผู้ใช้ที่มีหมายเลขประจำตัว 501 มาจากโฮสต์ที่มีหมายเลขไอพีแอดเดรสเป็น 161.246.2.201 ข้อมูลพฤติกรรมจะถูกเก็บอยู่ในไดเรกทอรีที่มีชื่อเป็น “161.246.10.21” และชื่อไฟล์ “501” โดยข้อมูลจะถูกเก็บเป็นเรคอร์ดต่อกันไปเรื่อยๆ

ส่วนการวิเคราะห์ข้อมูล

ลักษณะการวิเคราะห์ข้อมูลทำการวิเคราะห์ลักษณะการใช้งานคำสั่งโดยดูจากอาร์กิวเมนต์ และลำดับการใช้งานคำสั่ง

ในส่วนการวิเคราะห์ข้อมูลจะแบ่งเป็น 2 ขั้นตอนคือ

1. *ขั้นตอนเตรียมข้อมูลเพื่อการวิเคราะห์*

เป็นขั้นตอนที่เตรียมข้อมูลพฤติกรรมที่รับมาให้อยู่ในรูปแบบที่เหมาะสมและง่ายต่อการวิเคราะห์โดยวิธีการคือนำข้อมูลคำสั่งที่เก็บอยู่ในฟิล์ดข้อมูล command[MAX_COMM_LEN] แล้วแทนที่อาร์กิวเมนต์ที่ไม่ใช่ออปชัน(อาร์กิวเมนต์ที่เป็นออปชันจะขึ้นต้นด้วยตัวอักษร ‘-’) ด้วยข้อความ “<f>” เหตุผลต้องเปลี่ยนแปลงข้อมูลดังกล่าวออกไปเนื่องมาจากว่าสิ่งที่บอกถึงพฤติกรรมของผู้ใช้งานจะเป็นอาร์กิวเมนต์ที่เป็นออปชันเสียมากกว่า ยกตัวอย่างเช่น เมื่อเราใช้คำสั่ง emacs ซึ่งเป็นโปรแกรมเท็กซ์เอดิเตอร์โดยเรียกจากเชลล์

#emacs –nw a.txt b.txt

ชื่อไฟล์ที่ตามหลังออปชันนั้นจะไม่แทนถึงพฤติกรรมผู้ใช้มากนักเพราะในแต่ละครั้งที่ผู้ใช้เรียกโปรแกรมนี้ขึ้นใช้งานก็มักจะเปลี่ยนชื่อไฟล์ที่เรียก แต่ออปชันยังคงเดิม ดังนั้นจากคำสั่งที่ผู้ใช้งานป้อนเข้ามาเมื่อผ่านส่วนขั้นตอนการเตรียมข้อมูลเพื่อการวิเคราะห์แล้วก็จะได้เป็นข้อความดังนี้

emacs –nw <f> <f>

แต่การเตรียมข้อมูลด้วยวิธีนี้ยังมีจุดอ่อนคือ มีบางคำสั่งที่ออปชันการใช้งานนั้นอาจไม่ได้ขึ้นต้นด้วยตัวอักษร '-' เช่น คำสั่ง tar เป็นต้น

#tar xvfz global-4.2.tar.gz

เมื่อผ่านขั้นตอนการเตรียมข้อมูลเพื่อการวิเคราะห์ผลลัพธ์ที่ได้คือ

tar <f> <f> <f>

เห็นได้ว่าออปชัน 'xvfz' นั้นหายไปด้วยเพราะไม่ได้ขึ้นต้นด้วยตัวอักษร '-' แต่อย่างไรก็ตาม คำสั่งที่ใช้ออปชันที่ไม่ได้ขึ้นต้นด้วยตัวอักษร '-' มีอยู่เป็นส่วนน้อย

2. ขั้นตอนการวิเคราะห์ข้อมูล

วิธีการที่ใช้วัดค่าความเหมือนจะใช้การวัดค่าความเหมือน(similarity measure) ของข้อมูลโดยค่าค่านี้จะมีค่ามากเมื่อข้อมูลคู่ที่นำมาเปรียบเทียบนั้นมีความคล้ายกันมาก และมีค่าน้อยลงเมื่อข้อมูลคู่นั้นๆ มีความต่างกันออกไปโดยค่ายิ่งมากก็ยิ่งแสดงว่าคล้ายกันมากและในทางกลับกันถ้าข้อมูลมีความต่างกันมากก็จะมีค่าน้อยลง

โดยอัลกอริทึมความเหมือนที่ใช้มีขั้นตอนดังนี้

ให้ Seq_1 , Seq_2 แทนลำดับข้อมูลชุดที่ 1 และ 2 ตามลำดับ

Seq_i (i), แทนข้อมูลตัวที่ i ใน Seq_j

ให้ c เป็นค่าคงที่มีค่าเริ่มต้นที่ 1 และ Sim แทนค่าความเหมือนมีค่าเริ่มต้นที่ 0

เริ่มขั้นตอนการเปรียบเทียบความเหมือนของลำดับ Seq_1 , Seq_2

1. ในแต่ละตำแหน่งของ i ในลำดับที่ทำการเปรียบเทียบ
   - ถ้า Seq_1 (i) = Seq_2 (i) ให้ Sim:=Sim+c และเพิ่มค่าโดย c:=c+1
   - ถ้าไม่เท่ากันให้ค่า c=1
2. หลังจากทุกตำแหน่งทำการเปรียบเทียบแล้วค่า Sim ที่ได้มาคือค่าความเหมือนของลำดับทั้งสอง

คะแนนค่าความเหมือนจะมีค่าระหว่าง 0 ถึง n*(n+1)/2 โดย n คือความยาวของลำดับที่ทำการตรวจสอบ ยกตัวอย่างเพื่อให้เห็นภาพได้ชัดขึ้นดังนี้

ให้ Seq_1 = { "ls", "- -color", "<f>", "vi" , "<f>"}

และ Seq_2 = { "ls", "-p" , "<f>", "cat", "<f>"}

จะได้ค่าความเหมือนของลำดับทั้งสองดังตารางโดยการเปรียบเทียบลำดับทั้งสองดูจากซ้ายไปขวา

Seq_1		ls	--color	<f>	vi	<f>	คะแนนหลังการเปรียบเทียบ
Seq_2		ls	-p	<f>	cat	<f>	
C	ค่าเริ่มต้น=1	1	2	1	2	1	-
Sim	ค่าเริ่มต้น=0	0	1	1	2	2	3

ตารางที่ 4.1 แสดงการคำนวนค่าความเหมือนของลำดับ

จะได้ว่าลำดับทั้งสองนี้มีค่าความเหมือน 3

3. ค่าความเหมือนของลำดับ Seq_i ลำดับเดียวกับเซตของลำดับ L ได้ดังนี้

$$Sim(Seq_i, L) = \max_{Seq_j \varepsilon L} \{ Sim(Seq_i, Seq_j) \}$$

หมายความว่าค่าความเหมือนของลำดับ Seq_i เมื่อเปรียบเทียบกับเซตของลำดับ L แล้วคือค่า Sim ที่มีค่ามากที่สุดเมื่อทำการเปรียบเทียบ Seq_i กับทุกๆ ลำดับในเซต L

จากอัลกอริทึมด้านบนได้แสดงให้เห็นว่าในการทำการวิเคราะห์นั้นต้องมีลำดับอยู่ 2 ชุดนำมาเปรียบเทียบกัน เมื่อนำมาเปรียบเทียบกับลักษณะของฐานข้อมูลพฤติกรรมผู้ใช้งานของเราแล้วเราจะแบ่งฐานข้อมูลออกเป็น 2 ส่วนคือ

1. ฐานข้อมูลการใช้งานในช่วงระยะเวลาปัจจุบัน
2. ฐานข้อมูลการใช้งานในช่วงระยะเวลาที่ผ่านมา

แล้วนำฐานข้อมูลทั้ง 2 มาเปรียบเทียบค่าความเหมือนกันโดยในอัลกอริทึมขั้นตอนที่ 3 จะต้องนำลำดับพฤติกรรมการใช้งานในช่วงระยะเวลาปัจจุบันจำนวน 1 ลำดับมาทำการเปรียบเทียบกับฐานข้อมูลการใช้งานในช่วงระยะเวลาที่ผ่านมาทั้งหมดแล้วหาค่าสูงสุด และเมื่อได้ค่านั้นๆ ก็นำมาบวกเก็บไว้และทำการหาค่าความเหมือนของลำดับต่อไปของฐานข้อมูลการใช้งานในช่วงระยะเวลาปัจจุบันกับฐานข้อมูลการใช้งานในช่วงระยะเวลาที่ผ่านมาทั้งหมด จนกระทั่งสิ้นสุดข้อมูลของฐานข้อมูลการใช้งานในช่วงระยะเวลาปัจจุบัน ค่ารวมที่ได้คือค่าความเหมือนของพฤติกรรมการใช้งานในช่วงระยะเวลาปัจจุบันเทียบกับช่วงระยะเวลาที่ผ่านมา ซึ่งเราใช้ค่าค่านี้ในการวัดการเปลี่ยนแปลงไปของพฤติกรรมของผู้ใช้แต่ละคน หากค่านี้มีค่าสูงแสดงว่าพฤติกรรมผู้ใช้คนนั้นๆ ค่อนข้างคงที่ แต่ถ้าหากค่านี้มีค่าต่ำแสดงว่าพฤติกรรมของผู้ใช้มีการเปลี่ยนแปลงและหากมีค่าต่ำมากจนถึงระดับหนึ่งที่เราตั้งไว้ก็จะแสดงว่าพฤติกรรมผู้ใช้คนนั้นมีพิรุธ เข้าข่ายการบุกรุก

จากอัลกอริทึมข้างบนสังเกตได้ว่ามีตัวแปรที่สำคัญอยู่ 4 ตัวแปรคือ

1. จำนวนสมาชิกใน Seq แต่ละ Seq
2. จำนวนข้อมูลที่ใช้เปรียบเทียบจากฐานข้อมูลการใช้งานในช่วงระยะเวลาปัจจุบัน
3. จำนวนข้อมูลที่ใช้เปรียบเทียบจากฐานข้อมูลการใช้งานในช่วงระยะเวลาที่ผ่านมา
4. ค่าต่ำสุดที่ยอมรับได้ของการเปลี่ยนแปลงพฤติกรรม

ซึ่งค่าทั้ง 4 นี้จำเป็นต้องมีการปรับให้เหมาะสมในแต่ละระบบซึ่งจะได้อธิบายต่อไป

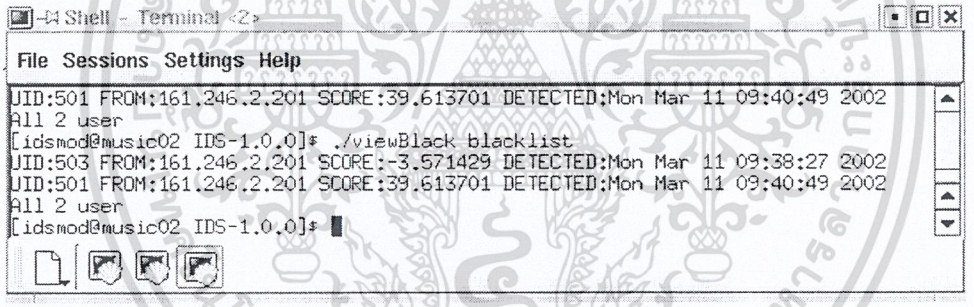
4.2.3 ส่วนตอบสนองการวิเคราะห์

เป็นส่วนที่ถูกเรียกใช้เมื่อระบบตรวจจับได้ว่ามีผู้ใช้ที่มีพฤติกรรมเปลี่ยนไปจนถึงระดับที่ผิดปรกติโดยจะแบ่งการทำงานเป็น 2 ประเภทคือ

1. ทำการเก็บล็อกไว้ใน IDS โฮสต์ โดยระบบจะมีอยู่ 2 ชนิดคือ

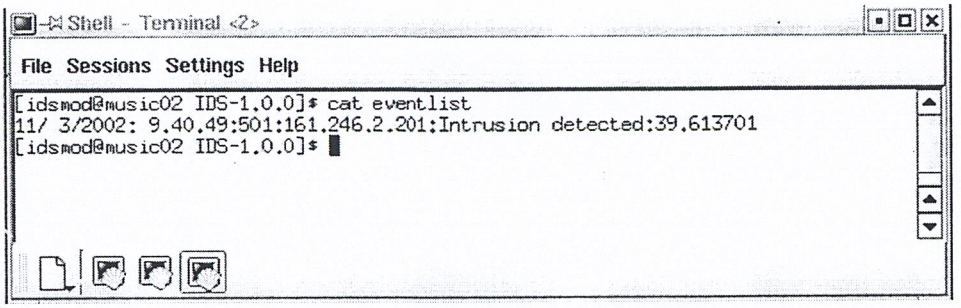
- Blacklist เก็บข้อมูลผู้ใช้งานที่เข้าข่ายมีพฤติกรรมการบุกรุกในขณะนั้นและเมื่อพฤติกรรมของผู้ใช้คนนั้นๆ เปลี่ยนแปลงไปในทางที่ดีขึ้นจนมีค่าพฤติกรรมที่เกินค่าที่กำหนดไว้ก็จะถูกลบออกจากรายชื่อนี้ โครงสร้างของไฟล์นี้จะเป็นเรคอร์ดต่อๆ กันโดยมีโครงสร้างดังนี้

```
struct blackUser{
  int uid;                      // uid ของผู้ใช้
  char hostIP[20];              // หมายเลขไอพีแอดเดรสที่ผู้ใช้ใช้งานอยู่
  float score;                  // ค่าคะแนนการบุกรุก
  unsigned long detectTime;     // เวลาที่ตรวจพบการบุกรุก
}
```



รูปที่ 4.4 แสดงข้อมูลที่อยู่ในไฟล์ blacklist

- Eventlist เก็บข้อมูลเหตุการณ์ต่างๆ ทั้งหมดที่เกิดขึ้นเช่นมีผู้ใช้เริ่มมีพฤติกรรมเข้าข่ายการบุกรุก มีผู้ใช้ที่มีพฤติกรรมดีขึ้นที่เปลี่ยนสถานะจากผู้บุกรุกเป็นผู้ใช้ปรกติ มีความผิดพลาดจากการตรวจสอบความถูกต้องของฐานข้อมูล เป็นต้น ไฟล์นี้มีโครงสร้างเป็นเท็กซ์ไฟล์ปรกติ



รูปที่ 4.6 แสดงข้อมูลที่อยู่ในไฟล์ eventlist

2. แจ้งเตือนผู้ดูแลระบบ โดยการส่งข้อความเตือนไปยังผู้ดูแลระบบ

บทที่ 5
การปรับค่าพารามิเตอร์ของระบบและผลการทดลอง

5.1 การปรับค่าพารามิเตอร์ของระบบ

จากที่ได้กล่าวมาแล้วในบทที่ 3 ว่าในอัลกอริทึมวัดค่าพฤติกรรมการใช้งานนั้นมีตัวแปรที่สำคัญอยู่ 4 ตัวแปรคือ

1. จำนวนสมาชิกใน Seq แต่ละ Seq
2. จำนวนข้อมูลที่ใช้เปรียบเทียบจากฐานข้อมูลการใช้งานในช่วงระยะเวลาปัจจุบัน
3. จำนวนข้อมูลที่ใช้เปรียบเทียบจากฐานข้อมูลการใช้งานในช่วงระยะเวลาที่ผ่านมา
4. ค่าต่ำสุดที่ยอมรับได้ของการเปลี่ยนแปลงพฤติกรรม

ภาพรวมของการปรับพารามิเตอร์ทั้ง 4 ตัวคือ การหาพารามิเตอร์ 3 ค่าแรกนั้นต้องทำการทดลองและเปรียบเทียบค่าเพื่อหาค่าที่เหมาะสมแล้วจึงใช้ค่าที่ได้ทั้ง 3 ไปช่วยหาค่าตัวแปรตัวที่ 4 ต่อไป

5.1.1 การทดลองหาค่าพารามิเตอร์ 3 ค่าแรกที่เหมาะสม

กำหนดตัวแปร

- ให้ SEQ_LEN แทนจำนวนสมาชิกใน Seq แต่ละ Seq
- ให้ R_WINDOW_SIZE แทนจำนวนข้อมูลที่ใช้เปรียบทียบจากฐานข้อมูลการใช้งานในช่วงระยะเวลาปัจจุบันให้
- ให้ DB_WINDOW_SIZE แทนจำนวนข้อมูลที่ใช้เปรียบทียบจากฐานข้อมูลการใช้งานในช่วงระยะเวลาที่ผ่านมา

ในการทดลองนี้ใช้ผู้ใช้งาน 3 คนเป็นตัวอย่างทดสอบโดย

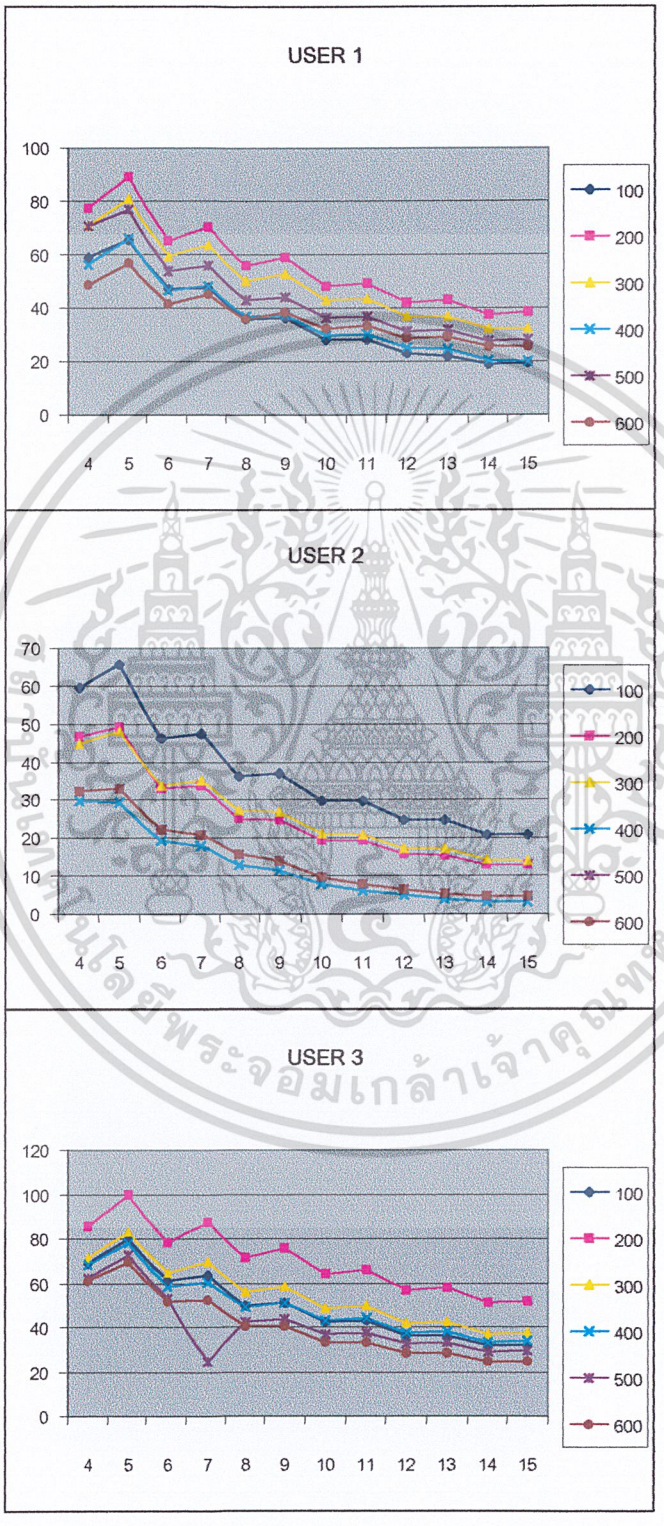
- ผู้ใช้คนที่ 1 มีจำนวนข้อมูลทั้งหมด 10580 เรคอร์ด
- ผู้ใช้คนที่ 2 มีจำนวนข้อมูลทั้งหมด 709 เรคอร์ด
- ผู้ใช้คนที่ 3 มีจำนวนข้อมูลทั้งหมด 14580 เรคอร์ด

โดยในผู้ใช้แต่ละคนจะทำการเปรียบเทียบข้อมูลดังนี้

1. SQ_LEN ที่ค่าระหว่าง 4 ถึง 15
2. R_WINDOW_SIZE ที่ค่า 100, 200, 300, 400, 500, 600
3. DB_WINDOW_SIZE ที่ค่า 100, 200, 400, 700

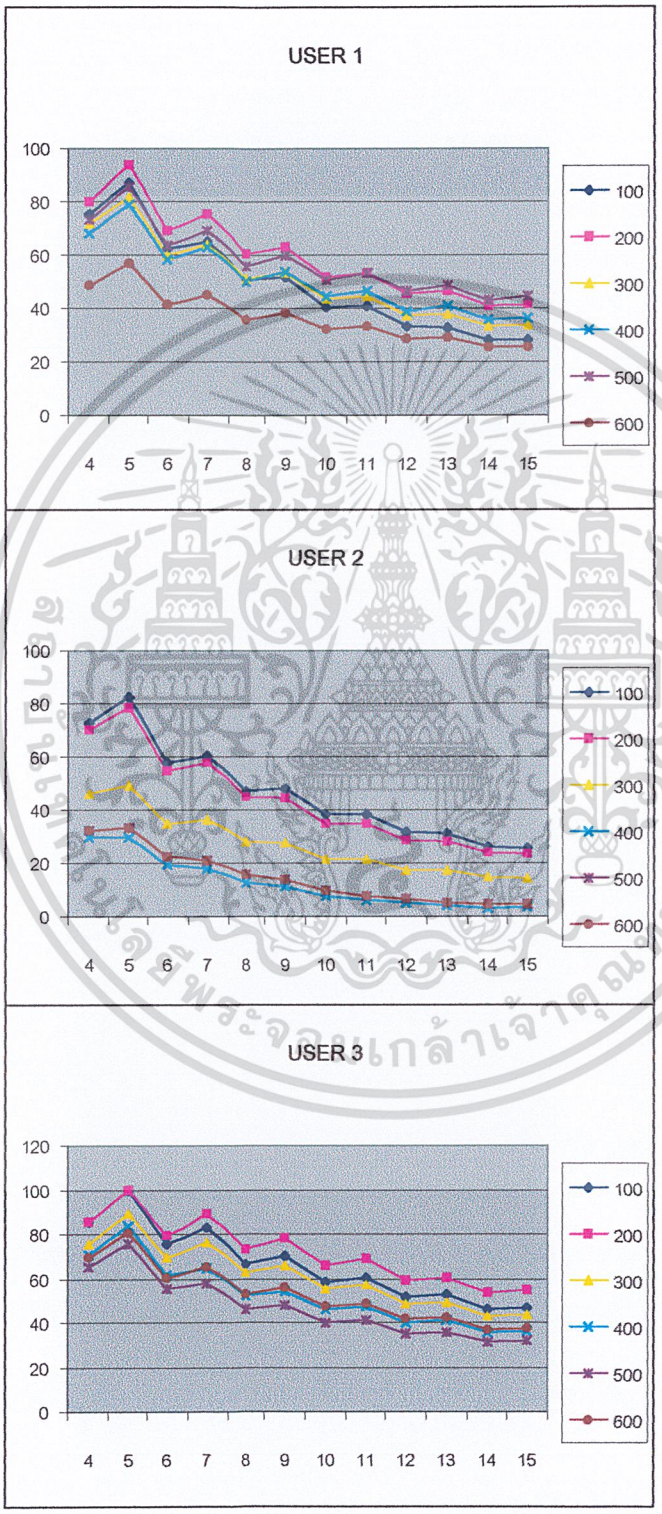
โดยในแต่ละกราฟผลการทดลองนั้นแกนนอน(x) หมายถึงค่า SEQ_LEN แกนตั้ง(y) หมายถึงค่าความเหมือนของพฤติกรรม ชุดข้อมูลแต่ละชุดในกราฟนั้นๆ จะเป็นการเปรียบเทียบค่า R_WINDOW_SIZE ตามที่กำหนดไว้ข้างต้น และในกราฟหนึ่งๆ จะมีค่า DB_WINDOW_SIZE คงที่

การทดลองครั้งที่ 1 ค่า DB_WINDOW_SIZE = 100



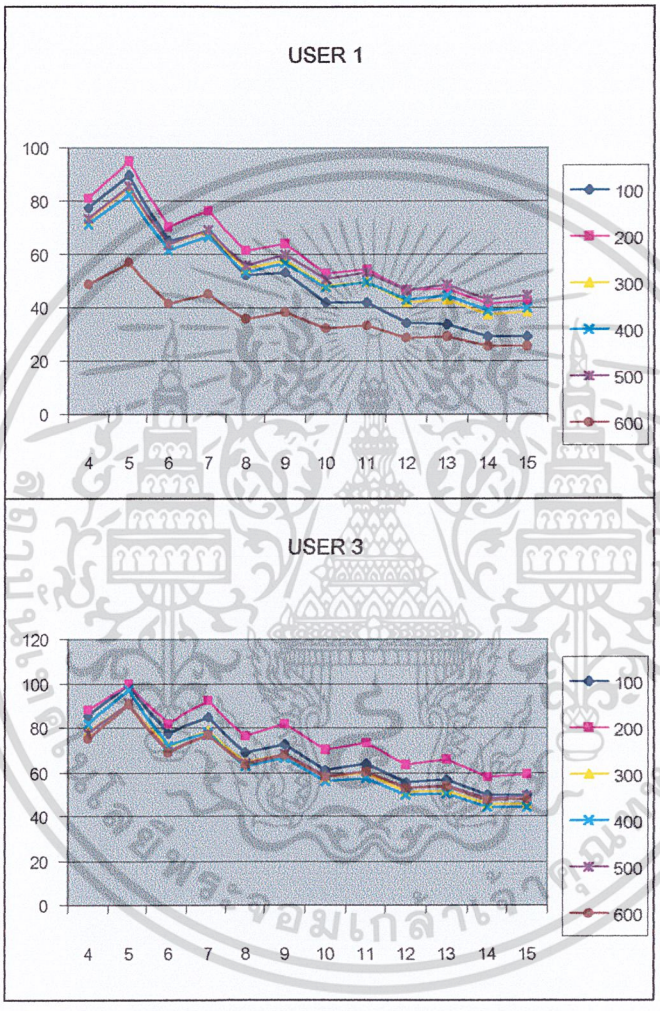
รูปที่ 5.1 การทดลองหาค่าพารามิเตอร์โดยให้ค่า DB_WINDOW_SIZE=100

การทดลองครั้งที่ 2 ค่า DB_WINDOW_SIZE = 200



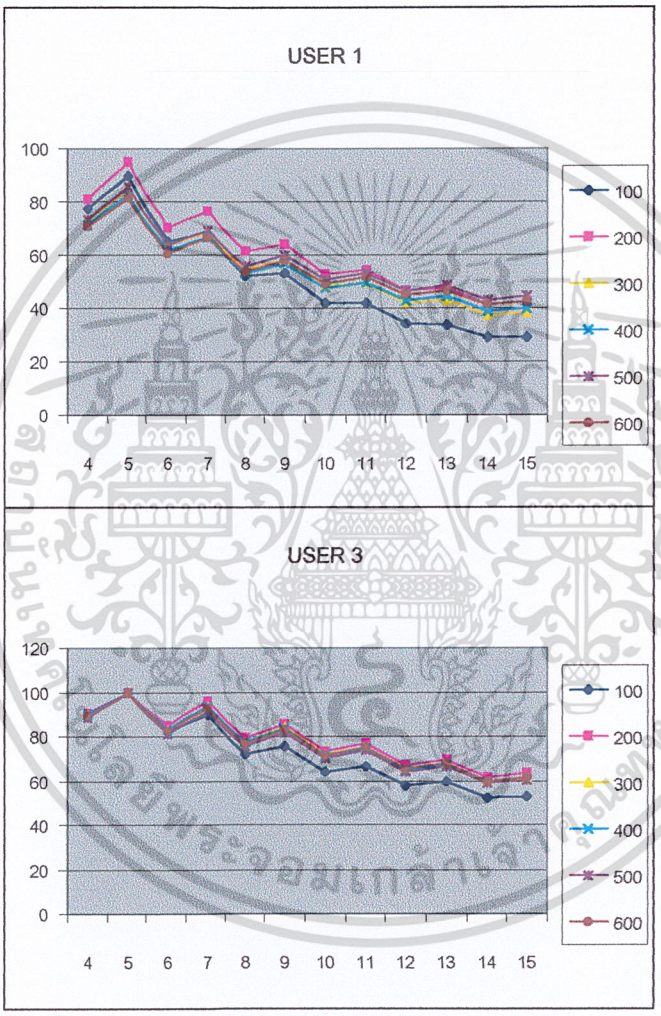
รูปที่ 5.2 การทดลองหาค่าพารามิเตอร์โดยให้ค่า DB_WINDOW_SIZE=200

การทดลองครั้งที่ 3 ค่า DB_WINDOW_SIZE = 400



รูปที่ 5.3 การทดลองหาค่าพารามิเตอร์โดยให้ค่า DB_WINDOW_SIZE=400

การทดลองครั้งที่ 4 ค่า DB_WINDOW_SIZE = 700



รูปที่ 5.4 การทดลองหาค่าพารามิเตอร์โดยให้ค่า DB_WINDOW_SIZE=700

สรุปและอธิบายผลการทดลอง

จากกราฟการทดลองทั้งหมดเราเลือกค่าตัวแปรทั้ง 3 ตัวได้ดังนี้

1. ค่า SEQ_LEN เลือกค่า 9

   เหตุผล สังเกตจากกราฟจะเห็นว่ายิ่งค่า SEQ_LEN มีค่ายิ่งน้อย ผลลัพธ์ก็จะยิ่งมีค่าความเหมือนมากเนื่องมาจากถ้า SEQ_LENน้อยๆ จะทำให้จำนวนคำสั่งที่นำมาเทียบในแต่ละครั้งนั้นมีจำนวนน้อยดังตัวอย่างข้างล่างนี้

   ข้อมูล: "ls", "- -color", "-la", "<f>", "vi", "<f>", "cc", "<f>", "-o", "<f>"

   กรณี SEQ_LEN=4 จะได้ Seq={ "ls", "- -color", "-la", "<f>" }

   กรณี SEQ_LEN=9 จะได้ Seq={ "ls", "- -color", "-la", "<f>", "vi", "<f>", "cc", "<f>", "-o" }

   ถ้าค่า SEQ_LEN ยิ่งมีค่าน้อยหมายความว่าในการเทียบลำดับข้อมูลในแต่ละครั้งจะเป็นการเทียบการใช้งานเพียงคำสั่งเดียว ซึ่งไม่ตรงกับจุดประสงค์ที่ต้องการเพราะเราต้องการวัดค่าลำดับการใช้งานคำสั่งด้วยจึงต้องเทียบกันหลายๆ คำสั่งในครั้งหนึ่งๆ เราจึงไม่เลือกค่า SEQ_LEN น้อยๆ มาใช้งาน

2. ค่า R_WINDOW_SIZE เลือกค่า 200

   เห็นได้อย่างชัดเจนว่าส่วนใหญ่ของกราฟผลลัพธ์ค่า R_WINDOW_SIZE มีค่า 200 ให้ผลลัพธ์ค่าความเหมือนที่สูงกว่าค่าอื่น ๆ

3. ค่า DB_WINDOW_SIZE เลือกค่า 400

   ค่า DB_WINDOW_SIZE ที่น้อยกว่านี้ให้ค่าความเหมือนที่ต่ำเกินไป และค่าที่สูงกว่านี้ให้ค่าความเหมือนที่ใกล้เคียงกัน ดังนั้นจึงเลือกค่า 400 เพราะจะใช้เวลาในการวิเคราะห์น้อยกว่าใช้เวลาในการเก็บฐานข้อมูลการวิเคราะห์น้อยกว่า

หมายเหตุ หากพิจารณาการทดลองที่มีค่า DB_WINDOW_SIZE=200 ของผู้ใช้คนที่ 2 จะเห็นว่ากราฟค่าความเหมือนมีค่าต่ำลงเรื่อยอย่างเห็นได้ชัด มีสาเหตุมาจากจำนวนข้อมูลฐานข้อมูลไม่เพียงพอเพราะเป็นผู้ใช้ใหม่ ดังนั้นการทดลองถัดไปจึงไม่ได้นำผู้ใช้คนที่ 2 เข้ามาทดลองด้วย

5.1.2 การหาค่าต่ำสุดที่ยอมรับได้ของการเปลี่ยนแปลงพฤติกรรม

จากการทดลองที่ผ่านมาเราได้ค่าพารามิเตอร์ที่ใช้ในการวิเคราะห์ครบแล้วทั้ง 3 ค่า ซึ่งจะนำมาใช้ในการหาค่าพารามิเตอร์ต่ำสุดที่ยอมรับได้ของการเปลี่ยนแปลงพฤติกรรมโดยมีวิธีการทดลองดังนี้

- ใช้ข้อมูลพฤติกรรมของผู้ใช้ 3 คนในการทดลอง
- ทดลองพฤติกรรมที่ทำให้เกิดการบุกรุกขึ้นโดยนำฐานข้อมูลการใช้งานในช่วงระยะเวลาปัจจุบันของแต่ละคน(R_DATA) มาเทียบกับฐานข้อมูลการใช้งานในช่วงระยะเวลาที่ผ่านมา (DB_DATA) ของทุกคน เสมือนกับว่าผู้ใช้คนหนึ่งถูกผู้ใช้คนอื่นนำแอคเคาท์ของตนมาใช้งานแล้วหาค่าความเหมือนของพฤติกรรม

- บันทึกและวิเคราะห์ผลการทดลอง

ซึ่งจากการทดลองได้ผลลัพธ์ดังนี้

DB_DATA / R_DATA	ผู้ใช้งานคนที่ 1	ผู้ใช้งานคนที่ 2	ผู้ใช้งานคนที่ 3
ผู้ใช้งานคนที่ 1	60.02	35.59	14.81
ผู้ใช้งานคนที่ 2	22.22	58.50	14.41
ผู้ใช้งานคนที่ 3	16.79	21.58	62.09

ตารางที่ 5.1 ผลการทดลองการเปรียบเทียบพฤติกรรม

สรุปผลการทดลอง

สังเกตว่าในระบบนี้ค่าพฤติกรรมของผู้ใช้ค่อนข้างไม่คงที่ โดยค่าความเหมือนพฤติกรรมของตัวผู้ใช้งานเองจะอยู่ที่ประมาณ 60 และค่าความเหมือนของพฤติกรรมเมื่อเทียบกับผู้ใช้คนอื่นๆ จะอยู่ที่ 35 ลงมา แสดงว่าช่วงค่าข้อมูลที่เราจะเลือกไปใช้เป็นค่าต่ำสุดที่ยอมรับได้ของการเปลี่ยนแปลงพฤติกรรมน่าจะอยู่ระหว่าง 35-59 โดยในที่นี้จะยึดค่า 40 เป็นค่าที่ใช้ในการทดสอบระบบต่อไป

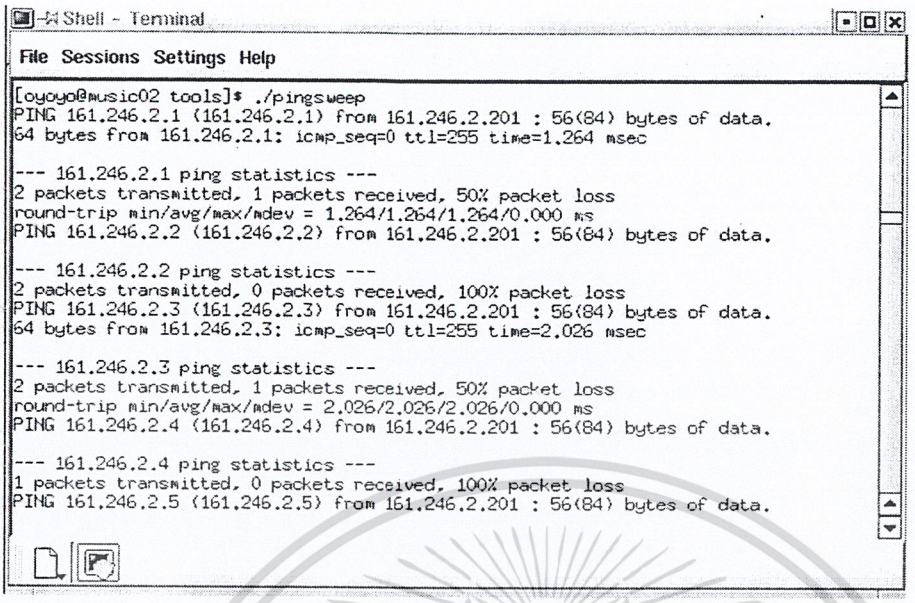
5.2 ทดลองบุกรุกโดยให้มีพฤติกรรมการใช้งานที่ผิดปรกติของผู้ใช้

ในหัวข้อนี้เป็นการทดสอบการใช้งานที่ผิดปรกติไปเพื่อทดสอบระบบตรวจจับผู้บุกรุก

Ping sweeps

เป็นหนึ่งในพื้นฐานการบุกรุกที่นิยมกระทำกัน ก็คือ การ ping ไปยังเครื่องเป้าหมายจำนวนมากพร้อม ๆ กัน ในลักษณะคล้ายกับการกวาดหรือกราดยิง ซึ่งเรียกว่าเป็นการทำ ping sweep เพื่อตรวจสอบว่าเครื่องปลายทางใดบ้างที่ยังเปิดทำงานอยู่ซึ่งจะได้ข้อมูลของระบบ โดยปรกติ ถ้าใช้คำสั่ง ping ธรรมดา, ping จะมีการส่งแพ็คเก็ต ICMP ECHO (Type 8) ออกไปยังเครื่องปลายทางและรอคอย ICMP ECHO_REPLY (Type 0) ที่ถูกส่งกลับมา

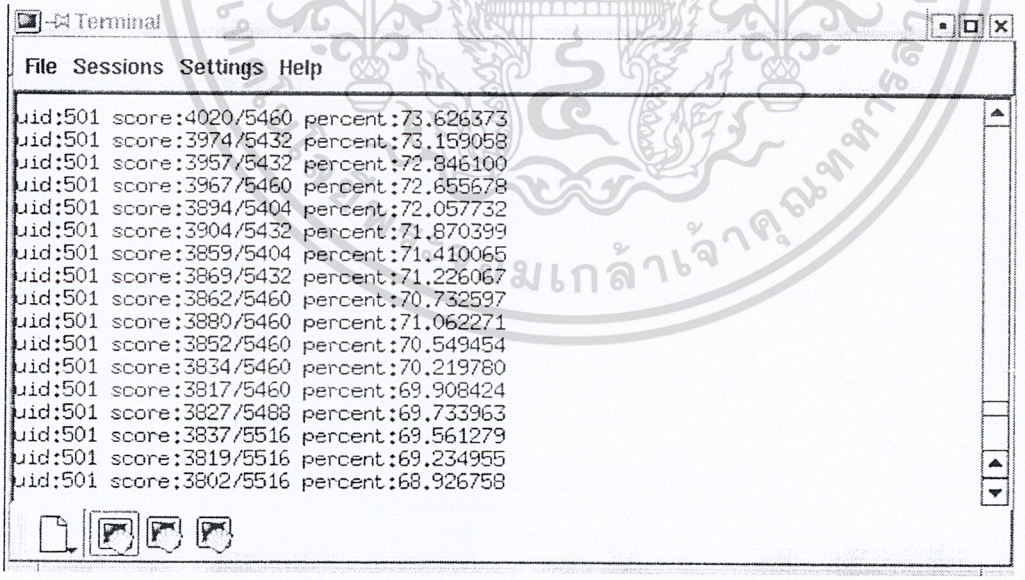
จากรูปที่ 5.5 สังเกตว่าจากการทำงานของ ping sweeps จะทำการ ping ไปยังเครื่องเป้าหมายตั้งแต่ IP หมายเลข 161.246.2.1 จนถึง IP หมายเลข 161.246.2.120 เพื่อสำรวจว่ามีเครื่องใดเปิดให้บริการอยู่บ้างเพื่อเป็นแนวทางในการติดต่อกับเครื่องนั้น ๆ เพื่อทำการบุกรุกต่อไป



รูปที่ 5.5 แสดงผลของการทำงาน ping sweeps

การทดลอง

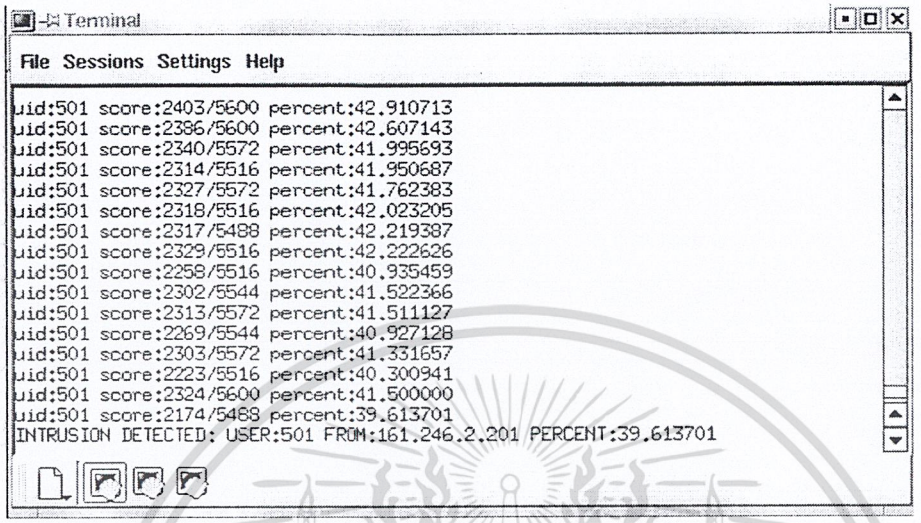
จากการทดลองโดยให้ผู้ใช้ทำการ ping sweeps ไปยังเครื่องเป้าหมาย โดยระบบตรวจจับผู้บุกรุกคอมพิวเตอร์สามารถรายงานความเปลี่ยนแปลงที่เกิดขึ้นจากการใช้คำสั่ง ping ดังรูปข้างล่างนี้



รูปที่ 5.6 แสดงผลการทำงานของ IDS โฮสต์

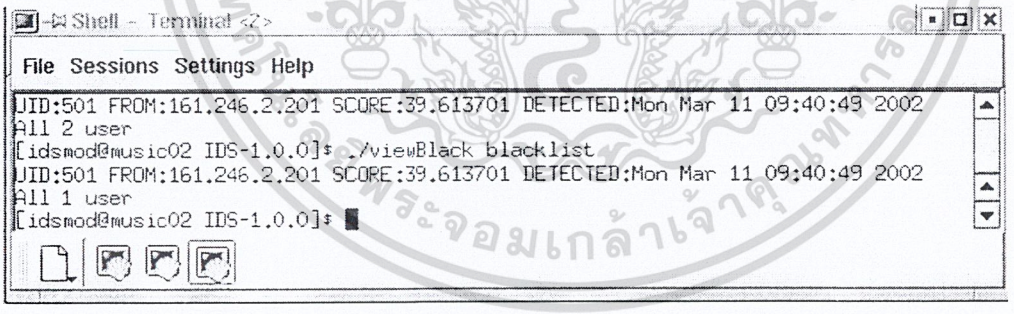
จากรูปที่ 5.6 เป็นผลการทำงานของ IDS โฮสต์ โดยแสดงหมายเลขประจำตัวของผู้ใช้งาน (User [illegible] ใช้คำสั่ง ping และมีค่าของเปอร์เซ็นต์โดยเปรียบเทียบค่าความเหมือนของคำสั่ง หากคำสั่งมีความ

เหมือนกับการใช้งานในช่วงระยะที่ผ่านมามากจะมีค่าสูง แต่หากมีความเหมือนกับการใช้งานในช่วงระยะผ่านมาน้อยจะมีค่าต่ำ ซึ่งสังเกตว่ามีผู้ใช้ที่มีค่า UID = 501 ใช้คำสั่ง ping ติดต่อกันเป็นจำนวนมากทำให้ค่าความเหมือนของการใช้งานลดลงเรื่อย ๆ



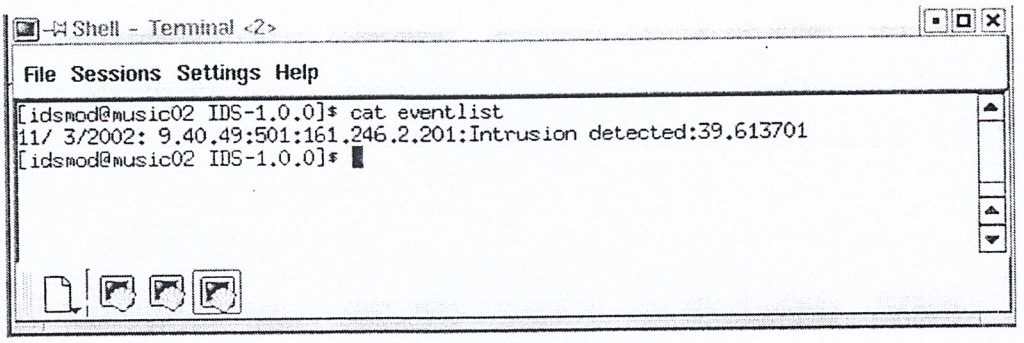
รูปที่ 5.7 แสดงผลการทำงานของ IDS โฮสต์ เมื่อตรวจพบการบุกรุก

จากรูปที่ 5.7 เมื่อผู้ใช้มีค่าความเหมือนต่ำกว่าค่าที่ตั้งไว้ (ในที่นี้ตั้งค่าไว้ที่ 40) IDS โฮสต์ ก็ทำการตอบสนองจากการวิเคราะห์ว่ามีการบุกรุกเกิดขึ้นให้แก่ผู้ดูแลระบบได้ทำการติดตามต่อไป



รูปที่ 5.8 บันทึกข้อมูลการบุกรุกลงไฟล์ blacklist

จากรูปที่ 5.8 จะบันทึกข้อมูลต่าง ๆ ของผู้บุกรุกไว้ในไฟล์ที่ชื่อ blacklist โดยในไฟล์นี้จะบันทึกข้อมูลแบบลักษณะของไดนามิก คือ ปรับเปลี่ยนขนาดได้เมื่อมีผู้บุกรุกเพิ่มขึ้นหรือลดลง ข้อมูลที่ทำการบันทึก เช่น หมายเลขประจำตัวผู้ใช้ หมายเลขไอพี (IP Address) ค่าของความเหมือน วัน เดือน ปี และเวลาที่เกิดการบุกรุก เพื่อใช้ในการตรวจสอบในภายหลังว่าเมื่อมีการติดต่อจากผู้ใช้นี้ให้ติดตามพฤติกรรมการใช้งานว่ามีพฤติกรรมที่เป็นปรกติมากขึ้นหรือไม่ หากมีพฤติกรรมที่เป็นพฤติกรรมปรกติมากขึ้นก็ทำการลบรายชื่อออกจากไฟล์ blacklist



รูปที่ 5.9 แสดงไฟล์ eventlist

จากรูปที่ 5.9 แสดงไฟล์ eventlist ที่เก็บรวมรวมข้อมูลทั้งหมดเป็นล็อกไฟล์ คือ เมื่อเกิดพฤติกรรมการบุกรุกขึ้น ระบบตรวจจับผู้บุกรุกคอมพิวเตอร์จะทำการบันทึกประวัติการบุกรุกทั้งหมดไว้ในไฟล์นี้ โดยแตกต่างจากไฟล์ blacklist คือ เมื่อผู้บุกรุกมีพฤติกรรมการใช้งานที่ดีขึ้นแล้ว ระบบจะทำการลบรายชื่อผู้ใช้งานนั้นออก แต่ในไฟล์ eventlist ยังคงเก็บไว้อยู่ เพื่อมีไว้ให้ผู้ดูแลระบบจะทำการตรวจสอบในภายหลังได้

บรรณานุกรม

เอกสารอ้างอิง

[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] Aha, D., Kibler. D., & Albert. M (1991). Instance-based learning algorithms. *Machine Learning, 6,* 37-66

[4] Anderson, J.P. (1980). *Computer security threat monitoring and surveillance*, (Technical Report), Washington, PA, James P. Anderson Co.

[5] Denning, D.E. (1987). An intrusion-detection model. *IEEE Transactions on Software Engineering, 13,222-232*

[6] Dietterich, T.G. , & Michalski, R.S. (1986). Learning to predict sequences. In Michalski, Carbonell & Mitchell (Eds.), *Machine Learning: An artificial intelligence approach*. San Mateo, CA: Morgan Kaufmann

[7] Frank, J. (1994). Machine learning and intrusion detection: *Current and future directions. Proc. of the 17th National Computer Security Conference.*

[8] Smaha, S.E. (1988). Haystack: An intrusion detection system. *Proceedings of the Fourth Aerospace Computer Security Applications Conference* (pp.37-44).

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

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

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

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

[13] Smaha, S.E. (1988). Haystack: An intrusion detection system. *Proceedings of the fourth Aerospace Computer Security Applications Conference* (pp. 37-44).

[14] Daniel P. Bovet ,Marco Cesti: *"Understanding the Linux Kernel"* , O'Reilly & Associates, Inc,2001

[15] Syngress: *"Hack proofing your network"* , Syngress Media, Inc.,2000

[16] Stefan Axelsson, Ulf Lindqvist, Ulf Gustafson, Erland Johnsson: *"An Approach to UNIX Security Logging"*. Department of Computer Engineering, Chalmers University of Technology

[17] Terran Lane,Carla E. Bordley: *"An Application of Machine Learning to Anomaly Detection"*,School of Electrical and Computer Engineering, Purdue University,West Lafayette,1997

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

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

[20] มนตรี พจนารถลาวัณย์ : *"การเขียนโปรแกรมคอมพิวเตอร์ด้วยเทอร์โบซี"*, บริษัท ซีเอ็ดยูเคชัน จำกัด (มหาชน), 2540

[21] (ผู้แปล) เอกสิทธิ์ วิริยจารี : "ปีดทางแฮกเกอร์", บริษัท ซีเอ็ดยูเคชัน จำกัด (มหาชน), 2544

เว็บไซต์อ้างอิง

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

[39] http://www.hackingexposed.com : Hacking exposed book web site