ระบบรักษาความปลอดภัยอิงกฎสำหรับยูนิกซ์

RULE-BASE UNIX SECURITY PROTECTION SYSTEM

อนิราช มิ่งขวัญ
ANIRACH MINGKHWAN

วิทยานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิทยาศาสตรมหาบัณฑิต
สาขาวิชาวิทยาการคอมพิวเตอร์และเทคโนโลยีสารสนเทศ
บัณฑิตวิทยาลัย
สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง
พ.ศ. 2542
ISBN 974-622-435-2

ระบบรักษาความปลอดภัยอิงกฎสำหรับยูนิกซ์

RULE – BASE UNIX SECURITY PROTECTION SYSTEM

อนิราช มิ่งขวัญ

ANIRACH MINGKHWAN

วิทยานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิทยาศาสตรมหาบัณฑิต
สาขาวิชาวิทยาการคอมพิวเตอร์และเทคโนโลยีสารสนเทศ
บัณฑิตวิทยาลัย
สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง
พ.ศ. 2542

ISBN 974-622-435-2

RULE – BASE UNIX SECURITY PROTECTION SYSTEM

ANIRACH MINGKHWAN

A THESIS SUBMITTED IN PARTIAL FULFILLMENT
OF THE REQUIREMENT FOR THE DEGREE OF
MASTER OF SCIENCE IN COMPUTER SCIENCE
AND INFORMATION TECHNOLOGY
SCHOOL OF GRADUATE STUDIES
KING MONGUT'S INSTITUTE OF TECHNOLOGY LADKRABANG
1999

ISBN 974-622-435-2

หัวข้อวิทยานิพนธ์	ระบบรักษาความปลอดภัยแบบอิงกฎสำหรับยูนิกซ์
นักศึกษา	นายอนิราช มิ่งขวัญ
รหัสประจำตัว	35628019
ปริญญา	วิทยาศาสตรมหาบัณฑิต
สาขาวิชา	วิทยาการคอมพิวเตอร์และเทคโนโลยีสารสนเทศ
พ.ศ.	2542
อาจารย์ผู้ควบคุมวิทยานิพนธ์	อาจารย์สุรสิทธิ์ วรรณไกรโรจน์
อาจารย์ผู้ควบคุมวิทยานิพนธ์ร่วม	ผศ.ดร.เอื้อน ปิ่นเงิน

บทคัดย่อ

งานวิจัยฉบับนี้ได้ทำการศึกษาค้นคว้าเพื่อนำเสนอแนวทางในการพัฒนาและใช้งานโปรแกรม Agent ในระบบ Automatic Intrusion Detection System (AIDS) ของระบบ UNIX ได้อย่างไร ทั้งในด้านทฤษฎีของการทำงาน สถาปัตยกรรม และสภาพแวดล้อมที่สนับสนุนการทำงานของระบบ

ระบบรักษาความปลอดภัยแบบอิงกฎสำหรับยูนิกซ์นี้ เป็นระบบที่ทำงานใช้คำสั่งพื้นฐานของระบบ UNIX มาประกอบกันขึ้นเป็นชุดคำสั่ง โดยจะมีโปรแกรมคอยตรวจสอบและจ่ายงานให้กับแต่ละโปรแกรมย่อยซึ่งจะวิเคราะห์และตอบโต้กับสถานการณ์ที่ได้รับมอบหมายแบบอัตโนมัติตามความรู้ที่ได้รับจากผู้ควบคุมระบบ

จากผลที่ได้จะเห็นได้ว่าเราสามารถที่จะใช้รูปแบบการทำงานของ Agent เพื่อปรับปรุงการทำงานของระบบ Automatic Intrusion Detection System ในระบบ UNIX ได้

Thesis Title	Rule – Base UNIX Security Protection System
Student	Mr.Anirach Mingkhwan
Student ID.	35628019
Degree	Master of Science
Programme	Computer Science and Information Technology
Year	1999
Thesis Advisor	Mr.Surasit Vannakrairojn
Thesis Co-Advisor	Asst. Prof. Dr.Ouen Pin-ngern

ABSTRACT

The major research problem in this paper is how to use software agents to implement an AIDS in UNIX operating system. Both theoretical foundation and an architecture for the implementation of such a system shall be presented.

Rule-Base UNIX Security protection is the system that uses basic command of UNIX system. The system has the control module to assign a task of any happening event to Agent that can analyst and act autonomously by using knowledge that provide by system administration.

The results of this research show that we can used Agent to improve the AIDS environment of UNIX system.

กิตติกรรมประกาศ

ในที่สุดวันนี้ก็มาถึง วิทยานิพนธ์ฉบับนี้สำเร็จลุล่วงได้ด้วยความเข้าใจและคำปรึกษาในการแก้ปัญหาดีจาก อาจารย์สุรสิทธิ์ วรรณไกรโรจน์ และ ผศ.ดร.เอื้อน ปิ่นเงิน ขอบคุณท่านอาจารย์ทั้งสองที่เข้าใจ ให้โอกาส และให้คำปรึกษาที่ดีเสมอมา ขอบคุณ บิดา-มารดาผู้ให้กำเนิด นี่เป็นอีกโอกาสหนึ่งที่ได้ตอบแทนพระคุณ ขอบคุณเพื่อน-น้อง-พี่ ที่คอยเป็นกำลังใจถามไถ่ช่วยเหลือและให้กำลังใจ ขอบคุณเจ้าหน้าที่ของคณะที่คอยติดตามส่งข่าว ขอบคุณจดหมายฉบับเล็กๆ ก่อนไปออสเตรเลียจากผู้ที่ห่วงใย คอยเตือนสติให้ต้องทำให้สำเร็จ ขอบคุณคนรู้ใจที่คอยเป็นกำลังใจและอยู่เคียงข้างเสมอ

อนิราช มิ่งขวัญ

สารบัญ

หน้า

สารบัญ(ต่อ)

หน้า

สารบัญ(ต่อ)

หน้า

สารบัญตาราง

สารบัญภาพ

ภาพที่ **หน้า**

บทที่ 1
บทนำ

1.1 ความเป็นมาและความสำคัญของปัญหา

ระบบปฏิบัติการ UNIX เป็นระบบปฏิบัติการที่ได้รับความนิยม และใช้งานกันอย่างแพร่หลายในปัจจุบัน โดยเฉพาะเมื่อเครื่องคอมพิวเตอร์ทั่วโลกเชื่อมกันเป็นเครือข่ายระดับชาติหรือที่รู้จักกันในชื่อของ Internet เมื่อเราสามารถเชื่อมต่อเข้ากับระบบเครือข่ายและรับข้อมูลข่าวสารได้จากทั่วโลกนั่นก็หมายความว่าผู้ใช้จากทั่วโลกก็สามารถที่จะเข้าถึงระบบของเราเช่นเดียวกัน และถ้ามองในแง่ร้ายก็หมายถึงผู้ที่ต้องการจะเจาะเข้าสู่ระบบของเราก็สามารถนั่งอยู่ที่ใดของโลกก็ได้

ถึงแม้ว่าผู้ออกแบบระบบ UNIX จะกล่าวว่าระบบ UNIX ถูกออกแบบมาโดยไม่ได้คำนึงถึงระบบความปลอดภัยก็ตาม [Richie,1986] แต่ในทางปฏิบัติ ผู้บริหารระบบสามารถที่จะควบคุมระดับความปลอดภัยได้ในระดับที่ตนเองต้องการ [Grampp, Morris, 1987] แต่เนื่องจากระบบ UNIX เป็นระบบที่ซับซ้อน ผู้จัดการระบบอาจมองข้ามในบางจุดไป โดยเฉพาะผู้ที่ยังไม่มีประสบการณ์เพียงพอ

งานวิจัยฉบับนี้จะนำเสนอแนวทางในการพัฒนาระบบตรวจสอบความไม่น่าไว้วางใจทางด้านความปลอดภัยของระบบยูนิกซ์ เพื่อให้ผู้ที่เป็นผู้บริหารระบบสามารถที่จะตรวจสอบความปลอดภัยของระบบได้ตลอดเวลาและสามารถแก้ปัญหาของระบบที่เกิดขึ้นได้อย่างทันท่วงที โดยใช้เทคโนโลยีของ Intelligence Agent

1.2 วัตถุประสงค์ของการศึกษา

1. เพื่อศึกษาแนวทางในการสร้างระบบการตรวจสอบความไม่น่าไว้วางใจและควบคุมความปลอดภัยของระบบ UNIX เพื่อให้สามารถทำการตอบโต้, ป้องกันการบุกรุก หรือการใช้งานที่ไม่ถูกต้องได้โดยอัตโนมัติ
2. เพื่อศึกษารูปแบบและความเหมาะสมของการทำงานของ Agent กับระบบ UNIX ว่ามีความเหมาะสมและสามารถนำมาใช้งานได้หรือไม่อย่างไร
3. เพื่อศึกษาแนวทางในการพัฒนาระบบรักษาความปลอดภัยที่เป็นแบบรวมศูนย์โดยมีการวิเคราะห์สถานการณ์ และการตอบโต้กับเหตุการณ์ ต่างๆ เป็นแบบขนาน

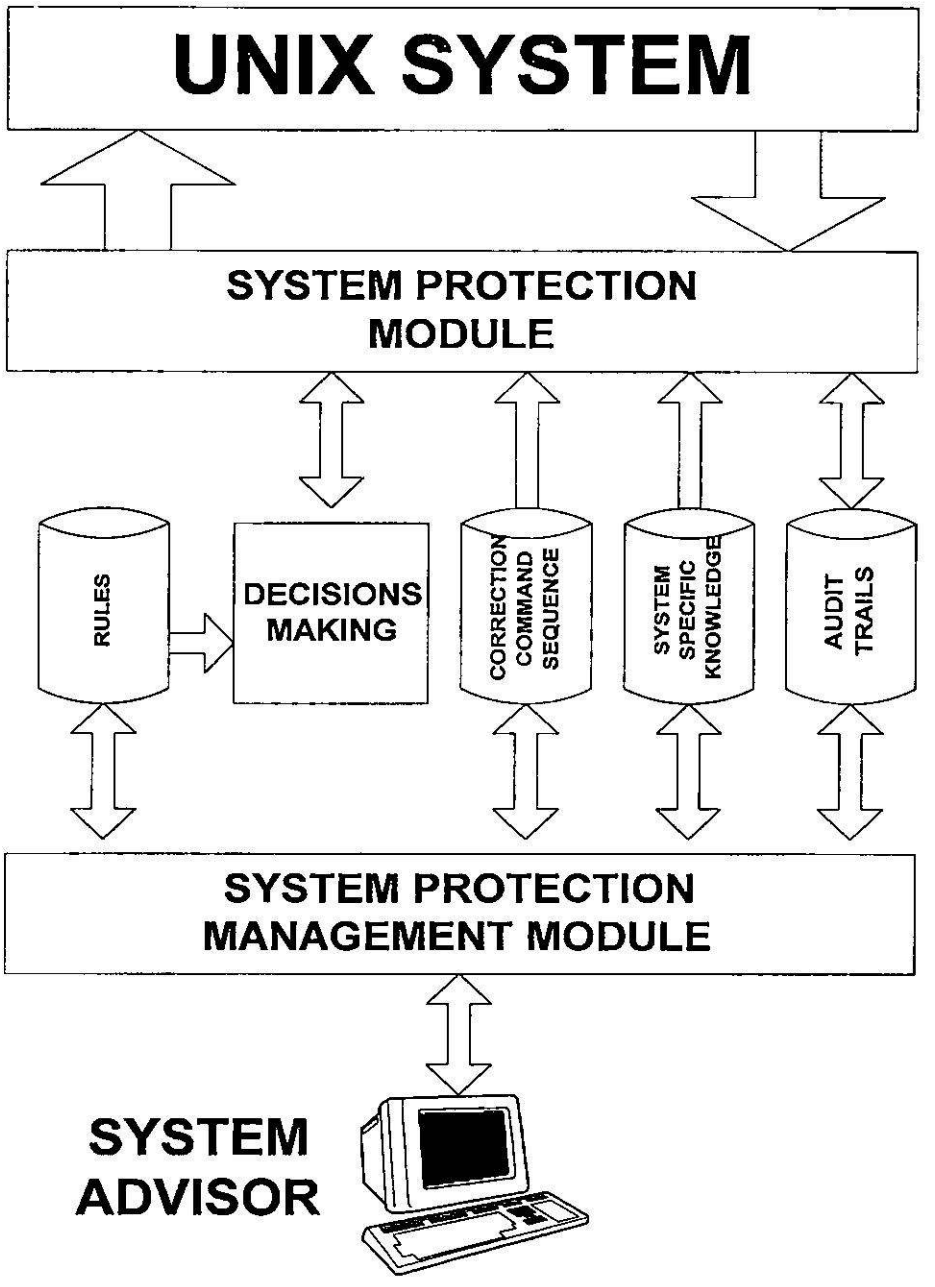
4. เพื่อเป็นแนวทางในการลดภาระหน้าที่ และปริมาณงานของบุคลากร หรือผู้เชี่ยวชาญที่ต้องคอยดูแลระบบ UNIX ซึ่งมีความยุ่งยากและซับซ้อน
5. เพื่อให้ได้ระบบ UNIX ที่มีความปลอดภัยและสามารถให้ผู้ใช้ใช้งานได้อย่างมั่นใจ
6. เพื่อเป็นแนวทางในการเริ่มต้นสำหรับผู้ที่จะต้องดูแลในเรื่องความปลอดภัยของระบบ UNIX และการพัฒนาระบบสำหรับ การตรวจสอบ ป้องกันและตอบโต้ในเรื่องของความปลอดภัยสำหรับระบบ UNIX รวมทั้งเป็นข้อมูลสำหรับผู้สนใจต่อไป

1.3 ทฤษฎีหรือแนวความคิดที่ใช้ในการศึกษา

ในงานวิจัยนี้จะเป็นการนำเสนอรูปแบบที่จะปรับปรุงการทำงานของการป้องกันระบบในด้านความปลอดภัยของระบบที่เรียกว่า "ระบบการตรวจจับการบุกรุก (Intrusion Detection System :IDS)" โดยใช้เทคโนโลยีใหม่ทางด้านปัญญาประดิษฐ์ (Artificial Intelligence :AI) ที่เรียกว่าโปรแกรม Agent (Software Agent) หรือที่เรียกกันทั่วๆ ไปว่า softbot เพื่อพัฒนาให้เป็นระบบตรวจจับการบุกรุกแบบอัตโนมัติ (Automated Intrusion Detection System: AIDS) ซึ่งมี Agent ทำหน้าที่ในการตรวจสอบและตอบโต้กับความไม่น่าไว้วางใจรูปแบบต่างๆ ที่เกิดขึ้นในระบบ โดยงานวิจัยนี้จะเป็นการพัฒนาระบบตัวอย่างขึ้นใช้ระบบ UNIX ซึ่งจะแบ่งส่วนการทำงานของระบบออกเป็น 4 ส่วนคือ

1. ส่วนของการจัดการและเก็บรวบรวมข้อมูลที่เกี่ยวกับความปลอดภัยของระบบ UNIX เพื่อใช้เป็น ข้อมูลในการตัดสินใจสำหรับการป้องกันและตอบโต้
2. ส่วนของการดักจับและบันทึกเหตุการณ์ที่ไม่น่าไว้วางใจ
3. ส่วนของการวิเคราะห์ความไม่น่าไว้วางใจ
4. ป้องกันและตอบโต้กับสภาวะการณ์ต่าง ๆ

ซึ่งโครงสร้างของส่วนการทำงานต่างๆ ของระบบจะถูกแบ่งออกเป็นส่วนประกอบย่อยๆ ดังแสดงในภาพที่ 1.1



ภาพที่ 1.1 แสดงโครงสร้างโดยรวมของระบบ

1.4 ขอบเขตของการศึกษา

ศึกษาถึงแนวทางในการพัฒนาระบบ IDS ให้เหมาะสมกับเทคโนโลยีในปัจจุบันโดยนำเอาแนวความคิดของระบบ Intelligence Agent มาประยุกต์เพื่อนำเสนอแนวทางและรูปแบบวิธีการทำงานของระบบ IDS รูปแบบใหม่ โดยจะทำการทดลองโดยการจำลองรูปแบบการทำงานของ

ระบบที่นำเสนอ เพื่อให้เห็นความเป็นไปได้ในการนำมาใช้งานกับระบบ UNIX เพื่อลดภาระงานของผู้บริหารระบบ ในการคอยติดตามตรวจสอบความไม่น่าไว้วางใจในด้านความปลอดภัย และสามารถตอบโต้กับสถานการณ์ที่ไม่น่าไว้วางใจต่างๆ แบบอัตโนมัติ ซึ่งมีขอบเขตของระบบที่จะทำการวิจัยดังนี้คือ

1. งานวิจัยนี้เป็นการศึกษาเพื่อหาแนวทางในการจัดทำต้นแบบของระบบรักษาความปลอดภัยที่สามารถวิเคราะห์และตอบโต้ การกระทำที่ไม่ปลอดภัยต่อระบบ UNIX
2. ทำการพัฒนาระบบตามรูปแบบที่ได้นำเสนอ และทดสอบการทำงานของส่วนประกอบต่างๆ ของระบบเพื่อให้เห็นถึงความเป็นไปได้ของระบบที่นำเสนอ ว่าสามารถใช้งานได้

1.5 ขั้นตอนของการศึกษา

1. ศึกษาการทำงานของระบบตรวจจับการบุกรุก (Intrusion Detection System :IDS) ต่างๆ ที่มีอยู่ และปัญหาของการพัฒนาระบบเหล่านั้น
2. ศึกษาทฤษฎี โครงสร้าง และการทำงาน ของโปรแกรม Agent ประเภทต่างๆ เพื่อหาแนวทางที่เหมาะสมสำหรับการพัฒนาเป็นระบบ IDS
3. ศึกษาข้อมูลที่สามารถใช้ในการตรวจสอบความปลอดภัยของระบบ UNIX และข้อจำกัดของระบบ IDS
4. ออกแบบ Architecture สำหรับส่วนต่างๆ ของระบบ จัดทำผังการทำงานโดยรวมและความสัมพันธ์กันของระบบต้นแบบ
5. จัดทำโปรแกรมเพื่อทดสอบการทำงานในส่วนต่างๆ ตามรูปแบบที่นำเสนอว่าสามารถทำงานได้จริงตามนั้นหรือไม่
6. สรุปผลการวิจัย ประเมินผล และเสนอแนะแนวทางในการวิจัยต่อไป

1.6 ประโยชน์ที่คาดว่าจะได้รับจากงานวิจัย

1. ได้ทราบถึงลักษณะและแนวทางของการจัดทำระบบการตรวจจับการบุกรุกสำหรับระบบ UNIX, การรักษาความปลอดภัยที่มีอยู่ในระบบ UNIX รวมทั้งจุดบกพร่องต่างๆ ของระบบเพื่อเป็นประโยชน์ต่อผู้ที่ต้องรับผิดชอบและดูแลความปลอดภัยของระบบ UNIX

2. ได้ทราบถึงโครงสร้างและรูปแบบในการทำงานของโปรแกรมประเภท Intelligence Agent ซึ่งสามารถใช้งานเป็นระบบ ตรวจจับการบุกรุกในการใช้งานระบบ UNIX แบบอัตโนมัติ
3. เป็นแนวทางในการพัฒนาระบบตรวจจับการบุกรุกที่เกี่ยวข้องกับความปลอดภัยที่มีประสิทธิภาพและทันต่อเหตุการณ์

บทที่ 2
ทฤษฎีและงานวิจัยที่เกี่ยวข้อง

2.1 ระบบตรวจจับการบุกรุก (Intrusion Detection System)

ความหมายของคำว่าการบุกรุก (Intrusion) ในระบบคอมพิวเตอร์จะหมายถึงการที่บุคคลใดบุคคลหนึ่งพยามที่จะเจาะเข้าสู่ระบบ หรือบุคคลที่เป็นผู้ใช้ในระบบอยู่แล้วที่ใช้งานในรูปแบบที่ไม่ได้รับอนุญาต ซึ่งการตีความหมายของการกระทำเหล่านี้จะขึ้นอยู่กับแต่ละระบบ

ระบบการตรวจจับการบุกรุก (Intrusion Detection System) หรือที่เรียกย่อๆ ว่า IDS มีวัตถุประสงค์ในการตรวจสอบการเจาะเข้าสู่ระบบหรือการใช้งานที่ไม่ได้รับอนุญาตของผู้ใช้ โดยระบบ IDS จะทำงานอยู่ในระบบคอมพิวเตอร์ตลอดเวลาในลักษณะการทำงานแบบเบื้องหลัง (Background process) และจะทำการแจ้งเตือนเมื่อตรวจสอบพบความไม่น่าไว้วางใจ ซึ่งรูปแบบของการตรวจสอบหรือการแจ้งเตือนก็จะขึ้นอยู่กับการติดตั้งและการกำหนดของผู้ที่ดูแลหรือบริหารระบบนั้นๆ

2.2 ประเภทของผู้บุกรุก (Intruders)

ประเภทของผู้ไม่น่าไว้วางใจหรือผู้ที่จะเป็นผู้บุกรุกจะแบ่งออกเป็น 2 กลุ่มใหญ่ๆ คือ

ผู้บุกรุกจากภายนอก(Outside Intruders)

คนส่วนใหญ่เข้าใจว่าผู้ที่พยามจะเจาะข้อมูลความลับของตนจะเป็นบุคลภายนอก และจะมีการป้องกันการบุกรุกของบุคคลกลุ่มนี้อย่างเข้มแข็ง ดังนั้นผู้ที่จัดอยู่ในกลุ่มนี้จะเป็นพวกนักเจาะระบบที่ต้องตั้งใจอย่างสูง (hackers)

ผู้บุกรุกที่อยู่ในระบบ(Inside Intruders)

จากการศึกษาของ FBI เปิดเผยว่าผู้ที่บุกรุกและประสบความสำเร็จในการเจาะเข้าสู่ระบบ 80% เป็นผู้ที่อยู่ในองค์กร ถ้าไตร่ตรองให้ดีจะเห็นว่าผู้ที่อยู่ในองค์กรจะทราบเกี่ยวกับรูปแบบของระบบ ที่อยู่ของข้อมูลที่มีความสำคัญและรูปแบบของการป้องกันต่างๆ จึงง่ายกว่าผู้บุกรุกจากภายนอก

จากเหตุผลที่มาตรการในการป้องกันต่างๆ ซึ่งกำหนดขึ้นเพื่อใช้ป้องกันข้อมูลภายในจากผู้ประสงค์ร้ายภายนอก แต่ผู้ที่ประสบความสำเร็จในการเจาะระบบส่วนใหญ่มาจากบุคคล

ภายใน ดังนั้นระบบตรวจจับที่ดีจึงจำเป็นที่จะต้องตรวจพบความไม่น่าไว้วางใจทั้งสองรูปแบบคือจากผู้ที่พยายามเจาะภายนอกและผู้ที่เป็นผู้ใช้ภายใน

2.3 แนวทางในการจัดระบบความปลอดภัย (Security Policy)

แนวทางในการจัดระบบความปลอดภัย จะเป็นการนิยามว่าอะไรคือสิ่งที่อนุญาต และอะไรคือไม่อนุญาต ซึ่งมีแนวทางในการกำหนดอยู่สองรูปแบบคือ

1. Probative คือการห้ามทุกอย่างที่ไม่ได้ระบุว่าอนุญาต
2. Permissive คือทุกอย่างที่ไม่ได้ระบุว่าห้ามหมายถึงอนุญาต

โดยทั่วไปหน่วยงานที่เข้มงวดเกี่ยวกับความปลอดภัยมักจะใช้รูปแบบแรก การกำหนดมาตรการความปลอดภัยต่างๆ จะระบุอย่างชัดเจนว่าสามารถทำอะไรได้บ้างในระบบ และทุกส่วนที่ไม่ได้ระบุไว้จะหมายถึงห้าม ระบบในรูปแบบนี้มักจะใช้กับการรักษาความปลอดภัยทางด้านความมั่นคงของประเทศหรือในหน่วยทหาร

การใช้งานในรูปแบบที่สองเป็นการใช้งานระบบคอมพิวเตอร์เพื่อให้เกิดประโยชน์สูงสุด โดยทุกคนสามารถใช้งานระบบได้อย่างเต็มที่ ทุกเรื่องสามารถยอมรับได้ถ้างานสำเร็จ ไม่มีผู้ใดเสียหายและทุกคนมีความสุขในการใช้งาน ซึ่งการใช้งานในรูปแบบนี้ไม่ค่อยจะประสบความสำเร็จนักในสังคมปัจจุบัน เนื่องจากผู้ใช้ไม่ได้เข้ามาสู่ระบบและเรียนรู้ในการเคารพสิทธิของผู้ใช้อื่น การแข่งขันกันทำให้ผู้ใช้ลืมจรรยาบรรณที่ดีในการใช้งานระบบร่วมกัน รวมไปถึงการละเมิดสิทธิในข้อมูลของผู้ใช้อื่น ยิ่งไปกว่านั้นในบางกลุ่มของงานยังอาจมีการกลั่นแกล้งหรือการจารกรรมเพื่อความอยู่รอดซึ่งมีแนวโน้มสูงขึ้นในสังคมยุคข่าวสาร

ผู้ใช้ส่วนใหญ่จะใช้ระบบตามกฎของการใช้งานที่ระบุไว้ ซึ่งจะกำหนดให้ต้องเคารพในความเป็นส่วนตัวของผู้ใช้อื่น เมื่อการใช้งานของผู้ใช้ตั้งอยู่บนความไว้วางใจกัน ซึ่งง่ายต่อการเปลี่ยนแปลงและถูกปองร้ายจากผู้ไม่หวังดี

จากตัวอย่างทั้งหมดที่กล่าวมา ระบบที่ดีจึงต้องมีการตั้งกฎเกณฑ์ในการใช้งานต่างๆ และบังคับใช้ เนื่องจากไม่มีประโยชน์ที่จะมีกฎแล้วไม่สามารถบังคับได้ และต้องแสดงให้เห็นว่ากฎเหล่านั้นใช้บังคับได้จริง ดังที่มีผู้กล่าวว่า "Justice must be done, and it must be seen to be done"

2.4 องค์ประกอบของความปลอดภัยของระบบ

องค์ประกอบที่ต้องคำนึงถึงในการออกแบบความปลอดภัยของระบบคอมพิวเตอร์ มีดังต่อไปนี้

1. Availability - ระบบจะต้องพร้อมเสมอเมื่อผู้ใช้ต้องการใช้งาน
2. Utility - ระบบและข้อมูลที่เก็บอยู่ในระบบต้องมีประโยชน์
3. Integrity - ระบบและข้อมูลจะต้องครบถ้วนสมบูรณ์และเข้าใจได้
4. Authenticity - ระบบจะต้องสามารถตรวจสอบว่าผู้ใช้เป็นใครได้ และผู้ใช้ก็สามารถตรวจสอบว่าระบบเป็นระบบที่ต้องการใช้หรือไม่ด้วย
5. Confidentially – ข้อมูลที่เป็นความลับต้องรู้ได้เฉพาะผู้ที่เป็นเจ้าของหรือผู้ที่เจ้าของข้อมูลอนุญาตเท่านั้น
6. Possession - ผู้ที่เป็นเจ้าของระบบต้องสามรถควบคุมได้ การเสียการควบคุมระบบให้กับผู้ที่ประสงค์ร้ายเท่ากับว่าข้อมูลของผู้ใช้ทั้งระบบตกอยู่ในอันตราย

2.5 ประเภทของการบุกรุก (Intrusion Classification)

ก่อนที่เราจะกล่าวถึงการตรวจจับการบุกรุก สิ่งแรกที่ควรรู้คือนิยามของคำว่าการบุกรุก (Intrusion) ซึ่งการบุกรุกจะไปเกี่ยวข้องกับข้อกำหนดต่างๆ ในเรื่องของความปลอดภัย เช่นอะไรที่ห้ามหรืออะไรที่อนุญาตให้สามารถทำได้ในระบบ

การบุกรุก (Intrusion) อาจจะนิยามได้ดังนี้ [HeadyLugerEtAl:90]

"การกระทำใดๆ ที่พยามที่จะล่วงล้ำ integrity, confidentiality หรือ availability ของ ทรัพยากรในระบบ"

ซึ่งสามารถจำแนกได้เป็น 2 ประเภทคือ

1. Misuse intrusions - เป็นการตั้งใจเจาะเข้าสู่ระบบในจุดบกพร่องของระบบโดยมีการไตร่ตรองไว้แล้ว ซึ่งสามารถตรวจสอบได้โดยการติดตามดูรูปแบบการทำงาน

2. Anomaly intrusions - เป็นการตรวจจับการบุกรุกในรูปแบบของการใช้งานที่ผิดไปจากรูปแบบที่เคยใช้ ซึ่งสามารถตรวจสอบได้จากข้อมูลรูปแบบการใช้งานเดิมที่มีการบันทึกไว้

เนื่องจากการตรวจจับการบุกรุกของการเจาะเข้าสู่ระบบแบบ Misuse intrusions มีรูปแบบและวิธีการที่แน่นอนฉะนั้นการตรวจจับสามารถทำได้โดยการเทียบข้อมูลที่จับบันทึกไว้ (audit-trail) กับรูปแบบวิธีในการเจาะที่บันทึกไว้ เช่น การสร้างโปรแกรม setuid สามารถตรวจสอบได้โดยดูจากผลที่บันทึกไว้ของการทำ System calls ซึ่งสามารถทำได้โดยใช้วิธีการ pattern matching เช่นในงานวิจัยของ [KumarSpafford:94]

ความไม่น่าไว้วางใจที่เกิดจากความผิดไปจากรูปแบบเดิม สามารถตรวจสอบได้จากการเปรียบเทียบกับรูปแบบปกติของระบบ ต้นแบบของระบบสร้างเป็น metrics จากข้อมูลการทำงานของระบบ metrics ในที่นี้ [Denning:87]

"เป็นการสุ่มตัวแปร x ที่แสดงปริมาณที่เพิ่มขึ้นในช่วงเวลาที่กำหนด"

ซึ่ง metrics นี้จะได้จากการคำนวณการทำงานเฉลี่ยของ CPU, จำนวนครั้งที่ติดต่อกับเครือข่ายต่อนาที หรือปริมาณของ process ต่อผู้ใช้ ความผิดปกตินี้อาจเป็นผลจากการเจาะเข้าสู่ระบบดังนั้นถ้าเรามี metrics ของระบบปกติเราจะสามารถตรวจสอบถึงความผิดปกติได้ [Denning:87]

"การวิเคราะห์การเจาะเข้าสู่ระบบซึ่งทำให้การทำงานของระบบผิดไปจากรูปแบบปกติจะถูกพบเมื่อเปรียบเทียบกับรูปแบบการทำงานปกติที่บันทึกไว้"

การตรวจจับการบุกรุกจากรูปแบบที่ผิดไปจากปกตินี้เป็นเรื่องที่ค่อนข้างยากเนื่องจากไม่มีรูปแบบที่แน่นอนในการตรวจจับจึงต้องใช้ระบบแบบ "fuzzy" เข้ามาช่วยในการวิเคราะห์ ระบบเป้าหมายควรจะสามารถทำงานได้เช่นเดียวกับการตรวจสอบของมนุษย์ซึ่งคอยตรวจสอบระบบว่าเกิดความไม่น่าไว้วางใจขึ้นหรือไม่ และสามารถที่จะละเลยหรือยอมรับความผิดปกติที่เกิดจากการทำงานของผู้ใช้ที่ได้รับอนุญาตได้

2.6 การตรวจจับการบุกรุก (Intrusion Detection)

ระบบตรวจจับการบุกรุกหลายๆ ระบบทำงานโดยการวิเคราะห์ข้อมูลการทำงานของระบบปฏิบัติการที่ได้มีการบันทึกไว้ (audit-trail) ข้อมูลนี้จะเป็นการบันทึกการทำงานของระบบในช่วงเวลาต่างๆ ซึ่งเป็นข้อมูลที่สามารถใช้งานได้ง่ายและมีอยู่ในเกือบทุกระบบ จากข้อสังเกตนี้ ระบบการตรวจจับการบุกรุกจะทำการวิเคราะห์ข้อมูลการทำงานของระบบและสร้าง metrics ของสภาพการทำงานของระบบในรูปแบบปกติ และทำให้สามารถตัดสินใจได้ว่าเมื่อใดที่ระบบมีการทำงานที่แปลกไป

ระบบ IDS สามารถที่จะคอยตรวจสอบการทำงานของระบบด้วยก็ได้ โดยการเก็บข้อมูลสถิติที่ได้รับการกำหนดไว้ซึ่งอาจจะได้จากการทำงานของ CPU, disk I/O, หน่วยความจำ, การทำงานของผู้ใช้, จำนวนที่ Login ไม่สำเร็จ เป็นต้น เนื่องจากข้อมูลเหล่านี้เมื่อมีความผิดปกติเกิดขึ้นระบบ IDS จะตรวจสอบได้ทันที

- **คุณลักษณะของระบบตรวจจับการบุกรุกที่ดี** (Characteristics of a Good Intrusion Detection System)

ระบบการตรวจจับการบุกรุกที่ดีควรมีคุณสมบัติดังต่อไปนี้ ซึ่งจะขึ้นอยู่กับกลไกที่ระบบนั้นพัฒนาขึ้น

1. จะต้องทำงานอยู่ตลอดเวลาโดยไม่ต้องมีการควบคุมของผู้ดูแลระบบ ในลักษณะที่เป็นการทำงานอยู่เบื้องหลัง (background process) แต่ผู้ที่ควบคุมระบบจะต้องสามารถตรวจสอบการทำงานจากภายนอกได้ตลอดเวลา
2. จะต้องเป็นระบบที่เป็น fault tolerant ในความหมายที่ว่าสามารถที่จะทำงานต่อไปได้โดยไม่ต้องทำการสร้างฐานความรู้ (knowledge-base) ทุกครั้งที่เริ่มระบบ
3. ระบบจะต้องมีการตรวจสอบตนเองเพื่อไม่ให้ถูกลบ ทำลาย หรือแทนที่ด้วยโปรแกรมอื่นได้
4. ต้องใช้ทรัพยากรของระบบให้น้อยที่สุดเท่าที่จะทำได้ เนื่องจากระบบที่ทำให้การทำงานของเครื่องช้าลงจะไม่ได้รับการยอมรับ
5. จะต้องมีการตรวจสอบการทำงานที่ผิดไปจากรูปแบบปกติ
6. จะต้องสามารถปรับแก้ให้เข้ากับระบบได้ง่าย เนื่องจากระบบคอมพิวเตอร์แต่ละระบบมีรูปแบบคำสั่งและกลไกในการป้องกันระบบที่แตกต่างกัน

7. ระบบจะต้องสามารถปรับการทำงานให้สอดคล้องกับการเปลี่ยนแปลงของระบบได้ เช่นเมื่อมีการติดตั้งอุปกรณ์ใหม่หรือลงโปรแกรมใหม่ ระบบก็จะต้องเปลี่ยนแปลงข้อมูลตามไปด้วย
8. ต้องมีความชาญฉลาดในการวิเคราะห์

- **ความผิดพลาดจากการตรวจจับที่เกิดขึ้นในระบบสามารถที่จะจำแนกออกเป็นประเภทต่างๆ ได้ดังนี้คือ**
    - false positive คือการวิเคราะห์ว่ามีความไม่น่าไว้วางใจเกิดขึ้น ซึ่งแท้ที่จริงเป็นการทำงานโดยถูกต้องของผู้ใช้
    - false negative คือการวิเคราะห์ว่าเป็นการทำงานที่ถูกต้องทั้งๆ ที่เป็นการบุกรุกระบบและระบบปล่อยให้ผ่านไป
    - subversion errors คือการที่ผู้บุกรุกเปลี่ยนแปลงการทำงานของระบบ IDS ให้ไม่สามารถตรวจจับได้ว่ามีความผิดปกติเกิดขึ้น

ความผิดพลาดในทางบวก (false positive) จะทำให้ผู้ใช้ระบบ Intrusion Detection System ต้องคอยระบุให้ระบบละเลยการตรวจสอบผิดพลาด ซึ่งน่าจะมีการผิดพลาดเช่นนี้เกิดขึ้นให้น้อยที่สุด (อาจจะเป็นไปไม่ได้ที่จะไม่ให้เกิดเลย) ข้อพึงระวังคือถ้ามีการเกิดขึ้นของ false positive มากๆ ผู้ใช้ระบบอาจจะระบุให้ระบบตรวจจับละเลยความไม่น่าไว้วางใจหรือการบุกรุกที่เกิดขึ้นจริงไปก็ได้

ความผิดพลาดในทางลบ (false negative) เกิดขึ้นเมื่อระบบไม่ทำอะไรเลยแม้จะตรวจสอบพบการบุกรุก ซึ่งเป็นความผิดพลาดที่ร้ายแรงกว่าในรูปแบบแรกเนื่องจากทำให้ผู้ที่ตรวจสอบระบบเข้าใจว่าการบุกรุกนั้นเป็นการกระทำที่ถูกต้องโดยปล่อยให้เหตุการณ์เหล่านั้นเกิดขึ้นได้ ซึ่งเป็นสภาวะที่แย่กว่าก่อนที่จะมีระบบตรวจจับ

Subversion errors เป็นความผิดพลาดที่ซับซ้อน และมักจะทำในรูปแบบของความผิดพลาดในทางลบ (false negative) โดยการใช้ความรู้ปรับเปลี่ยนการทำงานของระบบตรวจจับให้ไม่สามารถที่จะตรวจสอบการกระทำผิดที่กำลังจะทำได้

- **ลักษณะของระบบตรวจจับการบุกรุกจำแนกตามแหล่งที่มาของข้อมูล**
  - Host based - ใช้ข้อมูล audit ในการพิจารณาจาก Host เพียงเครื่องเดียว
  - MultiHost based - ใช้ข้อมูล audit ในการพิจารณาจาก Host หลายๆ เครื่อง
  - Network based - พิจารณาจากข้อมูลที่มีการส่งในเครือข่ายร่วมกับข้อมูล audit ใน host

ลักษณะของระบบตรวจจับการบุกรุกจำแนกตามรูปแบบของการกระทำ

การตรวจจับจากเหตุการณ์ผิดปกติ (anomaly detection model) ระบบตรวจจับจะคอยตรวจสอบการทำงานที่ผิดไปจากรูปแบบเดิมของการใช้งาน

การตรวจจับจากการใช้งานที่ไม่ได้รับอนุญาต (misuse detection model) ระบบจะทำการตรวจสอบการกระทำที่อาจจะส่อได้ว่าเป็นการทำงานที่ไม่น่าไว้วางใจตามรูปแบบของการเจาะเข้าสู่ระบบที่บันทึกไว้

2.7 ข้อมูลที่สามารถใช้ในการตรวจสอบความปลอดภัยในระบบ UNIX ได้

แนวคิดในการตรวจสอบความปลอดภัยในระบบ UNIX นั้นสามารถที่จะทำได้โดยการตรวจสอบจากข้อมูลที่ระบบทำการบันทึกไว้หรือที่เรียกว่า Audit trail โดยเราสามารถใช้ข้อมูลในการตรวจสอบการทำงานในระบบ UNIX ได้จาก 2 แหล่งคือ

- ข้อมูลทั่วไปที่เก็บไว้ (General log files)
- ข้อมูลที่ได้จากโปรแกรม SYSLOG (SYSLOG generated log files)

ซึ่งข้อมูลประเภท General log files ที่เก็บไว้สามารถที่จะจำแนกออกได้อีกดังนี้คือ

- แฟ้ม log แบบธรรมดา (Basic log files)
- แฟ้ม log เฉพาะสำหรับโปรแกรมบางตัว (Program specific log files)
- แฟ้ม log เฉพาะสำหรับผู้ใช้บางคน (User specific log files)

- **ข้อมูลทั่วไปที่เก็บไว้** (General log files)

แฟ้ม log **แบบธรรมดา** (Basic log files)

แฟ้มข้อมูลต่อไปนี้สามารถที่จะพบได้ภายใต้ directory /var/adm.

Lastlog

เป็นแฟ้มข้อมูลที่เก็บ record ของการใช้งานครั้งสุดท้ายของผู้ใช้ที่เข้ามาใช้ระบบ โดยเก็บข้อมูลดังต่อไปนี้คือ

- Timestamp
- Terminal line number

  Format : Binary

  To turn it on : Self

utmp & wtmp

utmp จะเก็บข้อมูลของผู้ใช้ทุกคนที่กำลังใช้งานอยู่ในระบบ ในขณะที่ wtmp จะเก็บข้อมูลการเข้าใช้และออกจากระบบของผู้ใช้ทุกคน โดยจะบันทึกข้อมูลเหล่านี้คือ

- Username
- Terminal line number
- Device name
- Process ID of the login shell
- Code that denotes the type of entry
- Exit status of the process
- Time that the entry was made

  Format : Binary

  To turn it on : Self

ซึ่งข้อมูลที่อยู่ในแฟ้ม utmp นี้จะถูกใช้งานโดยโปรแกรม who, whodo, w, และ finger ซึ่งจะรายงานข้อมูลของผู้ใช้ที่ใช้งานระบบอยู่

utmpx & wtmpx

แฟ้มข้อมูลนี้จะมีอยู่ในระบบ Solaris, IRIX และ ระบบปฏิบัติการที่เป็นมาตรฐาน SVR4 UNIX โดยเป็นการเพิ่มเติมข้อมูลไปจาก utmp & wtmp เดิม โดยเพิ่มเติมข้อมูลเหล่านี้คือ

- Session ID
- Unused bytes for future expansions
- Remote host name (for logins that originated over a network)

Format : Binary

To turn it on : Self

ข้อมูลที่อยู่ในแฟ้มข้อมูล wtmpx นี้จะถูกใช้งานโดย คำสั่ง last

loginlog

แฟ้มข้อมูลนี้จะพบในระบบ UNIX ที่เป็น System-V (รวมทั้ง Solaris) โดยจะบันทึกข้อมูลของการ login เข้าสู่ระบบที่ไม่สำเร็จ โดยแฟ้มข้อมูลนี้จะบันทึกข้อมูลเหล่านี้คือ

- User name
- Terminal line number
- Timestamp

Format : Text

To turn it on : Self

acct or pacct

ระบบ UNNIX สามารถที่จะเก็บข้อมูลคำสั่งเดี่ยวทุกคำสั่งที่ผู้ใช้ใช้งานได้ การเก็บข้อมูลประเภทนี้เรียกว่า process accounting โดยแฟ้มข้อมูล acct และ pacct นี้ สามารถที่จะใช้ในการตรวจสอบการเจาะเข้าสู่ระบบเพื่อดูว่าผู้ที่เจาะเข้ามานั้นทำอะไรบ้าง โดยมีข้อมูลที่บันทึกไว้ดังนี้คือ

- Command name
- User name
- Terminal line number
- Amount of CPU time used
- Time that the process exited
- Flags

Format : Binary

To turn it on : Execute /usr/lib/acct/startup

คำสั่งที่ใช้ดูข้อมูลของแฟ้มข้อมูลในแบบ text คือ คำสั่ง lastcomm หรือ acctcom

messages

แฟ้มข้อมูลนี้จะเก็บข้อมูลของทุกๆ ข้อความที่แสดงออกทาง system console แฟ้มข้อมูลนี้สามารถเรียกดูเพื่อใช้ประโยชน์ได้ง่ายเนื่องจากเป็น text ธรรมดา

Format : Text

To turn it on : Self

แฟ้ม log เฉพาะสำหรับโปรแกรมบางตัว (Program specific log files)

แฟ้มข้อมูลเหล่านี้จะพบได้ใน /var/adm นอกจากจะระบุเป็นอย่างอื่น

aculog

คำสั่ง tip และคำสั่ง UUCP ของ Berkeley version จะเก็บข้อมูลทุกครั้งของการหมุนโทรศัพท์ออก โดยข้อมูลที่เก็บไว้เป็นดังนี้คือ

- Account name
- Timestamp
- Entry in the /etc/remote file that was used to place the call
- Phone number dialed
- Actual device used
- Result of the call, success/failure

Format : Text

To turn it on : Self

sulog

แฟ้มข้อมูลนี้จะใช้ในการบันทึกข้อมูลทุกครั้งที่มีการเรียกใช้คำสั่ง su โดยเก็บข้อมูลดังนี้คือ

- Username
- Result of execution
- Terminal line number
- Timestamp

Format : Text

To turn it on : Set SULOG=/var/adm/sulog in the /etc/defaults/su file.

security

แฟ้มข้อมูลนี้จะบันทึกการทำงานที่พยามจะฝ่าฝืนการใช้ ระบบ UUCP ซึ่งจะพบได้ใน /var/spool/uucp/.Admin ข้อมูลที่น่าสนใจจะเป็นข้อมูล 2 ประเภทต่อไปนี้คือ

1. ข้อมูลที่ขึ้นต้นด้วย "xfer" ซึ่งจะเป็นการระบุการพยายามลักลอบในการถ่ายโอนข้อมูล ซึ่งจะเก็บข้อมูลดังนี้คือ
   - User on the requesting host
   - User on the destination host
   - Information to identify file name and size
   - Time stamp
2. ข้อมูลที่ขึ้นต้นด้วย "rexec" ซึ่งหมายถึงการพยามใช้คำสั่งที่ไม่ได้รับอนุญาต โดยเก็บข้อมูลเหล่านี้คือ
   - User on the request host
   - User on the destination host
   - Timestamp
   - The command and the options involved

     Format : Text

     To turn it on : Self

access_log

แฟ้มข้อมูลนี้จะปรากฏในระบบที่มีการใช้งาน NCSA HTTPD server สำหรับระบบ WWW ซึ่งปกติจะอยู่ใน /usr/local/etc/http/logs โปรแกรม HTTPD server จะระบุตำแหน่งของแฟ้มนี้ไว้ ซึ่งจะทำให้ทราบว่า ที่ใดบ้างที่เข้ามาและเอาข้อมูลใดไปบ้าง โดยเก็บข้อมูลเหล่านี้คือ

- Name of the remote computer that initiated the transfer
- Remote login name
- Remote user name
- Timestamp
- HTTP command that was executed
- Status code that was returned
- Number of bytes that were transferred

  Format : Text

  To turn it on : To be found

โปรแกรมหนึ่งที่ใช้ในการวิเคราะห์แฟ้มนี้คือ โปรแกรมที่ชื่อ getstats ซึ่งจะบอกได้ว่ามีใครที่เข้ามาใช้งานบ้าง, มาจากไหน, แฟ้มไหนที่ได้รับความสนใจสูง และ ข้อมูลที่น่าสนใจทางสถิติอื่นๆ

inetdlog

แฟ้มข้อมูลนี้จะทำการบันทึกทุกการติดต่อที่เป็น TCP โดยจะบันทึกข้อมูลเหล่านี้คือ

- Timestamp
- Remote host name
- Remote host's IP address
- Program name (telnet, finger, systat, etc.)
- Local port number

Format : Text

To turn it on : Launch inetd with the -t option

แฟ้มข้อมูลนี้จะอยู่ที่ /var/log

แฟ้ม log เฉพาะสำหรับผู้ใช้บางคน (User specific log files)

แฟ้มข้อมูลเหล่านี้มักจะพบที่ home directory ของผู้ใช้ ซึ่งผู้ใช้อาจจะทำการลบทิ้งก่อนที่จะออกจากระบบได้ ซึ่งสามารถแก้ไขได้โดยการทำ hard link ของแฟ้มข้อมูลนั้นไปยังส่วนของ disk ที่ผู้ใช้นั้นไม่สามารถเข้าถึงได้

.history

แฟ้มข้อมูลนี้จะบันทึกข้อมูลของทุกคำสั่งที่ผู้ใช้สั่งให้ทำงานจาก shell prompt ซึ่งอาจเป็นประโยชน์ในกรณีที่ต้องการสร้างการทำงานเรียนแบบการทำงานเดิม หรืออาจจะเป็นประโยชน์ในการพิจารณาร่วมกับ log file อื่นเพื่อตรวจสอบว่าผู้ที่บุกรุกเข้ามาทำอะไรที่ไม่น่าไว้วางใจกับระบบบ้าง

mail

ผู้ใช้สามารถที่จะกำหนดบันทึกการเข้าออกของ mail โดยข้อมูลการบันทึกนี้สามารถใช้ตรวจสอบได้ว่ามี mail ใดเข้า-ออก จากระบบบ้าง

.rhosts & .netrc

แฟ้มนี้เป็นการบันทึกข้อกำหนดของระบบเครือข่าย เพื่อให้สามารถใช้คำสั่งได้ง่าย ซึ่งอาจเป็นช่องทางให้ผู้ที่ประสงค์ร้ายเจาะเข้าสู่ระบบได้

- ข้อมูลที่ได้จากโปรแกรม SYSLOG (SYSLOG generated log files)

นอกจาก log file ที่กล่าวถึงในตอนต้นแล้วระบบ UNIX ยังมีโปรแกรมที่ทำการเก็บ log ทั่วไปที่เรียกว่า syslog ซึ่งสามารถกำหนดรูปแบบการทำงานได้

ระบบจะใช้ระบบ logging จากส่วนกลาง (centralized logging) ซึ่งเรียกใช้โปรแกรม /usr/sbin/syslogd (ในระบบของเรา) โปรแกรมที่ถูกกำหนด (รวมทั้งโปรแกรมของผู้ใช้) ที่ต้องการเก็บข้อมูล (log) จะต้องส่งข้อมูลให้ syslog ข้อความจะสามารถบันทึกลงได้หลายแฟ้ม, อุปกรณ์ หรือ เครื่องอื่นที่ระบุ ขึ้นอยู่กับตัวที่ส่งข้อความและความสามารถในการเป็น server

เมื่อเริ่มใช้งาน syslog โปรแกรมจะอ่านข้อกำหนดการทำงานจาก /etc/syslog.conf เพื่อที่จะดูว่าเหตุการณ์แบบใดบ้างที่จะต้องบันทึกลงแฟ้มข้อมูล

จากนั้นโปรแกรมจะคอยดักจับข้อความจาก 3 แหล่งด้วยกันคือ

- /dev/klogd : เป็น device พิเศษ ที่คอยอ่านข้อความจาก kernel
- /dev/log : UNIX domain socket, ใช้ในการอ่านข้อความซึ่งสร้างโดย process ในเครื่อง
- UDP port 514 : Internet domain socket, ใช้ในการอ่านข้อความที่ สร้างขึ้นจากเครื่องอื่นๆ ในระบบ LAN

โดยทั้งหมดนี้จะขึ้นอยู่กับการกำหนดในแฟ้ม configuration file ที่จะตัดสินใจว่าจะทำอย่างไรเมื่อได้รับข้อความต่างๆ

2.8 ปัญหาของการตรวจสอบความปลอดภัยในระบบ UNIX

การพัฒนาระบบการตรวจจับการบุกรุกที่เกี่ยวกับความปลอดภัยในระบบ UNIX (Intrusion Detection System) หรือที่เรียกว่าระบบ IDS เดิมใช้โปรแกรมเพียงโปรแกรมเดียวในการตรวจสอบความปลอดภัยของระบบซึ่งโปรแกรมเหล่านั้น มีปัญหาดังนี้คือ

1) มีปริมาณข้อมูลที่ต้องการตรวจสอบในแต่ละเหตุการณ์มีเป็นจำนวนมากทำให้ไม่สามารถเสร็จสิ้นการตรวจสอบข้อสงสัยได้ทันก่อนที่จะมีความผิดปกติใหม่เกิดขึ้น
2) โปรแกรมที่ใช้ในการตรวจสอบมีขนาดใหญ่ และ ใช้ทรัพยากรของระบบมาก
3) รูปแบบของระบบที่ใช้ตรวจสอบแบบเดิมมีลักษณะเป็นระบบที่ซับซ้อนปรับเปลี่ยนการทำงานได้ยาก และไม่ได้รวมเป็นระบบเดียว

2.9 การประยุกต์ใช้งานโปรแกรม Agent ในระบบ IDS

โปรแกรม Agent สามารถประยุกต์ใช้งานกับการตรวจจับการบุกรุกทางด้านความปลอดภัยของระบบ UNIX ได้เป็นอย่างดีเนื่องจากสามารถที่จะมอบหมายให้ Agent แต่ละตัวตรวจสอบแต่ละข้อสงสัยได้โดย Agent สามารถที่จะใช้กฎของการตรวจสอบร่วมกัน และสามารถที่จะทำงานตามข้อกำหนดของผู้บริหารระบบ (Rule) ได้ และเมื่อทำงานเสร็จสิ้นแล้วก็จะหายไปจากระบบทำให้ไม่สิ้นเปลืองทรัพยากรของระบบ

โดยแนวทางในการประยุกต์ใช้โปรแกรม Agent เพื่อตรวจสอบความปลอดภัยในระบบ UNIX สามารถทำได้ยึดหลักการออกแบบให้สนับสนุนการทำงานดังนี้คือ

1) ต้องเป็นโปรแกรมที่ไม่ใช้ทรัพยากรมาก
2) Agent ต้องมีขนาดเล็กเพื่อสามารถพัฒนาให้เคลื่อนที่ไปตรวจสอบเครื่องอื่นๆ ในระบบได้ในอนาคต
3) การสั่งงานของผู้ใช้ (ผู้ควบคุมระบบ) ต้องไม่ยุ่งยาก
4) ปรับเปลี่ยนขั้นตอนการทำงานได้เองเมื่อมีวิธีใหม่ๆ ที่ดีกว่า
5) การตรวจสอบและแก้ไขความผิดปกติต้องเป็นไปในลักษณะที่เป็นอิสระต่อกัน

2.10 โปรแกรม Agent (Software Agent)

โปรแกรม Agent เป็นแนวความคิดใหม่ของการทำงานของโปรแกรมคอมพิวเตอร์ โดยพัฒนามาจากสาขาวิชาของวิทยาการคอมพิวเตอร์คือ Artificial Intelligence (AI) เพื่อพัฒนาความสามารถของโปรแกรมให้สามารถทำการเรียนรู้และตัดสินใจได้เอง โดยเลียนแบบการทำงานของมนุษย์ (Human-like agent) ในบทนี้จะกล่าวถึงนิยาม ของโปรแกรม Agent โดยอ้างอิงจากงานวิจัยต่าง ๆ

2.11 ความแตกต่างระหว่างโปรแกรม และ โปรแกรม Agent

โปรแกรม Agent แตกต่างจากความเป็นโปรแกรมแบบธรรมดาเนื่องจากการทำให้เป็นโปรแกรมที่สามารถคิดและทำงานตอบโต้กับสภาพแวดล้อมได้เองโดยไม่ต้องคอยสั่งงานโดยผู้ใช้คล้ายกับมนุษย์ซึ่งอาจจะเปรียบเทียบแบบง่ายๆ ได้ว่า

โปรแกรม Agent = โปรแกรมธรรมดา + ความสามารถทางด้าน AI

การตัดสินใจตอบโต้กับสภาพการณ์ต่างๆ ของโปรแกรม Agent จะขึ้นอยู่กับสภาพแวดล้อมและฐานความรู้ที่มีอยู่ เมื่อฐานความรู้และสภาพแวดล้อมเปลี่ยนไปการตัดสินใจก็อาจจะเปลี่ยนไปด้วย

2.12 ประเภทต่าง ๆ ของโปรแกรม Agent

ในส่วนนี้จะกล่าวถึงประเภทต่างๆ ของโปรแกรม Agent ที่มีอยู่ โดยสามารถแบ่งกลุ่มของโปรแกรม Agent ได้ดังต่อไปนี้เช่น

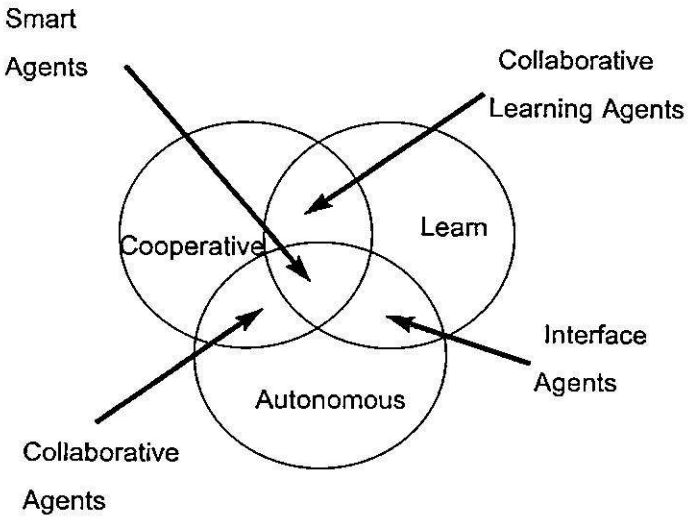
- แบบแรกอาจจะแบ่งตามความสามารถในการเคลื่อนย้าย (Mobility) คือโปรแกรมสามารถที่จะเคลื่อนย้ายตัวเองไปทำงานยังเครื่องคอมพิวเตอร์ต่างๆ ในเครือข่ายได้หรือไม่ ซึ่งก็จะแบ่งได้เป็น Agent ที่เคลื่อนย้ายได้ (mobile agents) และที่ทำงานอยู่ในระบบเดียวไม่ได้เคลื่อนย้ายไปยังระบบอื่นไม่ได้ (static agent)

- แบบที่สองอาจแบ่งเป็น ประเภทให้คำปรึกษา (deliberative) หรือ ประเภทตอบสนอง (reactive) โดยประเภทให้คำปรึกษาจะทำการวิเคราะห์หาเหตุผลตามโครงสร้างแนวคิดที่กำหนดไว้ (Thinking paradigm) โดยจะทำการวางแผน, ต่อรอง หรือ ตกลงกับ Agent อื่น

เพื่อให้สามารถทำงานที่ได้รับมอบหมายให้ลุล่วงไปได้ ส่วน Agent ประเภทตอบสนองเริ่มมีการค้นคว้าโดย Brooks (1986) และ Agre & Chapman (1987) ซึ่งสามารถทำงานตอบสนองกับสภาพแวดล้อมในสถานะปัจจุบันได้โดยไม่ต้องมี Model ของสภาพแวดล้อมนั้นอยู่ภายใน ซึ่ง Brooks ได้แย้งว่าลักษณะของความเป็นอัจฉริยะ (Intelligent behavior) นั้นสามารถทำได้โดยไม่จำเป็นที่จะต้องมีการกำหนด หรือแทนความหมายใดไว้ก่อนเหมือนในระบบ AI ทั่วๆ ไป [Brooks, 1991b]

- แบบที่สามอาจแบ่งตามคุณลักษณะของโปรแกรม Agent ซึ่งมีหัวข้อใหญ่ในการแบ่งดังนี้
  - ความสามารถในการทำงานแบบอัตโนมัติ (Autonomy) ซึ่งทำให้ Agent สามารถทำงานได้ด้วยตัวของมันเองโดยไม่ต้องการคำแนะนำจากมนุษย์ โดยทำงานเพื่อให้บรรลุเป้าหมายของงานที่ได้รับมอบหมายมาตั้งแต่ต้น ซึ่งในการทำงานนั้นเป็นลักษณะของการทำงานแบบที่เป็นการริเริ่ม (initiative) ไม่ใช่เป็นการตอบสนองต่อสภาวะแวดล้อมแบบธรรมดา [Wooldridge & Jenning, 1995a]
  - ความสามารถในการทำงานร่วมกับผู้อื่น (Cooperation) การทำงานร่วมกับผู้อื่นเป็นความคุณสมบัติหนึ่งที่สำคัญของโปรแกรม Agent เนื่องจากสุดท้ายแล้วในสภาพการทำงานจริงคงจะเต็มไปด้วยโปรแกรม Agent มากมาย ซึ่งจำเป็นที่โปรแกรม Agent จะต้องสามารถเรียนรู้และทำงานร่วมกับ Agent อื่นรวมทั้งมนุษย์ด้วย
  - ความสามารถในการเรียนรู้ (Learning) เป็นคุณสมบัติที่จำเป็นอย่างยิ่งในการที่จะทำให้ Agent มีความฉลาด ซึ่ง Agent จะต้องทำการเรียนรู้สิ่งที่มันจะต้องตอบสนองด้วย และจากการวิเคราะห์สิ่งที่เรียนรู้นี้ จะทำให้ Agent สามารถที่จะปรับปรุงประสิทธิภาพการทำงานของตัวมันเองให้ดีขึ้นได้ด้วย

จากคุณสมบัติขั้นต่ำทั้ง 3 ข้อนี้เราจะสามารถที่จะแบ่ง Agent ออกเป็น 4 ประเภทดังแสดงในภาพที่ 2.1 คือ collaborative agents, collaborative learning agents, interface agents and truly smart agent.



ภาพที่ 2.1 แสดงการแบ่งประเภทของโปรแกรม Agent ตามคุณลักษณะ

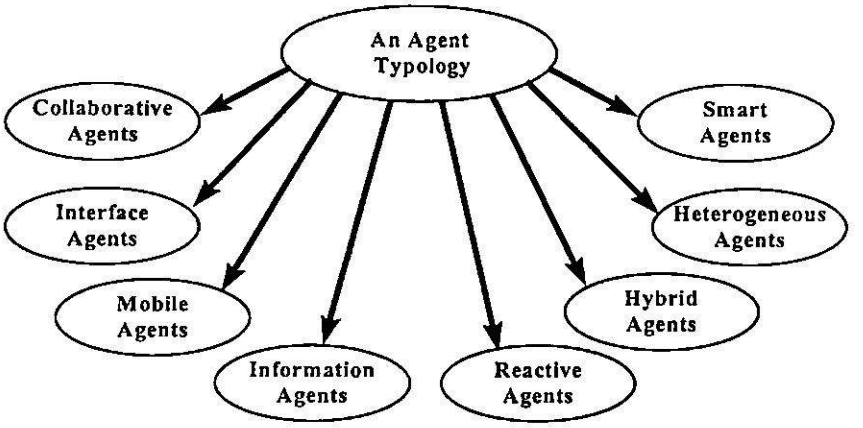
จากที่กล่าวมาบางทีอาจเกิดการนำเอาคุณสมบัติของการแบ่งแยกประเภทของโปรแกรม Agent มารวมกันเช่น static deliberative collaborative agents, mobile reactive collaborative agents, static deliberative interface agents, mobile reactive interface agents ตัวอย่างเช่น Lashkari et al.(1994) เสนอผลงานที่ AAAI ในเรื่อง Collaborative interface agents ซึ่งจะจัดอยู่ในกลุ่มของ static collaborative interface agents

- แบบที่ 4 อาจแบ่งได้ตามหน้าที่การทำงานเช่น world wide web (WWW) information agents ซึ่งใช้ในการค้นหาข้อมูลในระบบเครือข่าย Internet เช่น WebCrawlers, Lycos และ Spiders. และเช่นเดียวกัน information agent อาจเป็น static, mobile หรือ deliberative

- ในแบบที่ 5 เราอาจจะรวมเอากลุ่มของ hybrid agents ซึ่งเป็นโปรแกรม Agent ที่รวมเอาหลายๆ ความสามารถเข้าด้วยกัน

ในความเป็นจริงแล้วโปรแกรม Agent นั้นเป็นประเภท multi-dimensional space ซึ่งเราไม่สามารถที่จะใช้คุณสมบัติเพียง 2-3 อย่างมาจัดกลุ่มได้ ในที่นี้เราจึงจัดกลุ่มของ

โปรแกรม Agent ใหม่ ซึ่งคิดว่าจะครอบคลุมประเภทของโปรแกรม Agent ส่วนใหญ่ ดังภาพที่ 2.2



ภาพที่ 2.2 การจัดกลุ่มของโปรแกรม Agents

2.13 ตัวอย่างโครงสร้างของโปรแกรม Agent ประเภทต่างๆ

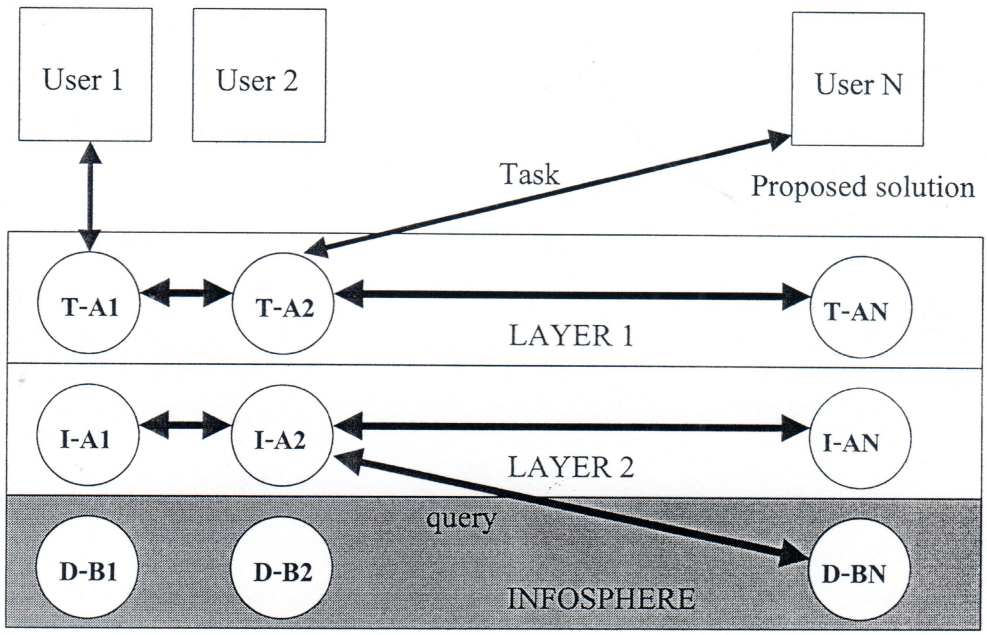
Collaborative Agents

เป็นลักษณะการทำงานร่วมกันของโปรแกรม Agent กับ Agents อื่นๆ เพื่อให้สามารถทำงานตามที่ได้รับมอบหมายได้สำเร็จ แรงจูงใจในการสร้าง Collaborative Agents

- เพื่อแก้ปัญหาที่ใหญ่เกินกว่าที่ agent เพียงตัวเดียวจะทำการแก้ปัญหาได้, ทรัพยากรที่จำกัด หรือเพื่อลดอัตราเสี่ยงในการมี agent เพียงตัวเดียว
- เพื่อให้สามารถเชื่อมต่อการทำงานของระบบเดิมๆ ที่มีอยู่ก่อนได้ เช่น expert system, decision support system เป็นต้น
- เพื่อเป็นทางออกให้กับปัญหาของระบบการทำงานแบบ inherently distributed problem เช่น distributed sensor networks หรือ ระบบ air-traffic control
- เพื่อเป็นการแก้ปัญหาในการค้นหาข้อมูลจากระบบ distributed information source
- เพื่อใช้กับปัญหาการกระจายกันอยู่ของผู้เชี่ยวชาญ เช่นในระบบ health care provisioning

- เพื่อขยายการทำงานแบบ modularity (ลดความซับซ้อน), ความเร็ว (parallelism), reliability (redundancy)

ระบบตัวอย่างของ Collaborative Agent เช่น CMU Pleiades System ซึ่งสถาปัตยกรรมนี้ใช้ในการพัฒนาระบบ visitor hosting system ซึ่งเป็นระบบสำหรับการให้บริการกับแขกที่มาเช่น การนัดหมาย, การหาข้อมูลที่ต้องการ โดยการแบ่งการทำงานออกเป็นระดับชั้น (layer) ดังแสดงในภาพที่ 2.3



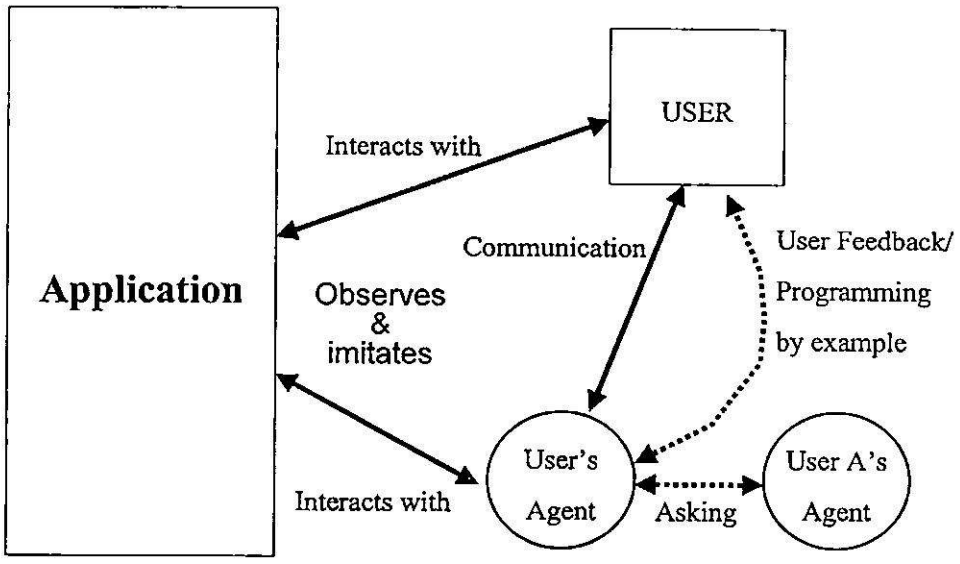
ภาพที่ 2.3 สถาปัตยกรรม ของ CMU Pleiades System

Interface Agents

เป็นโปรแกรม Agent ที่พัฒนาขึ้นเพื่อที่จะปรับปรุงระบบและลักษณะการทำงานของ User Interface ให้ดีกว่าที่เป็นอยู่ หรือในอีกชื่อที่รู้จักกันคือระบบ human-computer interfaces (HCI)

ซึ่ง Interface Agent สามารถที่จะทำการปรับปรุงประทธิภาพของระบบสามารถทำได้โดย

- การเลียนแบบการทำงานของผู้ใช้
- การพิจารณาจาก feedback ที่ได้จากผู้ใช้
- รับวิธีการทำงานจากผู้ใช้โดยตรง
- สอบถามหรือเรียนรู้จากโปรแกรม Agent อื่นๆ ในระบบ



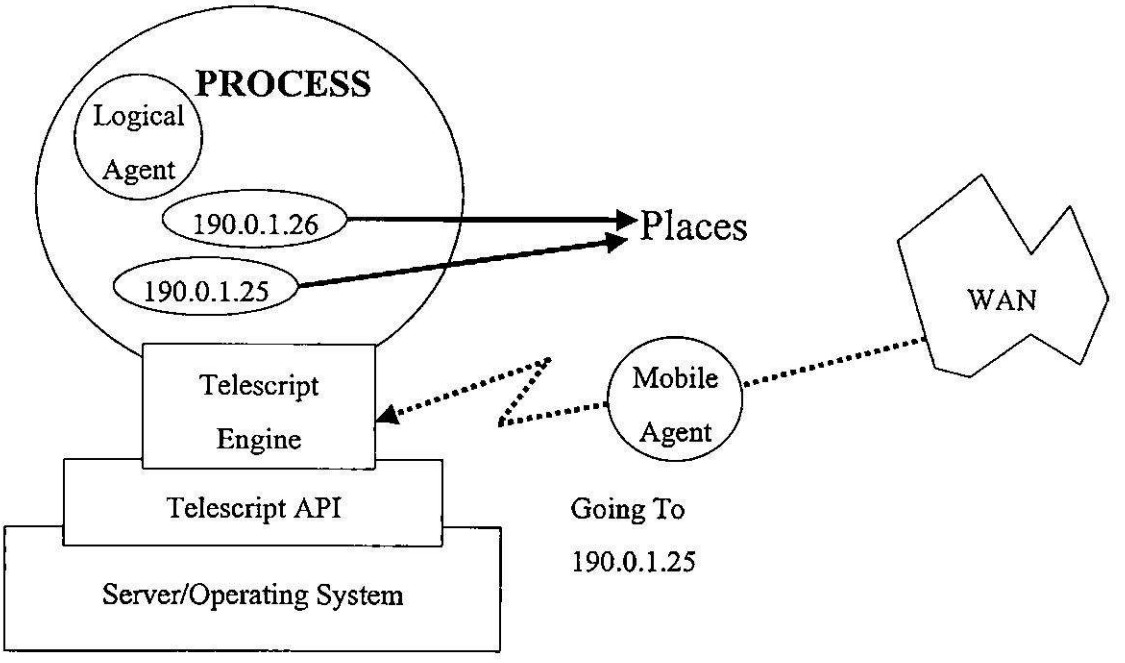
ภาพที่ 2.4 แสดงการทำงานของ Interface Agent

Mobile Agents

Mobile Agents เป็นโปรแกรม Agent อีกประเภทหนึ่งที่ได้รับความสนใจอย่างรวดเร็ว โดยเฉพาะอย่างยิ่งเมื่อ General Magics ได้ทำการเผยแพร่ Telescript Development Environment แต่การใช้งาน Mobile Agent ในทางการค้าครั้งแรกได้แก่ระบบ Magic Link PDA หรือ personal intelligent communication (PIC) ของบริษัท Sony ซึ่งอุปกรณ์นี้จะทำงานในลักษณะของการจัดการ E-mail, FAX รวมทั้งการบริการค้นหาข้อมูลที่เป็นข้อความ,ภาพ หรือ เสียง ที่ผู้ใช้ต้องการจาก America Online และ AT&T PersonaLink Services โดยการใช้ Telescript

ประโยชน์ของ Mobile Agent

- ลดค่าใช้จ่ายของการสื่อสาร
- ใช้ทรัพยากรในเครื่องตนเองน้อย (สามารถส่ง Agent ออกไปทำงานที่เครื่องอื่นแล้วเอาผลกลับมาแสดงได้)
- สามารถทำงานในแบบ Asynchronous computing (ผู้ใช้สามารถที่จะส่ง Agent ออกไปทำงาน ผู้ใช้สามารถที่ปิดเครื่อง หรือออกจากระบบแล้วค่อยกลับมาดูผลอีกทีในภายหลัง)
- เป็นสถาปัตยกรรมที่ดีสำหรับการทำงานในแบบ distributed computing

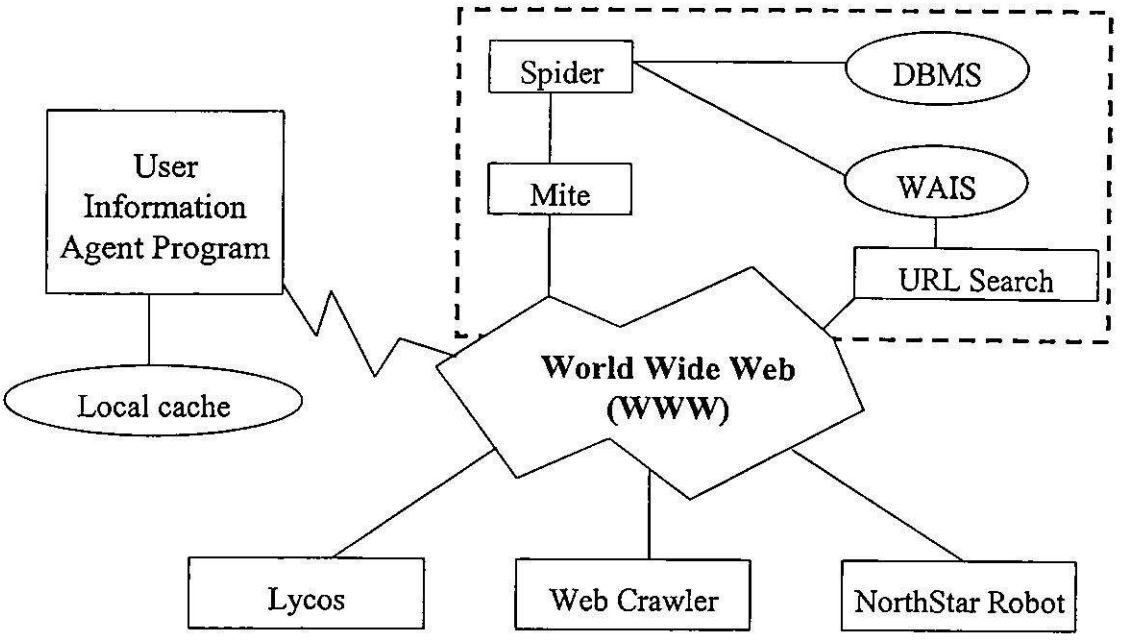


ภาพที่ 2.5 แสดงการทำงานของ Mobile Agent โดยใช้ Telescript

Information Agents

Information agents เกิดขึ้นเนื่องจากความต้องการในการจัดการกับข้อมูลที่เกิดขึ้นอย่างรวดเร็วในปัจจุบันซึ่งจะเห็นได้จาก Davies&Weeks (1995) ที่ได้เก็บรวบรวมการพยากรณ์เกี่ยวกับการเพิ่มขึ้นของข้อมูลทางด้านเทคนิคเอาไว้ในปี 1982 ว่าจะเพิ่มขึ้นเป็น 2 เท่าทุกๆ 5 ปี, ในปี 1988 กล่าวว่าจะเพิ่มขึ้นเป็น 2 เท่าทุกๆ 2.2 ปี และในปี 1992 ประมาณว่าเป็น 1.6 ปี และยิ่งในปัจจุบันการใช้งาน หรือสืบค้นข้อมูลที่กระทำผ่านระบบ WWW ซึ่ง Nicholas Negroponte แห่งห้องทดลองที่ MIT กล่าวไว้ว่าปริมาณของ WEB Site เพิ่มขึ้นเป็น 2 เท่า ทุกๆ 15 วัน

การทำงาน Information agents จะทำการรวบรวมและจัดการกับข้อมูลจากแหล่งข้อมูลต่างๆ เพื่อนำเสนอผู้ใช้ตามรูปแบบที่ต้องการ



ภาพที่ 2.6 แสดงการทำงานของ Information Agents

จากภาพแสดงการทำงานของ Spider ซึ่งเป็น agent ที่จัดการเกี่ยวกับ index ซึ่งสามารถที่จะค้นหาข้อมูลจาก WWW ในลักษณะของ depth-first และทำการเก็บข้อมูลโครงสร้างของ WWW ลงในตัวจัดการฐานข้อมูล(DBMS) และสารบัญของ WWW เหล่านั้นจะถูกบันทึกไว้ใน WAIS ตัวอย่างอื่นที่เห็นได้เช่น Lycos หรือ Webcrawler ซึ่งสามารถใช้ในการสร้างสารบัญสำหรับการค้นหาข้อมูลจาก Web site ได้เช่นกัน ซึ่งในปัจจุบันมี spider ทำงานอยู่ในระบบ WWW มากกว่า 20 ตัว

Information agent จะรับข้อมูลที่ผู้ใช้ต้องการ และทำการค้นหาข้อมูลใน search engine ซึ่งอาจจะเป็นที่เดียวหรือหลายที่ หรือข้อมูลที่เก็บไว้จนได้ข้อมูลที่ผู้ใช้ต้องการ

2.14 สรุปคุณลักษณะทั่วไปของโปรแกรม Agent

จากข้อมูล และนิยามของโปรแกรม Agent จากงานวิจัยและบทความต่างๆ ที่กล่าวมาทั้งหมดเราจะสามารถสรุปคุณลักษณะของโปรแกรม Agent ที่ดีได้ดังต่อไปนี้คือ

1. สามารถเป็นตัวแทนได้ (Delegation) เป็นความสามารถของโปรแกรม agent ที่สามารถที่จะเข้าใจและทำงานที่ได้รับมอบหมายให้ลุล่วงไปได้โดยไม่ต้องให้ผู้ใช้คอยสั่งงาน

2. สามารถวิเคราะห์ข้อมูลเพื่อหาแนวทางในการทำงานได้ (Data-directed Execution) เป็นความสามารถในการตรวจสอบและวิเคราะห์ข้อมูลที่มีอยู่ เช่น การเปลี่ยนแปลงของฐาน

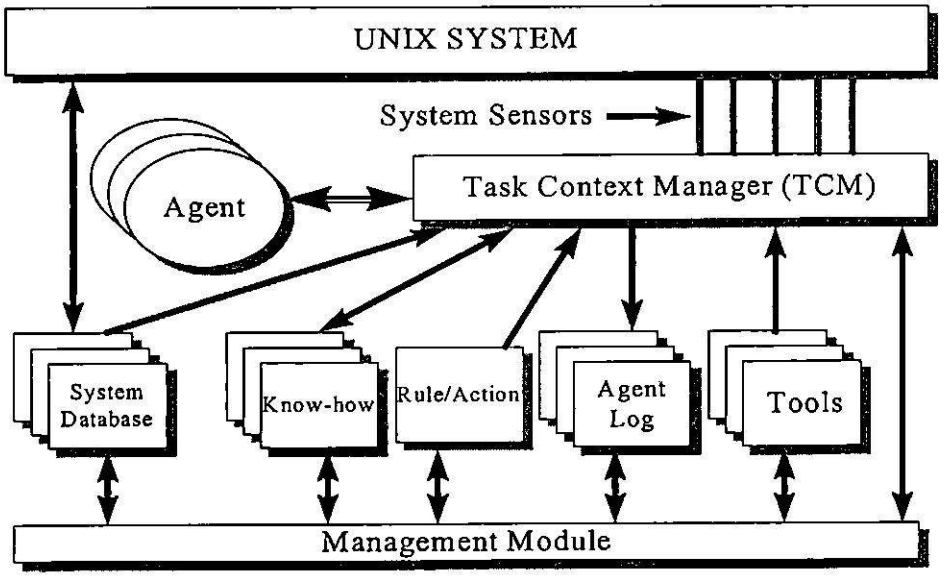
ข้อมูล, ระบบปฏิบัติการ หรือ เครือข่าย เพื่อกำหนดรูปแบบในการทำงานให้เหมาะสมกับสภาวะแวดล้อมได้ตลอดเวลา

3. ความสามารถในการสื่อสาร (Communication) เป็นความสามารถของโปรแกรม agent ในการที่จะทำการติดต่อกับโปรแกรม agent อื่น หรือผู้ใช้อื่นเพื่อแลกเปลี่ยนข้อมูล ข่าวสาร ที่จำเป็นต้องใช้ในการปฏิบัติงานลุล่วงตามที่ได้รับมอบหมาย
4. การใช้เหตุผล (Reasoning) ความสามารถนี้เป็นความสามารถที่โปรแกรม agent สามารถที่จะทำการวิเคราะห์แยกแยะการขอให้ทำงานได้ว่าความต้องการของคำขอนั้นคืออะไร และ หาข้อมูลต่างๆ ที่ใช้เป็นข้อมูลสำหรับทำงานตามคำขอนั้นๆ ได้
5. การวางแผน (Planning) เป็นความสามารถของโปรแกรม Agent ในการจัดระบบและวางแผนการทำงานเพื่อให้บรรลุภารกิจที่ได้รับมอบหมาย
6. การปรับปรุงตัวของมันเอง (Self-modification) เป็นความสามารถในการปรับปรุงเปลี่ยนแปลงการทำงานของตัวโปรแกรม Agent เองเพื่อให้สามารถทำงานได้ดีขึ้น
7. ความสามารถในการต่อเชื่อมกับ Module ต่างๆ โดยไม่ต้องทำการ compile ใหม่ (Plug&Play properties) เป็นความสามารถในการเลือกใช้และเข้าใจ Tools ต่างๆ ที่มีอยู่ในระบบได้โดยไม่ต้องมีการแก้ไขหรือปรับเปลี่ยนโปรแกรม
8. เป็นโปรแกรมที่มีขนาดเล็ก (Lightweight) เป็นคุณสมบัติที่ตัวโปรแกรม Agent จะต้องมีขนาดที่เล็กมากๆ เนื่องจากจะได้ไม่ต้องใช้ resource ในการทำงานมาก และไม่ทำให้เกิดความล่าช้ากับระบบในกรณีที่ต้องย้ายไปทำงานยังเครื่อง หรือ เครือข่ายอื่น โดย Module หรือ tools ส่วนใหญ่จะถูกออกแบบให้อยู่บนเครื่องที่เป็น Host ให้มากที่สุด

บทที่ 3
สถาปัตยกรรมของระบบ
(System Architecture)

3.1 องค์ประกอบของระบบ

จากการแนวทางในการประยุกต์ใช้งานโปรแกรม Agent งานวิจัยต่างๆ และข้อมูลที่สามารถใช้ในการตรวจสอบความปลอดภัยในระบบยูนิกซ์ในบทที่ผ่านมา เราสามารถที่จะออกแบบสถาปัตยกรรมของระบบรักษาความปลอดภัยแบบอิงกฎสำหรับยูนิกซ์ และสภาวะแวดล้อมที่ใช้ในการตรวจจับการบุกรุกที่เกี่ยวข้องกับความปลอดภัยของระบบ UNIX เพื่อให้เหมาะสมกับสภาพการทำงาน โดยโปรแกรม Agent ที่ทำงานอยู่ในระบบรักษาความปลอดภัยนี้จะต้องมีขนาดเล็กและสามารถที่จะขยายการทำงานให้เป็นลักษณะของ Mobile Agent ได้ในอนาคตในกรณีที่ต้องการขยายระบบให้สามารถตรวจสอบความปลอดภัยใน Host ต่างๆ ที่ต่ออยู่ในระบบเครือข่าย ดังนั้นงานวิจัยนี้จึงออกแบบให้โปรแกรม Agent ใช้ทรัพยากรร่วมกันที่ส่วนกลางให้มากที่สุดซึ่งจะมีองค์ประกอบของสภาพแวดล้อมในการทำงานของระบบโดยรวมดังแสดงในภาพที่ 3.1



ภาพที่ 3.1 สถาปัตยกรรมของระบบรักษาความปลอดภัยแบบอิงกฎสำหรับยูนิกซ์

องค์ประกอบหลักในการทำงานของระบบสามารถแบ่งได้ดังต่อไปนี้คือ

- ส่วนจัดการงาน (Task Context Manager)
- ฐานข้อมูลของระบบ (System Database)
- ฐานข้อมูลความชำนาญ (Know-how Database)
- เครื่องมือ (Tools)
- บันทึกการทำงานของ Agent (Agent log)
- กฎ และ แนวทางปฏิบัติ (Rule/Action)
- ส่วนควบคุม (Management Module)
- ตัวตรวจสอบระบบ (System Sensors)
- Agent

ส่วนควบคุมการทำงาน (Task context Manager)

Task Context Manager (TCM) มีหน้าที่ในการทำงานคล้ายกับองค์กรที่คอยจ่ายงานให้กับ Agent ในการตรวจสอบข้อสงสัยที่คาดว่าจะเป็นอันตรายกับระบบ รวมทั้งให้บริการกับ Agent ที่ต้องการใช้งาน Tools ต่างๆ เพื่อปฏิบัติงานให้ลุล่วงไปได้ โดยหน้าที่หลักๆ ของ TCM มีดังต่อไปนี้คือ

- ตรวจสอบความผิดปกติต่างๆ ที่อาจนำไปสู่ความไม่ปลอดภัยของระบบ (Sensor)
- สร้างและมอบหมายงานให้ Agent ในการตรวจสอบข้อสงสัยนั้นๆ (Agent Launch)
- จัดการและบริหารทรัพยากรต่างๆ ที่ Agent ต้องเรียกใช้ (Tools Manager)

โครงสร้างในการส่งผ่านข้อมูลของ TCM จะเป็น blackboard architecture ซึ่งง่ายต่อการเปลี่ยนแปลงและเพิ่มเติมรูปแบบของการทำงานใหม่ๆ เข้าไปโดยไม่ต้องทำการเปลี่ยนแปลง หรือกระทบกับส่วนอื่นของระบบ

การทำงานของ TCM จะเป็นการตรวจสอบระบบอยู่ตลอดเวลาโดยเมื่อเกิดเหตุการณ์ต่างๆ ที่ไม่น่าไว้วางใจก็จะทำการสร้างและส่ง Agent ออกไปตรวจสอบข้อสงสัยต่างๆ โดยมีตัวอย่างของการตรวจสอบดังต่อไปนี้

รายการตัวอย่างการตรวจสอบสภาพแวดล้อมของความปลอดภัย (Security Check List)

- ใช้โปรแกรม Password generator ในการควบคุมการเปลี่ยน password ของผู้ใช้
- กำหนดอายุการใช้งานของ password
- มีการใช้งานระบบ auditing และ accounting
- ไม่ควรอนุญาตให้ใช้ .rhosts, /etc/hosts.equiv, .netrc
- ตรวจสอบ home directories ของผู้ใช้ทุกคน ไม่ควรให้สิทธิในการ อ่าน/เขียน มากกว่า 755 (r- - r-x r-x)
- กำหนดค่า umask ที่เหมาะสมของผู้ใช้ทุกคนผ่านทาง system profile

ตัวอย่างรายการตรวจสอบผู้ใช้ที่เพิ่งเข้าสู่ระบบ (New LoginUser CheckList)

- Login เข้าสู่ระบบในช่วงเวลาที่กำหนดหรือไม่
- Login มาจาก เครือข่ายที่ไม่ได้รับอนุญาตหรือไม่
- เตือนการ Login ครั้งสุดท้ายจากเครื่องที่ไม่น่าไว้ใจ
- Login Session มากกว่าที่กำหนดไว้หรือไม่

ฐานข้อมูลของระบบ (System Database)

ในระบบนี้ system database จะเป็นฐานข้อมูลที่เก็บบันทึกข้อมูลต่างๆ ที่ใช้เป็นสำหรับการทำงานของ Agent โดยได้แก่ข้อมูล Audit trial และ log file ต่างๆ ซึ่งแยกออกเป็น 2 ประเภท คือ

- log file ที่ได้จากการทำงานของระบบ UNIX อยู่แล้ว
- แฟ้มข้อมูลที่เป็นข้อกำหนดเฉพาะของระบบ และผู้ใช้

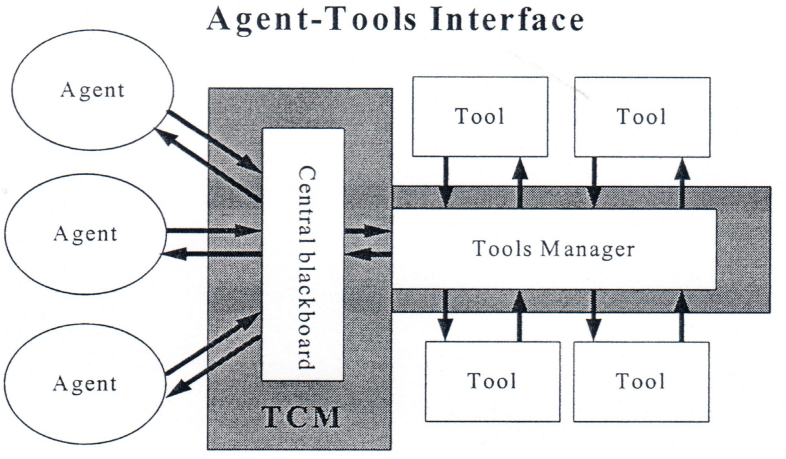
ฐานข้อมูลความชำนาญ (Know-how Database)

Know-how เป็นข้อมูลที่ตัว Agent ใช้ในการถ่ายทอดความรู้และแนวทางในการทำงานระหว่าง Agent ด้วยกัน (Knowledge sharing) เพื่อเพิ่มประสิทธิภาพในการทำงานและสามารถบรรลุภารกิจที่ได้รับมอบหมายให้เร็วขึ้นรวมทั้งยังเป็นการเรียนรู้และพัฒนาการทำงานของระบบ ซึ่งสามารถกระทำได้โดยเมื่อ Agent ถูกสร้างขึ้นในระบบ และได้รับมอบหมายภาระกิจที่จะต้องปฏิบัติ Agent จะทำการวิเคราะห์แนวทางในการทำงานตามความต้องการนั้นโดยการตรวจสอบความต้องการนั้นกับแผนงานที่มีอยู่แล้วว่ามีแผนงานตามที่ต้องการอยู่แล้วหรือไม่ ถ้ามีและระบบไม่ได้มีการเพิ่ม Tool ใหม่เข้ามาในระบบก็จะทำงานตามแผนงานนั้นซึ่งเป็นการใช้ความรู้ร่วมกัน

แต่ถ้ามีการเพิ่ม Tool ใหม่ขึ้นมา ก็จะต้องทำการวิเคราะห์แผนการทำงานใหม่อีกครั้ง ซึ่งถ้าได้แผนที่ดีกว่าก็จะทำการบันทึกไว้ แต่ถ้าไม่ได้แผนที่ดีกว่าเดิมก็จะใช้ความรู้เดิม โดยข้อมูลพื้นฐานสำหรับการวิเคราะห์ know-how ที่จะต้องมีเพื่อให้ Agent สามารถทำงานได้จะประกอบด้วยข้อมูลหลักๆ ดังนี้คือ

- รายการของอุปกรณ์ที่มีให้ Agent ใช้ และรูปแบบการใช้งาน (Tools Table)
- แนวทางในการปฏิบัติงาน ที่ Agent อื่นเคยปฏิบัติมาแล้วและผลรวมของค่าตัวถ่วงน้ำหนักที่ได้ในการทำงานตามแนวทางนั้น

เครื่องมือ (Tools)



ภาพที่ 3.2 รูปแบบการใช้งาน Tools ของโปรแกรม Agent

Tools จะเป็นที่รวมของชุดคำสั่งต่างๆ ที่ Agent จะต้องใช้ในการทำงาน โดย Agent ทราบว่ามี Tools ใดบ้าง มีวิธีการใช้งานอย่างไรโดยได้ข้อมูลจาก Know-how ในส่วนของ Tools table ด้วยเหตุผลทางด้านความปลอดภัย เมื่อต้องทำงานร่วมกับ Mobile Agent ที่มาจากระบบอื่น (สำหรับการพัฒนาต่อไปในอนาคต) ดังนั้นการขอใช้งาน Tools จะกระทำโดยการระบุความต้องการใช้งาน Tools ใน Blackboard ของ TCM โดยจะมี Tools Manager คอยตรวจสอบและให้บริการกับทุกๆ ความต้องการที่บันทึกลงใน Central blackboard และส่งผ่านผลการทำงานให้โดยบันทึกกลับไปที่ Central Blackboard เหมือนเดิม

ผู้บริหารระบบสามารถที่จะเพิ่มเติมการทำงาน เปลี่ยนแปลงการทำงาน หรือเพิ่มความสามารถใหม่ๆ ในการทำงานของระบบได้โดยการใส่ Tool ใหม่ๆ ลงไป แล้วไปบันทึกวิธีการใช้งาน

ไว้ใน KnowHow Table ซึ่งจะทำให้ง่ายในการใช้ปรับปรุงและเปลี่ยนแปลงความสามารถของโปรแกรม Agent

บันทึกการทำงานของ Agent (Agent log)

เป็นฐานข้อมูลที่บันทึกทุกๆ การทำงานของ Agent เพื่อประโยชน์ในการตรวจสอบการทำงานและการวิเคราะห์รูปแบบการทำงานของ Agent ซึ่งสามารถนำกลับมาวิเคราะห์สร้างเป็นรูปแบบหรือวิธีลัดในการทำงานได้

กฎ และแนวทางปฏิบัติ (Rule / Action)

Rule/Action เป็นฐานข้อมูลที่ใช้ประกอบในการตัดสินใจของ Agent ในการทำงาน ข้อมูลที่อยู่ในฐานข้อมูลนี้เป็นข้อมูลที่ได้จากผู้บริหารระบบ

Rule จะเป็นกฎต่างๆ ที่ต้องที่เกี่ยวข้องกับความไม่น่าไว้วางใจต่าง ๆ หรือเป็นข้อห้ามที่มีอยู่แล้วในระบบตัวอย่างเช่น การห้าม login มากกว่า 1 session การห้ามผู้ใช้ธรรมดา login หลัง 16.00 น. เป็นต้น

Action เป็นแนวทางปฏิบัติเมื่อตรวจสอบได้ว่ามีการละเมิด Rule นั้น ๆ เช่นเมื่อผู้ใช้ธรรมดา login เข้ามาหลังจาก 16.00 น. ให้ส่งข้อความเตือนและตัดผู้ใช้ออกจากระบบ

ส่วนควบคุม (Management Module)

Management Module เป็นชุดโปรแกรมส่วนที่ใช้ในการตรวจสอบ, ควบคุม, และเปลี่ยนแปลงรูปแบบในการทำงานของระบบ รวมทั้งการเพิ่มเติม Tools, การเปลี่ยนแปลง Rule / Action โดยมีการทำงานหลักๆ ดังนี้คือ

- เปลี่ยนและกำหนดรูปแบบการทำงานของ TCM
- เพิ่ม / ลบ / แก้ไข เครื่องมือ (Tools)
- เพิ่ม / ลบ / แก้ไข กฎและแนวทางปฏิบัติ (Rule/Action)
- เพิ่ม / ลบ / แก้ไข ฐานข้อมูลความชำนาญ (Know-how)
- เพิ่ม / ลบ / แก้ไข / วิเคราะห์ ฐานข้อมูลระบบSystem Database
- แสดงข้อมูลการทำงานของ Agent (AgentLog)

ตัวตรวจสอบระบบ (System Sensors)

System Sensors เป็นส่วนประกอบหนึ่งที่ใช้ในการเฝ้าระวังความเป็นไปของระบบ ในด้านความปลอดภัย ซึ่งผู้ที่บริหารระบบจะสามารถ Enable/Disable Sensor ต่างๆ ได้ตามความต้องการ และสามารถเพิ่ม Sensors สำหรับตรวจสอบเหตุการณ์ใหม่ๆ ได้ตามที่ต้องการ โดยการเรียกผ่าน Management Module

Sensors ต่างๆ จะทำงานอย่างอิสระไม่ขึ้นต่อกัน ตามเหตุการณ์ที่ได้ระบุไว้ โดยใช้กลไกที่มีอยู่แล้วในระบบ UNIX เช่น crontab, startup profile โดยตัวตรวจจับนี้จะแบ่งออกเป็นสองกลุ่มใหญ่ ๆ คือ

- ตัวตรวจสอบตามเหตุการณ์ (Event Sensor) เช่น เมื่อผู้ใช้ Login เมื่อเปิดเครื่อง เมื่อมีการเปลี่ยน Password การเปลี่ยนเป็นผู้ใช้อื่น
- ตัวตรวจสอบตามระยะเวลาที่กำหนด (Periodical Sensor) ซึ่งแบ่งออกเป็น 2 กลุ่มคือ
  - ตัวตรวจสอบตามเวลาที่ระบุคงที่ (Fixed Periodical Sensor) เช่น ตรวจสอบหาแฟ้ม coredump หรือ แฟ้มข้อมูลที่มีการ setuid
  - ตัวตรวจสอบที่กำหนดเวลาการตรวจสอบครั้งต่อไป (At Next Sensor) เช่น การตรวจสอบว่ามี Session ที่ Idle เกินเวลาที่กำหนดหรือไม่ ซึ่งการตรวจสอบครั้งต่อไปก็คือ

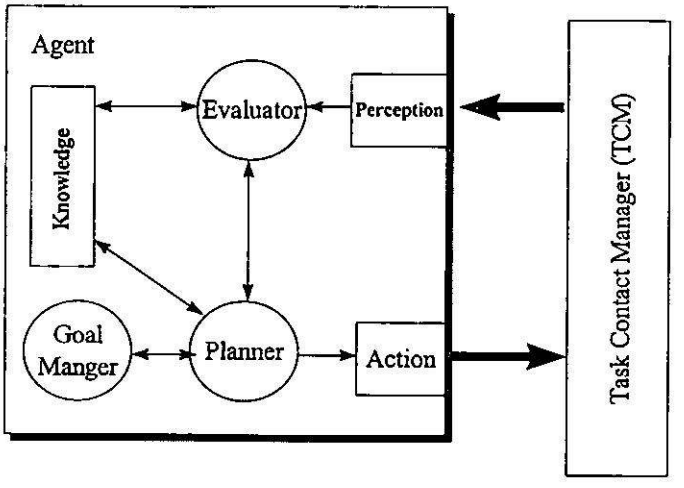
    ค่าเวลาที่อนุญาต – เวลาของ session ที่ Idle สูงสุด(ที่ไม่เกินกำหนด)

Agent

Agent จะเป็นโปรแกรมที่ถูกเรียกขึ้นมาใช้งานโดย TCM โดยคุณสมบัติของ Agent ที่เกิดขึ้นทุกตัวจะเหมือนกันหมด แต่การทำงานและวิธีการทำงานจะขึ้นอยู่กับภาระกิจที่ได้รับมอบหมาย

3.2 สถาปัตยกรรมของโปรแกรม Agent (Agent architecture)

จากการพิจารณาสภาพแวดล้อมของระบบ UNIX และองค์ประกอบต่างๆ ในการทำงานของโปรแกรม Agent ตามที่ได้ออกแบบไว้ข้างต้น จึงได้ทำการออกแบบสถาปัตยกรรมของตัวโปรแกรม Agent ที่ใช้สำหรับระบบตรวจจับความไม่น่าไว้วางใจในสภาพแวดล้อมของระบบ UNIX ตามภาพประกอบต่อไปนี้ ทั้งนี้เพื่อประโยชน์ในการปรับปรุงประสิทธิภาพหรือโครงสร้างของ Agent ในภายหลังได้ง่าย



ภาพที่ 3.3 สถาปัตยกรรมของโปรแกรม Agent

องค์ประกอบหลักของโปรแกรม Agent ได้แก่

- ส่วนความรู้ (Knowledge)
- ส่วนจัดการเป้าหมาย (Goal Manager)
- ส่วนวางแผน (Planner)
- ส่วนประเมินผล (Evaluator)
- ส่วนรับข้อมูล (Perception)
- ส่วนตอบโต้ (Action)

ส่วนความรู้(Knowledge)

จะเป็นฐานข้อมูลที่ถูกบรรจุให้กับตัว Agent ในตอนที่ TCM สร้างตัว Agent ขึ้น โดยจะประกอบไปด้วยข้อมูลความรู้ต่อไปนี้

1. ความรู้ของ Agent (Agent Knowledge) ที่เกี่ยวข้องกับขั้นตอนการทำงานของ Agent เองเพื่อใช้ในการควบคุมตัวเอง
2. ความรู้เกี่ยวกับระบบ และการทำงานกับระบบ ซึ่งเป็นข้อมูลที่ได้รับจาก TCM
3. ความรู้และเทคนิคในการทำงาน และเครื่องมือต่างๆ ซึ่งได้จาก Know-how database
4. Rule/Action ที่ใช้ในการตัดสินใจตอบโต้กับสภาวะการต่างๆ ในการปฏิบัติงาน

ส่วนจัดการเป้าหมาย (Goal Manager)

เป็นส่วนที่ใช้ในการควบคุมเป้าหมายที่ Agent จะต้องปฏิบัติให้ลุล่วงตามที่ได้รับมอบหมาย โดยจะได้รับความต้องการจากผู้ใช้ หรือระบบว่าเป้าหมายหรือ goal คืออะไร และทำให้อยู่ในรูปของ standard goal เพื่อส่งให้กับ Planner ต่อไป

ส่วนวางแผน (Planner)

Planner หรือส่วนใช้วางแผนในการทำงานโดยมีหน้าที่วางแผนและควบคุมการทำงานให้บรรลุตามเป้าหมายที่ได้รับมาจาก Goal Manager ทำการแยก goal ออกเป็นหลายๆ sub goal และพยายามที่จะทำการเปรียบเทียบ sub goal เหล่านั้นกับผลการทำงานของ Tool และความสามารถต่างๆ ที่ Agent มีอยู่เพื่อทำการจัดเรียงลำดับการทำงานจนสามารถบรรลุเป้าหมายได้ Planner จะสามารถบอกได้ว่า Tool หรือความสามารถใดของ Agent จะสามารถตอบสนอง sub goal ใดได้โดยอาศัยข้อมูลจาก Know-how ซึ่งเมื่อวางแผนและทำงานจนครบทุก sub goal แล้วก็หมายความว่าเป้าหมายหรือ goal ที่ได้รับมอบหมายได้เสร็จสิ้นไปแล้ว Planner ก็จะสามารถที่จะรับเป้าหมายใหม่จาก Goal Manager มาทำได้ต่อไป

ส่วนประเมินผล (Evaluator)

Evaluator เป็นส่วนที่ใช้ในการวิเคราะห์ข้อมูลที่ได้จาก Perception ว่าข้อมูลที่ได้เป็นข้อมูลอะไร โดยถ้าเป็นข้อมูลความรู้เกี่ยวกับระบบก็จะส่งให้กับ Knowledge แต่ถ้าเป็นผลที่ได้จากการทำงานก็จะส่งให้กับ Planner เพื่อใช้ในการควบคุมการทำงานต่อไป ซึ่งประเภทของข้อความต่างๆ ที่จะได้รับอาจแบ่งได้ดังนี้

- ผลหรือข้อมูลที่ได้จากการเรียกให้ Tools ทำงาน
- ข้อมูลที่ TCM แจ้งให้กับ agent ทราบในลักษณะของข่าวสาร
- ข้อมูลที่ได้จาก agent อื่น
- ข้อมูลที่ไม่เกี่ยวข้องกับตัว agent
- อื่นๆ ...

ส่วนรับรู้ (Perception)

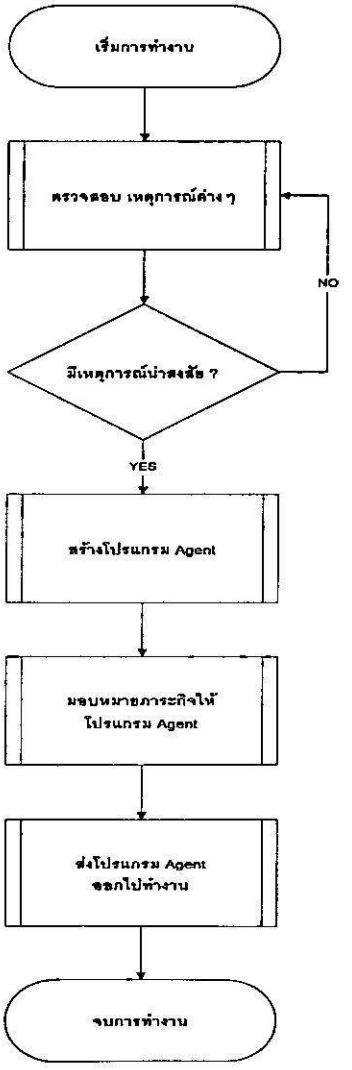
Perception เปรียบเสมือนเป็นส่วนรับรู้ ที่จะต้องคอยตรวจสอบข่าวสารต่างๆ ที่ถูกบันทึกลงไปใน Central Blackboard ของ TCM เพื่อเก็บข่าวสารต่างๆ นั้นส่งให้กับ Evaluator เพื่อนำไปใช้ในการพิจารณาว่าข่าวสารใดควรส่งให้กับส่วนประกอบใดของ โปรแกรม Agent

ส่วนตอบโต้ (Action)

Action จะเป็นส่วนของการทำงานที่ใช้ในการควบคุมการส่งคำขอที่ได้รับจาก Planner ให้กับ TCM ในการเรียกใช้ Tools ต่างๆ โดยการแจ้งไว้ใน Central Blackboard ในรูปแบบการสื่อสารที่เป็นมาตรฐานเพื่อการติดต่อกับ TCM, Agent อื่น, หรือผู้ใช้

3.3 การทำงานของระบบ

การทำงานของระบบ Rule - Base UNIX Security Protection System นี้จะทำงานโดยเริ่มการควบคุมจาก TCM ซึ่งจะคอยตรวจจับการงานที่น่าสงสัยว่าจะเกิดเป็นอันตรายกับระบบก็จะทำการสร้าง Agent ขึ้นเพื่อมอบหมายข้อสงสัยนั้นให้ตรวจสอบ



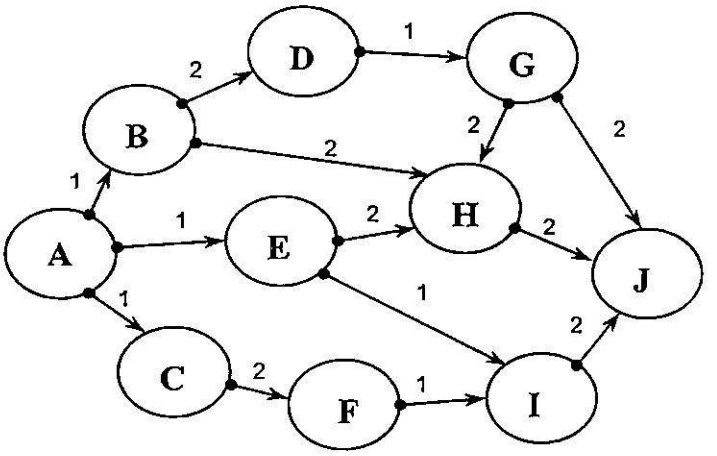
ภาพที่ 3.4 การทำงานของ Task Context Manager (TCM)

3.4 การทำงานของโปรแกรม Agent

โปรแกรม Agent จะถูกสร้างโดย TCM พร้อมกับได้รับการมอบหมาย ความไม่น่าไว้วางใจที่ TCM ต้องการให้ตรวจสอบ จะเริ่มทำงานโดยการรับความต้องการจาก TCM มาแล้วทำการวิเคราะห์ว่าผิดต่อกฎข้อใดหรือไม่โดยจะเริ่มจากการวิเคราะห์หาแผนการทำงานก่อน

3.5 ขั้นตอนการวิเคราะห์แผนการทำงานของ Agent

การวิเคราะห์ความไม่น่าไว้วางใจของระบบตามที่ได้รับมอบหมายมาจาก TCM นั้น Agent จะทำการวิเคราะห์หาแผนการปฏิบัติงานโดยการ map เอา Tools ต่างๆ ที่มีอยู่ในตาราง Tools Table ในลักษณะโครงสร้างของกราฟ พร้อมทั้งกำหนดค่าน้ำหนักในแต่ละเส้นทางไว้ดังแสดงในภาพที่ 3.5



ภาพที่ 3.5 การวิเคราะห์หาแนวทางในการใช้งานเครื่องมือต่าง ๆ

เมื่อทำการวิเคราะห์แนวทางในการวิเคราะห์แผนการทำงานจากรูปจะได้แผนการทำงานสำหรับเส้นทางจาก A -> J ต่อไปนี้

เส้นทางที่ 1	A -> B-> D-> G-> J	= 6
เส้นทางที่ 2	A -> B -> H -> J	= 5
เส้นทางที่ 3	A -> E -> H -> J	= 5
เส้นทางที่ 4	A -> E -> I -> J	= 4
เส้นทางที่ 5	A -> C -> F -> I -> J	= 6

จากทั้ง 5 เส้นทางที่วิเคราะห์ได้จะเห็นได้ว่าเมื่อพิจารณาจากค่าน้ำหนักที่ให้ไว้แล้ว เส้นทางที่น่าจะเลือกคือเส้นทางที่ 4 ซึ่งมีค่าน้ำหนักต่ำสุดคือ 4 จากวิธีการพิจารณาในรูปแบบนี้เองที่เราสามารถนำมารวบรวมสร้างเป็นแผนการปฏิบัติงานสำหรับ Agent ในระบบ UNIX โดยตัวอย่างการเรียกใช้ Agent ในการตรวจสอบเหตุการณ์เช่น

Format : Agent(EventNo,Parameter)

Agent(E002,rach)

จะหมายถึงให้ทำการตรวจสอบผู้ใช้ที่ ทำการ login เข้าสู่ระบบ ตามเงื่อนไขของกฎที่ต้องตรวจสอบตามที่ได้ระบุไว้สำหรับเหตุการณ์นี้ว่ามีการผิดกฎข้อใดหรือไม่

บทที่ 4
ระบบรักษาความปลอดภัยแบบอิงกฎสำหรับยูนิกซ์
(Rule – Base UNIX Security Protection System)

4.1 การสร้างฐานข้อมูลสำหรับระบบ

ในการพัฒนาระบบ Rule – Base UNIX Security Protection System นั้นส่วนสำคัญของระบบที่จะทำให้ระบบสามารถทำงานได้ตามที่ออกแบบในบทที่ผ่านมานั้น คือส่วนของฐานข้อมูลที่ใช้ในการทำงานโดยการบรรจุเอาความรู้และวิธีการต่างๆ ซึ่งได้จากการนำเอาความรู้และประสบการณ์ในการทำงานของผู้บริหารระบบมาใส่ไว้ในฐานข้อมูลเพื่อให้ Agent เลือกวิเคราะห์และทำตามขั้นตอนที่ดีที่สุดในการปฏิบัติงานตามเป้าหมายที่ได้รับ โดยเป้าหมายของการออกแบบจะอยู่ที่การปรับเปลี่ยนขั้นตอนในการทำงานของ Agent ได้ตามสภาพของความรู้, แนวทางในการตรวจสอบ หรือเครื่องมือใหม่ที่ผู้ดูแลระบบทำการปรับเปลี่ยน

การกำหนดรูปแบบและการสร้างส่วนประกอบต่างๆ ที่ใช้สำหรับทำงานร่วมกับระบบนี้ถือเป็นขั้นตอนที่สำคัญในระบบตรวจสอบความปลอดภัยของ UNIX โดยใช้ Agent ซึ่งจำเป็นที่จะต้องทำงานภายใต้สภาวะแวดล้อมที่เหมาะสม

ตามแนวทางของการออกแบบระบบในบทที่ผ่านมา ในสภาพแวดล้อมนี้จะหมายถึงการจัดรูปแบบของแฟ้มข้อมูลของระบบ UNIX ที่เกี่ยวข้องกับการทำงาน และฐานข้อมูลที่สร้างขึ้นเพื่อเก็บข้อมูลการทำงานต่าง ๆ โดยในฐานข้อมูลจะมี Attribute ของข้อมูลที่ใช้ในระบบดังต่อไปนี้

ตารางที่ 4.1 Attribute ของข้อมูลที่ใช้ในระบบ

Attribute Name	Type	Description
RuleNo	Char(4)	รหัสกฎเกณฑ์
RuleName	Char(25)	ชื่อกฎเกณฑ์
AllowValue	Unsigned Int	ค่าที่ยอมให้ได้
RulePurpose	Char(50)	วัตถุประสงค์ของกฎเกณฑ์
ActNo	Char(4)	รหัสการตอบโต้
ActName	Char(25)	ชื่อการตอบโต้

ตารางที่ 4.1 Attribute ของข้อมูลที่ใช้ในระบบ (ต่อ)

Attribute Name	Type	Description
ActSeq	Unsigned Int	ลำดับการตอบโต้
KnowHowNo	Char(4)	รหัสความรู้
KnowHowStat	Char(1)	สถานะความรู้
Weight	Unsigned Int	ค่าน้ำหนัก
ToolNo	Char(4)	รหัสเครื่องมือ
ToolName	Char(25)	ชื่อเครื่องมือ
Command	Char(50)	รูปแบบคำสั่ง
ToolSeq	Unsigned Int	ลำดับการใช้เครื่องมือ
PreValue	Char(25)	ค่าที่ต้องส่งให้กับเครื่องมือ
PostValue	Char(25)	ค่าที่ได้หลังจากการใช้เครื่องมือ
EventNo	Char(4)	รหัสเหตุการณ์
EventName	Char(25)	ชื่อเหตุการณ์
Enable	Char(1)	สถานะของการทำงาน Y/N
SensorCommand	Char(25)	คำสั่งในการติดตั้ง Sensor
SensorRemove	Char(25)	คำสั่งยกเลิกการติดตั้ง Sensor
RuleSeq	Unsigned Int	ลำดับการตรวจสอบกฎเกณฑ์
LogTime	Time	เวลาที่บันทึก
LogDate	Date	วันที่บันทึก
AgentNo	Unsigned Int	รหัส Agent
Knowledge	Char(25)	ความรู้ในการทำงาน
WorkSeq	Unsigned Int	ลำดับการทำงาน
User	Char (15)	รหัสผู้ใช้

ตารางข้อมูลต่างๆ ที่ประกอบขึ้นเป็นฐานข้อมูลที่ใช้สำหรับการทำงานของโปรแกรม Agent จะประกอบด้วยตารางดังต่อไปนี้

ตารางเหตุการณ์ที่ต้องตรวจสอบ (Event Table)

ตารางเหตุการณ์ที่ต้องตรวจสอบ (Event Table) เป็นตารางที่บันทึกเหตุการณ์ต่างๆ ที่จะต้องทำการตรวจสอบเช่น การ Login เข้าสู่ระบบของผู้ใช้

- EventNo
- EventName
- SensorCommand
- Enable

ตารางที่ 4.2 ตัวอย่างข้อมูลในตารางเหตุการณ์ที่ต้องตรวจสอบ

EventNo	EventName	SensorAdd	Enable
E001	LOGIN	EventLogin	Y
E002	STARTUP	EventStart	Y
E003	CHKIDLE	EventChkIdle	Y

ตารางกฎที่ต้องตรวจสอบในแต่ละเหตุการณ์ (EventRule Table)

ตารางกฎที่ต้องตรวจสอบในแต่ละเหตุการณ์ (EventRule Table) จะเป็นตารางที่ระบุว่ามีกฎใดบ้างที่จะต้องตรวจสอบในแต่ละเหตุการณ์ ซึ่งใช้เป็นข้อมูลสำหรับ TCM ในการมอบหมายงานให้กับ Agent

- EventNo
- RuleNo

ตารางที่ 4.3 ข้อมูลในตารางกฎที่ต้องตรวจสอบในแต่ละเหตุการณ์

EventNo	RuleNo
E001	R001
E001	R002
E001	R006

ตารางกฎเกณฑ์ (Rule Table)

ตารางกฎเกณฑ์ (Rule Table) เป็นตารางที่จะบันทึกกฎต่างๆ ที่จะเป็นข้อกำหนดในการใช้งานระบบเช่น การห้ามผู้ใช้ Login พร้อมๆ กันมากกว่า 2 Session.

- RuleNo
- RuleName
- ActNo
- AllowValue
- RulePurpose

ตารางที่ 4.4 ตัวอย่างข้อมูลในตารางกฎเกณฑ์

RuleNo	RuleName	ActNo	Allow Value	RulePurpose
R001	NoRhost	A005	0	Not allow .rhost file
R002	LimitLoginSession	A001	2	Can't Login more than Allow Session
R003	NoSetUIDRoot	A002	0	Not allow SetUID Root file

ข้อมูลในฟิลด์ ActNo จะเป็นเงื่อนไขในการตรวจสอบ ตัวอย่างเช่นการทำตาม Action ที่ A001 ถ้าผลลัพธ์ที่ได้มีค่ามากกว่า 2 ก็จะหมายถึงมีการละเมิดกฎเกณฑ์ R002 คือมีการ Login เข้ามาในระบบของผู้ใช้มากกว่า 2 Session ระบบจะทำงานโดยการตอบโต้ตามรายการ Action ที่ระบุในตาราง FalseRuleAction ต่อไป ซึ่งค่า AllowValue ในตาราง Rule นี้สามารถที่จะเปลี่ยนแปลงได้ตามความต้องการของผู้ดูแลระบบ

ตารางกฎการตอบโต้ (FalseRuleAction Table)

ตารางกฎการตอบโต้ (FalseRuleAction Table) เป็นตารางที่ใช้เก็บข้อมูลของรายการและลำดับของการตอบโต้ (Action) เมื่อมีการละเมิดกฎเกิดขึ้น

- RuleNo
- ActNo
- ActSeq

ตารางที่ 4.5 ตัวอย่างข้อมูลในตารางกฎการตอบโต้

RuleNo	ActNo	ActSeq
R002	A101	1
R002	A102	2
R002	A103	3

จากตาราง เมื่อมีการละเมิดกฎหมายเลข R002 ระบบ จะทำการตอบโต้ โดยทำตาม Action ที่ A101, A102 และ A103 ตามลำดับ

ตารางการทำงาน (Action Table)

ตารางการทำงาน (Action Table) เป็นตารางสำหรับเก็บข้อมูลการดำเนินการตอบโต้ประเภทต่างๆ ที่จะเกิดขึ้นเช่น วิธีการหาว่ามี Sesion ของผู้ใช้กี่ Session

- ActNo
- ActName
- KnowHowNo
- KnowHowStat
- Weight

ตารางที่ 4.6 ตัวอย่างข้อมูลในตาราง Action

ActNo	ActName	KnowHowNo	KnowHowStat	Weight
A001	KillUser	K001	Y	
A002	MailAdmin	K002	Y	
A003	RemoveFile	K003	Y	
A006	NUMLOGIN	K009	Y	

ตารางถ่ายทอดความรู้ (KnowHow Table)

ตารางถ่ายทอดความรู้ (KnowHow Table) เป็นตารางที่ใช้ในการบันทึกขั้นตอนในการทำงานตาม Action ต่างในตาราง Action ว่ามีขั้นตอนอย่างไรบ้าง

- KnowHowNo
- ToolNo
- ToolSeq

ตารางที่ 4.7 ตัวอย่างข้อมูลในตารางถ่ายทอดความรู้

KnowHowNo	ToolNo	ToolSeq
K001	T001	1
K001	T002	2
K001	T003	3
K009	T017	1

ตารางเครื่องมือ (Tool Table)

ตารางเครื่องมือ (Tool Table) เป็นตารางที่ใช้ในการเก็บรายการของเครื่องมือที่มีอยู่ในระบบ ซึ่งอาจจะเป็นชื่อคำสั่งหรือชุดคำสั่งที่ได้จากการบันทึกไว้ของผู้บริหารระบบ

- ToolNo
- ToolName
- Command
- PreValue
- PostValue
- Weight

ตารางที่ 4.8 แสดงตัวอย่างข้อมูลในตารางเครื่องมือ

ToolNo	ToolName	Command	PreValue	PostValue	Weight
T001		ps -ef | grep $UNAME	-	LISTPSEF	
T002		awk '{print $2}'	LISTPSEF	PID	
T003		xargs –I kill –9 {}	PID	KILLUSER	

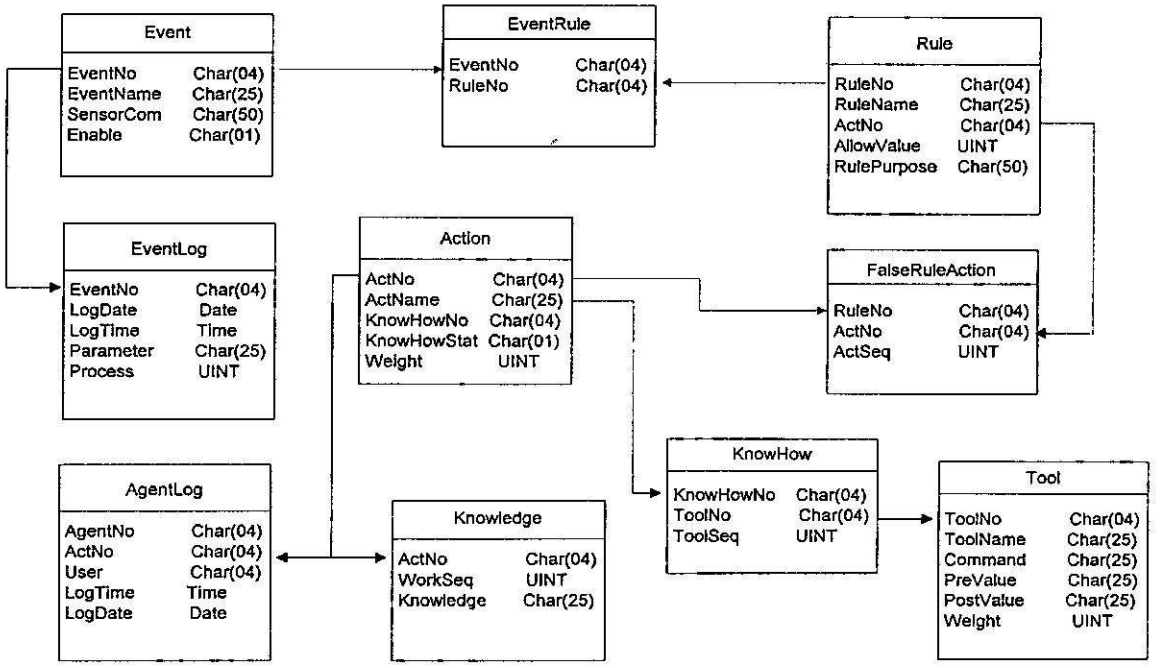
ตารางบันทึกการทำงานของ Agent (AgentLog Table)

ตารางบันทึกการทำงานของ Agent (AgentLog Table) จะเป็นตารางที่เก็บข้อมูลของ Agent แต่ละตัวที่ทำงานอยู่ในระบบว่าทำอะไรไปบ้างเมื่อไหร่

ตารางที่ 4.9 แสดงตัวอย่างข้อมูลในตารางบันทึกการทำงานของ Agent

AgentNo	ActNo	User	LogDate	LogTime
1	A001	rach	1-OCT-1996	12:13:01
2	A002	iti94080	1-OCT-1996	13:15:01
3	A003	somchai	1-OCT-1996	13:18:01

ซึ่งความสัมพันธ์ของตารางข้อมูลต่างๆ จะเป็นดังที่แสดงในภาพต่อไปนี้



ภาพที่ 4.1 ความสัมพันธ์ของข้อมูลที่ใช้ในระบบ

4.2 การพัฒนาส่วน Task Contact Manager (TCM)

การพัฒนาในส่วนของ Task Contact Manager (TCM) นั้นจะทำโดยการใช้กลไกต่างๆ ที่มีอยู่แล้วในระบบ UNIX เป็นหลัก โดยจากโครงสร้างของ TCM ในบทที่ผ่านมาเราสามารถประยุกต์ใช้ส่วนต่างๆ ของระบบได้ดังนี้คือ

- **ตัวดักจับเหตุการณ์** (Sensor)
- Tools Manager

ตัวดักจับเหตุการณ์ Sensor

จะเป็นส่วนที่ใช้การดักจับเหตุการณ์ต่างๆ ตามที่มีระบุอยู่ในตาราง Event ซึ่งจะเป็น Script ในการแทรกขั้นตอนการตรวจสอบไว้ในระบบเช่น

- crontab ใช้ในการตรวจสอบเหตุการณ์ตามระยะเวลาที่กำหนด
- /etc/profile, /csh.login ใช้ในการตรวจสอบการ Login เข้าสู่ระบบของผู้ใช้
- system startup file ใช้ในการตรวจสอบระบบหลังการเปิดเครื่อง

โดยเมื่อมีเหตุการณ์ตามที่ระบุเหล่านั้นเกิดขึ้น ก็จะทำการสร้าง Agent ขึ้นมาเพื่อทำการตรวจสอบทันที โดยที่ Agent ที่สร้างขึ้นนั้นจะทำการตรวจสอบอะไรบ้างก็ขึ้นอยู่กับรายการของกฎเกณฑ์ที่ต้องตรวจสอบในแต่ละเหตุการณ์ โดยเมื่อกฎเกณฑ์ใดถูกละเมิด Agent ก็จะทำการตอบโต้ตามที่ระบุไว้ต่อไป

ตัวจัดการเครื่องมือ (Tools Manager)

ในส่วนของการจัดการเครื่องมือจะใช้ความสามารถของตัวจัดการฐานข้อมูลเข้ามาทำหน้าที่นี้ โดยหน้าที่หลักของตัวจัดการเครื่องมือจะเป็นการจัดเก็บและเรียกใช้เครื่องมือ และข้อมูลต่างๆ ที่มีอยู่ในระบบตามความต้องการของ Agent

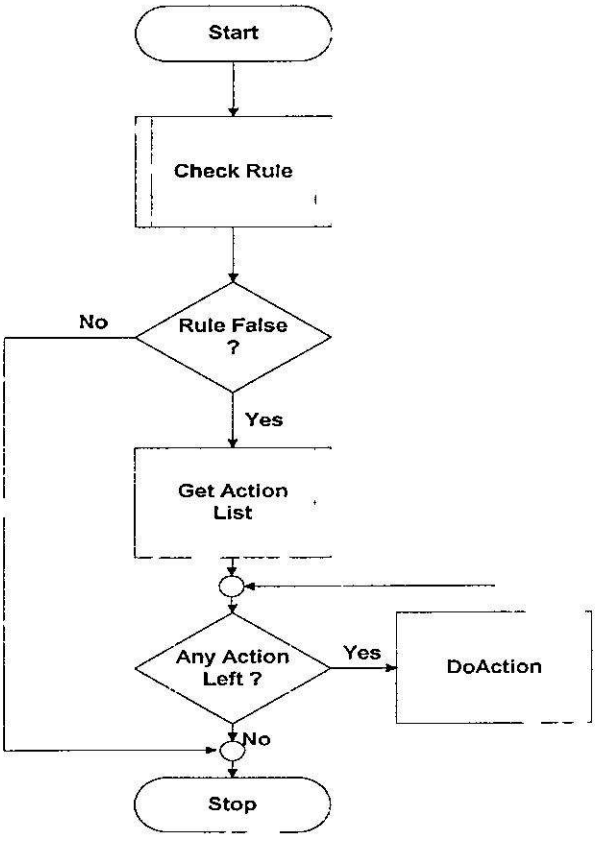
ตัวสร้าง Agent (Agent Launch)

ตัวสร้าง Agent นี้จะเป็นส่วนที่ สร้างและมอบหมายงาน ให้ Agent ไปปฏิบัติ โดยภาระกิจที่จะมอบหมายให้ ก็คือ การตรวจสอบกฎเกณฑ์ตามที่ระบุไว้ในเหตุการณ์นั้น ว่ามีกฎเกณฑ์ใดบ้างที่จะต้องตรวจสอบในเหตุการณ์ที่เกิดขึ้น ซึ่งจะมากหรือน้อยก็ขึ้นอยู่กับผู้ที่บริหารระบบ โดยตัวสร้าง Agent

ก็จะทำการสร้าง Agent เท่ากับจำนวนกฎเกณฑ์ที่ต้องตรวจสอบนั้น เมื่อทำการตรวจสอบพบว่ามีการละเมิดกฎเกณฑ์ก็จะทำการตอบโต้ แต่ถ้าไม่พบก็จะจบการทำงานไป

4.3 การพัฒนาตัวโปรแกรม Agent

ตัวโปรแกรม Agent จะเขียนด้วย Script ที่ไม่มีส่วนของคำสั่งในการทำงานตอบโต้กับระบบหรือเป้าหมายในการทำงานอยู่เลย ซึ่งจะเป็นการสร้างให้ตัว Agent นี้ทำงานให้บรรลุผลตามเป้าหมายที่ได้รับตามแนวทางที่วิเคราะห์ได้เท่านั้น ซึ่งการออกแบบในลักษณะนี้จะทำให้สามารถเปลี่ยนแปลงเป้าหมายในการทำงานและขั้นตอนในการปฏิบัติงานได้ตามรูปแบบใหม่ๆ ที่ได้จากฐานข้อมูลของระบบ ส่วนการทำงานของระบบสามารถเปลี่ยนแปลงได้ตลอดเวลาโดยไม่ต้องแก้ไขตัว Agent เลย ลักษณะการทำงานของ Agent ที่พัฒนาขึ้นจะเป็นดังรูปแบบต่อไปนี้คือ



ภาพที่ 4.2 การโต้ตอบของ Agent

4.4 ตัวอย่างการทำงานของระบบที่พัฒนาขึ้น

ในส่วนนี้จะเป็นตัวอย่างการทำงานร่วมกันของระบบที่ได้ทำการพัฒนาขึ้นตามที่ได้ออกแบบไว้ในบทที่ผ่านมา โดยจะเป็นการตรวจสอบในเหตุการณ์ การ Login เข้าสู่ระบบของผู้ใช้คนหนึ่ง ซึ่ง Login เข้ามาใช้ระบบเป็น session ซึ่งระบบอนุญาตให้มีได้เพียง 2 session เท่านั้น โดยสามารถอธิบายขั้นตอนการทำงานของระบบได้ดังต่อไปนี้

- เมื่อผู้ใช้ Login เข้าสู่ระบบ Sensor สำหรับตรวจจับการ Login เข้ามาใช้งานของผู้ใช้ จะบันทึกข้อมูลการเข้ามาใช้งานของผู้ใช้ว่า เป็นใคร ข้ามาใช้งานเมื่อใด โดยจะบันทึกข้อมูลนี้ลงใน ตาราง EventLog (ตารางที่ใช้บันทึกเหตุการณ์ต่างๆ ที่ตรวจจับได้โดย Sensors) โดยจะบันทึกข้อมูลดังต่อไปนี้ลงไป

EventNo	:	E001
Edate	:	21:Jan:1999
Etime	:	17:32
Parameter	:	rach
Process	:	0

- เมื่อ Sensor ได้ทำการบันทึกข้อมูลลงไปแล้ว Task Context Manager จะอ่านข้อมูล Event ที่เกิดขึ้นนั้นขึ้นมาแล้ว เรียกใช้ส่วนของ Agent Launch (ส่วนของการมอบหมายงานให้กับโปรแกรม Agent) เพื่อมอบหมายงานให้กับ Agent แต่ละตัว โดยดูว่ามี Rule ใดบ้างที่จะต้องตรวจสอบสำหรับ Event ที่เกิดขึ้นนี้ โดยอ่านข้อมูลจากตาราง EventRule ซึ่งจะได้ข้อมูลดังนี้

ตารางที่ 4.10 ข้อมูลที่ต้องตรวจสอบในเหตุการณ์

EventNo	RuleNo
E001	R001
E001	R002
E002	R011

ซึ่งจากข้อมูลในตารางจะเห็นได้ว่า กฎที่ต้องตรวจสอบสำหรับเหตุการณ์ E001 มีด้วยกัน 2 กฎคือ R001 (NoRhost) และ R002 (LimitLoginSession) ส่วน Agent Launch จะมอบหมายงานให้กับ Agent 2 ตัวด้วยกันดังนี้

```
Agent("R001","rach");        /* Agent หมายเลข 1 */
Agent("R002","rach");        /* Agent หมายเลข 2 */
```

Agent ทั้งสองจะแยกกันทำงานในลักษณะที่เป็น Parallel ไม่ขึ้นแก่กัน ดังต่อไปนี้

Agent หมายเลข 1 และ Agent หมายเลข 2 จะทำการตรวจสอบขั้นตอนการทำงานว่าต้องปฏิบัติงานอย่างไรบ้าง โดยดูจากข้อมูลการทำงานในตาราง Rule

ตารางที่ 4.11 ตัวอย่างข้อมูลในตาราง Rule ซึ่งบอกว่าใช้ Action ใดตรวจสอบ

Rule No	Rule Name	Action No	Allow Value	Rule Purpose
R001	NoRhost	A005	0	No Rhost File
R002	LimitLoginSession	A001	2	Check over limit login sessions
R003	NoSetUIDRoot	A002	0	Not Allow any SetUID Root File

จากข้อมูลที่กำหนดในตาราง Agent หมายเลข 1 ซึ่งต้องทำการตรวจสอบกฎหมายเลข R001 จะต้องปฏิบัติตามขั้นตอนใน Action A005

จากข้อมูลที่กำหนดในตาราง Agent หมายเลข 2 ซึ่งต้องทำการตรวจสอบกฎหมายเลข R002 จะต้องปฏิบัติตามขั้นตอนใน Action A001

- เมื่อ Agent ทราบว่าจะต้องปฏิบัติตามขั้นตอนของ Action ใดแล้วก็จะนำหมายเลข Action ที่ได้นั้นไปหาขั้นตอนการทำงาน โดยดูจากตาราง Action ว่าต้องใช้ Know How ใดในการทำงาน ตัวอย่างเช่น Agent หมายเลข 2 ซึ่งจะ

ตารางที่ 4.12 แสดงตัวอย่างข้อมูล KnowHow สำหรับ Action

Action No	Action Name	KnowHow No	KnowHowStat	Weigth
A001	CheckNoOfSession	K001	N	2

โดยจากข้อมูลในตาราง Agent จะปฏิบัติตาม Know How หมายเลข K001

- จากข้อมูลในตาราง Action ทำให้ Agent ทราบว่าต้องปฏิบัติตาม Know How หมายเลข K001 แต่จากตารางค่าของ KnowHowStat มีค่าเป็น N ซึ่งหมายความว่าฐานข้อมูล KnowHow ที่มีอยู่ก่อนแล้วนั้นมีการเพิ่มเครื่องมือใหม่เข้ามาในระบบ (Tools) ถ้ามีค่าเป็น Y หมายถึงยังไม่มีการเพิ่มเครื่องมือใหม่ใด ๆ นั่นหมายความว่าสามารถที่จะใช้ความรู้ที่ Agent อื่นเคยปฏิบัติงานมาก่อนหน้านี้แล้วได้ แต่สำหรับในสถานการณ์เช่นนี้ Agent จะต้องทำการวิเคราะห์เครื่องมือที่มีอยู่ทั้งหมดเพื่อหาแนวทางที่เหมาะสมที่สุดสำหรับการทำงาน

- การวิเคราะห์ความรู้หรือการใช้งาน Tools ของ Agent จะเริ่มจากการพิจารณาความรู้ที่มีอยู่ว่าจะสามารถทำงานให้บรรลุวัตถุประสงค์ได้อย่างไร โดยความรู้ในการทำงานนั้นจะเก็บอยู่ในตาราง Knowledge

ตารางที่ 4.13 ความรู้และลำดับขั้นการทำงานของ Action

Action No	Work Sequence	Knowledge
A001	1	ListUserSession
A001	2	NumberOfLine

จากข้อมูลในตาราง Knowledge จะทำให้ Agent ทราบว่าขั้นตอนในการ ตรวจสอบจำนวนของ Session ที่ผู้ใช้ Login เข้ามาใช้งานระบบนั้นสามารถทำได้โดยใช้ 3 ขั้นตอนตามลำดับของการทำงาน (Work Sequence) คือ

- แสดงรายการ Process ทั้งหมดของผู้ใช้ (List User Session)
- นับจำนวนรายการที่เหลืออยู่ (Count Line)

- จากความรู้ในการทำงานที่ Agent ได้จาก Knowledge ทำให้สามารถที่เลือกใช้ Tools ที่มีอยู่ในตาราง Tools เพื่อที่จะหาทางทำให้บรรลุเป้าหมายได้ตาม Action ได้แล้วโดยเมื่อดูจาก ตาราง Tools

ตารางที่ 4.14 แสดงตัวอย่างข้อมูลที่มีอยู่ในตาราง Tool

Tool No	Tool Name	Command	Pre Value	Post Value	We
T001	ListUserProc	ps au|grep $USER	USER	ListUserProc	3
T002	FilterLoginProc	grep login	ListUserProc	ListUserSession	2
T003	CountLine	wc –l	ListLine	NumberOfLine	2
T004	ListUserSession	who | grep $USER	USER	ListUserSession	2
T005	ListUserSession	w | grep $USER	USER	ListUserSession	1

จากตัวอย่างข้อมูลในตาราง Agent จะคำนวณหาแนวทางในการใช้เครื่องมือเพื่อให้บรรลุเป้าหมายโดยสร้างทางเลือกที่เป็นไปได้ดังนี้

T001(3) ⟶ T002(2) ⟶ T003(2) = 7

T004(2) ⟶ T003(2) = 4

T005(1) ⟶ T003(2) = 3

แต่เมื่อพิจารณาจาก ค่าน้ำหนักของการใช้ Tools ในแต่ละทางเลือกแล้วจะเห็นได้ว่าทางเลือกสุดท้ายเป็นทางเลือกที่ได้ค่าน้ำหนักน้อยที่สุด ซึ่ง Agent จะเลือกเส้นทางนี้ในการทำงานตามเป้าหมาย

- เมื่อได้ Tools Agent จะทำการต่อ Tools ทั้งหมดเข้าเป็นข้อความเดียวกันโดยใช้ | (pipe) เป็นตัวเชื่อมระหว่างชุดคำสั่ง

  w | grep $USER | wc –l

  ซึ่งจะได้จำนวน Session ที่ผู้ใช้ Login เข้ามาใช้งานระบบ คือ 3

- เมื่อได้ค่าจากการทำงานแล้วนำไปเปรียบเทียบกับค่าที่อนุญาตให้มีได้ จะเห็นได้ว่าเกินที่กำหนด ซึ่ง Agent ต้อง ตรวจสอบว่ามีมาตรการอย่างไรในการตอบโต้กับสถานการณ์ ของการละเมิดกฎ

นั้น ๆ โดยดูจากตาราง FalseRuleAction ซึ่งจะเก็บแนวทางในการตอบโต้กับการละเมิดกฎต่าง ๆ เอาไว้

ตารางที่ 4.15 แสดงลำดับขั้นการตอบโต้ของ Agent เมื่อผิดกฎ R002

Rule No	Action No	Action Seq
R002	A101	1
R002	A102	2
R002	A103	3

จากตารางจะเห็นได้ว่าการละเมิดกฎนี้จะมีการตอบโต้ทั้งหมด 3 รายการด้วยกัน คือ A101, A102, A103 ตามลำดับ

- เมื่อได้ลำดับและแนวทางของการตอบโต้แล้ว Agent ก็จะทำการ ตอบโต้ตามรูปแบบที่ได้กำหนดไว้ตามลำดับ ซึ่งผู้ดูแลระบบจะต้องเป็นผู้ที่กำหนดลำดับความสำคัญของการตอบโต้เอง

4.5 ตัวอย่างการสร้างกฎการตรวจสอบระบบ

ในตัวอย่างนี้จะเป็นการเพิ่มข้อมูลสำหรับ เหตุการณ์ของการ Login เข้าสู่ระบบของผู้ใช้เพื่อตรวจสอบว่าผู้ใช้มีการ Login เข้ามาในระบบมากกว่า 2 session หรือไม่ หากผู้ใช้มีการ Login เข้ามาใช้งานมากกว่า 2 session ให้ระบบตอบโต้โดย

- ตัดผู้ใช้ออกจากระบบ
- ส่ง E-mail แจ้งให้ผู้ใช้ทราบว่าได้มีการใช้งานมากกว่าจำนวน session ที่กำหนด

ซึ่งสามารถสร้างรูปแบบของการทำงานในระบบนี้ได้โดยปฏิบัติตามขั้นตอนต่อไปนี้

- เพิ่มข้อมูลในตาราง Event

การเพิ่มข้อมูลในตาราง Event เป็นการเพิ่มข้อมูลสำหรับการตรวจสอบในเหตุการณ์ที่กำหนด ซึ่งถ้ามีการตรวจสอบอยู่แล้วก็ไม่จำเป็นที่จะต้องเพิ่ม Event อีกแต่ถ้ายังไม่มีก็จะต้องมีการเพิ่มในตาราง Event ดังคำสั่งต่อไปนี้

```
INSERT INTO Event VALUES ('E001','LOGIN','EventLogin','Y')
```

- เพิ่มข้อมูลในตาราง EventRule

ตาราง EventRule เป็นตารางที่จะบอกว่าเมื่อเกิดเหตุการณ์นี้ขึ้นแล้วจะต้องมีการตรวจสอบกฎใดบ้างในที่นี้เราจะให้ตรวจสอบกฎหมายเลข R002 ในเหตุการณ์ของการ Login ของผู้ใช้ด้วยจะสามารถทำได้โดย

```
INSERT INTO EventRule  VALUES ('E001','R002')
```

- เพิ่มข้อมูลในตาราง Rule

ตาราง Rule จะมีรายละเอียดข้อมูลของกฎ ว่าเป็นกฎเกี่ยวกับอะไรมีขอบเขตอย่างไรและใช้ Action ใดในการตรวจสอบ ในที่นี้ระบุว่า การ Login มากกว่าจำนวน session ที่กำหนด เป็นกฎหมายเลข R002 ซึ่งสามารถตรวจสอบได้โดยใช้ Action หมายเลข A001 โดยผลที่ได้จากการตรวจสอบต้องไม่เกิน 2 session หากมากว่า 2 จะหมายถึงผิดกฎ

```
INSERT INTO Rule  VALUES ('R002', 'LimitLoginSession', 'A001' ,2,
'Check over limit sessions')
```

- เพิ่มข้อมลในตาราง FalseRuleAction

ในตารางนี้จะเป็นการกำหนดว่าหากมีการผิดกฎแล้วจะให้ระบบตอบโต้ด้วย Action หมายเลขใดบ้าง ซึ่งในที่นี้กำหนดให้ว่าหากมีการละเมิดกฎหมายเลข R002 ให้ตอบโต้โดย Action หมายเลข A901 และ A903

```
INSERT INTO FalseRuleAction  VALUES ('R002','A901',1)
INSERT INTO FalseRuleAction  VALUES ('R002','A903',2)
```

- เพิ่มข้อมูลในตาราง Action

ในตาราง Action นี้จะต้องทำการเพิ่ม Action ทั้งหมด 3 Action โดย A001 เป็น Action ที่ใช้ในการตรวจสอบ ส่วน Action หมายเลข A901 และ A903 เป็น Action ที่ใช้สำหรับการตอบโต้ โดยเมื่อเพิ่ม Action ใหม่จะกำหนดให้ค่าของ KnowHowStat เป็น 0 ก่อน เนื่องจากต้องการให้ Agent ทำการวิเคราะห์ KnowHow ใหม่

```
INSERT INTO Action  VALUES ('A001','CheckNoOfSesstion',0,'N',10)
INSERT INTO Action  VALUES ('A901','KillAllUserProc',0,'N',10)
```

INSERT INTO Action VALUES ('A903','MailWarning',0,'N',10)

- เพิ่มข้อมูลในตาราง ActionKnowledge

ข้อมูลในตาราง ActionKnowledge นี้เป็นข้อมูลที่บอกว่ามีความรู้ใดบ้างที่ใช้ในการทำตาม Action ต่างๆ ในที่นี้ Action A001 มีความรู้หรือแนวทางในการทำงานอยู่สองความรู้ด้วยกันคือ K001 และ K002 ส่วน A901 มีแนวทางในการทำงานตามความรู้หมายเลข K004 และ Action A903 ทำได้โดยใช้ความรู้หมายเลข K004

INSERT INTO ActionKnowledge VALUES ('A001','K001')

INSERT INTO ActionKnowledge VALUES ('A001','K002')

INSERT INTO ActionKnowledge VALUES ('A903','K003')

INSERT INTO ActionKnowledge VALUES ('A901','K004')

- เพิ่มข้อมูลในตาราง Knowledge

ตาราง Knowledge จะบันทึกข้อมูลซึ่งเป็นขั้นตอนในการทำงานซึ่ง Agent จะใช้ความรู้นี้ในการเลือกใช้ Tool จากตาราง Tool ผู้ควบคุมระบบจะต้องทำการบันทึกความรู้นี้โดยใส่ลำดับขั้นของการทำงานด้วยเช่น การตรวจสอบจำนวน session ที่ Login เข้ามาใช้งานในระบบสามารถตรวจสอบได้โดยใช้คำสั่ง ps au | grep USER | wc –l ก็จะประกอบด้วย 2 ขั้นตอนด้วยกันคือ ps au | grep USER (ListUserSession) และ wc –l (NumberOfLine) ซึ่งความรู้เหล่านี้จะถูกบันลงในตาราง Knowledge โดยใช้คำสั่ง

INSERT INTO Knowledge VALUES ('K001',1,'ListUserSession')

INSERT INTO Knowledge VALUES ('K001',2,'NumberOfLine')

INSERT INTO Knowledge VALUES ('K002',1,'ListUserProc')

INSERT INTO Knowledge VALUES ('K002',2,'GrepUserSession')

INSERT INTO Knowledge VALUES ('K002',3,'NumberOfLine')

INSERT INTO Knowledge VALUES ('K003',1,'MailLimitLogin')

INSERT INTO Knowledge VALUES ('K004',1,'ListUserProc')

INSERT INTO Knowledge VALUES ('K004',2,'Column2List')

INSERT INTO Knowledge VALUES ('K004',3,'KillPIDList')

- เพิ่มข้อมูลในตาราง Tool

การเพิ่มข้อมูลในตาราง Tool เป็นขั้นตอนสุดท้ายของการทำงานโดยจะเป็นการเพิ่มคำสั่งสำหรับการทำงานตามความรู้ต่าง ๆ ดังต่อไปนี้

```
INSERT INTO Tool VALUES ('T003','CountLine','wc –l '
,'ListLine','NumberOfLine',3)
INSERT INTO Tool VALUES ('T005','wListUserSession','w | grep
Parameter','USER',ListUserSession',1)
```

บทที่ 5
สรุปผลการวิจัยและข้อเสนอแนะ

5.1 สรุปผลการวิจัย

จากผลการทำงานของของระบบตัวอย่างที่พัฒนาขึ้นจะเห็นได้ว่า เราสามารถที่จะตรวจสอบความไม่น่าไว้วางใจ และดำเนินการตอบโต้กับเหตุการณ์นั้นๆ ได้ทันทีที่เกิดเหตุการณ์นั้นๆ ขึ้นในระบบและเป็นการตรวจสอบแบบทันทีในลักษณะที่เป็นการทำงานพร้อมๆ กันในทุกเงื่อนไขที่ต้องการตรวจสอบ (Parallel check) ซึ่งเป็นแนวทางที่มีความเหมาะสมกับการทำงานของระบบตรวจจับความไม่น่าไว้วางใจมากขึ้นกว่าระบบในรูปแบบที่เป็นการตรวจสอบตามลำดับขั้น เพื่อให้ทันกับการพัฒนาของเทคโนโลยีเมื่อระบบคอมพิวเตอร์มีความเร็วในการประมวลผลมากขึ้นเช่นในปัจจุบัน และระบบคอมพิวเตอร์ที่ใช้งานอยู่มีการต่อเชื่อมกันเป็นเครือข่ายออกไปนอกองค์กรซึ่งถือเป็นความเสี่ยงที่เพิ่มขึ้นในแง่ของความปลอดภัย

นอกจากการตรวจสอบที่เป็นลักษณะของการทำไปพร้อมๆ กันแล้วนั้น ระบบนี้ยังได้พัฒนาให้ตัวระบบสามารถที่จะเรียนรู้และปรับเปลี่ยนการทำงานได้ เมื่อมีการเปลี่ยนหรือเพิ่มความรู้ใหม่ๆ ให้กับตัวระบบ ระบบก็จะสามารถพัฒนาการทำงานของตัวมันเองให้เหมาะสมและ

ผลของระบบที่พัฒนาขึ้นเมื่อเปรียบเทียบกับคุณลักษณะของระบบตรวจจับความไม่น่าไว้วางใจที่ดี (Characteristics of a Good Intrusion Detection System) โดยระบบการตรวจจับความไม่น่าไว้วางใจที่ดีมีคุณสมบัติดังต่อไปนี้

1. จะต้องทำงานอยู่ตลอดเวลาโดยไม่ต้องมีการควบคุมของผู้ดูแลระบบ ในลักษณะที่เป็นการทำงานอยู่เบื้องหลัง (background process) แต่ผู้ที่ควบคุมระบบจะต้องสามารถตรวจสอบการทำงานจากภายนอกได้ตลอดเวลา

   <u>ระบบที่พัฒนาขึ้น</u> สามารถทำได้โดยใช้ความสามารถของ corn ในระบบ UNIX ผู้ควบคุมระบบสามารถตรวจสอบการทำงานได้โดยดูจากข้อมูล Log ต่าง ๆ ที่เก็บไว้ในฐานข้อมูลของระบบ

2. จะต้องเป็นระบบที่เป็น fault tolerant ในความหมายที่ว่าสามารถที่จะทำงานต่อไปได้โดยไม่ต้องทำการสร้างฐานความรู้ (knowledge-base) ทุกครั้งที่เริ่มระบบ

   ระบบที่พัฒนาขึ้น สามารถทำได้โดยใช้ระบบฐานข้อมูลเข้ามาช่วยในการเก็บและเรียกใช้ข้อมูลตามที่ต้องการ ทำให้ไม่ต้องทำการสร้างฐานความรู้ใหม่ทุกครั้งที่เริ่มระบบ โดยระบบจะเป็นโปรแกรมที่เริ่มทำงานเองเมื่อเปิดเครื่อง และการเปลี่ยนแปลงเครื่องมือหรือความรู้ใดๆ ของระบบไม่จำเป็นที่จะต้องเริ่มระบบใหม่

3. ระบบจะต้องมีการตรวจสอบตนเองเพื่อไม่ให้ถูกลบ ทำลาย หรือแทนที่ด้วยโปรแกรมอื่นได้

   ระบบที่พัฒนาขึ้น ไม่มี

4. ต้องใช้ทรัพยากรของระบบให้น้อยที่สุดเท่าที่จะทำได้ เนื่องจากระบบที่ทำให้การทำงานของเครื่องช้าลงจะไม่ได้รับการยอมรับ

   ระบบที่พัฒนาขึ้น ใช้โปรแกรมที่มีอยู่แล้วในระบบ UNIX มาทำงานร่วมกันเพื่อให้เป็นมาตรฐาน สามารถปรับปรุง เปลี่ยนแปลงวิธีการและขั้นตอนในการทำงานได้ง่าย

5. จะต้องมีการตรวจสอบการทำงานที่ผิดไปจากรูปแบบปกติ

   ระบบที่พัฒนาขึ้น มีการสร้าง และเก็บข้อมูลของระบบปกติเพื่อนำไปเปรียบเทียบกับการทำงานของระบบ ในสถานการที่สงสัยว่ามีเหตุผิดปกติเกิดขึ้น

6. จะต้องสามารถปรับแก้ให้เข้ากับระบบได้ง่าย เนื่องจากระบบคอมพิวเตอร์แต่ละระบบมีรูปแบบคำสั่งและกลไกในการป้องกันระบบที่แตกต่างกัน

   ระบบที่พัฒนาขึ้น สามารถแก้ไขและปรับเข้ากับระบบได้ง่ายเนื่องจากคำสั่งต่างๆ ที่ใช้ในระบบเป็นคำสั่งพื้นฐานในระบบ UNIX รวมทั้งตัวโปรแกรมที่ใช้งานก็เป็นโปรแกรมที่ RUN ได้บนระบบ UNIX หลายระบบ

7. ระบบจะต้องสามารถปรับการทำงานให้สอดคล้องกับการเปลี่ยนแปลงของระบบได้ เช่นเมื่อมีการติดตั้งอุปกรณ์ใหม่หรือลงโปรแกรมใหม่ ระบบก็จะต้องเปลี่ยนแปลงข้อมูลตามไปด้วย

ระบบที่พัฒนาขึ้น สามารถปรับการทำงานของระบบได้อย่างง่ายดายโดยการเปลี่ยนข้อมูลในตารางความรู้ (Knowledge table)

8. ต้องมีความชาญฉลาดในการวิเคราะห์

ระบบที่พัฒนาขึ้น ระบบนี้จะมีความชาญฉลาดในการวิเคราะห์และปฏิบัติตามรูปแบบโดยเมื่อมีการเปลี่ยนแปลง หรือมีเครื่องมือใหม่ๆ เข้ามาในระบบ ระบบจะทำการวิเคราะห์ และปรับปรุงขั้นตอนการทำงานให้เหมาะสมอยู่เสมอ

5.2 ประเมินผลการวิจัย

จากรูปแบบในการออกแบบระบบที่ได้นำเสนอนี้ จะเห็นได้ว่าเป็นระบบที่สามารถจะตรวจสอบพบข้อผิดพลาด และสามารถที่จะเรียนรู้ ที่จะใช้ เครื่องมือ หรือแนวทางการตรวจสอบใหม่ๆ ได้ ซึ่งจะเป็นผลดีกับผู้ดูแลระบบในการที่จะช่วยให้ผู้ที่ดูแลระบบสามารถทำงานได้ง่ายขึ้น นอกจากนั้นยังสามารถแลกเปลี่ยนข้อมูลกับผู้ดูแลระบบคนอื่นๆ โดยการแลกเปลี่ยนข้อมูลระหว่างกันนี้ทำได้โดยการแลกเปลี่ยนข้อมูลที่บันทึกลงในตารางต่างๆ ที่มีอยู่ในระบบ ได้อีกด้วย

5.3 ข้อเสนอแนะ

จากผลของการทำงานของระบบและการศึกษาระหว่างการพัฒนาระบบตามรูปแบบที่กำหนดขึ้น จะเห็นได้ว่าระบบยังไม่ได้ระบบที่ดีนักเพียงแต่เป็นการพิสูจน์ให้เห็นว่าสามารถพัฒนาระบบตามรูปแบบที่ได้

1. ในการพัฒนาระบบ IDS ที่เหมาะสมนั้นจำเป็นที่จะต้องคำนึงถึงความเหมาะสมในแง่ของการทำงานของระบบและความเร็วในการทำงานของระบบด้วยซึ่งจากระบบที่พัฒนาขึ้นนี้อาจใช้ค่าน้ำหนักเป็นตัวเลือก เพื่อหาความเหมาะสมได้
2. การรับรู้ข้อความใน Message Board ของการส่งผ่านงานระหว่าง Module ควรเป็นรูปแบบของการ Interrupt ไม่ใช่การ pulling เหมือนในระบบที่พัฒนาขึ้น โดยต้องเป็นการทำงานในรูปแบบนั้นเนื่องจากฐานข้อมูลที่ใช้ (MiniSQL Version 2.6) ไม่สนับสนุนการทำงานแบบ Trigger
3. ควรมีการพัฒนาให้ระบบสามารถที่จะทำการตรวจสอบเครื่องที่เกี่ยวข้องกับเครื่องของตนเองได้ด้วย เช่นมีการส่ง Agent ของตนข้ามไปยังเครื่องอื่นเพื่อตรวจสอบหาข้อมูลที่จะใช้วิเคราะห์เพิ่มเติมตามภารกิจที่ได้รับมอบหมายได้ด้วย (Mobile Agent)

4. ควรพัฒนารูปแบบของการส่งผ่านข้อมูล หรือการแลกเปลี่ยนข้อมูลที่เกี่ยวข้องกับการทำงานของ Agent ที่ปฏิบัติงานอยู่กับ Agent อื่นๆ ที่ได้ผ่านขั้นตอนการทำงานนั้นมาแล้ว (Knowledge sharing) เพื่อจะได้ไม่ต้องเปลือง ทรัพยากรในการทำงานนั้นซ้ำอีก ซึ่งอาจกระทำได้เช่น การแลกเปลี่ยน Know how ในตัวอย่างระบบที่พัฒนาขึ้น แต่ควรลงลึกในระหว่างปฏิบัติงานไม่ใช่รอเมื่อการทำงานสิ้นสุดลงแล้ว
5. ควรมีการแก้ไขแนวทางของการตรวจสอบผลว่าละเมิดกฎในระบบที่พัฒนาใหม่โดยอาจสร้างให้สามารถใส่เงื่อนไขของการตรวจสอบได้มากกว่าระบบนี้ ซึ่งใส่ได้เฉพาะเงื่อนไข มากกว่าค่า AllowValue เท่านั้น

เอกสารอ้างอิง

[1] Larry J. Hughes, Jr "Actually useful Internet security techniques" ,New Riders Publishing ,1995

[2] Bryant R. Bringle, "UNIX Security for the Organization" ,SAMS Publishing,1994

[3] Farrow, R.,"UNIX System Security How to Protect Your Data and Prevent Intruders" Addison-Wesley Publishing Company,Inc.1990

[4] Ritchie, Dennis M. "On the Security of UNIX" May 1975. Reprint in UNIX System Manager's Manual, 4.3 Berkeley Software Distribution. University of California, Berkeley. April 1986

[5] Grammp,F. T., and R. H. Morris, "UNIX Operating System Security" AT&T Bell Laboratories Technical Jounal, 63(8): 1649-1672, October 1984

[6] David A. Curry ,"IMPROVING THE SECURITY OF YOUR UNIX SYSTEM", Information and Telecommunications Sciences and Technology Division Final Report ,April 1990

[7] Federic J. Cooper; Chris Goggans; John K. Halvey; Larry Hughes; Lisa Morgan; Karanjit Siyan; William Stallings; Peter Stephenson, "Implementing Internet Security", New Riders Publishing, 1995

[8] Mark F. Komarinski, Cary Collett, "Linux System Administration Handbook", Prentice Hall PTR, 1998

[9] Etzioni, Oren etc, "OS Agent: Using AI Techniques in the Operating System Environment", University of Washington, 1994

[10] Hayes-Roth. B., "An Architecture for Adaptive Intelligent Systems". Artificial Intelligence: Special Issue on Agents and Interactivity, 72:329-365, 1995.

[11] Michael Wooldrige, Nicolas R. Jennings, "Intelligent Agents: Theory and Practice", Knowledge Engineering Review, January 1995.

[12] Colin G. Harrison, David M. Chess, Aaron Kershenbaum, "Mobile Agents: Are they a good idea", Research report, IBM Research Division, 1995

[13] Brian Jepson, David J. Hughes, "OFFICIAL GUIDE TO MiniSQL 2.0", John Wiley & Sons Inc, 1998

ภาคผนวก

บทความที่ตีพิมพ์ในวารสารวิชาการ

ระบบรักษาความปลอดภัยแบบอัจฉริยะสำหรับยูนิกซ์

Intelligence UNIX Security Protection

อนิราช มิ่งขวัญ*

อาจารย์สุรสิทธิ์ วรรณไกรโรจน์**

ดร.เอื้อน ปิ่นเงิน***

บทคัดย่อ

ถึงแม้จะมีคำกล่าวที่ว่า "ความลับไม่มีในโลก" แต่ทุกองค์กรก็ย่อมจะมีความลับ หรือข้อมูลที่ต้องการใช้ภายในองค์กรเท่านั้น และเป็นที่แน่นอนว่าการรักษาความลับขององค์กรก็เป็นภารกิจหนึ่งที่สำคัญและต้องทำให้ดีที่สุด

บทความฉบับนี้กล่าวถึงมาตรการในการรักษาความปลอดภัยของระบบยูนิกซ์ โดยการรวบรวมชุดคำสั่งที่มีอยู่ในระบบและเครื่องมือต่างๆ จัดทำขึ้นเป็นระบบรักษาความปลอดภัยสำหรับระบบยูนิกซ์ โดยมีขั้นตอนของการตัดสินใจในการป้องกันระบบรวมทั้งการตอบโต้ผู้ที่บุกรุกเข้าสู่ระบบซึ่งใช้ระบบผู้เชี่ยวชาญเป็นตัววิเคราะห์

ABSTRACT

Even if there has one comparing sentence that is "no secret in the world" but every organizations also has a secret or data which is wanted to use only in the organization and absolutely that it is the most important work and have to do the best thing to protect the secret of the organization.

Also saying in this paper is a criterion for protecting security of UNIX System by collecting commands and tools within the system. Which has a step of protection and decision including with retorting the system attacker by using expert system to be analyst.

* นักศึกษาปริญญาโท สาขาวิทยาการคอมพิวเตอร์และเทคโนโลยีสารสนเทศ ภาควิชาคณิตศาสตร์ คณะวิทยาศาสตร์ สถาบันเทคโนโลยีพระจอมเกล้าฉลาดกระบัง

** อาจารย์ประจำ ภาควิชาวิศวกรรมคอมพิวเตอร์ คณะวิศวกรรมศาสตร์ สถาบันเทคโนโลยีพระจอมเกล้าฉลาดกระบัง

*** อาจารย์ประจำ ภาควิชาวิศวกรรมคอมพิวเตอร์ คณะวิศวกรรมศาสตร์ สถาบันเทคโนโลยีพระจอมเกล้าฉลาดกระบัง

1. บทนำ

ระบบปฏิบัติการ UNIX เป็นระบบปฏิบัติการที่ได้รับความนิยม และใช้งานกันอย่างแพร่หลายในปัจจุบันโดยเฉพาะเมื่อเครื่องคอมพิวเตอร์ทั่วโลกเชื่อมกันเป็นเครือข่ายระดับชาติหรือที่รู้จักกันในชื่อของ Internet เมื่อเราสามารถเชื่อมต่อเข้ากับระบบเครือข่ายและรับข้อมูลข่าวสารได้จากทั่วโลกนั่นก็หมายความว่าผู้ใช้จากทั่วโลกก็สามารถที่จะเข้าถึงระบบของเราเช่นเดียวกันและถ้ามองในแง่ร้ายก็หมายถึงผู้ที่จะเจาะเข้าสู่ระบบของเราก็สามารถนั่งอยู่ที่ใดของโลกก็ได้

ถึงแม้ว่าผู้ออกแบบระบบ UNIX จะกล่าวว่าระบบยูนิกซ์ถูกออกแบบมาโดยไม่ได้คำนึงถึงระบบความปลอดภัยก็ตาม (Richie,1986) แต่ในทางปฏิบัติผู้จัดการระบบสามารถที่จะควบคุมระดับความปลอดภัยได้ในระดับที่ตนเองต้องการ (Grampp, Morris, 1984) แต่เนื่องจากระบบยูนิกซ์เป็นระบบที่ซับซ้อน ผู้จัดการระบบอาจมองข้ามในบางจุดไป โดยเฉพาะผู้ที่ยังไม่มีประสพการณ์เพียงพอ

บทความฉบับนี้จะนำเสนอตัวอย่างรูปแบบของโปรแกรมบริหารระบบยูนิกซ์เพื่อให้ผู้ที่เป็นผู้บริหารระบบสามารถที่จะตรวจสอบความปลอดภัยของระบบได้ตลอดเวลาและสามารถแก้ปัญหาของระบบได้อย่างทันท่วงที

2. หลักการ

สิ่งสำคัญสิ่งหนึ่งสำหรับการรักษาความปลอดภัยของระบบยูนิกซ์คือการตรวจสอบความปลอดภัยของระบบซึ่งประกอบไปด้วย

2.1 การตรวจสอบ log file ต่าง ๆ ของระบบเพื่อตรวจสอบหาการเข้าสู่ระบบหรือการกระทำใดๆ ของผู้ใช้ที่ไม่ได้รับอนุญาต

2.1.1. Account Security

Account Security จะต้องตรวจสอบ 2 เรื่องต่อไปนี้คือ การเข้ามาใช้ระบบของผู้ใช้ในช่วงเวลาที่ไม่ปกติ เช่น ในช่วงวันหยุด หรือ ช่วงกลางดึก และ ใช้คำสั่งที่ไม่ค่อยได้ใช้ตามปกติ

2.1.1.1. แฟ้มข้อมูล lastlog

แฟ้มข้อมูล /var/adm/lastlog [Solaris,1993] จะเก็บ วัน, เวลา และสถานที่ ที่ผู้ใช้ login เข้าสู่ระบบ โดยจะแสดงรายละเอียดของการ login ครั้งสุดท้ายที่ผู้ใช้เข้ามาใช้ระบบเพื่อให้ผู้ใช้ตรวจสอบความถูกต้อง เช่น

Last login: Tue Nov 7 02:44:29 from leo.kmitnb.ac.th

2.1.1.2 แฟ้มข้อมูล utmp และ wtmp

แฟ้มข้อมูล /etc/utmp [Solaris, 1993] เป็นแฟ้มข้อมูลที่ใช้เก็บว่าผู้ใช้ใดบ้างที่กำลังใช้ระบบอยู่ โดยใช้คำสั่ง who

```
$ who
chit pts/0 Nov  6 17:27   (202.44.41.18)
ttm37snj  pts/3 Nov 7 03:09   (termsrv.kmitnb.ac.th)
rach pts/6 Nov 7 03:13   (godii.kmitnb.ac.th)
sak pts/1      Nov  6 13:40   (kmitnb04.kmitnb.ac.th)
```

ซึ่งจะเก็บข้อมูล login name, terminal ที่ใช้, เวลาที่เข้าใช้, และใช้จากเครื่องใด

แฟ้มข้อมูล /var/adm/wtmp ซึ่งสามารถที่จะให้ข้อมูลของผู้ใช้ในส่วนของเวลาที่เข้ามาใช้ระบบ (login) ช่วงเวลาที่ใช้ และเวลาที่เลิกใช้ระบบ (logout) โดยใช้คำสั่ง last ดังตัวอย่าง

```
#last
root  console  Sun Aug 20 15:28 - 04:26 (2+12:57)
reboot system boot             Sun Aug 20   15:27
root  console Thu Aug 17  23:37 - 23:37  (00:00)
surasak ftp 202.44.38.213 Thu Aug 17 04:13 - 04:19  (00:05)
surasak pts/2 crimson.kmitnb.a  Thu Aug 17 04:12 - 04:19  (00:07)
pradit  pts/0  202.14.164.123   Mon Aug 14 10:59 - down  (6+04:27)
pradit  pts/1  kmitnb03.kmitnb.  Mon Aug 14 10:58 - 11:00  (00:01)
reboot system boot             Fri Aug      11  00:57
pradit  pts/0 kmitnb03.kmitnb.  Thu Aug 10  03:09 - down  (21:48)
aong    pts/0  202.44.41.2 Wed Aug  9  07:49 - 00:14  (16:24)
sak     pts/0  kmitnb04.kmitnb.  Fri Aug 4 13:55 - 07:49 (4+17:53)
root    console Thu Aug  3 01:25 - 01:26  (00:00)
wtmp begins Thu Aug  3 00:57
```

ซึ่งจะทำให้สามารถใช้ข้อมูลเหล่านี้เป็นข้อมูลพื้นฐานสำหรับการตรวจสอบผู้ใช้ที่ผิดปกติได้

2.1.1.3 แฟ้มข้อมูล acct

แฟ้มข้อมูล /var/adm/acct จะบันทึกข้อมูลของการใช้คำสั่งต่าง ๆ ของผู้ใช้ เช่น ใครคือผู้สั่ง, สั่งเมื่อไหร่, นานเท่าไหร่, ซึ่งข้อมูลเหล่านี้จะถูกบันทึกทุกครั้งที่สิ้นสุดคำสั่ง โดยข้อมูลเหล่านี้จะบันทึกได้บนระบบที่มีการรัน Accounting เท่านั้น

เราสามารถที่จะดูข้อมูลจากแฟ้มข้อมูล acct ได้ โดยใช้คำสั่ง lastcomm ซึ่งจะแสดงข้อมูลของการใช้คำสั่ง เช่น

```
$ lastcomm
pico rach    pts/5  0.22 secs Tue Nov  7 13:31
stty rach    pts/5  0.14 secs Tue Nov  7 13:31
lastcomm rach pts/5  2.83 secs Tue Nov  7 13:31
ls rach      pts/5      0.17 secs Tue Nov  7 13:31
stty rach    pts/5      0.16 secs Tue Nov  7 13:29
mail rach    pts/5      0.25 secs Tue Nov  7 13:29
cat rach     pts/5      0.17 secs Tue Nov  7 13:29
quota rach   pts/5      0.30 secs Tue Nov  7 13:29
pt_chmod S root   __     0.17 secs Tue Nov  7 13:29
in.telne          root   __     2.92 secs Tue Nov  7 13:04
ksh        S      rach   __     1.45 secs Tue Nov  7 13:04
ksh        F      rach pts/6 0.05 secs Tue Nov  7 13:29
sh         S      root pts/6 0.88 secs Tue Nov  7
    13:14
```

ซึ่งจะแสดงข้อมูลของคำสั่งตามลำดับคือ คำสั่ง สถานะของคำสั่ง "F" หมายถึงมีการสร้าง process ขึ้นมา "S" หมายถึงเป็น process ที่เป็น set-user-id "D" หมายถึงการออกจาก process เนื่องจาก core dump "X" หมายถึงการที่ process จบแบบไม่ปกติ ส่วนที่เหลือจะเป็น ชื่อผู้ใช้, terminal, CPU times, วันที่ และเวลาที่ process เริ่มทำงาน

2.2 ความปลอดภัยของระบบเครือข่าย

การตรวจสอบความปลอดภัยของระบบเครือข่ายเป็นเรื่องที่ยุ่งยากและซับซ้อนเนื่องจากผู้ที่ต้องการเจาะเข้าสู่ระบบสามารถที่จะใช้วิธีการได้หลายรูปแบบ และขึ้นอยู่กับตัวระบบเครือข่ายด้วย อย่างไรก็ตามเราก็สามารถที่จะใช้ข้อมูลบางอย่างเพื่อตรวจสอบความปลอดภัยของระบบเครือข่ายได้

2.2.1 ความสามารถของ syslog

syslog เป็นการเก็บข้อมูลความผิดพลาดของระบบ ซึ่งจะแสดงออกที่ console และจะเก็บลงในแฟ้มข้อมูล /var/adm/messages ซึ่งจะมีข้อมูลของความผิดพลาดคือ วัน, เวลา, ชื่อของโปรแกรมที่ส่งข้อความมา และโดยปกติจะมีหมายเลขของ process มาด้วย ตัวอย่างเช่น

```
Nov  5 06:00:38 gemini su: 'su root' failed for surasak on
    /dev/pts/3
Nov  7 13:03:57 gemini login: REPEATED LOGIN
    FAILURES ON /dev/pts/5 FROM 202.44.34.89
Nov  7 13:07:01 gemini su: 'su root' failed for rach on
    /dev/pts/6
Nov  7 13:07:11 gemini login: REPEATED LOGIN
    FAILURES ON /dev/pts/5 FROM 202.44.34.89
Nov  7 13:14:45 gemini su: 'su root' failed for rach on
    /dev/pts/6
Nov  7 13:42:27 gemini login: REPEATED LOGIN
    FAILURES ON /dev/pts/6 FROM 202.44.34.88
Nov  8 01:30:16 gemini su: 'su root' failed for rach on
    /dev/pts/5
```

ข้อมูลในส่วนที่เราสนใจจากตัวอย่างก็คือข้อความที่มาจากโปรแกรม login และ su เมื่อมีผู้ใช้ login เป็น root โปรแกรม login จะแสดงข้อความของการ login นี้ ซึ่งส่วนใหญ่แล้วการ login เป็น root จากที่อื่นมักจะถูกกำหนดให้ไม่สามารถใช้งานได้ เนื่องจากจะทำให้ยากที่จะระบุได้ว่าผู้ใช้คนใด

Login ยังสามารถที่จะเก็บข้อมูลของผู้ที่ login เข้าสู่ระบบไม่สำเร็จถึง 3 ครั้ง ซึ่ง program login จะไม่อนุญาตให้ login อีก การตรวจสอบข้อความ REPEATED LOGIN FAILURES จึงเป็นการตรวจสอบว่ามีใครพยามที่จะเจาะเข้าสู่ระบบโดยการเดารหัสผ่านหรือไม่

สุดท้าย เมื่อมีใครใช้คำสั่ง su ไม่ว่าจะเป็นการเปลี่ยนเป็น root หรือ ผู้ใช้อื่น ได้สำเร็จหรือไม่จะมีการบันทึกไว้ซึ่งทำให้สามารถที่จะตรวจสอบได้ว่ามีผู้ใดพยามที่จะเปลี่ยนเป็นผู้ใช้อื่นหรือไม่

2.2.2 ความสามารถของคำสั่ง showmount

คำสั่ง showmount สามารถใช้แสดงข้อมูลของการ mount ข้อมูลของ NFS server ถ้าไม่มี option จะแสดงข้อมูลของ host ทั้งหมด -a จะแสดงข้อมูลของ host และ directory -d จะแสดงข้อมูลของ directory ทั้งหมดที่มีการ mount ใช้อยู่

ข้อมูลของคำสั่ง showmount จะสามารถตรวจสอบได้ 2 อย่าง คือ เฉพาะ host ที่ได้รับอนุญาตเท่านั้นที่จะปรากฏ จะต้องไม่มีการ mount directory ที่ไม่ได้อนุญาต

2.3 ความปลอดภัยของระบบแฟ้มข้อมูล

การตรวจสอบจุดที่ไม่ปลอดภัยของระบบแฟ้มข้อมูลเป็นอีกส่วนหนึ่งที่สำคัญสำหรับระบบรักษาความปลอดภัย เริ่มจากการตรวจสอบแฟ้มข้อมูลที่สามารถแก้ไขได้โดยผู้ที่ไม่ได้รับอนุญาตซึ่งสามารถที่จะทำให้ผู้ใช้นั้นมีสิทธิสูงขึ้น แฟ้มข้อมูลที่จะทำให้ผู้ที่เจาะเข้าสู่ระบบมีสิทธิสูงขึ้น และควรจะตรวจสอบได้ว่ามีการเปลี่ยนแปลงแฟ้มข้อมูลที่ไม่ได้รับอนุญาต รวมทั้งสามารถที่จะนำแฟ้มข้อมูลเดิมก่อนการเปลี่ยนแปลงกลับมาใช้ได้

2.3.1 คำสั่ง find

คำสั่ง find เป็นคำสั่งที่มีประโยชน์มากในการค้นหาข้อมูลที่ต้องการในระบบยูนิกซ์ ในที่นี้จะใช้คำสั่งนี้เพื่อตรวจสอบหาข้อมูลของจุดที่ไม่ปลอดภัยที่อาจจะเกิดขึ้นได้ในระบบแฟ้มข้อมูล

2.3.1.1 การตรวจสอบแฟ้มข้อมูล setuid และ setgid

หนทางหนึ่งที่ผู้เจาะเข้าสู่ระบบจะสามารถเพิ่มสิทธิของตนได้คือการเรียกใช้โปรแกรมที่จะให้สิทธิพิเศษแก่ผู้ใช้ (setuid or setgid) ซึ่งโดยทั่วไปผู้ที่เคยเจาะเข้าสู่ระบบแล้วมักจะทิ้งโปรแกรมประเภทนี้เอาไว้ที่ใดที่หนึ่งเพื่อการเข้ามาครั้งต่อไป การใช้คำสั่ง find เพื่อช่วยในการตรวจสอบแฟ้มข้อมูลที่ให้สิทธิพิเศษแก่ผู้ใช้ (setuid or setgid) สามารถทำได้ดังนี้

```
# find / -type f -a \( -perm -4000 -o -perm -2000 \) -print
```

ซึ่งหลังจากที่ทำคำสั่งนี้แล้วจะได้รายชื่อของแฟ้มข้อมูลที่มีการให้สิทธิพิเศษแก่ผู้ใช้ ซึ่งจะต้องตรวจสอบโดยเฉพาะแฟ้มข้อมูลที่ไม่ได้อยู่ใน directory ต่อไปนี้

/bin

/etc

/usr/bin

/usr/ucb

/usr/etc

2.3.1.2 การตรวจสอบแฟ้มข้อมูลที่เขียนได้โดยผู้ใช้ทั่วไป

แฟ้มข้อมูลที่สามารถเขียนได้โดยผู้ใช้ทั่วไป จะเป็นหนึ่งในจุดที่ผู้ที่เจาะเข้าสู่ระบบสามารถที่จะแก้ไขหรือเปลี่ยนแปลงการทำงานของแฟ้มข้อมูลได้ตามต้องการซึ่งสามารถตรวจสอบได้โดยการใช้คำสั่ง

```
# find / -perm -2 -print
```

2.3.1.3 การตรวจสอบหาแฟ้มข้อมูลที่ไม่ปกติ

การตรวจสอบแฟ้มข้อมูลที่ไม่ได้มีผู้เป็นเจ้าของแล้ว ซึ่งนอกจากจะเป็นการป้องกันผู้เจาะเข้าสู่ระบบแล้ว ยังเป็นการกำจัดแฟ้มข้อมูลที่ไม่ได้ใช้แล้วให้กับระบบอีกด้วย ซึ่งสามารถหาแฟ้มข้อมูลเหล่านี้ได้โดยการใช้คำสั่ง

```
# find / -nouser -print
```

การใช้ -nouser จะเป็นการแสดงแฟ้มข้อมูลของผู้ใช้ซึ่งไม่มีชื่ออยู่ในแฟ้มข้อมูล /etc/passwd ทางเลือก -nogroup จะเป็นการแสดงแฟ้มข้อมูลที่ไม่มีกลุ่มผู้ใช้ใดเป็นเจ้าของ โดยถ้าต้องการค้นหาแฟ้มข้อมูลที่ไม่มีเจ้าของ หรือ ไม่มีกลุ่มผู้ใช้ใดเป็นเจ้าของก็สามารถกระทำได้โดยการใช้ทางเลือก -o

```
# find / -nouser -o -nogroup -print
```

2.3.1.4 การตรวจสอบแฟ้มข้อมูล .rhosts

ผู้ใช้ไม่ควรที่จะมีการสร้างแฟ้มข้อมูล .rhosts ไว้ใน directory ของตนเองซึ่งสามารถตรวจสอบในส่วนที่เป็น directory ของผู้ใช้ (ไม่ต้องตรวจสอบใน / หรือ /usr) โดยใช้คำสั่ง

```
# find /home -name .rhosts -print
```

ซึ่งจะแสดงแฟ้มข้อมูล .rhosts ที่มีอยู่ภายใต้ /home ออกมา

2.3.2 การสร้างข้อมูลอ้างอิงเพื่อการตรวจสอบ

การสร้างข้อมูลอ้างอิงของระบบเพื่อใช้ในการตรวจสอบในภายหลังสามารถที่ จะช่วยให้การรักษาความปลอดภัยของระบบมีประสิทธิภาพมากขึ้น โดยการเก็บรายละเอียดของแฟ้มข้อมูลที่ไม่อนุญาตให้ผู้ใดแก้ไขเอาไว้แล้วเก็บซ่อนไว้ในส่วนที่ผู้ที่เจาะเข้าสู่ระบบไม่สามารถที่จะค้นพบได้ง่าย ๆ ซึ่งข้อมูลเหล่านี้จะถูกใช้เพื่อการเปรียบเทียบกับการเปลี่ยนแปลงของระบบ โดยการทำข้อมูลอ้างอิงและตรวจสอบนี้สามารถทำได้โดยใช้คำสั่งพื้นฐานของระบบ UNIX คือคำสั่ง ls และคำสั่ง diff

ขั้นตอนแรกคือการใช้คำสั่ง ls เพื่อสร้างแฟ้มข้อมูลไว้สำหรับอ้างอิง ถ้าเป็นการทำไว้ทันทีเมื่อลงระบบในครั้งแรกได้จะเป็นการดี รูปแบบคำสั่ง ls ที่ใช้คือ

```
# ls -aslgr /bin /etc /usr > MasterChecklist
```

แฟ้มข้อมูล MasterChecklist นี้จะเก็บข้อมูลทั้งหมดของ directory ทั้งหมดที่ระบุ ซึ่งจะสามารถตัดเอาแฟ้มข้อมูลที่มีการแก้ไขบ่อย ๆ ออกได้ (เช่น /etc/utmp /usr/adm/acct) โดยจะต้องเก็บแฟ้มข้อมูลนี้ให้ปลอดภัย ซึ่งอาจจะเป็นในแผ่น disk หรือ tape ก็ได้

การตรวจสอบการเปลี่ยนแปลงของแฟ้มข้อมูลที่เปลี่ยนไปในปัจจุบัน สามารถทำได้โดยการสร้างแฟ้มข้อมูลที่เก็บข้อมูลของรายการแฟ้มข้อมูลปัจจุบันแล้วนำไปเปรียบเทียบกับแฟ้ม MasterChecklist ที่เก็บไว้ โดยการใช้คำสั่ง diff ในการเปรียบเทียบ

```
# diff MasterChecklist CurrentList
```

ซึ่งจะบอกถึงส่วนที่เปลี่ยนไปของแฟ้มข้อมูลซึ่งสามารถตรวจสอบได้ การตรวจสอบลักษณะนี้จะทำให้ผู้ที่ดูแลระบบทราบโดยตลอดถึงการเปลี่ยนไปของแฟ้มข้อมูลต่าง ๆ เมื่อตรวจสอบแฟ้มข้อมูล CurrentList จนเป็นที่แน่ใจว่าไม่มีแฟ้มข้อมูลที่สำคัญใดเปลี่ยนแปลงไป หรือมีแฟ้มข้อมูลใดที่เป็นอันตรายต่อระบบแปลกปลอมเข้ามา ก็สามารถที่จะทำการเปลี่ยนชื่อของแฟ้มข้อมูล CurrentList ให้เป็น MasterChecklist เพื่อใช้ในการตรวจสอบครั้งต่อไปได้

2.3.3 การสำรองข้อมูล

การทำการสำรองแฟ้มข้อมูลกระบวนการหนึ่งในระบบรักษาความปลอดภัยที่ไม่ควรจะมองข้าม เนื่องจากการสำรองแฟ้มข้อมูลจะช่วยในการป้องกันแฟ้มข้อมูลที่สำคัญไม่ให้เสียหายจากผู้ที่เจาะเข้าสู่ระบบแล้วยังเป็นการป้องกันการเสียหายเนื่องจากอุบัติเหตุ การเสียหายของอุปกรณ์เครื่อง หรือแม้แต่ความผิดพลาดของผู้ดูแลระบบเอง

การทำการสำรองแฟ้มข้อมูลที่ดีที่สุดคือการสำรองแฟ้มข้อมูลทั้งหมด ซึ่งควรทำการสำรองข้อมูลทั้งระบบอย่างน้อยเดือนละครั้ง การสำรองข้อมูลเฉพาะที่มีการเปลี่ยนแปลงควรทำอย่างน้อย 2 ครั้งต่อสัปดาห์ หรือจะให้ดีควรทำการสำรองข้อมูลทุกวันในตอนท้ายของวัน คำสั่งที่เหมาะสมควรจะใช้คำสั่ง dump มากกว่าคำสั่งอื่น ๆ เช่น tar หรือ cpio เนื่องจากคำสั่ง dump สามารถที่จะนำข้อมูลกลับมาอีกครั้งในสภาวะที่เหมือนกับตอนที่นำขึ้นไป ซึ่งคำสั่งอื่นอาจทำไม่ได้

2.4 ข้อมูลเฉพาะของระบบ

การที่จะตรวจสอบส่วนต่าง ๆ ของระบบดังที่ได้กล่าวมาแล้วนั้น ผู้ที่บริหารระบบจำเป็นที่จะต้องรู้ข้อมูลต่าง ๆ ที่มีอยู่ในระบบเป็นอย่างดีเช่น มี Process อะไรที่ run อยู่บ้าง เวลาที่ผู้ใช้คนใดคนหนึ่งจะ login เข้ามาใช้ และอื่น ๆ อีกมากมายที่ได้จากการเฝ้าดูระบบ ของผู้บริหารระบบ

คำสั่งที่จะใช้ในการตรวจสอบความเป็นไปของระบบ ที่มีในระบบ UNIX มีดังต่อไปนี้

2.4.1 คำสั่ง ps

คำสั่ง ps เป็นคำสั่งที่ใช้แสดงข้อมูลของ process ต่าง ๆ ที่ run อยู่ในระบบ และใครที่เป็นผู้เรียกใช้คำสั่งนั้น ซึ่งจะสามารถเห็นได้ว่า process ใดที่เป็น proceces ที่ไม่สมควรที่จะมีในเวลานั้น

2.4.2 คำสั่ง who และ w

คำสั่ง who เป็นคำสั่งที่ใช้เพื่อแสดงว่าในขณะนั้นมีผู้ใช้ใดที่ใช้ระบบอยู่บ้าง ซึ่งการที่ผู้บริหารระบบใช้คำสั่งนี้เพื่อตรวจสอบบ่อย ๆ จะทำให้ผู้บริหารทราบได้ว่าผู้ใช้คนใดที่เข้ามาใช้งานบ้าง และ มักจะเข้ามาในช่วงเวลาใด

คำสั่ง w เป็นคำสั่งที่รวมเอาความสามารถของ who และ ps เข้าไว้ด้วยกัน โดยจะแสดงข้อมูลของผู้ที่ใช้งานอยู่ เวลาที่ idle และกำลังใช้คำสั่งใดอยู่

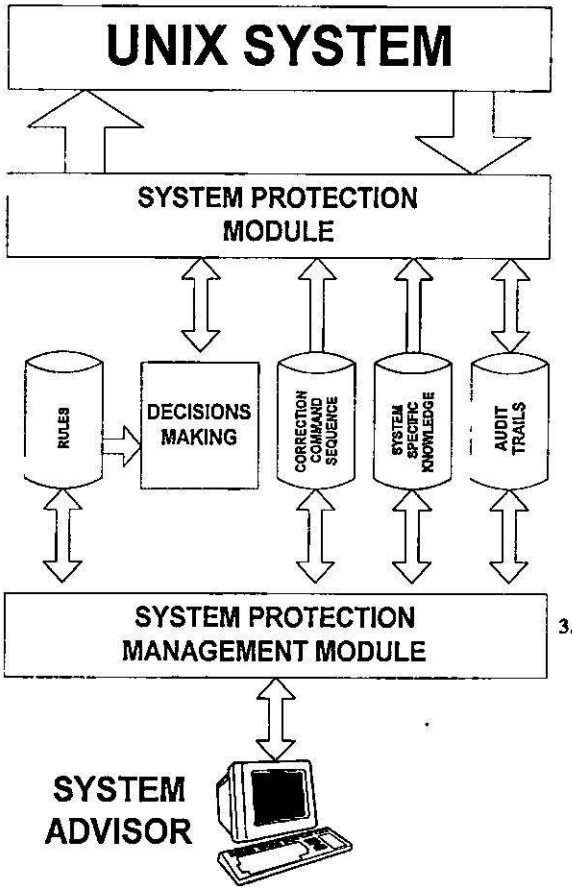
2.4.3 คำสั่ง ls

คำสั่ง ls เป็นคำสั่งที่ใช้ในการแสดงแฟ้มข้อมูลที่มีอยู่ใน directory ต่าง ๆ ซึ่งการตรวจสอบอยู่เป็นประจำจะทำให้ทราบได้ว่ามีแฟ้มข้อมูลใดที่แปลกปลอมเข้ามาบ้าง

ซึ่งในการใช้คำสั่ง ls นี้ควรจะใช้ option -a ซึ่งจะแสดงแฟ้มข้อมูลที่ขึ้นต้นด้วย . ซึ่งจะไม่แสดงผลด้วยคำสั่ง ls ธรรมดา และผู้ที่ต้องการจะทิ้งแฟ้มข้อมูลใดที่ไม่ถูกต้องเอาไว้ ก็มักจะใช้ชื่อแฟ้มข้อมูลประเภทนี้

3.รูปแบบของระบบรักษาความปลอดภัยแบบอัจฉริยะสำหรับระบบยูนิกซ์

ระบบรักษาความปลอดภัยแบบอัจฉริยะสำหรับระบบยูนิกซ์ ที่จะนำเสนอในบทความนี้มีส่วนประกอบดังที่แสดงในรูปที่ 3.1



รูป 3.1 แสดงส่วนประกอบต่างๆ ของระบบรักษาความปลอดภัยแบบอัจฉริยะสำหรับระบบยูนิกซ์

3.1 ส่วนของโปรแกรมสำหรับการตรวจสอบและป้องกันระบบ (System Protection Module)

ส่วนของโปรแกรมสำหรับการตรวจสอบและป้องกันระบบนี้จะเป็นส่วนที่ควบคุมโดยตรงกับโปรแกรมปฏิบัติการ (Operating System) ของระบบ UNIX โดยเน้นการใช้คำสั่งทั่ว ๆ ไปที่มีอยู่ในระบบอยู่แล้วมาสร้างเป็นส่วนป้องกันระบบ ซึ่งแยกได้เป็น 2 ส่วนใหญ่ๆ คือ

3.1.1 ส่วนของการป้องกันระบบ

3.1.2 ส่วนของการเก็บรวบรวมข้อมูลของระบบ

3.2 ส่วนของการตัดสินใจ (Decision Making)

ส่วนของการตัดสินใจนี้จะเป็นส่วนที่ถูกเรียกใช้โดยส่วนของโปรแกรมสำหรับการป้องกันระบบ (System Protection Module)เพื่อทำการวิเคราะห์ข้อมูลต่าง ๆ ที่เก็บมาได้ว่าระบบอยู่ในสภาวะที่ปลอดภัยหรือไม่โดยใช้ข้อมูล จากกฎของการตัดสินใจ (Rules) ประกอบกับ ข้อมูลเฉพาะของระบบ (System Specific Knowledge) และแฟ้มข้อมูลที่บันทึกการทำงานของระบบ(Audit Trails) เพื่อตัดสินใจว่าควรจะกระทำหรือตอบได้กับเหตุการณ์ที่เกิดขึ้นกับระบบอย่างไรดีเช่น

IF INSECURE=3

THEN CALL KILLUSER(%userid)

3.3 แฟ้มข้อมูลกฎการตัดสินใจ (Rules)

ส่วนของแฟ้มข้อมูลกฎการตัดสินใจ จะเป็นแฟ้มข้อมูลที่เก็บเอากฎต่าง ๆ ที่มีอยู่เอาไว้เพื่อวิเคราะห์ว่าเกิดอะไรขึ้น โดยกฎต่างๆ เหล่านี้จะได้จากการป้อนเข้าไปโดยผู้ที่บริหารระบบ เช่น

IF ผู้ใช้ไม่มีสิทธิ su เป็น root

THEN INSECURE=3

3.4 แฟ้มข้อมูลของชุดคำสั่งที่ใช้ในการแก้ไขระบบ (Correction Command Sequence)

ชุดคำสั่งที่ใช้ในการแก้ไขระบบ จะเป็นส่วนของการเก็บคำสั่งและรูปแบบ เพื่อใช้ตอบได้กับเหตุการณ์ต่าง ๆ โดยจะถูกเรียกใช้โดย ส่วนของการป้องกันระบบ (System Protection Module) ตามแนวทางที่ส่วนของการตัดสินใจวิเคราะห์ได้ เช่น

KILLUSER

kill -9 %1

3.5 แฟ้มข้อมูลเฉพาะของระบบ (System Specific Knowledge)

เป็นส่วนที่ใช้เก็บข้อมูลเฉพาะของระบบที่แตกต่าง จากระบบทั่ว ๆ ไป หรือข้อยกเว้นต่าง ๆ ที่มีอยู่ในระบบที่แตกต่างไปจากระบบ UNIX ปกติ

- ใครบ้างที่มีสิทธิในการ su เป็น root
- รายชื่อแฟ้มที่เป็น SUID SGID
- รายชื่อของผู้ใช้และช่วงเวลาที่เข้ามาใช้

3.6 แฟ้มข้อมูลที่บันทึกการทำงานต่าง ๆ ของระบบ (Audit Trails)

เป็นแฟ้มข้อมูลที่บันทึกการทำงานต่าง ๆ ของระบบที่ได้จากระบบ UNIX ทั่ว ๆ ไป เช่น

- การบันทึกการ login เข้ามาใช้งาน
- คำสั่งที่ผู้ใช้เรียกใช้

3.7 ส่วนของการบริหารระบบ (System Protection Management Module)

เป็นส่วนที่ติดต่อกับผู้บริหารระบบเพื่อการตรวจสอบสถานการณ์ของระบบ ป้อนข้อมูลต่างๆ ให้กับระบบ เช่น ชุดคำสั่ง หรือ กฎการตรวจสอบใหม่

4. บทสรุป

จากรูปแบบที่ได้นำเสนอจะเห็นได้ว่าระบบ UNIX เป็นระบบที่มีคำสั่งและแนวทางในการตรวจสอบมากมายซึ่งเป็นการแสดงให้เห็นว่าผู้ที่บริหารระบบสามารถที่จะตรวจสอบและระวังป้องกันการบุกรุกเข้าสู่ระบบได้เป็นอย่างดี โดยการรวบรวมคำสั่งพื้นฐานที่มีอยู่แล้วในระบบเพื่อตรวจสอบและติดตามค้นหาส่วนที่อาจจะเป็นอันตรายกับระบบหรือตัวผู้ใช้ จากรูปแบบของระบบที่ได้นำเสนอในส่วนที่ 3 นั้นเป็นแนวความคิดสำหรับองค์ประกอบของระบบที่จะทำให้ระบบยูนิกซ์สามารถที่จะป้องกันตัวเองได้ โดยสามารถรับเงื่อนไขจากผู้ที่เป็นผู้บริหารระบบซึ่งสามารถที่จะกำหนดเงื่อนไขต่างๆ ได้ตามที่ต้องการ, สามารถที่จะกำหนดระยะเวลาที่เหมาะสมในการตรวจสอบโดยขึ้นอยู่กับความเหมาะสมซึ่งแต่ละระบบไม่เหมือนกัน, สามารถที่จะกำหนดเงื่อนไขการตอบโต้กับผู้บุกรุกได้ ทำให้สามารถที่ป้องกันความเสียหายของระบบได้อย่างทันท่วงที รวมทั้งสามารถแลกเปลี่ยนแฟ้มข้อมูลของ กฎการตรวจสอบ และแนวทางป้องกันกับผู้ที่ใช้ระบบเหมือนกันได้

ปัญหาและอุปสรรคที่พบจากรูปแบบของระบบนี้ คือช่วงเวลาที่กำหนดในการตรวจสอบอาจจะมีเวลามากพอที่ผู้บุกรุกที่ทราบว่ามีระบบนี้ติดตั้งอยู่จะสามารถเปลี่ยนเป็น root และสั่งหยุดการตรวจสอบไปก่อน

เอกสารอ้างอิง

[1] Larry J. Hughes, Jr "Actually useful Internet security techniques" ,New Riders Publishing , 1995

[2] Bryant R. Bringle, "UNIX Security for the Organization" SAMS Publishing,1994

[3] Farrow, R.,"UNIX System Security How to Protect Your Data and Prevent Intruders" Addison-Wesley Publishing Company,Inc.1990

[4] Ritchie, Dennis M. "On the Security of UNIX" May 1975. Reprint in UNIX System Manager's Manual, 4.3 Berkeley Software Distribution. University of California, Berkeley. April 1986

[5] Grammp,F. T., and R. H. Morris, "UNIX Operating System Security" AT&T Bell Laboratories Technical Jounal, 63(8): 1649-1672, October 1984

[6] David A. Curry ,"IMPROVING THE SECURITY OF YOUR UNIX SYSTEM", Information and Telecommunications Sciences and Technology Division Final Report ,April 1990

ประวัติผู้เขียน

นายอนิราช มิ่งขวัญ เกิดเมื่อวันที่ 7 สิงหาคม พ.ศ. 2512 ที่อำเภอท้ายเหมือง จังหวัดพังงา เป็นบุตรคนแรกของนายรุ่ง และ นางประนอม มิ่งขวัญ สำเร็จการศึกษาระดับมัธยมศึกษาที่โรงเรียนเทพลีลา ในปีการศึกษา 2530 สำเร็จการศึกษาระดับปริญญาตรี หลักสูตรวิทยาศาสตรบัณฑิต สาขาวิทยาการคอมพิวเตอร์ ในปีการศึกษา 2534 จากสถาบันเทคโนโลยีพระจอมเกล้าพระนครเหนือ และเข้าศึกษาในหลักสูตรวิทยาการคอมพิวเตอร์และเทคโนโลยีสารสนเทศที่สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง เมื่อ พ.ศ. 2535