# บทที่ 2

# ทฤษฎี หลักการ และงานวิจัยที่เกี่ยวข้อง

## 2.1 บทนำ

ในบทนี้จะกล่าวถึงทฤษฎี หลักการและงานวิจัยที่เกี่ยวข้องกับการทำวิทยานิพนธ์โดยที่เนื้อหาในส่วนแรกจะเป็นการกล่าวถึงภัยคุกคามระบบคอมพิวเตอร์ ลักษณะการบุกรุกและผลกระทบที่เกิดขึ้นหลังจากนั้นจะเป็นการกล่าวถึงระบบตรวจจับการบุกรุก ประเภทของระบบตรวจจับการบุกรุก ค่าประจำตัวผู้ใช้ในระบบปฏิบัติการยูนิกซ์ (user credentials) เรื่องของ system call การเปลี่ยนแปลงสถานะ (state transition) และกฎที่ใช้สนับสนุนการตรวจจับการบุกรุกในวิทยานิพนธ์ชิ้นนี้

## 2.2 ภัยคุกคามระบบคอมพิวเตอร์ (Threat)

ภัยคุกคามระบบคอมพิวเตอร์ในที่นี้หมายถึงการกระทำอันก่อให้เกิดความเสียหายแก่ระบบคอมพิวเตอร์ ซึ่งไม่ว่าจะเป็นการขโมยข้อมูล การทำลายข้อมูล ไปจนกระทั่งการทำให้ระบบไม่สามารถใช้งานได้ ภัยคุกคามอาจจะเกิดขึ้นโดยอุบัติเหตุ เช่น ไฟไหม้ น้ำท่วม หรือเกิดขึ้นโดยเจตนาของบุคคลที่ต้องการก่อให้เกิดความเสียหายแก่ระบบโดยการโจมตี การกระทำที่ก่อให้เกิดภัยคุกคามเหล่านี้เป็นการกระทำที่ทำให้ระบบขาดความปลอดภัยซึ่งระบบที่มีความปลอดภัยจะหมายถึงระบบที่มีการให้บริการในลักษณะอย่างน้อย 3 ประการคือ [9]

1. การรักษาความลับ (confidentiality) คือการรับรองว่าจะมีการเก็บข้อมูลไว้เป็นความลับ และผู้มีสิทธิเท่านั้นจึงจะเข้าถึงข้อมูลนั้นได้
2. การรักษาความสมบูรณ์ (integrity) คือการรับรองว่าข้อมูลจะไม่ถูกเปลี่ยนแปลงหรือถูกทำลายโดยผู้ที่ไม่มีสิทธิไม่ว่าจะเป็นโดยอุบัติเหตุหรือโดยเจตนา
3. ความพร้อมใช้ (availability) คือการรับรองว่าข้อมูลและบริการสื่อสารต่าง ๆ พร้อมที่จะใช้ได้ในเวลาที่ต้องการใช้งาน

ฉะนั้นหากมีพฤติกรรมใดพฤติกรรมหนึ่งที่เป็นผลทำให้คุรสมบัติข้อใดข้อหนึ่งในลักษณะที่กล่าวมาข้างต้นขาดหายไปเราจะเรียกพฤติกรรมเหล่านั้นว่า "การบุกรุก" (intrusion) และเรียกผู้ที่กระทำพฤติกรรมดังกล่าวว่า "ผู้บุกรุก" (intruder) ภัยคุกคามระบบคอมพิวเตอร์แบ่งออกเป็นสองกลุ่มใหญ่ ๆ คือ ภัยคุกคามแบบพาสซีฟ (passive threat) และภัยคุกคามแบบแอ็คทีฟ (active threat)

**ภัยคุกคามแบบพาสซีฟ**

ภัยคุกคามแบบพาสซีฟ เป็นภัยคุกคามที่ไม่ก่อให้เกิดความเสียหายแก่ระบบคอมพิวเตอร์โดยตรงบุคคลที่กระทำให้เกิดภัยคุกคามหรือผู้บุกรุก จะใช้วิธีการใดวิธีการหนึ่งเพื่อให้ได้มาซึ่งข้อมูลที่ต้องการอาทิเช่น การติดตั้งโปรแกรมตรวจจับแพ็กเกต (packet sniffing) ซึ่งคล้ายกับลักษณะของการดักฟังตัวอย่างการกระทำหรือการโจมตีในลักษณะนี้ได้แก่ การขโมยชื่อบัญชีและรหัสผ่านของผู้ใช้ซึ่งอาจทำได้โดยการดักจับข้อมูลในขณะที่ผู้ใช้ใส่ชื่อบัญชีและรหัสผ่านเพื่อติดต่อขอใช้บริการไปยังเครื่องเซิร์ฟเวอร์ใด ๆ ชื่อบัญชีและรหัสผ่านเหล่านั้นก็จะถูกส่งไปยังผู้บุกรุกเนื่องจากข้อมูลที่วิ่งอยู่บนระบบเครือข่ายโดยส่วนใหญ่มักจะเป็นข้อมูลดิบที่ไม่มีการเข้ารหัสทำให้ผู้บุกรุกสามารถดักจับข้อมูลเหล่านี้ได้ ลักษณะการโจมตีแบบนี้จะรู้จักกันในชื่อที่เรียกว่า "สนิฟเฟอร์ (sniffer)" อีกตัวอย่างหนึ่งของภัยคุกคามแบบพาสซีฟได้แก่ การโจมตีแบบ social engineering โดยที่ social engineering เป็นการหลอกให้ผู้ที่เป็นเป้าหมายของการโจมตีเปิดเผยข้อมูลที่ต้องการเช่นหมายเลขบัตรเครดิตหรือรหัสผ่านบัตรธนาคาร ฯลฯ ลักษณะการโจมตีแบบนี้อาจจะส่งมาทางอีเมล์โดยที่จะมีเนื้อความและชื่ออีเมล์ ตลอดจนชื่อผู้ส่งที่ปลอมขึ้นมาเพื่อให้ดูน่าเชื่อถือ การโจมตีที่ก่อให้เกิดภัยคุกคามแบบพาสซีฟอื่น ๆ ได้แก่ eavesdropping, wiretap, port scanning เป็นต้น

**ภัยคุกคามแบบแอคทีฟ**

ภัยคุกคามแบบแอคทีฟ เป็นภัยคุกคามที่ก่อให้เกิดความเสียหายแก่ข้อมูลในระบบ ทำให้ระบบหยุดทำงาน หรือทำให้การติดต่อสื่อสารระหว่างเครื่องคอมพิวเตอร์ติดขัดหรือขาดการติดต่อ ภัยคุกคามประเภทนี้เช่น การโจมตีที่เรียกว่า denial of service attack (DoS) การโจมตีแบบนี้ผู้บุกรุกจะใช้วิธีการขอใช้บริการที่ให้บริการอยู่ในระบบ โดยการขอจองทรัพยากรที่มีในระบบแบบสะสมด้วยอัตราที่รวดเร็ว จนกระทั่งระบบไม่มีทรัพยากรเหลือพอที่จะให้บริการผู้ใช้อื่น วิธีการที่นิยมใช้เป็นการสร้างแพ็กเกตขอเชื่อมต่อผ่านทางระบบเครือข่ายโดยการใช้โปรโตคอลทีซีพีจำนวนมาก

เรียกว่า TCP SYN Flooding และ ping of death ซึ่งเป็นการสร้างแพ็กเกตขนาดใหญ่ส่งไปยังบริการไอซีเอ็มพีด้วยคำสั่ง "ping" ตัวอย่างอื่น ๆ ของภัยคุกคามประเภทแอคทีฟได้แก่ การโจมตีโดยการใช้โปแกรมประเภทไวรัส (virus program) หนอนอินเตอร์เน็ต (Internet worm) ม้าโทรจัน (trojan horse) อีเมล์บอมบ์ (e-mail bomb) ประตูกล (backdoors) สปายแวร์ (spyware) ลอจิกบอมบ์ (logic bomb) ซึ่งจะขอกล่าวรายละเอียดเฉพาะการโจมตีที่พบกันแพร่หลายซึ่งได้แก่ โปรแกรมประเภทไวรัส หนอนอินเตอร์เน็ต และม้าโทรจัน ดังนี้

**โปรแกรมประเภทไวรัส**

โปรแกรมประเภทไวรัส เป็นโปรแกรมคอมพิวเตอร์ประเภทหนึ่งที่ถูกออกแบบมาให้แพร่กระจายตัวเองจากแฟ้มหนึ่งไปยังแฟ้มอื่นๆ ภายในเครื่องคอมพิวเตอร์ ไวรัสจะแพร่กระจายตัวเองอย่างรวดเร็วไปยังทุกแฟ้มภายในคอมพิวเตอร์แต่ไวรัสจะไม่สามารถแพร่กระจายจากเครื่องหนึ่งไปยังอีกเครื่องหนึ่งได้ด้วยตัวมันเองโดยทั่วไปเกิดจากการที่ผู้ใช้เป็นพาหะนำไวรัสจากเครื่องหนึ่งไปยังอีกเครื่องหนึ่ง เช่นผู้ใช้ส่งอีเมล์โดยแนบเอกสารหรือแฟ้มที่มีไวรัสไปด้วย การทำสำเนาแฟ้มที่ติดไวรัสไปไว้บนไฟล์เซริฟเวอร์ การแลกเปลี่ยนแฟ้มที่ติดไวรัสโดยใช้แผ่นดิสก์เก็ตเมื่อผู้ใช้ทั่วไปรับแฟ้มจากดิสก์เก็ตมาใช้งานไวรัสก็จะแพร่กระจายภายในเครื่องคอมพิวเตอร์นั้น ๆ และจะเป็นวงจรในลักษณะนี้ต่อไป

**หนอนอินเตอร์เน็ต**

หนอนอินเตอร์เน็ตโดยทั่วไปคล้ายกับโปรแกรมประเภทไวรัสแต่หนอนอินเตอร์เน็ตเป็นโปรแกรมที่ถูกออกแบบมาให้สามารถแพร่กระจายตัวเองจากเครื่องคอมพิวเตอร์เครื่องหนึ่งไปยังอีกเครื่องหนึ่งโดยอาศัยระบบเครือข่ายและจะแพร่กระจายได้อย่างรวดเร็วและทำความเสียหายรุนแรงกว่าไวรัสมาก

**ม้าโทรจัน**

ม้าโทรจันเป็นโปรแกรมคอมพิวเตอร์ที่ถูกออกแบบมาให้แฝงตัวเองเข้าไปในระบบเพื่อให้ทำงานตามที่ผู้เขียนโปรแกรมม้าโทรจันกำหนด อาทิเช่น โปรแกรมคอมพิวเตอร์ที่แฝงตัวในระบบและทำงานโดยการดักจับเอารหัสผ่านที่ผู้ใช้ใช้ในการเข้าสู่ระบบต่างๆ แล้วส่งกลับไปยังผู้

บุกรุก เพื่อเข้าใช้หรือโจมตีระบบในภายหลัง โดยที่โปรแกรมเหล่านี้จะแฝงมาในหลาย ๆ รูปแบบ อย่างเช่น เกมส์ การ์ดอวยพร หรือจดหมายต่าง ๆ โปรแกรมม้าโทรจันไม่ได้ถูกออกแบบมาเพื่อทำลายระบบ หรือสร้างความเสียหายต่อระบบคอมพิวเตอร์ โปรแกรมม้าโทรจันต่างจากโปรแกรมประเภทไวรัสและหนอนอินเตอร์เน็ตคือมันไม่สามารถทำสำเนาตัวเองและแพร่กระจายตัวเองได้แต่สามารถที่จะอาศัยตัวกลางซึ่งอาจเป็นโปรแกรมต่าง ๆ จดหมาย หรือการที่ผู้ใช้ไปดาวน์โหลดแฟ้มข้อมูลจากแหล่งต่างๆ เมื่อเรียกใช้งานแฟ้มเหล่านี้ โปรแกรมม้าโทรจันที่แฝงตัวมาก็จะทำงานและจะเปิดช่องทางต่างๆให้ผู้บุกรุกเข้าโจมตีระบบได้

## 2.3 ช่องทางส่วนใหญ่ที่ผู้บุกรุกใช้ในการเข้าสู่ระบบ

รายละเอียดในหัวข้อนี้จะกล่าวถึงช่องทางที่ผู้บุกรุกสามารถเข้าสู่ระบบซึ่งได้แก่ จุดอ่อนของระบบ (vulnerabilities) ข้อบกพร่องในการตั้งรหัสผ่านของผู้ใช้

### 2.3.1 จุดอ่อนของระบบ

จุดอ่อนของระบบเป็นปัญหาที่มีอยู่ในการกำหนดค่าในระบบปฏิบัติการ โปรแกรมในระบบหรือสคริปที่ถูกเขียนขึ้น ซึ่งจุดอ่อนเหล่านี้ได้แก่

**การล้นของบัฟเฟอร์ (Buffer overflows)**

การล้นของบัฟเฟอร์เป็นจุดอ่อนที่เกี่ยวเนื่องกับเทคนิคในการโปรแกรมอาทิเช่น การใช้ฟังก์ชัน sprintf(), vsprintf(), scanf() หรือ gets() โดยไม่มีการตรวจสอบขนาดของอากิวเมนต์ที่ส่งมา ผู้บุกรุกจะใช้วิธีการส่งค่าที่มีขนาดใหญ่กว่าขนาดของบัฟเฟอร์ที่ผู้พัฒนากำหนดไว้ ทำให้การชี้ของพอยน์เตอร์เปลี่ยนตำแหน่งมายังตำแหน่งที่ผู้บบุกรุกต้องการและรันคำสั่งอื่นผู้บุกรุกสามารถรันโปรแกรมที่ต้องการได้ โดยทั่วไปเป้าหมายของผู้บุกรุกในการใช้จุดอ่อนนี้ต้องการจะให้ได้มาซึ่งสิทธิของผู้ใช้ที่มีสิทธิสูงสุด [10] ตัวอย่างการใช้จุดอ่อนในเรื่องของการล้นของบัฟเฟอร์ในการบุกรุกเช่น ผู้พัฒนาได้กำหนดขนาดของตัวแปรที่จะรับค่ารหัสประจำตัวผู้ใช้เป็น 250 ตัวอักษร แต่ผู้บุกรุกใส่ค่าในตัวแปรด้วยขนาด 300 ตัวอักษร ซึ่งใน 50 ตัวอักษรที่เกินมาประกอบด้วยโค้ดที่จะทำให้ผู้บุกรุกเข้าสู่ระบบได้

**Race Condition**

Race condition เป็นปัญหาที่เกิดขึ้นโดยอาศัยการให้บริการของระบบปฏิบัติการที่ทำงานแบบ multitasking หรือ multithread นั่นคือสามารถทำงานมากกว่าหนึ่งอย่างในเวลาเดียวกันได้ race condition จะเกิดขึ้นเมื่อมีโปรเซสตั้งแต่สองโปรเซสขึ้นไปเรียกใช้ทรัพยากรเดียวกันและมีการแก้ไขทรัพยากรของระบบเช่น แฟ้มข้อมูล ทำให้มีโอกาสที่เกิดความผิดพลาดแก่แฟ้มนั้นได้ ตัวอย่างเช่น [11] programs A และ B จำเป็นที่จะต้องแก้ไขแฟ้มเดียวกันในการที่จะแก้ไขแฟ้มนี้ โปรแกรมจะต้องอ่านแฟ้มข้อมูลไปเก็บไว้ในหน่วยความจำแล้วจึงทำการแก้ไขข้อมูลในหน่วยความจำหลังจากนั้นจึงคัดลอกข้อมูลในหน่วยความจำนี้กลับเข้าสู่แฟ้ม race condition จะเกิดขึ้นเมื่อ program A อ่านแฟ้มเข้าสู่หน่วยความจำแล้วทำการแก้ไขแต่ก่อนที่โปรแกรม A จะเขียนกลับลงสู่แฟ้ม program B ได้ทำการอ่าน แก้ไข และเขียนลงสู่แฟ้มก่อน หลังจากนั้นโปรแกรม A ทำการเขียนทับกลับไปในแฟ้มจะเห็นได้ว่าการแก้ไขแฟ้มทั้งหมดของ program B จะหายไปแต่ยังไงก็แล้วแต่ race condition ค่อนข้างจะเกิดขึ้นได้ยากเนื่องจากผู้บุกรุกต้องทำการทดลองเป็นพันๆครั้งจึงจะสามารถเจาะเข้าสู่ระบบได้

**ประตูกล (Black door)**

ประตูกลเป็นข้อบกพร่องที่เกิดขึ้นโดยผู้ที่พัฒนาโปรแกรมในระบบทำขึ้นเพื่อใช้เป็นช่องทางในการเข้าไปทำงานในการติดตามและทดสอบโปรแกรม และไม่ได้ปิดประตูนี้เมื่อใช้งานเสร็จ นอกจากนี้ยังมีข้อบกพร่องที่เกิดจากผู้ดูแลระบบในเรื่องของการกำหนดค่า system configuration โดยไม่มีการกำหนดรหัสผ่าน ซึ่งเป็นช่องทางหนึ่งที่ทำให้ผู้บุกรุกเข้าสู่ระบบได้

### 2.3.2 ข้อบกพร่องในการตั้งรหัสผ่านของผู้ใช้

ข้อบกพร่องนี้เกิดจากการที่ผู้ใช้ในระบบละเลยความสำคัญของการตั้งรหัสผ่านโดยเลือกใช้รหัสผ่านที่เป็นชื่อของตนเอง สมาชิกในครอบครัว หรือคำที่มีความหมายสามารถเดาได้ง่าย ผู้บุกรุกสามารถใช้วิธีการเปรียบเทียบรหัสผ่านของผู้ใช้กับคำที่มีอยู่ในดิกชันนารี วิธีการนี้ทำให้ผู้บุกรุกได้มาซึ่งบัญชีผู้ใช้และรหัสผ่านของระบบเพื่อเข้าใช้ระบบได้การป้องกันไม่ให้เกิดพฤติกรรมเหล่านี้เป็นการป้องกันวิธีหนึ่งแต่ถึงกระนั้นจะอาศัยเพียงการป้องกันอย่างเดียวไม่เพียงพอ

การตรวจจับ (detection) พฤติกรรมเหล่านั้นจึงเข้ามามีบทบาทสำคัญในเรื่องของการรักษาความปลอดภัย ในหัวข้อถัดไปจะกล่าวถึงเรื่องของลักษณะการบุกรุกและผลกระทบที่เกิดขึ้น

## 2.4 ลักษณะการบุกรุกและผลกระทบ

ในหัวข้อนี้จะกล่าวถึงลักษณะการบุกรุกรวมถึงผลกระทบที่ตามมาเมื่อผู้บุกรุกสามารถบุกรุกระบบได้สำเร็จ การบุกรุกที่เกิดขึ้นโดยส่วนมากมักจะเกิดจากการบุกรุกที่อาศัยจุดอ่อนหรือข้อ บกพร่องของโปรแกรมที่มีอยู่ในระบบ เนื่องผู้บุกรุกเองสามารถหาข้อมูลเหล่านี้ได้จากข้อมูลที่เผยแพร่อยู่บนอินเทอร์เน็ต ข้อบกพร่องของโปรแกรมซึ่งถือว่าเป็นข้อบกพร่องที่ผู้บุกรุกใช้เป็นช่องทางในการบุกรุกระบบมากอย่างหนึ่งคือปัญหาการล้นของบัฟเฟอร์ได้มีการอ้างอิงว่าการล้นของบัฟเฟอร์เป็นข้อ บกพร่องที่ก่อให้เกิดปัญหาในเรื่องของความปลอดภัยมากที่สุด [12] เนื่องจากการโจมตีประเภทนี้เป็นวิธีหนึ่งที่จะช่วยทำให้ผู้บุกรุกสามารถควบคุมการใช้งานทรัพยากรต่าง ๆ ของระบบที่ถูกโจมตีนั้นได้หากผู้บุกรุกกระทำการโจมตีได้สำเร็จ ตารางที่ 2.1 – 2.3 เป็นข้อมูลของผลกระทบเมื่อผู้บุกรุกสามารถบุกรุกระบบได้สำเร็จโดยเน้นเฉพาะผลกระทบที่เกิดกับระบบปฏิบัติการยูนิกซ์เท่านั้น การแบ่งกลุ่มของผลกระทบจะแบ่งเป็นกลุ่มใหญ่ 3 กลุ่มคือผลกระทบที่เกิดขึ้นเมื่อผู้บุกรุกได้มาซึ่งบัญชีผู้ใช้ในระบบ ผลกระทบที่เกิดขึ้นเมื่อผู้บุกรุกสามารถเข้าถึงแฟ้มในระบบ และกลุ่มสุดท้ายเป็นผลกระทบอื่น ๆ

ตารางที่ 2.1 ผลกระทบเมื่อผู้บุกรุกได้มาซึ่งบัญชีผู้ใช้ในระบบ

| ผู้บุกรุกได้มาซึ่งบัญชีผู้ใช้ในระบบ | |
|---|---|
| ผลกระทบ | จุดอ่อนในระบบ |
| 1. ผู้บุกรุกสามารถใช้งานคำสั่งด้วยสิทธิของผู้ใช้สูงสุด | - Integer overflow in Sun RPC XDR library routines<br>- Buffer Overflow in CDE ToolTalk<br>- Integer Overflow In XDR Library<br>- Exploitation of Vulnerability in CDE Subprocess Control Service<br>- Buffer Overflow in CDE Subprocess Control Service<br>- Two Input Validation Problems In FTPD<br>- Remotely Exploitable Buffer Overflow Vulnerability in mountd talkd Vulnerability<br>- Four Vulnerabilities in the Common Desktop Environment<br>- Vulnerability in suidperl(sperl)<br>- Multiple vulnerabilities in Mozilla products<br>- A Exploitation of phpBB highlight parameter vulnerability<br>- Double Free Bug in zlib Compression Library<br>- Multiple Vulnerabilities in PHP fileupload<br>- Multiple Vulnerabilities in ISC DHCP 3<br>- Buffer Overflow in Sendmail<br>- Buffer Management vulnerability in OpenSSH<br>- Multiple Vulnerabilities in Snort Preprocess VU#139129 and VU#16785<br>- Buffer Overflow in ISC DHCPD Minires Library<br>- Multiple Vulnerabilities in SSH Implementations<br>- Buffer Overflow in Solaris X Window Font Service<br>- Multiple Vulnerabilities in BIND(VU#852283)<br>- Apache/mod_ssl Worm |
| 2. ผู้บุกรุกเรียกใช้งานโปรแกรมด้วยสิทธิของเจ้าของโปรแกรมที่เป็นจุดอ่อน ซึ่งโดยทั่วไปจะเป็นผู้ใช้ที่มีสิทธิสูงสุด หรือผู้ใช้ที่มีสิทธิพิเศษ | |

ตารางที่ 2.1 (ต่อ)

| ผู้บุกรุกได้มาซึ่งบัญชีผู้ใช้ในระบบ | |
|---|---|
| ผลกระทบ | จุดอ่อนในระบบ |
| | - Multiple Vulnerabilities In OpenSSL<br>- Vulnerability in PHP<br>- Buffer Overflows in Multiple DNS Resolver Libraries<br>- OpenSSH Vulnerabilities in Challenge Response Handling<br>- Input Validation Problem in rpc.stated<br>- Format String Vulnerability in ISC DHCPD<br>- Heap Overflow in Cachefs Daemon (cachefsd)<br>- Format String Vulnerability in rpc.rwalld<br>- Recent Activity Against Secure Shell Daemons<br>- Multiple Vulnerabilities in WU-FTPD<br>- Multiple Vulnerabilities in lpd Vu#39001<br>- Oracle9iAS Web Cache vulnerable to buffer overflow<br>- Buffer Overflow in Gauntlet Firewall allows intruders to execute arbitrary code<br>- Format String Vulnerability in CDE ToolTalk<br>- Buffer Overflow Vulnerability in Calendar Manager Service Daemon, rpc.cmsd<br>- Vulnerability in statd exposes vulnerability in automountd<br>- Vulnerability in xlock<br>- Vulnerability in Nature Language Service<br>- SATAN Vulnerability: Password Disclosure<br>- Buffer Overflow in Sun Solstice AdminSuite Daemon sadmind<br>- Buffer Overflows in SSH daemon and RSAREF2 Library |

ตารางที่ 2.1 (ต่อ)

| ผู้บุกรุกได้มาซึ่งบัญชีผู้ใช้ในระบบ | |
|---|---|
| ผลกระทบ | จุดอ่อนในระบบ |
| 2. ผู้บุกรุกได้มาซึ่งสิทธิของเจ้าของโปรแกรมหรือคำสั่งที่เป็นจุดอ่อน สิทธิของผู้ใช้ที่มีสิทธิพิเศษ สิทธิอนุญาตในการใช้งานของกลุ่มผู้ใช้อื่น รวมทั้งสิทธิของผู้ใช้ที่มีสิทธิสูงสุด | - Vulnerability in webdist.cgi<br>- Buffer Overflow In Sun Solaris in.lpd Print Daemon<br>- Remote Buffer Overflow in Sendmail<br>- Buffer Overflow in Some Implementations of IMAP Servers<br>- Buffer Overrun Vulnerability in Count.cgi cgi-bin Program<br>- Buffer Overflow in Kerberos Administration Daemon<br>- Recent Activity Against Secure Shell Daemons<br>VU#157447<br>HP-UX newgrp Buffer Overrun Vulnerability<br>- MIME Conversion Buffer Overflow in Sendmail Versions 8.8.3 and 8.8.4<br>- HP-UX newgrp Buffer Overrun Vulnerability<br>- Sendmail Group Permissions Vulnerability<br>- Sendmail Daemon Mode Vulnerability<br>- Sendmail Vulnerabilities<br>- Vulnerability in expreserve<br>- Vulnerability in rdist<br>- Vulnerability in the dip program<br>- Vulnerability in suidperl<br>- NIS+ Configuration Vulnerability<br>wu-ftpd Misconfiguration Vulnerability<br>- SGI lp Vulnerability<br>- Telnetd Environment Vulnerability<br>- Sun 4.1.X Loadmodule Vulnerability<br>- Sun Sendmail Vulnerability |

ตารางที่ 2.1 (ต่อ)

| ผู้บุกรุกได้มาซึ่งบัญชีผู้ใช้ในระบบ | |
|---|---|
| ผลกระทบ | จุดอ่อนในระบบ |
| 3. สร้างความเสียหายแก่ระบบ หรือได้รับสิทธิของผู้ใช้ที่มีสิทธิสูงสุดvulnerability can crash the server, or be leveraged to gain root access<br>ผู้บุกรุกมีสิทธิเท่ากับเจ้าของคำสั่งหรือโปรแกรมที่เป็นจุดอ่อนทำให้สามารถติดตั้งและเรียกใช้โปรแกมได้ตามสิทธิที่ได้รับ | - Solaris ps Vulnerab<br>- Spoofing Attacks and Hijacked Terminal Connections<br>- NFS Vulnerability<br>- IBM AIX bsh Vulnerability<br>- ftpd Vulnerabilities<br>- Writable /etc/utmp Vulnerability<br>- SunOS rdist Vulnerability<br>- Solaris System Startup Vulnerability<br>- xterm Loggong Vulnerability<br>- SunOS/Solbourne loadmodule and modload Vulnerability<br>- UMN UNIX gopher and gopher+ Vulnerabilities<br>- SunOS Expreserve Vulnerability -<br>- SunOS File/Directory Permissions<br>- AIX crontab Vulnerability<br>- AIX Anonymous FTP Vulnerability<br>- AIX /bin/passwd Vulnerability<br>- AIX uucp Vulnerability<br>- AT&T System V Release 4 /bin/login Vulnerability<br>- DEC Ultrix Vulnerability<br>- SunOS in.telnetd Vulnerability<br>- Trojan Horse in IRC Client for UNIX<br>- Buffer Overflow in telnetd<br>- Syslog Vulnerability - A Workaround for Sendmail<br>- Majordomo Vulnerabilities |

ตารางที่ 2.2 ผลกระทบที่เกิดขึ้นเมื่อผู้บุกรุกเข้าถึงแฟ้มข้อมูล

| ผู้บุกรุกเข้าถึงแฟ้มข้อมูลที่อยู่ในระบบ | |
|---|---|
| ผลกระทบ | จุดอ่อนในระบบ |
| 1. ผู้บุกรุกเปิดเผยข้อมูลที่สำคัญ<br><br>2. ผู้บุกรุกสามารถ อ่านหรือเขียนแฟ้มด้วยสิทธิของผู้ใช้สูงสุด หรือ อ่าน แก้ไข ลบแฟ้มตามสิทธิของเจ้าของคำสั่งหรือโปรแกรมที่เป็นจุดอ่อนของระบบ<br><br>ผู้บุกรุกสามารถสร้างแฟ้ม บัญชีผู้ใช้หรือทำลายแฟ้มบันทึกข้อมูลการบุกรุก ผู้บุกรุกสามารถสร้าง เขียน ลบแฟ้ม ในบางกรณีสามารถทำให้ได้สิทธิของผู้ใช้สูงสุดซึ่งสามารถ สร้างหรือแก้ไขแฟ้มซึ่งผู้ใช้สูงสุดเป็นเจ้าของ รวมทั้งสามารถอนุญาตให้ผู้ใช้ที่ไม่มีสิทธิอ่านและเขียนแฟ้มระบบได้ | - Integer overflow in Sun RPC XDR library routines<br>- Integer Overflow In XDR Library<br>- Multiple Vulnerabilities in Oracle Servers<br>- Double-Free Bug in CVS Server<br>- ftpd Signal Handling Vulnerability<br>- Multiple Vulnerabilities in Oracle Servers<br>- Oracle9iAS Web Cache vulnerable to buffer overflow<br>- Oracle9iAS Web Cache vulnerable to buffer overflow<br>- Continuing Compromises of DNS Servers<br>- Sun Sendmail Vulnerability<br>- Commodore Amiga UNIX finger Vulnerability<br>- Vulnerability in the httpd nph-test-cgi script<br>- Ghostscript Vulnerability<br>- Vulnerability in WorkMan<br>- Multiple Vulnerabilities in CDE ToolTalk<br>- Multiple Vulnerabilities in Oracle Servers<br>- Double Free Bug in zlib Compression Library<br>- Multiple Vulnerabilities in PHP fileupload<br>- Vulnerability in SGI login LOCKOUT<br>- Vulnerability in rpc.statd<br>- Vulnerability in IRIX csetup<br>- Multi-platform Unix FLEXlm Vulnerabilities<br>- Vulnerability in Solaris 2.5 KCMS programs<br>- Vulnerability in Solaris admintool<br>- Vulnerability in Solaris vold |

ตารางที่ 2.2 (ต่อ)

| ผู้บุกรุกเข้าถึงแฟ้มข้อมูลที่อยู่ในระบบ | |
|---|---|
| ผลกระทบ | จุดอ่อนในระบบ |
| | - Vulnerability in fm_fls<br>- Vulnerability in expreserve<br>- Vulnerability in IRIX csetup<br>- Sendmail v.5 Vulnerability<br>- Vulnerabilities in /bin/mail<br>- Revised Patch for SunOS /usr/etc/rpc.mountd Vulnerability |

ตารางที่ 2.3 ผลกระทบอื่น ๆ ที่เกิดขึ้นเมื่อผู้บุกรุกบรุกระบบได้สำเร็จ

| ผลกระทบอื่น ๆ | |
|---|---|
| ผลกระทบ | จุดอ่อนในระบบ |
| ผู้บุกรุกสามารถเรียกใช้งานในระบบได้เทียบเท่าผู้ใช้ที่มีสิทธิสูงสุด | - Vulnerability in the at(1) program<br>- Vulnerability in suidperl(sperl) |
| ผู้บุกรุกสามารถติดตั้งโปรแกรมประเภท ม้าโทรจัน หรือโปรแกรมที่สามารถสร้างความเสียหายแก่ระบบ รวมถึงผู้บุกรุกสามารถเพิ่มโค้ดในแฟ้มระบบ เพื่อให้ผู้ที่ไม่มีสิทธิในระบบสามารถเข้าสู่ระบบบได้ | - CVS Heap Overflow Vulnerability<br>- wuarchive ftpd Trojan Horse<br>- Continuing Compromises of DNS servers<br>- GNU Project FTP Server Compromise<br>- Multiple Vulnerabilities in Oracle Servers<br>- Double-Free Bug in CVS Server<br>- Multiple Vulnerabilities in SSH Implementations<br>- Buffer Overflow in Solaris X Window Font Service<br>- Buffer Overflow in CDE ToolTalk<br>- Multiple Vulnerabilities In OpenSSL |

ตารางที่ 2.3 (ต่อ)

| ผลกระทบอื่น ๆ | |
|---|---|
| ผลกระทบ | จุดอ่อนในระบบ |
| | - Integer Overflow In XDR Library<br>- Buffer Overflows in Multiple DNS Resolver Libraries<br>- OpenSSH Vulnerabilities in Challenge Response Handling<br>- Input Validation Problem in rpc.stated<br>- Multiple Vulnerabilities in Oracle Servers<br>- Double Free Bug in zlib Compression Library<br>- Buffer Overflow in System V Derived Login<br>- HP-UX Line Printer Daemon Vulnerable to Directory Traversal<br>- Multiple Vulnerabilities in lpd<br>- Buffer Overflow in telnetdCA-2001-05 Exploitation of snmpXdmid<br>- Multiple Buffer Overflows in Kerberos Authenticated Services<br>- FTP Buffer Overflows<br>- Buffer overflows in some POP servers<br>- Buffer Overflow in NIS+ |

จากตารางที่ 2.1 – 2.3 แสดงถึงผลกระทบที่เกิดขึ้นเมื่อผู้บุกรุกสามารถบุกรรุกระบบได้สำเร็จโดยใช้จุดอ่อนที่มีอยู่ในระบบ ตารางที่ 2.1 แสดงผลกระทบเมื่อผู้บุกรุกได้มาซึ่งบัญชีผู้ใช้ในระบบ ตาราง 2.2 แสดงถึงผลกระทบเมื่อผู้บุกรุกเข้าถึงแฟ้มในระบบและในตารางที่ 2.3 แสดงผลกระทบอื่น ๆ เราสามารถแยกวิเคราะห์การบุกรุกดังกล่าวได้ดังนี้

ในกรณีที่ผู้บุกรุกได้มาซึ่งบัญชีผู้ใช้ในระบบนั้นถ้าหากสิทธิที่ได้มาเป็นสิทธิของผู้ใช้ที่มีสิทธิสูงสุดจะทำให้ผู้บุกรุกมีสิทธิที่จะควบคุมทรัพยากรทั้งหมดในระบบ หรือในกรณีที่ผู้บุกรุกได้สิทธิเทียบเท่าเจ้าของโปรแกรมหรือคำสั่งที่เป็นจุดอ่อน ผู้บุกรุกจะมีสิทธิในการใช้ทรัพยากรในระบบได้เทียบเท่ากับเจ้าของโปรแกรมเหล่านั้นซึ่งในกรณีนี้สามารถนำไปสู่การได้มาซึ่งสิทธิเทียบเท่าผู้ใช้ที่มีสิทธิสูงสุดได้เช่นเดียวกันเนื่องจากโปรแกรมหรือคำสั่งส่วนใหญ่ที่เป็นจุดอ่อนหรือข้อบกพร่องเป็นโปรแกรมประเภท setuid

ในกรณีที่ผู้บุกรุกสามารถเข้าถึงแฟ้มได้นั้นผลกระทบที่เกิดขึ้นได้แก่ การเผยแพร่หรือเปิดเผยข้อมูลที่สำคัญ เป็นความลับ การเพิ่ม ลบ แก้ไขแฟ้มข้อมูลที่อยู่ในระบบ หรือการสร้างแฟ้มในระบบ ในขณะที่บางกรณีสามารถนำไปสู่การสร้างบัญชีรายชื่อผู้ใช้ใหม่หรือแม้กระทั่งสามารถทำการอนุญาตให้ผู้ใช้ที่ไม่มีสิทธิในการเข้าใช้งานในระบบสามารถเข้ามาใช้งานได้

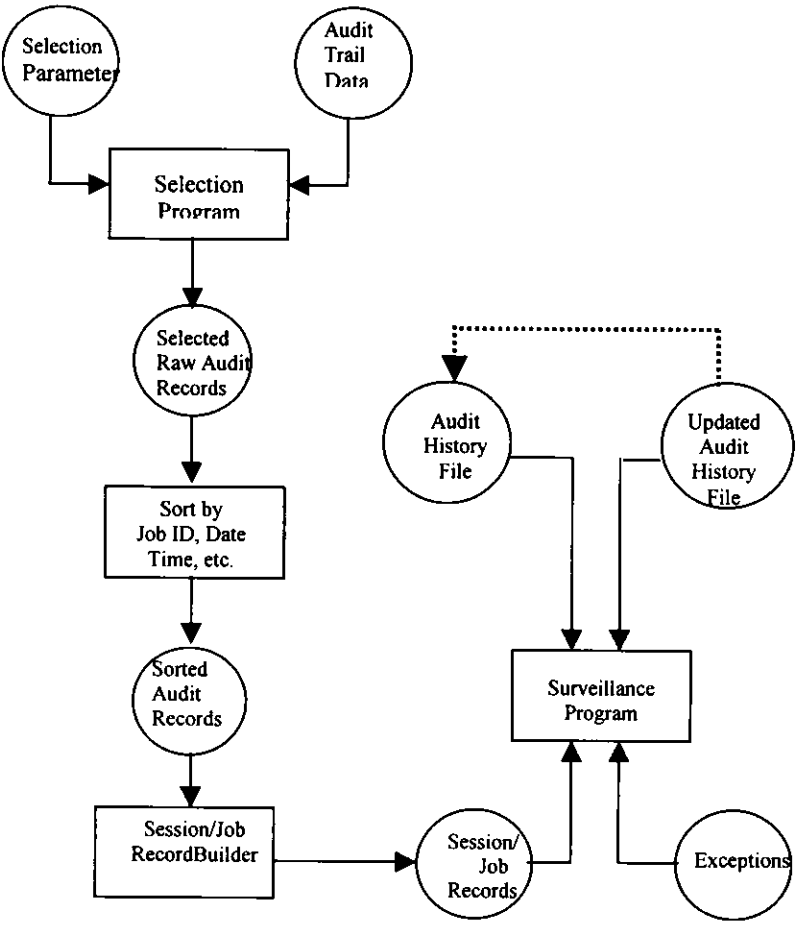
ในส่วนของผลกระทบอื่น ๆ ผู้บุกรุกสามารถใช้จุดอ่อนของโปรแกรมหรือคำสั่งเหล่านั้นในการติดตั้งโปรแกรมประเภทม้าโทรจัน หรือทำการแก้ไขโปรแกรมในระบบเพื่อให้ทำงานตามที่ผู้บุกรุกต้องการหรือแม้กระทั่งทำให้บริการที่เป็นข้อบกพร่องหรือระบบนั้นไม่สามารถให้บริการหรือทำงานได้ ในหัวข้อถัดไปจะกล่าวถึงระบบตรวจจับการบุกรุก

## 2.5 ระบบตรวจจับการบุกรุก (Intrusion Detection System : IDS)

ระบบตรวจจับการบุกรุก [13] คือ ระบบที่ประกอบด้วยฮาร์ดแวร์หรือซอฟต์แวร์สำหรับทำงานในกระบวนการตรวจสอบเหตุการณ์ต่าง ๆ ที่เกิดขึ้นในระบบคอมพิวเตอร์และเครือข่ายเพื่อวิเคราะห์หาร่องรอยของการบุกรุกโดยอัตโนมัติ

หลักการทำงานประการสำคัญของระบบตรวจจับการบุกรุกส่วนใหญ่ คือการตรวจจับพฤติกรรมที่ "ผิดปกติ" นั่นคือพฤติกรรมที่ต่างจากพฤติกรรมที่ควรจะเป็นของเหตุการณ์นั้น Anderson [14] ได้นำเสนอความคิดพื้นฐานไว้ว่าพฤติกรรมของผู้ใช้โดยปกติจะมีรูปแบบหรือกิจกรรมซ้ำ ๆ กันและคล้ายเดียงกันเสมอในการใช้ระบบแต่ละครั้ง และเมื่อพฤติกรรมของผู้ใช้แต่ละคนต่างไปจากเดิมมาก อาจจะสรุปได้ว่ามีความผิดปกติเกิดขึ้น ดังนั้นถ้าสามารถทำการบันทึกพฤติกรรมการใช้งานของผู้ใช้แต่ละคนไว้ได้ ก็อาจจะสามารถตรวจหาพฤติกรรมผิดปกติได้โดยการเปรียบเทียบพฤติกรรมต่าง ๆ เหล่านี้ และใน ค.ศ. 1980 Anderson ได้นำเสนอหลักการและรูปแบบ

การสร้างระบบเฝ้าติดตาม หรือตรวจสอบพฤติกรรมของผู้ใช้เพื่อตรวจหาการบุกรุกระบบคอมพิวเตอร์ จากแนวความคิดนี้เองได้นำมาสู่หลักการในการพัฒนา IDS ยุคแรก ๆ อย่างเช่น Intrusion Detection Expert System (IDES) [15] ซึ่งพัฒนาขึ้นที่ The Computer Science Laboratory of SRI International และ Haystack [4] ซึ่งพัฒนาสำหรับ Air Force Computer System




ภาพประกอบที่ 2.1 ระบบการเฝ้าติดตาม เสนอโดย Anderson

ที่มา: Anderson [14]

หลักเกณฑ์ที่นำเสนอโดย Anderson และแสดงโดยภาพประกอบที่ 2.1 ซึ่ง Anderson กล่าวว่า สิ่งที่จำเป็นในการวิเคราะห์ในรูปแบบนี้คือการเลือกเฉพาะสิ่งที่สำคัญ และตัวแปรที่เกี่ยวข้องกับการตรวจจับการบุกรุกจาก audit trail ซึ่งเป็นบันทึกพฤติกรรมหรือการทำงานตามลำดับเวลาของการใช้ทรัพยากรของระบบในระบบการรักษาความปลอดภัยคอมพิวเตอร์ซึ่งรวมถึงการล็อคอิน

เข้าสู่ระบบของผู้ใช้ การเข้าถึงแฟ้มกิจกรรมอื่น ๆ และการตรวจดูว่าได้มีการพยายามที่จะล่วงล้ำการรักษาความปลอดภัยหรือไม่

เนื่องจากข้อจำกัดในเรื่องของการจัดเก็บข้อมูลเกี่ยวกับพฤติกรรมการใช้งานของผู้ใช้และเพื่อลดจำนวนของข้อมูลซึ่งจะต้องนำมาวิเคราะห์โดยผู้ที่ทำหน้าที่ดูแลรักษาความปลอดภัย ดังนั้นการเลือกตัวแปรและการเลือกโปรแกรมจึงถูกรวมไว้ในโมเดลนี้ด้วย อย่างไรก็ตามหลักการตามรูปแบบที่ Anderson เสนอได้กลายมาเป็นพื้นฐานในการวิเคราะห์ audit trail และการสร้าง IDS ในเวลาต่อมา

## 2.6 ประเภทของระบบตรวจจับการบุกรุก

ประเภทของระบบตรวจจับการบุกรุกสามารถจำแนกได้แตกต่างกันออกไป ตามสิ่งที่นำมาพิจารณายกตัวอย่างเช่น การจำแนกระบบตรวจจับการบุกรุกตามวิธีการวิเคราะห์การบุกรุก การจำแนกระบบตรวจจับการบุกรุกโดยพิจารณาจากเทคนิคการออกแบบและเครื่องมือที่ใช้ และการจำแนกการระบบตรวจจับการบุกรุกโดยพิจารณาจากลักษณะของข้อมูลที่นำมาวิเคราะห์ ซึ่งสามารถแยกอธิบายได้ดังนี้

### 2.6.1 การจำแนกระบบตรวจจับการบุกรุกตามวิธีการวิเคราะห์การบุกรุก

การจำแนกระบบตรวจจับการบุกรุกตามวิธีการวิเคราะห์การบุกรุกสามารถจัดแบ่งออกได้เป็นสองประเภท [16] คือ Misuse Detection System และ Anomaly Detection System โดยที่

**Misuse Detection System**

ระบบตรวจจับการบุกรุกประเภทนี้จะวิเคราะห์พฤติกรรมการบุกรุกโดยจะทำการเปรียบเทียบพฤติกรรมของผู้ใช้ในขณะนั้นกับพฤติกรรมที่กำหนดไว้ หากพฤติกรรมการใช้งานผิดจากพฤติกรรมที่กำหนดไว้จะถือว่าเป็นการบุกรุก โดยทั่วไปแล้วการตรวจจับการบุกรุกแบบนี้จะใช้กับการบุกรุกที่เกิดจากภายในระบบเองอย่างเช่นผู้ใช้ที่มีสิทธิใช้งานระบบแต่ใช้สิทธิที่มีอยู่ไปในทางที่ผิด อาทิเช่นผู้ใช้ที่มีสิทธิใช้งานทั่วไปพยายามจะแก้ไขแฟ้มรหัสผ่านซึ่งอนุญาตให้ผู้ใช้ที่มี

สิทธิสูงสุดเท่านั้น พฤติกรรมเช่นนี้จะถือว่าเป็นพฤติกรรมที่ผิดจากสิทธิที่กำหนดให้และถือเป็นการบุกรุก ตัวอย่างการตรวจจับการบุกรุกที่ใช้เทคนิค misuse detection นี้ได้แก่ [17] - [19] ซึ่งเป็นการตรวจจับพฤติกรรมที่ใช้หลักการเปรียบเทียบรูปแบบหรือพฤติกรรมของผู้บุกรุกว่าเหมือนกันกับพฤติกรรมที่เป็นการบุกรุกหรือไม่ (pattern-matching) [20] - [22] เป็นลักษณะการตรวจจับพฤติกรรมที่ถือว่าเป็นการบุกรุกจากข้อมูลที่บันทึกการใช้งานในระบบ (audit data) ตัวอย่างงานวิจัยที่เกี่ยวข้องอื่น ๆ ได้แก่ [23] เป็นการตรวจจับการบุกรุกที่เน้นในเรื่องของการตรวจจับไวรัส [24] - [25] ตรวจจับพฤติกรรมที่ผิดปกติโดยเน้นในส่วนของการโจมตีแบบ DoS

**Anomaly Detection System**

ระบบตรวจจับการบุกรุกประเภทนี้เป็นวิธีการตรวจจับที่ตรวจสอบพฤติกรรมการใช้ทรัพยากรในระบบของผู้ใช้ว่าผิดไปจากพฤติกรรมปกติของผู้ใช้นั้นหรือไม่ อย่างเช่น ช่วงเวลาการใช้งานของผู้ใช้ผิดไปจากเดิมหรือไม่ หรือการใช้เนื้อที่ในหน่วยความจำ วิธีการตัดสินว่าการทำงานเป็นพฤติกรรมการบุกรุกหรือไม่อาจใช้หลักการทางสถิติเข้ามาช่วย เช่นผู้ใช้หรือโปรแกรมที่ถูกตรวจสอบมีการใช้เนื้อที่ในหน่วยความจำมากผิดปกติ ระบบการตรวจจับแบบนี้ค่อนข้างซับซ้อนเนื่องจากจะต้องแยกให้ชัดเจนระหว่างสิ่งที่เรียกว่า "ปกติ" และ "ไม่ปกติ" ซึ่งโดยทั่วไปแล้วถ้าเกิดกลุ่มพฤติกรรมที่ไม่แน่ใจจะถูกจัดเป็นพฤติกรรมในกลุ่มของพฤติกรรมที่ไม่ปกติหรือพฤติกรรมการบุกรุก นอกจากนี้ข้อมูลที่ถูกนำมาพิจารณาเปรียบเทียบในบางเทคนิคของวิธีการตรวจจับแบบ anomaly detection จะมีขนาดใหญ่เนื่องจากต้องเก็บข้อมูลการใช้งานของผู้ใช้แต่ละคนเพื่อใช้เปรียบเทียบในการใช้งานแต่ละครั้ง นอกจากนี้ในบางครั้งการเปรียบเทียบพฤติกรรมผู้ใช้กับข้อมูลในอดีตอาจก่อให้เกิดการตัดสินใจที่ผิดพลาดได้ เนื่องจากผู้ใช้อาจเปลี่ยนแปลงพฤติกรรมการใช้งานได้เช่นกัน ตัวอย่างระบบตรวจจับการบุกรุกที่ใช้เทคนิค anomaly detection ได้แก่ การตรวจจับซึ่งรวบรวมพฤติกรรมโดยดูจากลำดับของการเรียกใช้ system call เพื่อนำมาจำลองการทำงานของโปรแกรม รูปแบบนี้ถูกเสนอครั้งแรกโดย Forrest et al [26] งานวิจัยอื่น ๆ ที่เกี่ยวข้องได้แก่ [27] เป็นการตรวจจับที่มีการนำเอาเรื่องของ FSA (Finite State Automata) มาใช้ในการติดตามการเรียกใช้งานของ system call นอกจากนี้ยังมีการนำเอาเรื่องของ data-mining มาใช้ในการวิเคราะห์ลำดับการทำงานของ system call [28] ด้วย

**การจำแนกระบบตรวจจับการบุกรุกโดยพิจารณาจากเทคนิคการออกแบบและเครื่องมือที่ใช้**

การจำแนกระบบตรวจจับการบุกรุกโดยพิจารณาจากเทคนิคการออกแบบแบ่งได้เป็น ระบบผู้เชี่ยวชาญการตรวจจับการบุกรุก (Intrusion Detection Expert System: IDES) ระบบเปรียบเทียบกฎแบบปรับเปลี่ยนได้ (Rule-based Intrusion Detection System : RBID)

IDES [5], [29], [30] เป็นระบบผู้เชี่ยวชาญการตรวจจับการบุกรุกที่มีการสร้างแฟ้มข้อมูลพฤติกรรมของผู้ใช้เก็บไว้ก่อน เมื่อมีการใช้งานระบบก็จะมีการสร้างแฟ้มพฤติกรรมขึ้นใหม่จากต้นฉบับเดิมเมื่อข้อมูลเหล่านี้ผ่านการประมวลผลแล้วระบบตรวจจับการบุกรุกจะได้ค่าตัวเลขซึ่งใช้เป็นเกณฑ์ในการวัดความผิดปกติของแฟ้มข้อมูลพฤติกรรม ในการตัดสินพฤติกรรมที่ผิดปกติจะตัดสินจากการเปรียบเทียบแฟ้มพฤติกรรมปัจจุบันกับแฟ้มพฤติกรรมที่เก็บไว้ เนื่องจากการตรวจจับการบุกรุกวิธีนี้ใช้หลักสถิติและการสร้างแฟ้มพฤติกรรมในการตรวจจับจึงไม่สามารถเปลี่ยนแปลงกฎได้โดยอัตโนมัติ ทำให้ไม่สามารถวิเคราะห์การบุกรุกรูปแบบใหม่ ๆ ได้

RBID เป็นระบบเปรียบเทียบกฎแบบปรับเปลี่ยนได้เป็นวิธีการที่นำมาใช้เพื่อแก้ไขข้อจำกัดของระบบผู้เชี่ยวชาญซึ่งไม่สามารถเปลี่ยนแปลงกฎได้โดยอัตโนมัติ และไม่สามารถวิเคราะห์การบุกรุกรูปแบบใหม่ๆ ได้เพราะผู้ดูแลระบบต้องเพิ่มเอง วิธีการทำงานของระบบนี้คือการใช้โครงสร้างของข้อมูลในอดีตนำมาสร้างกฎโดยอัตโนมัติ

### 2.6.3 การจำแนกระบบตรวจจับการบุกรุกโดยพิจารณาจากลักษณะของข้อมูลที่นำมาวิเคราะห์

การจำแนกระบบตรวจจับการบุกรุกโดยการพิจารณาจากลักษณะของข้อมูลที่นำมาวิเคราะห์ สามารถจำแนกได้เป็นสองประเภทคือ ระบบตรวจจับการบุกรุกเฉพาะโฮสต์ (Host-based Intrusion Detection System: HIDS) และระบบตรวจจับการบุกรุกทุกสถานีงานในเครือข่าย (Network-based Intrusion Detection System: NIDS)

HIDS [31] – [32] เป็นระบบตรวจจับการบุกรุกที่วิเคราะห์ข้อมูลที่ถูกส่งเข้ามายังโฮสต์นั้น ๆ อาจเป็นแพคเก็ตที่ส่งเข้ามาเพื่อทำลายระบบโดยตรง หรืออาจวิเคราะห์จากแฟ้มบันทึกการทำงานของระบบปฏิบัติการนั้น การออกแบบในลักษณะนี้มักเป็นที่นิยมเนื่องจากง่ายต่อการ

พัฒนาและยังสามารถตรวจจับการบุกรุกได้อย่างรวดเร็ว งานวิจัยที่เกี่ยวข้องอย่างเช่น [32] เป็นการตรวจจับการบุกรุกที่ใช้วิธีการวิเคราะห์แบบ anomaly detection โดยวิเคราะห์จากความถี่ในการใช้คำสั่งของผู้ใช้ ตัวอย่างระบบตรวจจับประเภท HIDS ได้แก่ Computer Watch, Discovery, HAYSTACK, IDES, MIDAS (Multics Intrusion Detection Alerting System)

NIDS ระบบตรวจจับการบุกรุกแบบนี้จะรับข้อมูลจากทุกเครื่องในระบบเครือข่ายแล้ววิเคราะห์เพื่อมองหาความเป็นไปได้ที่อาจจะมีผู้บุกรุกเข้ามาทำลายในสถานีงานใด ๆ ใน เครือข่าย ระบบนี้อาจจะอาศัยการวิเคราะห์แพ็กเกตที่ส่งไปยังสถานีงานใดๆ ปริมาณแพ็กเกตที่ได้รับมานี้จะมีเป็นจำนวนมากดังนั้นระบบจึงมีความจำเป็นที่จะต้องใช้เทคนิคในการแบ่งข้อมูลเพื่อให้การทำงานมีความรวดเร็วมากขึ้น หากตรวจพบความผิดปกติจะต้องรายงานให้ผู้ดูแลระบบทราบ นอกจากนี้อาจจะใช้ร่วมกับระบบไฟล์วอลล์ โดยระบบตรวจสอบการบุกรุกส่งข้อมูลหรือสัญญาณไปยังระบบไฟล์วอลล์เพื่อให้ทำการปิดการเชื่อมต่อของโฮสต์ที่น่าสงสัยนั้นได้ทันที ด้วยรูปแบบการทำงานลักษณะนี้จะช่วยให้ผู้ดูแลระบบสามารถติดตามการทำงานในระบบเครือข่ายได้ตลอดเวลาโดยไม่จำเป็นต้องติดตั้งระบบตรวจจับการบุกรุกไว้ทุกสถานีงานเหมือนแบบแรก ตัวอย่างเครื่องมือที่ใช้การตรวจจับการบุกรุกแบบ NIDS ได้แก่

1. NADIR (Network Anomaly Detection and Intrusion Reporter) [33] ทำงานโดยการเปรียบเทียบการใช้งานบนเครือข่ายของผู้ใช้ในหนึ่งสัปดาห์ โดยการพิจารณาจากกฎที่ได้กำหนดไว้ก่อนหน้าในระบบผู้เชี่ยวชาญ
2. NSM (Network Security Monitor) [30] วิเคราะห์แพกเก็ตในเครือข่าย โดยพิจารณารูปแบบการใช้งานเครือข่ายตามชุดโปรโตคอลทีซีพี/ไอพี
3. DDIS (Distributed Intrusion Detection System) [29] มีสถาปัตยกรรมแบบกระจายซึ่งระบบตรวจจับแต่ละระบบจะเชื่อมต่อกันโดยผ่านเครือข่ายมีส่วนประกอบที่สำคัญสามส่วนคือ DDIS Director, Lan Monitor, Host Agent

อย่างไรก็ตามไม่ว่าระบบตรวจจับการบุกรุกจะถูกออกแบบด้วยเทคนิคใดก็ตามปัญหาที่พบส่วนใหญ่ก็คือจะมีกระบวนการอย่างไรที่จะตัดสินได้ว่าพฤติกรรมแบบใดเป็นการบุกรุกและพฤติกรรมใดเป็นพฤติกรรมปกติ ปัญหาที่พบอีกปัญหาหนึ่งก็คือปัญหาเกี่ยวกับการจัดการกับข้อมูลจำนวนมากที่ต้องวิเคราะห์ ข้อมูลเหล่านี้อาจเป็นแพกเก็ตที่รับมาจากเครือข่ายหรือข้อมูลที่ถูกอ่านมาจากแฟ้มบันทึกการทำงานของระบบปฏิบัติการซึ่งโดยทั่วไปจะมีปริมาณมาก ซึ่งจากปัญหาดัง

กล่าวทั้งในเรื่องของกระบวนการการตัดสินใจพฤติกรรมการบุกรุกหรือปัญหาเกี่ยวกับขนาดของข้อมูล ทำให้เกิดการผิดพลาดของการวิเคราะห์การบุกรุก กระบวนการตัดสินที่ผิดพลาดของระบบตรวจจับการบุกรุกมีสองแบบคือ การตัดสินผิดทางบวก (false positive) และการตัดสินผิดทางลบ (false negative) การตัดสินใจผิดทางบวกคือ การที่ระบบตรวจจับการบุกรุกวิเคราะห์ข้อมูลแล้วตัดสินว่ารูปแบบข้อมูลเกิดจากพฤติ-กรรมการบุกรุก ในขณะที่ในความเป็นจริงรูปแบบข้อมูลนั้นเกิดจากพฤติกรรมการใช้งานตามปกติหรือที่ได้รับอนุญาตจากระบบ ส่วนทางด้านของการตัดสินผิดในทางลบนั้น คือกรณีที่ระบบตรวจจับการบุกรุกวิเคราะห์ข้อมูลแล้วตัดสินว่าไม่ได้เกิดจากพฤติกรรมหรือการกระทำที่เป็นการบุกรุกทั้งที่ในความเป็นจริงเหตุการณ์หรือพฤติกรรมนั้นเป็นรูปแบบของการบุกรุก จะเห็นได้ว่าการตัดสินผิดทางลบจะก่อให้เกิดผลเสียมากกว่าการตัดสินผิดทางบวกเนื่องจากการบุกรุกนั้นสามารถทำลายระบบได้โดยไม่ถูกตรวจจับหรือเฝ้าระวัง

ในกรณีปัญหาที่เกี่ยวข้องกับปริมาณข้อมูลที่ระบบตรวจจับการบุกรุกจะต้องวิเคราะห์ที่มีจำนวนมากนั้น อาจเนื่องมาจากการวิเคราะห์ข้อมูลโดยที่ไม่ได้มีการจัดแบ่งหรือกรองข้อมูลก่อนการวิเคราะห์ทำให้การตรวจจับล่าช้าหรือทำงานผิดพลาดได้โดยเฉพาะในระบบตรวจจับการบุกรุกที่ทำงานแบบเวลาจริง (real-time IDS) ซึ่งมีข้อจำกัดของเวลาเป็นตัวแปรสำคัญ

อย่างไรก็ตามจนกระทั่งปัจจุบันนี้ยังไม่มีระบบตรวจจับการบุกรุกวิธีใดที่มีประสิทธิภาพในการทำงานได้ดีที่สุดโดยไม่มีข้อผิดพลาดในการพัฒนาโปรแกรมตรวจจับการบุกรุกสำหรับวิทยานิพนธ์นี้ใช้เทคนิคของระบบตรวจจับแบบ misuse detection system ซึ่งมีข้อดีในเรื่องของความผิดพลาดในการตัดสินมีน้อย เทคนิคที่ใช้ในการตัดสินว่าพฤติกรรมใดเป็นการบุกรุกอาศัยข้อมูลที่แสดงสิทธิในการใช้งานทรัพยากรของระบบโดยผ่านทางการเรียกใช้ system call ซึ่งในที่นี้คือวิเคราะห์จากการใช้การวิเคราะห์การเปลี่ยนแปลงสถานะของค่าประจำตัวของผู้ใช้ร่วมกับการวิเคราะห์ system call ที่ถูกเรียกใช้ พร้อมทั้งกฎที่ใช้สนับสนุนการตรวจจับ ซึ่งนอกจากจะช่วยเสริมในเรื่องของการตัดสินผิดทางบวกให้ลดน้อยลงแล้วยังช่วยในเรื่องของการกรองข้อมูลที่จะต้องนำมาวิเคราะห์พฤติกรรมการบุกรุกนอกจากนี้ยังใช้การวิเคราะห์ข้อมูลในลักษณะของ Host-based Intrusion Detection ซึ่งง่ายต่อการพัฒนาและยังสามารถตรวจจับการบุกรุกได้อย่างรวดเร็วในลักษณะของการตรวจจับแบบเวลาจริง ในการทำวิทยานิพนธ์นี้ได้จัดทำระบบตรวจจับการบุกรุกบนระบบปฏิบัติการยูนิกซ์ ซึ่งเป็นระบบปฏิบัติการที่ได้รับการนำมาใช้สำหรับเครื่องให้บริการ (server) ต่าง ๆ หลากหลาย ในหัวข้อถัดไปจะกล่าวถึงเนื้อหาที่เกี่ยวข้องกับงานวิจัยนี้และเป็นเนื้อหาสำคัญที่นำมาใช้ในการตัดสินว่าพฤติกรรมใดจัดเป็นการบุกรุกระบบคอมพิวเตอร์

## 2.7 ค่าประจำตัวผู้ใช้ (User credentials )

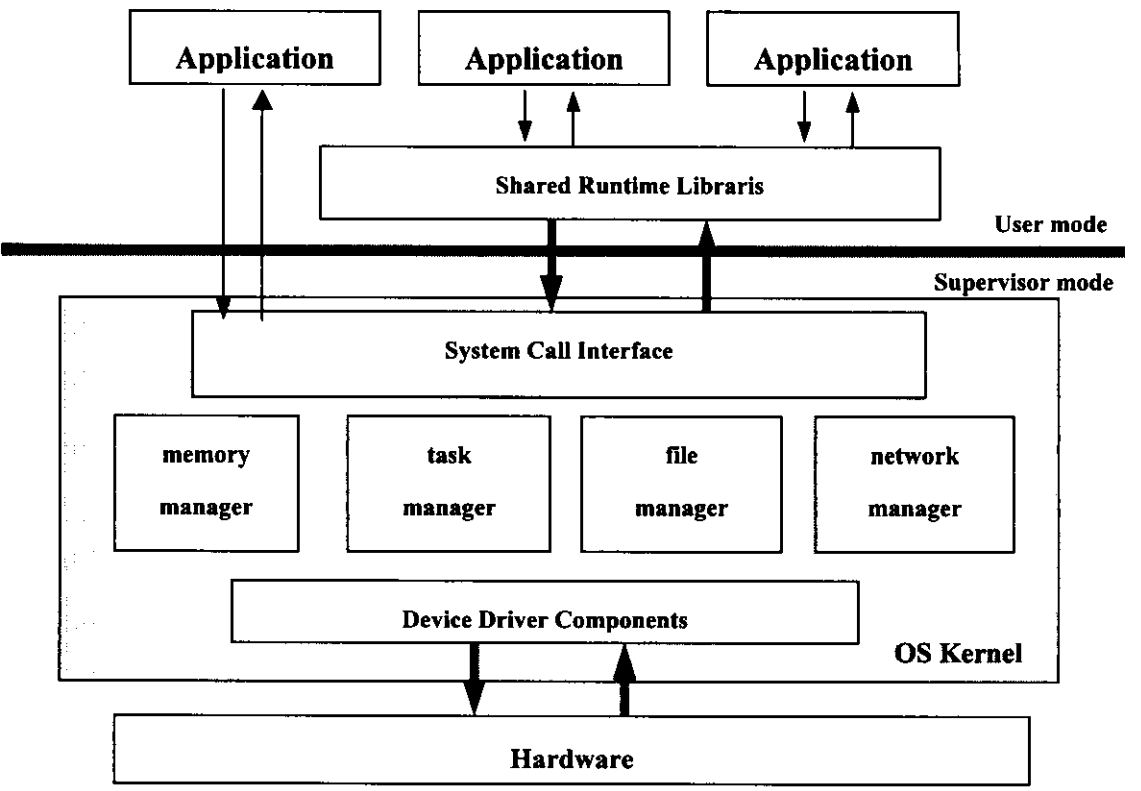
ในการใช้งานระบบปฏิบัติการยูนิกซ์นั้น ผู้ใช้ทุกคนจะสามารถใช้งานโดยการขอมีบัญชีผู้ใช้ (login หรือ account) และบัญชีผู้ใช้ที่ถูกสร้างขึ้นจะถูกกำหนดด้วยหมายเลขประจำตัวผู้ใช้ (User Identifier หรือ UID) และจะถูกกำหนดให้อยู่ในกลุ่มใดกลุ่มหนึ่งหรืออยู่ในหลายกลุ่มก็ได้ และแต่ละกลุ่มจะมีหมายเลขแทนกลุ่มที่เรียกว่า Group Identifier หรือเรียกย่อ ๆ ว่า GID นอกจากนี้ระบบปฏิบัติการยูนิกซ์ได้กำหนดค่า identifier อีกชนิดหนึ่งซึ่งเรียกว่า Effective Identifier หรือ EID ซึ่งใช้ในการตรวจสอบสิทธิการใช้งานของผู้ใช้ทุกคนในที่นี้คือค่า Effective User Identifier หรือ EUID และค่า Effective Group Identifier หรือ EGID ค่า ID เหล่านี้ใช้แทนด้วยค่าตัวเลขชนิด integer ซึ่งมีค่าตั้งแต่ 0 – ค่าสูงสุดของตัวแปรข้อมูลชนิด Integer ของแต่ละแบบ ในกระบวนการทำงานบนระบบปฏิบัติการยูนิกซ์ ทุกครั้งเมื่อผู้ใช้เรียกใช้คำสั่งต่าง ๆ จะมีการสร้างโปรเซสซึ่งเป็นกระบวนการทำงานของคำสั่งนั้น โดยที่ในโปรเซสนั้น ๆ จะมีการเรียกใช้ทรัพยากรในระบบผ่านทางบริการของระบบปฏิบัติการซึ่งเรียกว่า system call ในระบบปฏิบัติการยูนิกซ์ทรัพยากรของระบบทุก ๆ อย่างจะต้องมีเจ้าของขึ้นอยู่กับว่าเป็นทรัพยากรนั้นเป็นทรัพยากรอะไรอย่างเช่น แฟ้มข้อมูลที่ถูกสร้างขึ้นโดยผู้ใช้เอง เจ้าของทรัพยากรหรือเจ้าของแฟ้มนั้นก็คือตัวผู้ใช้ที่เป็นผู้สร้างแฟ้มนั้นขึ้นมา และในส่วนของแฟ้มที่เป็นแฟ้มระบบเจ้าของแฟ้มโดยทั่วไปคือ root ซึ่งเป็นผู้ที่มีสิทธิสูงสุดในระบบ โปรเซสเองก็ถือว่าเป็นส่วนหนึ่งของทรัพยากรและมีเจ้าของ อีกทั้งยังสามารถแสดงให้เห็นด้วยว่าโปรเซสนั้นกำลังเรียกใช้ทรัพยการใดและใครเป็นผู้กระทำ เมื่อโปรเซสกำลังทำงานในแต่ละโปรเซสจะมีค่า UID EUID GID EGID ติดอยู่เสมอ โดยได้มาจากค่า UID และ GID ของผู้ที่เรียกใช้หรือส่งโปรเซสเข้าไปทำงานในการเริ่มต้นทำงานของโปรเซสโดยทั่วไปค่าของ EUID/EGID จะมีค่าเท่ากับค่าของ UID/GID โดยที่ค่า UID และ GID จะใช้ระบุว่าผู้ใช้คนนั้นเป็นใครและอยู่ในกลุ่มใด ค่า EUID และค่า EGID จะถูกนำมาพิจารณาเมื่อโปรเซสเรียกใช้ทรัพยากรในระบบ นั่นคือเมื่อมีการเรียกใช้ทรัพยากรต่าง ๆ ในกระบวนการทำงานของคำสั่งหรือโปรเซสระบบจะมีขั้นตอนในการตรวจสอบสิทธิในการเข้าใช้ทรัพยากรนั้น ๆ โดยการตรวจสอบค่า EUID และ/หรือ ค่า EGID ของโปรเซสกับค่า UID และ/หรือ GID ของเจ้าของทรัพยากรที่ต้องการใช้ พร้อมทั้งพิจารณาค่า File/Directory Permission ของแฟ้มหรือไดเรกทอรีที่ถูกเรียกใช้ในขณะนั้น เมื่อมีการเรียกใช้ทรัพยากรต่าง ๆ ในกระบวนการทำงานของคำสั่งหรือโปรเซสระบบจะมีขั้นตอนในการตรวจสอบสิทธิในการเข้าใช้ทรัพยากรนั้น ๆ โดยการตรวจสอบค่า EUID และ/หรือ ค่า EGID ของ

โปรเซสกับค่า UID และหรือ GID ของเจ้าของทรัพยากรที่ต้องการใช้ พร้อมทั้งพิจารณาค่า File /Directory Permission หรือสิทธิในการเข้าถึงของแฟ้มหรือไดเรกทอรีที่ถูกเรียกใช้ในขณะนั้น [34] และเมื่อลองพิจารณาการเรียกใช้ทรัพยากรในระบบ จะเห็นได้ว่าค่าประจำตัวผู้ใช้ที่ติดมากับโปรเซสทั้งสี่ค่าคือ UID EUID GID EGID นั้น ค่าที่ถูกนำมาพิจารณาในลำดับแรกคือค่า EUID ดังที่ได้กล่าวไว้แล้วว่าในระบบปฏิบัติการยูนิกซ์จะมีผู้ใช้ซึ่งมีสิทธิสูงสุดหรือ root ซึ่งสามารถเข้าใช้งานทรัพยากรทั้งหมดในระบบได้ ค่าประจำตัวผู้ใช้หรือ User credentials ของ root บนระบบปฏิบัติการยูนิกซ์ มีค่า UID = 0 EUID = 0 GID = 0 และ EGID = 0 ซึ่งในวิทยานิพนธ์นี้เมื่อต้องการอ้างถึงค่าเหล่านี้จะเขียนในรูปแบบ (0,0,0,0) โดยแทน (UID,EUID,GID,EGID) ตามลำดับ ดังนั้นถ้าหากในขณะที่โปรเซสกำลังเรียกใช้ทรัพยากรในระบบและค่า EUID ที่ติดมากับโปรเซสในขณะนั้นมีค่าเป็น 0 ซึ่งเท่ากับค่า UID ของผู้ใช้ที่มีสิทธิสูงสุดเมื่อพิจารณาตามขั้นตอนการตรวจสอบสิทธิในการเข้าใช้ทรัพยากรโปรเซสนั้นก็จะได้รับอนุญาตจากระบบให้สามารถเรียกใช้ทรัพยากรในระบบได้เทียบเท่ากับสิทธิของผู้ใช้สูงสุดได้รับ นั่นหมายถึงสามารถเรียกใช้งานทรัพยากรทั้งหมดในระบบได้ เช่นเดียวกันกับค่า EGID ที่ติดมากับโปรเซส ถ้าหากมีค่าเท่ากับค่า GID ของ root คือมีค่าเป็นศูนย์ โปรเซสนั้นก็จะได้รับอนุญาตจากระบบให้สามารถเรียกใช้ทรัพยากรในระบบได้เทียบเท่ากับสิทธิของผู้ใช้สูงสุดได้รับเช่นเดียวกัน นั่นหมายถึงสามารถเรียกใช้งานทรัพยากรทั้งหมดในระบบได้ ฉะนั้นหากในสถานะหรือในขณะที่โปรเซสมีค่า EUID/EGID เป็นศูนย์ซึ่งมีสิทธิเทียบเท่า root ถือเป็นสถานะที่ควรเฝ้าระวังเนื่องจากเป็นสถานะที่ผู้บุกรุกสามารถก่อให้เกิดความเสียหายแก่ระบบได้ ในหัวข้อถัดไปจะกล่าวถึง system call และ system call ที่ทำให้ค่าของ EUID/EGID เปลี่ยนแปลง

## 2.8 System call

บนระบบปฏิบัติการยูนิกซ์เมื่อผู้ใช้เรียกใช้คำสั่งหรือโปรแกรมเพื่อให้ได้ผลลัพธ์ตามที่ต้องการ โปรแกรมหรือคำสั่งเหล่านั้นมีกระบวนการทำงานหรือเรียกอีกชื่อหนึ่งว่าโปรเซสเพื่อจัดการให้ได้ผลลัพธ์ตามที่ผู้ใช้ต้องการขึ้นอยู่กับคำสั่งหรือโปรแกรมที่ถูกเรียกใช้ โปรเซสของคำสั่งที่ถูกเรียกใช้จะมีการเรียกใช้ "system call" เพื่อติดต่อกับระบบปฏิบัติการดังแสดงในภาพประกอบที่ 2.2 ซึ่งแสดงความสัมพันธ์และกลไกในการทำงานของโปรเซสต่าง ๆ ในระบบปฏิบัติการยูนิกซ์ ส่วนของโปรแกรมที่ผู้ใช้เรียกใช้งานจะอยู่ในส่วนของพื้นที่ที่เรียกว่า user mode โปรแกรมจะติดต่อกับระบบปฏิบัติการโดยการเรียกใช้ system call ซึ่งอยู่ในส่วนของ kernel mode โดยตรงหรือผ่านทาง shared library อย่างเช่น library ของภาษาซี เมื่อระบบปฏิบัติการทำงานตามข้อมูลที่ได้รับมา

จาก system call แล้วก็จะส่งผลลัพธ์ผ่าน system call (return value) มายัง library ที่ถูกเรียกใช้เพื่อส่งค่ากลับมายังโปรแกรมหรือคำสั่งที่ผู้ใช้เรียกใช้อีกขั้นหนึ่ง หรือกล่าวโดยสรุป "system call" ก็คือตัวกลางในการติดต่อระหว่างผู้ใช้กับระบบปฏิบัติการและอุปกรณ์ที่ให้บริการเมื่อมีการเรียกใช้คำสั่งหรือโปรแกรมซึ่งหมายถึงไม่ว่าผู้ใช้จะเรียกใช้คำสั่งที่มีอยู่บนระบบหรือเขียนโปรแกรมเพื่อทำงานตามความต้องการของผู้ใช้เอง คำสั่งหรือโปรแกรมเหล่านั้นจะมีการเรียกใช้ system call เสมอและจากความสำคัญของค่า EUID และ EGID โดยเฉพาะเมื่อค่าเหล่านี้มีค่าเท่ากับศูนย์ ซึ่งเป็นค่าประจำตัวของ root มี system call บางตัวบนระบบปฏิบัติการยูนิกซ์สามารถเปลี่ยนแปลงค่าของ EUID/EGID ได้ซึ่งถ้าหากถูกนำไปใช้ในทางที่ผิดจะทำให้เกิดความเสียหายแก่ระบบได้นั่นคือถ้าหากโปรเซสของผู้ใช้ปกติซึ่งไม่ใช่ root มีการเรียกใช้ system call เพื่อเปลี่ยนค่าของ EUID หรือ EGID ให้เป็นศูนย์ และมีการเรียกใช้ทรัพยากรอย่างเช่น แฟ้ม /etc/passwd ซึ่งเป็นแฟ้มที่เก็บรหัสผ่านของผู้ใช้ในระบบเพื่อทำการแก้ไข โดยปกติแล้วผู้ใช้ปกติซึ่งไม่ใช่ root จะไม่มีสิทธิในการแก้ไขแฟ้มรหัสผ่านนี้ ในหัวข้อถัดไปจะกล่าวถึง system call ที่ใช้กำหนดค่าของ UID, EUID, GID และ EGID




ภาพประกอบที่ 2.3 แสดงการทำงานของ system call [35]

### 2.8.1 System call ที่ใช้กำหนดค่าของ UID, EUID, GID และ EGID

ในหัวข้อนี้จะกล่าวถึง system call บนระบบปฏิบัติการยูนิกซ์ซึ่งใช้กำหนดค่าของ UID EUID GID และ EGID ในระหว่างกระบวนการทำงานของคำสั่งหรือโปรเซส ซึ่งได้แก่ system call *setuid(), setgid(), setreuid(), setregid()* แต่ละตัวตามลำดับดังนี้

#### 2.8.1.1 *setuid(uid_t uid)* system call

*setuid(uid_t uid)* ถูกใช้เพื่อกำหนดค่าของ UID และ EUID ของโปรเซสที่เรียกใช้ให้เป็นไปตามค่า **uid** ที่ระบุในฟังก์ชัน *setuid()* โดยมีเงื่อนไขการทำงานคือ

1. ถ้าค่า EUID ของโปรเซสในขณะนั้นมีค่าเท่ากับศูนย์ ฟังก์ชัน setuid() จะเปลี่ยนค่าของ UID และ EUID ให้เท่ากับค่า **uid** ที่ระบุในฟังก์ชัน
2. ถ้าค่า EUID ของโปรเซสในขณะนั้นมีค่าไม่เท่ากับศูนย์ ฟังก์ชันนี้จะทำงานสำเร็จเมื่อค่า **uid** มีค่าเท่ากับค่า **UID** ของโปรเซสนั้น

#### 2.8.1.2 *setgid(gid_t gid)* system call

*setgid(gid_t gid)* ถูกใช้เพื่อกำหนดค่าของ GID และ EGID ของโปรเซสที่เรียกใช้ให้เป็นไปตามค่า **gid** ที่ระบุในฟังก์ชัน *setgid()* โดยรูปแบบการทำงานคล้ายกับฟังก์ชัน *setuid(uid_t uid)*

#### 2.8.1.3 *setreuid(uid_t ruid, uid_t euid)*

*setreuid(uid_t ruid, uid_t euid)* ใช้เพื่อสลับค่าของ UID และ EUID ของโปรเซสในขณะนั้น โดยที่ค่า UID จะถูกเปลี่ยนให้มีค่าเท่ากับค่า **ruid** ที่ระบุในฟังก์ชันในขณะที่ค่า EUID ของโปรเซสจะถูกเปลี่ยนให้มีค่าเท่ากับค่า euid ที่ถูกระบุเพื่อเปลี่ยนสิทธิในการเข้าใช้ชั่วคราว

#### 2.8.1.4 *setregid(gid_t rgid, gid_t egid)*

*setregid(gid_t rgid, gid_t tegid)* ถูกใช้เพื่อสลับค่าของ GID และ EGID ของโปรเซสในขณะนั้น โดยที่ค่าของ GID จะถูกเปลี่ยนให้มีค่าเท่ากับค่า rgid ที่ถูกระบุในฟังก์ชัน ในขณะที่ค่า EGID ของโปรเซสจะถูกเปลี่ยนให้มีค่าเท่ากับค่า **egid** ที่ระบุในฟังก์ชัน

จากหัวข้อ 2.8.1.1 และหัวข้อ 2.8.1.2 ซึ่งกล่าวถึง system call *setuid()* และ system call *setgid()* จะเห็นได้ว่าถ้าหากมีการเรียกใช้ system call *setuid()* หรือ system call *setgid()* ในขณะที่โปรเซสมีค่า EUID เท่ากับศูนย์ ระบบก็จะอนุญาตให้ใช้ system call *setuid()* เพื่อเปลี่ยนค่า UID และ EUID ตามค่าที่ระบุในฟังก์ชัน หากค่าที่ระบุในฟังก์ชันมีค่าเป็นศูนย์ นั่นก็หมายความว่าค่า UID ที่ติดมากับโปรเซสในขณะนั้นก็จะถูกเปลี่ยนเป็นศูนย์ ซึ่งหมายความว่าในขณะนั้นโปรเซสมีค่า UID และ EUID เท่ากับศูนย์ทำให้โปรเซสในขณะนั้นอยู่ในสถานะของ root เนื่องจากมีค่า UID เท่ากับศูนย์ ในทำนองเดียวกันกับ system call *setgid()* หากโปรเซสที่เรียกใช้ system call *setgid()* มีค่า EGID ที่ติดมากับโปรเซสในขณะนั้นเท่ากับศูนย์ ระบบก็จะอนุญาตให้ใช้ system call *setgid()* เพื่อเปลี่ยนค่า GID ตามค่าที่ระบุในฟังก์ชัน หากค่าที่ระบุมีค่าเป็นศูนย์นั่นก็หมายความว่าค่า GID และ EGID ที่ติดมากับโปรเซสในขณะนั้นก็จะถูกเปลี่ยนเป็นศูนย์ทั้งสองค่า ซึ่งก็คือโปรเซสในขณะนั้นอยู่ในสถานะของกลุ่ม root

นอกจากการใช้ system call ที่กล่าวมาข้างต้นในการเปลี่ยนค่า EUID และ EGID แล้วค่าทั้งสองนี้อาจจะสามารถถูกเปลี่ยนแปลงค่าได้ถ้าหากว่าคำสั่งหรือโปรแกรมที่ถูกเรียกใช้นั้น มีการกำหนด set-user-ID bit นั่นคือมีการระบุให้ในขณะที่มีการทำงานของคำสั่งหรือโปรแกรมเหล่านี้ค่า EUID ของโปรเซสจะมีค่าเท่ากับเจ้าของคำสั่งหรือโปรแกรมเหล่านั้น ตัวอย่างเช่น ผู้ใช้ปกติเรียกใช้คำสั่ง *passwd* ซึ่งมีการกำหนดสิทธิเป็น **r-sr-xr-x** นั่นคือมีกำหนดสิทธิในการรันโปรแกรมให้กับผู้ใช้อื่นดังนั้นผู้ใช้ปกติจึงสามารถเรียกใช้คำสั่งนี้ได้ แต่เนื่องจากมีการกำหนดสิทธิด้วยสัญลักษณ์ "s" ในส่วนของการรันโปรแกรมของผู้ใช้ประเภท user หรือเจ้าของแฟ้ม (หรือเรียกว่ามีการกำหนด set-user-ID bit) ดังนั้นเมื่อมีการเรียกใช้คำสั่งหรือแฟ้ม *passwd* ค่า EUID ของโปรเซสในขณะที่มีการเรียกใช้คำสั่ง *passwd* นี้จะมีค่าเท่ากับเจ้าของโปรแกรมหรือมีค่าเท่ากับศูนย์ เช่นเดียวกับ คำสั่งหรือโปรแกรมที่มีการกำหนด set-group-ID ในระหว่างการทำงานของคำสั่งประเภทนี้ค่า EGID ของโปรเซสในขณะนั้นจะมีค่าเท่ากับค่า GID ของเจ้าของแฟ้มหรือคำสั่งเช่นเดียวกัน ซึ่งหากมีการเรียกใช้ system call *setuid()*, *setgid()*, *setreuid()*, หรือ *setregid()* ในกระบวนการทำงานของคำสั่ง

ประเภท set-user-ID หรือ set-group-ID ก็ทำให้มีโอกาสที่จะมีการเปลี่ยนแปลงค่าของ UID หรือ GID เป็นศูนย์ ซึ่งก็คือโปรเซสในขณะนั้นอยู่ในสถานะของกลุ่ม root หรือกลุ่มระบบได้เช่นเดียวกัน จะเห็นได้ว่าเมื่อค่า EUID หรือ EGID ที่ติดมากับโปรเซสมีค่าเท่ากับศูนย์หรือมีค่าเท่ากับค่า UID หรือ GID ของ root โปรเซสในขณะนั้นมีโอกาสที่จะเข้าสู่สถานะของการเป็น root คือมีค่า UID เท่ากับศูนย์หรือเข้าสู่สถานะของกลุ่มระบบเนื่องจากมีค่า GID เท่ากับศูนย์นอกจากยังสามารถเรียกใช้งาน system call ใด ๆ ก็ได้ในระบบ รวมทั้งทรัพยากรต่าง ๆ ซึ่งโดยปกติแล้วผู้ใช้ปกติซึ่งไม่ใช่ root จะมีค่า EUID และ EGID เท่ากับค่า UID และ GID ของผู้ใช้เองซึ่งจะไม่มีค่าใดค่าหนึ่งมีค่าเท่ากับศูนย์ และผู้ใช้ปกติจะถูกกำหนดให้ใช้ทรัพยากรได้ตามที่ระบบกำหนดไว้เท่านั้น ฉะนั้นหากพบว่าผู้ใช้พยายามที่จะใช้งานทรัพยากรที่ไม่ได้ถูกกำหนดสิทธิไว้ หรือพยายามที่จะกระทำตัวเป็น root คือมีการเปลี่ยนแปลงค่าของ UID หรือ GID ในโปรเซสในขณะใดขณะหนึ่งจะถือว่าเป็นพฤติกรรมที่เข้าข่ายการบุกรุก ฉะนั้นหากเมื่อใดก็ตามที่ค่า EUID หรือ EGID ที่ติดมากับโปรเซสมีค่าเท่ากับศูนย์จึงเป็นสภาวะที่ควรเฝ้าระวังเนื่องจากมีโอกาสที่โปรเซสในขณะนั้นจะเรียกใช้ทรัพยากรที่เจ้าของโปรเซสไม่มีสิทธิ หรือมีโอกาสที่จะมีการเปลี่ยนแปลงค่าของ UID หรือ GID ของโปรเซสในขณะนั้นได้ ในหัวข้อถัดไปจะกล่าวถึงการเปลี่ยนแปลงสถานะของโปรเซส โดยการนิยามสถานะตามค่า UID EUID GID และ EGID ของโปรเซสในขณะนั้น

## 2.9 การเปลี่ยนแปลงสถานะ (State Transition)

จากงานวิจัยเรื่อง "A Process State-Transition Analysis and its Application to Intrusion Detection" [1] ซึ่งได้ทำการศึกษาเกี่ยวกับคุณสมบัติและความสัมพันธ์ของค่าประจำตัวของผู้ใช้และพฤติกรรมการบุกรุกโดยทั่วไป ได้สรุปและเสนอแนะการสร้างระบบตรวจจับการบุกรุกโดยการตรวจสอบจากการใช้ system call โดยโปรแกรมต่าง ๆ ซึ่งอาศัยรูปแบบของ State Transition Machine อธิบายถึงการเปลี่ยนแปลงค่าของ UID, EUID, GID และ EGID ที่ติดมากับโปรเซส ซึ่งสามารถแทนได้ด้วยสถานะต่าง ๆ หลายสถานะ โดยมีคำอธิบายคร่าว ๆ ดังนี้

### 2.9.1 คำนิยามของสถานะ (State Definitions )

คำว่าสถานะในที่นี้จะใช้อธิบายค่าของ UID, EUID, GID และ EGID ที่ติดมากับโปรเซส ณ ช่วงเวลาใดเวลาหนึ่ง โดยจะพิจารณาจากค่าของ UID, EUID, GID, และ EGID ยกตัวอย่างเช่น ในขณะหนึ่งโปรเซสมีค่า UID, EUID, GID และ EGID เป็น (100, ***100***, 20, 20) จะ

หมายถึง โปรเซสอยู่ในสถานะ A ต่อมาในอีกขณะเวลาหนึ่งโปรเซสมีค่า UID, EUID, GID, และ EGID เป็น (100, *0*, 20, 20) จะถือว่าโปรเซสอยู่ในสถานะ B นั่นคือเมื่อค่าใดค่าหนึ่งที่ติดมากับโปรเซสไม่ว่าจะเป็น UID, EUID, GID หรือ EGID มีการเปลี่ยนแปลงจะถือว่ามีการเปลี่ยนสถานะของโปรเซส ดังนั้นแต่ละสถานะในที่นี้จะประกอบด้วยชุดของตัวเลข 4 ตัว คือค่า UID, EUID, GID และ EGID ที่ติดมากับโปรเซส โดยที่ค่า UID และ EUID จะถูกแทนด้วยสัญลักษณ์ uid หรือ sid หรือ oid ขึ้นอยู่กับค่าของ UID หรือ EUID ในขณะนั้น และในขณะเดียวกันค่าของ GID และ EGID จะถูกแทนด้วยสัญลักษณ์ gid หรือ sgid หรือ ogid เช่นเดียวกันขึ้นอยู่กับค่าของ GID หรือ EGID ในขณะนั้นเช่นเดียวกัน ย่อหน้าถัดไปจะกล่าวถึงความหมายของสัญลักษณ์ uid, sid, oid, gid, sgid และ ogid ตามลำดับ

*uid* – (user's id) ใช้อ้างถึงค่า UID และ EUID ที่ติดมากับโปรเซสเมื่อค่าของ UID หรือ EUID ที่ติดมากับโปรเซสในขณะนั้นมีค่าเท่ากับค่า UID ของเจ้าของโปรเซสซึ่งได้รับการกำหนดให้โดยผู้ดูแลระบบ

*sid* – (special id) ใช้อ้างถึงค่า UID และ EUID ที่ติดมากับโปรเซสเมื่อค่าของ UID หรือ EUID ที่ติดมากับโปรเซสในขณะนั้นมีค่าเท่ากับค่า UID ของผู้ใช้ที่มีสิทธิพิเศษซึ่งในที่นี้จะหมายถึง root, daemon, operator, bin, news

*oid* – (other's id) ใช้อ้างถึงค่า UID และ EUID ที่ติดมากับโปรเซสเมื่อค่าของ UID หรือ EUID ที่ติดมากับโปรเซสในขณะนั้นมีค่าไม่เข้าข่ายในกรณีของ *uid* หรือ *sid*

*gid* – (group's id) ใช้อ้างถึงค่า GID และ EGID ที่ติดมากับโปรเซสเมื่อค่าของ GID หรือ EGID ที่ติดมากับโปรเซสในขณะนั้นมีค่าเท่ากับค่า GID ของเจ้าของโปรเซสซึ่งได้รับการกำหนดให้โดยผู้ดูแลระบบ

*sgid* – (special group id) ใช้อ้างถึงค่า GID และ EGID ที่ติดมากับโปรเซสเมื่อค่าของ GID หรือ EGID ที่ติดมากับโปรเซสในขณะนั้นมีค่าเท่ากับค่า GID ของกลุ่มสิทธิพิเศษ เช่น wheel, root, daemon, kmem, sys, tty, operator, bin และ news

*ogid* – (other's group id) ใช้อ้างถึงค่า GID และ EGID ที่ติดมากับโปรเซสเมื่อค่าของ GID หรือ EUID ที่ติดมากับโปรเซสในขณะนั้นมีค่าไม่เข้าข่ายในกรณีของ *gid* หรือ *sgid*

### 2.9.2 สถานะ (State)

ในขณะเวลาใดเวลาหนึ่งโปรเซสจะอยู่ในสถานะใดสถานะหนึ่งขึ้นอยู่กับค่าของ UID, EUID, GID และ EGID ขึ้นอยู่กับว่าในขณะนั้นค่า UID, EUID, GID และ EGID มีค่าเป็นอย่างไรสามารถแบ่งสถานะของโปรเซสออกได้เป็น 6 สถานะคือ สถานะปกติ (normal state) สถานะที่โปรเซสมีสิทธิพิเศษ (special privileged state) สถานะผู้ใช้สูงสุด (superuser state) สถานะกลุ่มระบบ (system group state) สถานะผู้ใช้อื่น (another user state) และสถานะสิ้นสุด (terminate state) โดยที่แต่ละสถานะมีความหมายดังต่อไปนี้

#### 2.9.2.1 สถานะปกติ (Normal state)

โปรเซสอยู่ในสถานะปกติก็ต่อเมื่อค่าของ UID, EUID, GID และ EGID ที่ติดมากับโปรเซสในขณะนั้นมีค่าเท่ากับค่า UID, EUID, GID และ EGID ของเจ้าของโปรเซสดังนั้นจากการนิยามสถานะในหัวข้อ 2.9.1 จะแทนค่าของ (UID, EUID, GID, EGID) ด้วย (***uid, uid, gid, gid***)

#### 2.9.2.2 สถานะที่มีสิทธิพิเศษ (Special Privileged State)

โปรเซสอยู่ในสถานะสิทธิพิเศษก็ต่อเมื่อค่า UID EUID ที่ติดมากับโปรเซสในขณะนั้นมีค่าใดค่าหนึ่งเท่ากับค่าของผู้ใช้ที่มีสิทธิพิเศษ เช่นเดียวกันกับค่า GID และ EGID ที่ติดมากับโปรเซสมีค่าใดค่าหนึ่งเท่ากับค่าของกลุ่มที่มีสิทธิพิเศษ ซึ่งการเปลี่ยนแปลงของค่าเหล่านี้จะขึ้นอยู่กับ system call ที่ถูกเรียกใช้ สามารถแบ่งออกได้เป็น 4 สถานะคือ

*setuid* ใช้เรียกสถานะที่เกิดจากการเรียกใช้ system call *setuid()* หรือสถานะที่เกิดจากการเรียกใช้คำสั่งประเภท set-user-ID

*setreuid* ใช้เรียกสถานะที่เกิดจากการเรียกใช้ system call *setreuid()*

*setgid* ใช้เรียกสถานะที่เกิดจากการเรียกใช้ system call *setgid()* หรือสถานะที่เกิดจากการเรียกใช้คำสั่งประเภท set-group-ID

*setregid* ใช้เรียกสถานะที่เกิดจากการเรียกใช้ system call *setregid()*

**2.9.2.3 สถานะผู้ใช้สูงสุด (Superuser State)**

โปรเซสอยู่ในสถานะผู้ใช้สูงสุดก็ต่อเมื่อค่าของ UID และ EUID ที่ติดมากับโปรเซสในขณะนั้นมีค่าเท่ากับค่า UID ของผู้ใช้ที่มีสิทธิพิเศษ นั่นคือสามารถอ้างถึงค่า UID, EUID, GID และ EGID ของโปรเซสในขณะนั้นด้วย ***(sid, sid, gid, gid)***

**2.9.2.4 สถานะกลุ่มระบบ (System Group State)**

โปรเซสอยู่ในสถานะกลุ่มระบบก็ต่อเมื่อค่าของ GID และ EGID ที่ติดมากับโปรเซสในขณะนั้นมีค่าเท่ากับค่าของกลุ่มสิทธิพิเศษ นั่นคือสามารถอ้างถึงค่า UID, EUID, GID, และ EGID ของโปรเซสในขณะนั้นด้วย ***(uid, uid, sgid, sgid)***

**2.9.2.5 สถานะผู้ใช้อื่น (Another User State)**

โปรเซสอยู่ในสถานะผู้ใช้อื่นก็ต่อเมื่อค่าของ UID, EUID, GID และ EGID ที่ติดมากับโปรเซสในขณะนั้นมีค่าเข้าข่ายต่อไปนี้

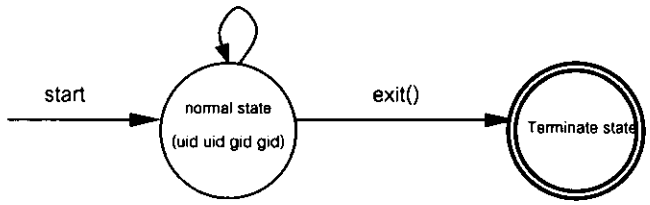
ค่า UID และ EUID ของโปรเซสขณะนั้นเปลี่ยนเป็นค่าของผู้ใช้อื่นที่ไม่ใช้เจ้าของโปรเซสหรือผู้ใช้ที่มีสิทธิพิเศษ นั่นคือสามารถอ้างถึงได้ด้วย ***(oid, oid, gid, gid)*** หรือค่า GID และ EGID ของโปรเซสขณะนั้นเปลี่ยนเป็นค่าของกลุ่มผู้ใช้อื่นที่ไม่ใช่เจ้าของโปรเซสหรือผู้ใช้ที่มีสิทธิพิเศษนั่นคือสามารถอ้างถึงด้วย ***(uid, uid, ogid, ogid)***

**2.9.2.6 สถานะสิ้นสุด (Terminate State)**

โปรเซสอยู่ในสถานะสิ้นสุดเมื่อ กระบวนการทำงานของคำสั่งนั้นกระทำจนสิ้นสุดกระบวนการและจบคำสั่งอย่างสมบูรณ์

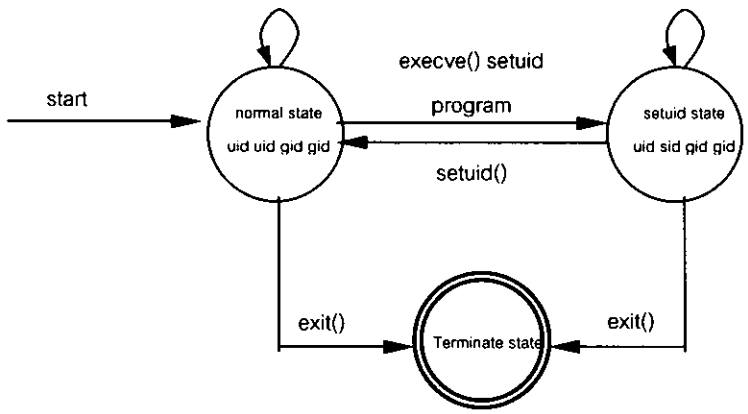
จากการนิยามสถานะและการจัดประเภทของสถานะจะเห็นได้ว่าสถานะที่ควรเฝ้าระวังหรือสถานะที่เสี่ยงต่อการก่อให้เกิดความเสียหายต่อระบบอันเกิดจากการเรียกใช้ทรัพยากรของระบบหรือการให้ได้สิทธิของ root ก็คือสถานะสิทธิพิเศษ ฉะนั้นสามาถตรวจสอบความผิดปกติ

ของระบบหรือตรวจสอบพฤติกรรมการบุกรุกๆได้จากสถานะของโปรเซสที่เปลี่ยนไป ภาพประกอบที่ 2.3 2.4 และ 2.5 จะแสดงให้เห็นถึงการเปลี่ยนสถานะเมื่อมีการเรียกใช้คำสั่งปกติซึ่งคือคำสั่งที่ไม่มีการกำหนด set-user-ID bit หรือ set-group-ID bit คำสั่งที่มีการกำหนด set-user-ID bit คำสั่งที่มีการกำหนด set-group-ID bit และในภาพประกอบที่ 2.6 จะแสดงให้เห็นภาพรวมของลักษณะการเปลี่ยนแปลงสถานะของทุกสถานะ



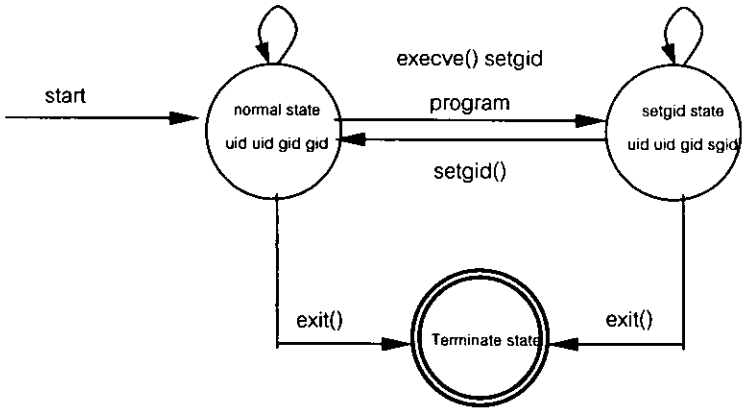

ภาพประกอบที่ 2.3 แสดงการเปลี่ยนแปลงสถานะเมื่อเรียกใช้คำสั่งปกติ

ที่มา: [36]

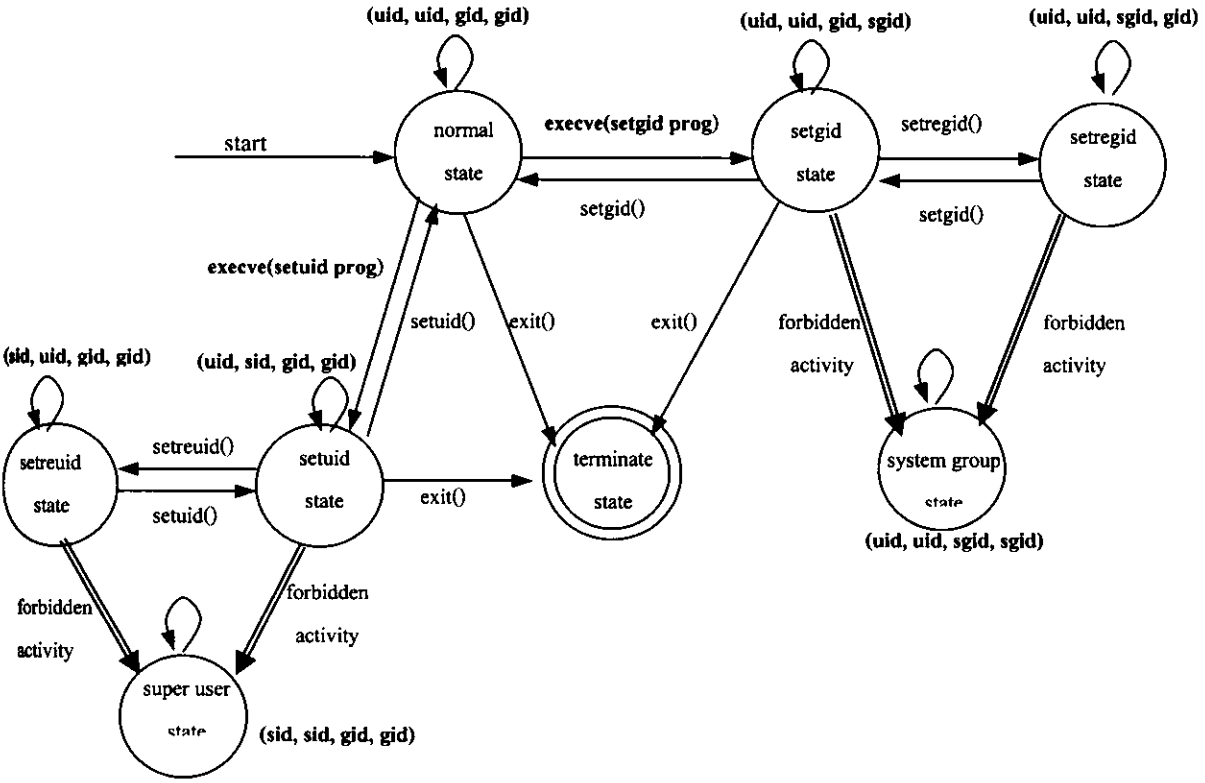



ภาพประกอบที่ 2.4 แสดงการเปลี่ยนแปลงสถานะเมื่อเรียกใช้คำสั่งที่มีการกำหนด set-user-ID bit

ที่มา: [36]




ภาพประกอบที่ 2.5 แสดงการเปลี่ยนแปลงสถานะเมื่อเรียกใช้คำสั่งที่มีการกำหนด set-group-ID bit ที่มา [36]




ภาพประกอบที่ 2.6 แสดงภาพรวมของลักษณะการเปลี่ยนแปลงสถานะของทุกสถานะ ที่มา: [36]

จากภาพประกอบที่ 2.3 – 2.5 จะสังเกตได้ว่าสถานะจะมีการเปลี่ยนแปลงเมื่อมีการเรียกใช้ system call *setuid(), setreuid(), setgid(), setregid()* และเมื่อมีการเรียกใช้โปรแกรม/คำสั่งที่มีการกำหนด set-user-ID bit (setuid program) หรือเมื่อมีการเรียกใช้โปรแกรม/คำสั่งที่มีการกำหนด set-group-ID bit (setgid program) นอกจากนี้ในภาพประกอบที่ 2.6 ยังแสดงให้เห็นว่าในขณะที่โปรเซสอยู่ในสถานะที่มีสิทธิพิเศษไม่ว่าจะเป็น setuid state, setreuid state, setgid state หรือ setregid state เนื่องจากสามารถทำให้โปรเซสเปลี่ยนสถานะไปเป็น superuser state หรือ system group state ซึ่งเป็นสถานะที่ไม่ได้รับการอนุญาตให้เกิดขึ้นในโปรเซสหรือกระบวนการทำงานของคำสั่งหรือโปรแกรมที่ถูกเรียกใช้โดยผู้ใช้ปกติ

ดังนั้นจะเห็นได้อย่างชัดเจนว่าถ้าหากสามารถที่จะติดตามกระบวนการทำงานของคำสั่ง/โปรแกรม โดยการพิจารณาการเปลี่ยนแปลงสถานะของโปรเซส จากการพิจารณาค่า UID EUID GID และ EGID ของโปรเซสขณะนั้น รวมทั้งค่า system call ที่ถูกเรียกใช้เราสามารถตรวจจับหรือวิเคราะห์ได้ว่าพฤติกรรมที่เกิดขึ้นนั้นเข้าข่ายการบุกรุกหรือไม่ ตัวอย่างเช่นผู้ใช้ปกติเริ่มต้นการเรียกใช้งานคำสั่งด้วยสถานะปกติแต่ในระหว่างการทำงานของโปรเซส สถานะของโปรเซสเปลี่ยนเป็นสถานะที่มีสิทธิพิเศษ และมีการเรียกใช้ system call *setuid()* เพื่อเปลี่ยนแปลงค่า UID ของโปรเซสในขณะนั้นเป็นศูนย์ พฤติกรรมเช่นนี้จะถูกระบุว่าเป็นการกระทำที่เข้าข่ายการบุกรุก

## 2.10 กฎที่ใช้สนับสนุนระบบตรวจจับการบุกรุก (Supporting Rules for Intrusion Detection)

กฎที่ใช้สนับสนุนระบบตรวจจับการบุกรุกที่จะกล่าวถึงนี้ ใช้ตัดสินพฤติกรรมที่เกิดขึ้นในขณะที่โปรเซสอยู่ในสถานะที่มีสิทธิพิเศษว่าเข้าข่ายการบุกรุก โดยจะพิจารณาจากสถานะของโปรเซสในขณะนั้นและ system call ที่ถูกเรียกใช้

**กฎข้อที่ 0** อนุญาตให้ system call ***setreuid()*** และ ***setregid()*** เท่านั้นที่สามารถเปลี่ยน UID หรือ GID ได้ ถ้ามีการใช้ system call ตัวอื่นหรือโปรแกรมอื่นเพื่อเปลี่ยน UID หรือ GID ให้ถือว่าเข้าข่ายการบุกรุก

**กฎข้อที่ 1** ในสถานะที่ต้องใช้ สิทธิพิเศษ ไม่อนุญาตให้มีการเรียกใช้ system call *execve()*

**กฎข้อที่2**ในขณะที่โปรเซสอยู่ในสถานะสิทธิพิเศษไม่อนุญาตให้มีการสร้างโปรแกรม setuid/setgid อนุญาตเฉพาะผู้ใช้ที่มีสิทธิสูงสุด เท่านั้น

**กฎข้อที่ 3** ไม่อนุญาตให้โปรเซสที่อยู่ในสถานะสิทธิพิเศษแก้ไขโปรแกรมใด ๆ ของระบบ

**กฎข้อที่ 4** ผู้ใช้ที่มีสิทธิสูงสุดเท่านั้นที่มีสิทธิในการสร้างบัญชีผู้ใช้ใหม่ได้

**กฎข้อที่ 5** System call เหล่านี้ *mount(), unmount(), nfssvc(), quotactl(), reboot(), settimeofday(), swapon()* สามารถเรียกใช้ได้โดยผู้ใช้ที่มีสิทธิสูงสุดเท่านั้น

จากกฎทั้งหมดนี้ ถ้ามีเหตุการณ์ใดที่ไม่ตรงตามกฎข้อใดข้อหนึ่งหรือพยายามจะทำลายกฎเหล่านี้จะถือว่าเป็นเหตุการณ์ที่ผิดปกติ หรือเป็นพฤติกรรมที่เข้าข่ายการบุกรุก วิทยานิพนธ์นี้จะใช้คำนิยามและกฎการตรวจจับที่กล่าวมาตั้งแต่หัวข้อที่ 2.7 ในการพัฒนาโปรแกรม

## 2.11 สรุป

ในบทนี้ได้กล่าวถึง ทฤษฎีและหลักการต่าง ๆ ที่เกี่ยวข้องกับการวิจัยอาทิเช่น ตัวอย่างการโจมตีระบบคอมพิวเตอร์ การตรวจจับการบุกรุก ค่าประจำตัวของผู้ใช้บนระบบปฏิบัติการยูนิกซ์ system call บนระบบปฏิบัติการยูนิกซ์ การนิยามสถานะของโปรเซส รวมทั้งกฎที่ใช้สนับสนุนในการตรวจจับ ซึ่งสรุปได้ว่า การพิจารณาพฤติกรรมว่าเข้าข่ายการบุกรุกหรือเป็นพฤติกรรมที่ผิดปกติหรือไม่นั้น สามารถพิจารณาได้จาก การเปลี่ยนแปลงสถานะซึ่งพิจารณาได้จากค่า UID EUID GID และ EGID ที่ติดมากับโปรเซส โดยที่สถานะที่ควรเฝ้าระวังคือสถานะที่มีสิทธิพิเศษ ค่า system call ที่ถูกเรียกใช้ในขณะนั้นจะถูกพิจารณาตามกฎที่ใช้สนับสนุนการตรวจจับ ในหัวข้อถัดไปจะกล่าวถึงการวิเคราะห์ ออกแบบและพัฒนาโปรแกรม โดยใช้หลักการการวิเคราะห์สถานะของโปรเซส การเรียกใช้ system call และกฎที่ใช้สนับสนุนการตรวจจับ