การใช้มิสยูสเพื่อประเมินความมั่นคงปลอดภัยในการติดตั้งใช้งานซอฟต์แวร์ประยุกต์

นางสาวขวัญชนก ลิ่มบัณฑิต

วิทยานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิทยาศาสตรมหาบัณฑิต
สาขาวิชาวิศวกรรมซอฟต์แวร์ ภาควิชาวิศวกรรมคอมพิวเตอร์
คณะวิศวกรรมศาสตร์ จุฬาลงกรณ์มหาวิทยาลัย
ปีการศึกษา 2554

MISUSE FOR SECURITY ASSESSMENT IN APPLICATION SOFTWARE DEPLOYMENT

Miss Kwanchanok Limbandit

A Thesis Submitted in Partial Fulfillment of the Requirements

for the Degree of Master of Science Program in Software Engineering

Department of Computer Engineering

Faculty of Engineering

Chulalongkorn University

Academic Year 2011

หัวข้อวิทยานิพนธ์	การใช้มิสยูสเพื่อประเมินความมั่นคงปลอดภัยในการติดตั้งใช้งานซอฟต์แวร์ประยุกต์
โดย	นางสาวขวัญชนก ลิ่มบัณฑิต
สาขาวิชา	วิศวกรรมซอฟต์แวร์
อาจารย์ที่ปรึกษาวิทยานิพนธ์หลัก	อาจารย์ ดร. ยรรยง เต็งอำนวย

คณะวิศวกรรมศาสตร์ จุฬาลงกรณ์มหาวิทยาลัย อนุมัติให้นับวิทยานิพนธ์ฉบับนี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญามหาบัณฑิต

..... คณบดีคณะวิศวกรรมศาสตร์
(รองศาสตราจารย์ ดร. บุญสม เลิศหิรัญวงศ์)

คณะกรรมการสอบวิทยานิพนธ์

..... ประธานกรรมการ
(ผู้ช่วยศาสตราจารย์ นครทิพย์ พร้อมพูล)

..... อาจารย์ที่ปรึกษาวิทยานิพนธ์หลัก
(อาจารย์ ดร. ยรรยง เต็งอำนวย)

..... กรรมการภายนอกมหาวิทยาลัย
(ผู้ช่วยศาสตราจารย์ ดร. เบญจพร ลิ้มธรรมาภรณ์)

ขวัญชนก ลิ่มบัณฑิต : การใช้มิสยูสเพื่อประเมินความมั่นคงปลอดภัยในการติดตั้งใช้งานซอฟต์แวร์ประยุกต์. (MISUSE FOR SECURITY ASSESSMENT IN APPLICATION SOFTWARE DEPLOYMENT) อ. ที่ปรึกษาวิทยานิพนธ์หลัก : อ. ดร. ยรรยง เต็งอำนวย, 82 หน้า.

ฮาร์ดเดนนิ่ง เป็นกระบวนการหนึ่งในการรักษาความมั่นคงปลอดภัยให้กับระบบ แต่การประเมินสถานะฮาร์ดเดนนิ่งของระบบเป็นเรื่องที่ทำได้ยากและต้องอาศัยผู้เชี่ยวชาญด้านความมั่นคงปลอดภัย อีกทั้งโดยปกติแล้ว ความต้องการด้านฮาร์ดเดนนิ่ง เป็นความต้องการที่ไม่ใช่เชิงหน้าที่ จึงทำให้ในบางครั้งอาจถูกละเลยไปได้ ดังนั้นจึงเป็นสิ่งสำคัญในการหาแนวทางในการแก้ไขปัญหานี้

งานวิจัยนี้ ใช้หลักการมิสยูสมาทำการวิเคราะห์ความต้องการด้านฮาร์ดเดนนิ่งของระบบโดยรวมเป็นส่วนหนึ่งในแผนภาพยูสเคส ซึ่งแสดงออกมาในรูปแผนภาพมิสยูสเคสอย่างเป็นทางการเพื่อช่วยในการออกแบบระบบ ทั้งนี้สามารถทำให้เปลี่ยนความต้องการที่ไม่ใช่เชิงหน้าที่ให้เป็นความต้องการเชิงหน้าที่ได้ โดยเข้าทำการประเมินสถานะฮาร์ดเดนนิ่งของระบบอย่างอัตโนมัติในขณะติดตั้งซอฟต์แวร์ประยุกต์ โดยไม่จำเป็นต้องอาศัยผู้เชี่ยวชาญด้านความมั่นคงปลอดภัย ทั้งนี้เพื่อเป็นการสร้างความเชื่อมั่นว่า จะไม่ติดตั้งซอฟต์แวร์ประยุกต์ในระบบที่ไม่มั่นคงปลอดภัย

ทั้งนี้งานวิจัยนี้ ได้วิเคราะห์ตัวอย่างฮาร์ดเดนนิ่งในสี่กรณี ได้แก่ การแยกหน่วยความจำชุดคำสั่งและหน่วยความจำข้อมูล การกำหนดสิทธิ์ของผู้ใช้ การตั้งค่าไฟร์วอลล์ และการเก็บบันทึกการใช้งานระบบ โดยได้นำเสนอเป็นแบบรูปความมั่นคงปลอดภัยเพื่อสะดวกต่อการประยุกต์ใช้งาน โดยสามารถนำไปรวมเป็นส่วนหนึ่งในการออกแบบร่วมกันกับความต้องการด้านอื่นของซอฟต์แวร์ประยุกต์ และพัฒนาเป็นโมดูลการประเมินความมั่นคงปลอดภัย โดยเป็นส่วนหนึ่งของโมดูลการติดตั้งของซอฟต์แวร์

ภาควิชา วิศวกรรมคอมพิวเตอร์ ลายมือชื่อนิสิต .....

สาขาวิชา วิศวกรรมซอฟต์แวร์ ลายมือชื่อ อ. ที่ปรึกษาวิทยานิพนธ์หลัก .....

ปีการศึกษา 2554

5270734021 : MAJOR SOFTWARE ENGINEERING

KEYWORDS : MISUSE / HARDENING / INSTALLATION / SECURITY PATTERN

KWANCHANOK LIMBANDIT : MISUSE FOR SECURITY ASSESSMENT IN APPLICATION SOFTWARE DEPLOYMENT. ADVISOR : YUNYONG TENG-AMNUAY, Ph.D., 82 pp.

Hardening is a process of enhancing the security of the system but is difficult to assess and usually depends on the expertise and care of system administrator. The hardening requirement is non-functional and can be overlooked.

This research employed misuse, in use case diagram, to analyse the hardening requirement to be included in misuse case diagram. This changes non-functional requirement to functional ones. The assessment will happen automatically during software deployment without relying on system administrator in order to make sure that software is not installed in unsecured system.

This work analysed four examples of system hardening: program and data memory separation, limiting user privileges, configuring firewall, and loggings. We also employed security patterns so it will be easy to be incorporated into the software design process and implemented as part of the deployment module.

Department : Computer Engineering Student's Signature .....

Field of Study : Software Engineering Advisor's Signature .....

Academic Year : 2011

กิตติกรรมประกาศ

วิทยานิพนธ์ฉบับนี้สำเร็จลุล่วงไปได้ด้วยความช่วยเหลือจาก อาจารย์ ดร. ยรรยง เต็งอำนวย อ.ที่ปรึกษาวิทยานิพนธ์ รวมถึงผู้ช่วยศาสตราจารย์ นครทิพย์ พร้อมพูล และผู้ช่วยศาสตราจารย์ ดร. เบญจพร ลิ้มธรรมาภรณ์ อ.กรรมการสอบวิทยานิพนธ์ ขอขอบพระคุณที่ท่านได้ให้คำปรึกษาแนะนำต่างๆ อันเป็นประโยชน์ยิ่ง ตลอดระยะเวลาของการจัดทำวิทยานิพนธ์

ขอขอบคุณพี่ๆ ห้องธุรการภาควิชาวิศวกรรมคอมพิวเตอร์ บัณฑิตคณะวิศวกรรมศาสตร์ และบัณฑิตวิทยาลัย ที่ให้ความอนุเคราะห์ในเรื่องต่างๆ และเป็นที่ปรึกษาในการจัดทำวิทยานิพนธ์

ขอขอบคุณห้อง ISEL และสมาชิก สำหรับอุปกรณ์ที่ใช้ประกอบการวิจัย และบรรยากาศดีๆ ของการทำงานวิจัย รวมถึงคำแนะนำ และข้อเสนอแนะต่างๆ

ขอขอบคุณ คุณพ่อ คุณแม่ ครอบครัว และเพื่อนๆ สำหรับการสนับสนุน และความเข้าใจในทุกๆ เรื่อง รวมถึงคำแนะนำดีๆ ที่ใช้ได้ผลเสมอมา

สุดท้ายนี้ ขอขอบคุณทุกกำลังใจอื่นๆ ที่ทำให้การทำงานวิจัยสำเร็จลงได้ด้วยดี

สารบัญ

หน้า

บทที่ หน้า

บทที่ หน้า

สารบัญตาราง

หน้า

สารบัญภาพ

หน้า

บทที่ 1
บทนำ

1.1. ความเป็นมาและความสำคัญของปัญหา

ด้วยความก้าวหน้าทางด้านเทคโนโลยีคอมพิวเตอร์ที่มีมากยิ่งขึ้น หลายๆ องค์กรทั้งภาครัฐและเอกชน จึงได้นำเทคโนโลยีคอมพิวเตอร์มาใช้ประโยชน์ในการทำงานขององค์กร สารสนเทศ และทรัพยากรขององค์กร จึงเป็นปัจจัยสำคัญสำหรับการทำงานขององค์กร เนื่องจากมีผลต่อการดำเนินงานขององค์กร แต่จากจำนวนภัยคุกคาม และการโจมตีระบบคอมพิวเตอร์ที่มีหลากหลายรูปแบบและมีจำนวนมากขึ้นทุกปี ทำให้ปัญหาด้านความมั่นคงปลอดภัยของระบบคอมพิวเตอร์ในองค์กรต่างๆ ทวีความรุนแรงมากขึ้น ดังนั้นระบบงานในปัจจุบันจึงจำเป็นที่จะต้องมีระบบรักษาความมั่นคงปลอดภัยที่แข็งแกร่ง เพื่อเป็นการปกป้องสารสนเทศ และทรัพยากรขององค์กร

หัวใจสำคัญประการหนึ่งในกระบวนการรักษาความมั่นคงปลอดภัยคือ การตรวจสอบความมั่นคงปลอดภัยของระบบที่จะนำซอฟต์แวร์ประยุกต์นั้นไปติดตั้ง ซึ่งความมั่นคงปลอดภัยของระบบที่จะรองรับซอฟต์แวร์ประยุกต์มีส่วนสำคัญในการเสริมสร้างความมั่นคงปลอดภัยให้กับซอฟต์แวร์ประยุกต์นั้นด้วย แม้ว่าในปัจจุบันจะมีแนวทางการพัฒนาซอฟต์แวร์ที่มั่นคงปลอดภัย เช่น ระบบ STRIDE และ DREAD ของบริษัทไมโครซอฟต์ [1] แต่การนำซอฟต์แวร์ประยุกต์เหล่านี้ไปติดตั้งกลับเป็นอีกขั้นตอนหนึ่งแยกจากการพัฒนาตัวซอฟต์แวร์เหล่านั้น เพราะความต้องการด้านความมั่นคงปลอดภัยเป็นความต้องการที่ไม่ใช่เชิงหน้าที่ ทั้งนี้โดยปกติแล้วความมั่นคงปลอดภัยของระบบจะตกอยู่ในความดูแลของผู้ดูแลระบบ (System Administrator) ซึ่งมีระดับทักษะความรู้ความชำนาญที่แตกต่างกันไป ดังนั้นจึงเป็นสิ่งสำคัญที่ต้องหาแนวทางในการเปลี่ยนความต้องการด้านความมั่นคงปลอดภัย ซึ่งเป็นความต้องการที่ไม่ใช่เชิงหน้าที่ ให้เป็นความต้องการเชิงหน้าที่

งานวิจัยนี้ ใช้หลักการมิสยูสในแผนภาพยูสเคสมาวิเคราะห์และรวบรวมความต้องการด้านความมั่นคงปลอดภัยเพื่อเปลี่ยนความต้องการที่ไม่ใช่เชิงหน้าที่เป็นความต้องการเชิงหน้าที่ในรูปแบบของแผนภาพมิสยูสเคส โดยมุ่งเน้นความต้องการด้านความมั่นคงปลอดภัยเกี่ยวกับสถานะฮาร์ดเดนนิ่งของระบบในขณะติดตั้งซอฟต์แวร์ประยุกต์เป็นกรณีศึกษา เพื่อเป็นการสร้างความเชื่อมั่นว่า มีการประเมินความมั่นคงปลอดภัยของระบบที่จะรองรับซอฟต์แวร์

ประยุกต์โดยอัตโนมัติในขณะติดตั้งใช้งานซอฟต์แวร์ประยุกต์ โดยไม่ต้องพึ่งพาผู้เชี่ยวชาญด้านความมั่นคงปลอดภัย ทั้งนี้จะได้สร้างความมั่นใจว่า ไม่ติดตั้งซอฟต์แวร์ประยุกต์ในระบบที่ไม่มั่นคงปลอดภัย

ในงานวิจัยนี้ได้วิเคราะห์ตัวอย่างฮาร์ดเดนนิ่งในสี่กรณี ได้แก่ การแยกหน่วยความจำชุดคำสั่งและหน่วยความจำข้อมูล การกำหนดสิทธิ์ของผู้ใช้ การตั้งค่าไฟร์วอลล์ และการเก็บบันทึกการใช้งานระบบ ซึ่งเกี่ยวข้องกับการโจมตีสี่กรณี ได้แก่ หน่วยความจำ การควบคุมการเข้าถึง การเข้าถึงระบบ และบันทึกการใช้งาน ตามลำดับ โดยมีแนวทางในการนำเสนอความต้องการเชิงหน้าที่ในรูปแบบของแบบรูปความมั่นคงปลอดภัยเกี่ยวกับฮาร์ดเดนนิ่งสี่กรณีเป็นตัวอย่าง ได้แก่ แบบรูปการแยกหน่วยความจำชุดคำสั่งและหน่วยความจำข้อมูล แบบรูปการกำหนดสิทธิ์ของผู้ใช้ แบบรูปการตั้งค่าไฟร์วอลล์ และแบบรูปการเก็บบันทึกการใช้งานระบบ ทั้งนี้เพื่อให้สะดวกต่อการประยุกต์ใช้แบบรูปความมั่นคงปลอดภัย ซึ่งเป็นความต้องการเชิงหน้าที่ที่เกี่ยวข้องกับการประเมินฮาร์ดเดนนิ่งของระบบในขณะติดตั้งซอฟต์แวร์ประยุกต์ให้สามารถรวมเข้ากับความต้องการเชิงหน้าที่ด้านอื่นๆของซอฟต์แวร์ประยุกต์ โดยแบบรูปความมั่นคงปลอดภัยนี้จะเป็นอิสระจากซอฟต์แวร์ประยุกต์นั้น

ทั้งนี้ได้กำหนดแบบรูปความมั่นคงปลอดภัยในรูปแบบของชื่อแบบรูปความมั่นคงปลอดภัย บริบท ปัญหาที่เกิดขึ้นในรูปแบบของแผนภาพมิสยูสเคส (Misuse Case Diagram) พร้อมคำอธิบายมิสยูสเคส (Misuse Case Description) การแก้ไขปัญหา โครงสร้างการแก้ไขปัญหาในรูปแบบแผนภาพคลาส (Class Diagram) พร้อมบัตรซีอาร์ซี (CRC Cards) ไดนามิกการทำงานเพื่อแก้ปัญหาในรูปแบบแผนภาพลำดับ (Sequence Diagram) พร้อมทั้งระบุตัวอย่างการแก้ปัญหา โดยสร้างเป็นโมดูลในการประเมินความมั่นคงปลอดภัยของระบบ ก่อนที่จะติดตั้งซอฟต์แวร์ประยุกต์ ข้อจำกัดของการใช้แบบรูปความมั่นคงปลอดภัย และผลที่ตามมาจากการใช้แบบรูปความมั่นคงปลอดภัย ซึ่งในงานวิจัยนี้ได้เสนอตัวอย่างการประยุกต์ใช้แบบรูปความมั่นคงปลอดภัยในกระบวนการพัฒนาซอฟต์แวร์ (Software Life Cycle) โดยได้แสดงผลการประยุกต์ใช้ในกรณีศึกษาสองกรณีเป็นอย่างน้อย พร้อมทั้งแสดงการทดสอบโมดูลการประเมินฮาร์ดเดนนิ่งของระบบ โดยเปรียบเทียบกับสภาพแวดล้อมจริงของการติดตั้งใช้งานซอฟต์แวร์ประยุกต์

1.2. วัตถุประสงค์ของการวิจัย

เพื่อนำเสนอระเบียบวิธีในการประยุกต์ใช้แผนภาพมิสยูสเคสในการประเมินฮาร์ดเดนนิ่งของระบบที่ใช้ในการรองรับการติดตั้งซอฟต์แวร์ประยุกต์

1.3. ขอบเขตของการวิจัย

1) ศึกษาข้อมูลการโจมตีระบบ เช่น หน่วยความจำ (Memory) การควบคุมการเข้าถึง (Access Control) การเข้าถึงระบบ (Access) และบันทึกการใช้งาน (Log)

2) ศึกษาข้อมูลฮาร์ดเดนนิ่งของระบบ เช่น การแยกหน่วยความจำชุดคำสั่งและหน่วยความจำข้อมูล การกำหนดสิทธิ์ของผู้ใช้ การตั้งค่าไฟร์วอลล์ และการเก็บบันทึกการใช้งานระบบ เป็นต้น

3) เครื่องมือที่ใช้ในการวิจัย จะใช้แผนภาพมิสยูสเคส เป็นพื้นฐานในการวิเคราะห์และรวบรวมความต้องการด้านความมั่นคงปลอดภัย

4) แบบรูปความมั่นคงปลอดภัยโดยใช้มิสยูส จะไม่ครอบคลุมความผิดพลาดในการออกแบบ

5) พัฒนาเครื่องมือในการประเมินฮาร์ดเดนนิ่งของระบบ

6) ระบบปฏิบัติการที่ใช้ในการทำวิจัย คือ ลีนุกซ์ (Linux)

1.4. ประโยชน์ที่ได้รับจากงานวิจัย

1) ทำให้เกิดเป็นระเบียบวิธีในการเปลี่ยนความต้องการที่ไม่ใช่เชิงหน้าที่ เช่น ความต้องการด้านความมั่นคงปลอดภัย ให้เป็นความต้องการเชิงหน้าที่

2) ทำให้เกิดเป็นระบบตรวจสอบอัตโนมัติผนวกเข้ากับการติดตั้งใช้งานซอฟต์แวร์ประยุกต์ โดยไม่ต้องพึ่งพาการตรวจสอบด้วยมือของผู้ดูแลระบบหรือผู้ทำการติดตั้งใช้งาน

3) ทำให้การพัฒนาระบบมีแบบรูปความมั่นคงปลอดภัยที่เกี่ยวเนื่องกับการติดตั้งใช้งานซอฟต์แวร์ประยุกต์ที่เหมาะสม

4) สะดวกต่อการประเมินความมั่นคงปลอดภัยของระบบที่จะนำซอฟต์แวร์ประยุกต์ไปติดตั้ง

5) แบบรูปความมั่นคงปลอดภัยที่เป็นทางการช่วยให้ปรับเปลี่ยนการพัฒนาซอฟต์แวร์ได้ตามเทคโนโลยีของการโจมตีและการป้องกันใหม่ๆ ได้โดยสะดวก

6) เป็นแนวทางให้กับผู้พัฒนาระบบได้สนใจและคำนึงถึงความมั่นคงปลอดภัยของระบบมากยิ่งขึ้น ไม่ถูกละเลย

7) เป็นแนวทางเพื่อให้กำหนดความต้องการด้านความมั่นคงปลอดภัยที่สอดคล้องกับนโยบายของการพัฒนาและการใช้ซอฟต์แวร์ประยุกต์มากขึ้น

1.5. วิธีดำเนินการทำวิจัย

1) ศึกษาข้อมูลช่องทางในการโจมตีระบบ และวิธีการป้องกันการโจมตี

2) ศึกษาข้อมูลฮาร์ดเดนนิ่งของระบบ

3) ศึกษาข้อมูลแผนภาพมิสยูสเคส แผนภาพคลาส และแผนภาพลำดับ

4) ศึกษาข้อมูลแบบรูปความมั่นคงปลอดภัย

5) ใช้มิสยูสวิเคราะห์การโจมตีระบบ และวิธีการป้องกัน

6) วิเคราะห์และออกแบบระบบ ในรูปแผนภาพมิสยูสเคส พร้อมคำอธิบายมิสยูสเคส แผนภาพคลาส พร้อมบัตรซีอาร์ซี และแผนภาพลำดับ

7) ออกแบบโครงสร้างแบบรูปความมั่นคงปลอดภัย

8) ประยุกต์ใช้แบบรูปความมั่นคงปลอดภัยในกรณีศึกษา โดยพัฒนาเป็นโมดูลการประเมินฮาร์ดเดนนิ่งของระบบ

9) ทดสอบโมดูลการประเมินฮาร์ดเดนนิ่งของระบบ และปรับปรุงแบบรูปความมั่นคงปลอดภัยเพื่อให้เหมาะสมกับระบบ

10) ตีพิมพ์ผลงานทางวิชาการ

11) เรียบเรียงและจัดทำวิทยานิพนธ์

1.6. โครงสร้างวิทยานิพนธ์

ในบทที่ 2 กล่าวถึง แนวคิด ทฤษฎี เอกสาร และงานวิจัยที่เกี่ยวข้อง ประกอบด้วยห้าหัวข้อ ได้แก่ วิศวกรรมด้านการรักษาความมั่นคงปลอดภัย ฮาร์ดเดนนิ่ง การติดตั้งใช้งานซอฟต์แวร์ แผนภาพมิสยูสเคส และแบบรูปความมั่นคงปลอดภัย ในบทที่ 3 กล่าวถึงระเบียบวิธีวิจัย ประกอบด้วยสองส่วน ได้แก่ การนำเสนอแบบจำลองแนวคิด และการนำเสนอระเบียบวิธีวิจัย ในบทที่ 4 กล่าวถึงแบบรูปความมั่นคงปลอดภัย ประกอบด้วยตัวอย่างสี่กรณี ได้แก่ แบบรูปการแยกหน่วยความจำชุดคำสั่งและหน่วยความจำข้อมูล แบบรูปการกำหนดสิทธิ์ของผู้ใช้ แบบรูปการตั้งค่าไฟร์วอลล์ และแบบรูปการเก็บบันทึกการใช้งานระบบ ในบทที่ 5 เป็นการประยุกต์ใช้แบบรูปความมั่นคงปลอดภัย ประกอบด้วยสองส่วน ได้แก่ การประยุกต์ใช้แบบรูปความมั่นคงปลอดภัยในกระบวนการพัฒนาซอฟต์แวร์ และกรณีศึกษา ในบทที่ 6 เป็นการทดสอบและอภิปรายผลการวิจัย ประกอบด้วยสองส่วน ได้แก่ การทดสอบการทำงานของโมดูลการประเมินฮาร์ดเดนนิ่งของระบบ และอภิปรายผลการวิจัย และในบทสุดท้าย บทที่ 7 เป็นการสรุปผลการวิจัย ประกอบด้วยสามส่วน ได้แก่ สรุปผลที่ได้รับจากงานวิจัย ข้อจำกัดของงานวิจัย และแนวทางการทำวิจัยในอนาคต

บทที่ 2
ทบทวนวรรณกรรม

เนื้อหาในบทนี้ กล่าวถึง แนวคิด ทฤษฎี เอกสาร และงานวิจัยต่างๆ ที่เกี่ยวข้อง โดยสามารถแบ่งออกเป็น 5 หัวข้อ ดังนี้

1) วิศวกรรมด้านการรักษาความมั่นคงปลอดภัย
2) ฮาร์ดเดนนิ่ง
3) การติดตั้งใช้งานซอฟต์แวร์
4) แผนภาพมิสยูสเคส
5) แบบรูปความมั่นคงปลอดภัย

ซึ่งในแต่ละหัวข้อ มีรายละเอียดของแนวคิด ทฤษฎี เอกสาร และงานวิจัยต่างๆ ที่เกี่ยวข้อง ดังต่อไปนี้

2.1. แนวคิดและทฤษฎีที่เกี่ยวข้อง

2.1.1. วิศวกรรมด้านการรักษาความมั่นคงปลอดภัย (Security Engineering)

วิศวกรรมด้านการรักษาความมั่นคงปลอดภัย คือ การนำความรู้และความเชี่ยวชาญในสาขาต่างๆ มาผสมผสานและประยุกต์ใช้ในกระบวนการพัฒนาระบบงานคอมพิวเตอร์ เพื่อให้ระบบสามารถป้องกันตนเองจากภัยคุกคาม และสามารถกู้คืนการทำงานและข้อมูลของระบบได้แม้จะถูกโจมตี

การรักษาความมั่นคงปลอดภัยของระบบ มีเป้าหมายสำคัญ 3 ประการ ได้แก่

1) การรักษาความลับของข้อมูล (Confidentiality)
2) การคงไว้ซึ่งบูรณภาพข้อมูล (Integrity)
3) การคงไว้ซึ่งความพร้อมใช้งานของระบบ (Availability)

รูปแบบของภัยคุกคาม (Threat) ประกอบด้วย 4 รูปแบบ [2] คือ

1) การดักจับข้อมูล (Interception)
2) การหยุดการทำงาน (Interruption)
3) การแก้ไขข้อมูล (Modification)
4) การสร้างข้อมูลปลอม (Fabrication)

2.1.2. **ฮาร์ดเดนนิ่ง** (Hardening)

ฮาร์ดเดนนิ่ง เป็นกระบวนการของการรักษาความมั่นคงปลอดภัยของระบบ โดยการลดพื้นผิวของช่องโหว่ทำให้ระบบมีความมั่นคงปลอดภัยสูงขึ้น [3]

การป้องกันระบบ แบ่งเป็น 4 ขั้นตอนหลัก [4] คือ

1) การลดโอกาสของการถูกโจมตี ประกอบด้วย การกำจัดซอฟต์แวร์ที่ไม่จำเป็น การกำจัดผู้ใช้งานระบบที่ไม่จำเป็น หรือการปิดการใช้งานการให้บริการที่ไม่จำเป็น

2) การอัพเดตแพทช์ให้กับเคอร์เนล เพื่อทำให้มั่นใจว่า ระบบได้ปิดช่องโหว่จุดอ่อน หรือแก้ไขจุดบกพร่องของซอฟต์แวร์และระบบปฏิบัติการ เช่น Exec Shield [5] PAX และ NXBit เป็นต้น

3) การปิดพอร์ตเน็ตเวิร์ก

4) การติดตั้งระบบการตรวจสอบการบุกรุก (IDS) ไฟร์วอลล์ และระบบการป้องกันการบุกรุก (IPS)

ตัวอย่างวิธีการในการทำฮาร์ดเดนนิ่ง เพื่อความมั่นคงปลอดภัยของลีนุกซ์ [6]

1) การเข้ารหัสข้อมูลที่ถูกส่งผ่านทางเครือข่ายด้วยการใส่รหัสผ่านหรือการใช้คีย์ เช่น การกำหนดค่าและติดตั้ง Apache SSL (Secure Server Layer) Https เป็นต้น

2) การลดซอฟต์แวร์ที่ไม่จำเป็นเพื่อหลีกเลี่ยงช่องโหว่ในซอฟต์แวร์ และใช้ตัวจัดการซอฟต์แวร์ เช่น yum เป็นต้น

3) การกำหนดให้หนึ่งบริการเครือข่ายต่อหนึ่งระบบ เช่น การติดตั้ง XEN Virtualization เป็นต้น

4) การอัพเดตแพทช์ให้กับลีนุกซ์เคอร์เนลและซอฟต์แวร์อย่างสม่ำเสมอ เช่น การใช้คำสั่ง apt-get เพื่ออัพเดตความมั่นคงปลอดภัย เป็นต้น

5) การใช้ส่วนขยายด้านความมั่นคงปลอดภัยลีนุกซ์ เช่น การใช้ SELinux ในการกำหนดสิทธิ์การเข้าใช้งานของผู้ใช้ เป็นต้น

6) การกำหนดนโยบายในส่วนบัญชีผู้ใช้และการกำหนดรหัสผ่าน เช่น การกำหนดอายุรหัสผ่าน และการไม่อนุญาตให้ตั้งรหัสผ่านเดิม เป็นต้น

7) การปิดการใช้งานผู้ใช้รูท

8) การกำหนดความมั่นคงปลอดภัยให้กับเซิร์ฟเวอร์ทางกายภาพ เช่น การกำหนดค่าไบออส และการปิดการใช้งานบูตจากอุปกรณ์ภายนอก เป็นต้น

9) การปิดการใช้งานการให้บริการต่างๆ ที่ไม่จำเป็น เช่น การใช้คำสั่ง service serviceName stop และ chkconfig serviceName off เป็นต้น

10) การลบเอ็กซ์วินโดว์ซิสเต็ม เช่น การใช้คำสั่ง yum groupremove "X Window System" เป็นต้น

11) การกำหนดค่า iptables เพื่อเปิดการใช้งานไฟร์วอลล์ และ TCPWrappers เพื่อกรองการเข้าถึงเครือข่ายอินเตอร์เน็ต

12) ฮาร์ดเดนนิ่งลีนุกซ์เคอร์เนล /etc/sysctl.conf เช่น การเปิดการใช้งาน ExecShield เพื่อแยกหน่วยความจำชุดคำสั่งและหน่วยความจำข้อมูลออกจากกัน เป็นต้น

13) การแบ่งพาร์ทิชันดิสก์ โดยแยกไฟล์ระบบออกจากไฟล์ผู้ใช้ และแยกไฟล์ระบบ /usr /home /var and /var/tmp และ /tmp ให้อยู่พาร์ทิชันอื่น

14) การปิดการใช้งาน Internet Protocol version 6 (IPv6) หากไม่จำเป็นต้องใช้งาน เพื่อเป็นการลดพื้นผิวช่องโหว่ของระบบ

15) ห้ามการใช้โปรแกรมที่มี SUID และ SGID

16) การใช้บริการการพิสูจน์ตัวจริงจากศูนย์กลาง เช่น การใช้ Kerberos เป็นต้น

17) การเก็บบันทึกการใช้งานและการตรวจสอบ เช่น การใช้ syslog ในไดเรกทอรี /var/log/ เพื่อเก็บบันทึกการเข้าใช้งานระบบ เป็นต้น

18) การใช้เซิร์ฟเวอร์ OpenSSH ที่ปลอดภัย

19) การติดตั้งและใช้ระบบการตรวจสอบการบุกรุก

20) การป้องกัน ไฟล์ ไดเรกทอรี และอีเมลล์ เช่น การใช้เครื่องมือ TrueCrypt ในการเข้ารหัสข้อมูล เป็นต้น

2.1.3. **การติดตั้งใช้งานซอฟต์แวร์** (Software Deployment)

การติดตั้งใช้งานซอฟต์แวร์ เป็นกิจกรรมที่เกิดขึ้นเพื่อทำให้ซอฟต์แวร์พร้อมใช้งาน ประกอบด้วย 7 กิจกรรมหลัก [7] คือ

1) การปล่อย (Release) เป็นการจัดเตรียมแพคเกจ โดยกำหนดทรัพยากรที่จำเป็นในการดำเนินการและเก็บรวบรวมข้อมูลสำหรับการดำเนินกิจกรรมต่อมาของการติดตั้งใช้งานซอฟต์แวร์

2) การติดตั้ง (Installation) เป็นการกำหนดค่าต่างๆ เพื่อเตรียมการสำหรับการเปิดใช้งานซอฟต์แวร์

3) การเปิดใช้งาน (Activation) เป็นการทำให้ซอฟต์แวร์สามารถใช้งานได้ในเวลาที่เหมาะสม

4) การปิดการใช้งาน (Deactivation) เป็นการทำให้ซอฟต์แวร์ไม่สามารถใช้งานได้ ซึ่งตรงข้ามกับการเปิดใช้งาน

5) การปรับปรุง (Updating) เป็นการปรับเปลี่ยนบางส่วนหรือทั้งหมดของซอฟต์แวร์ที่ติดตั้ง

6) การปรับตัว (Adaptation) เป็นการปรับเปลี่ยนซอฟต์แวร์ที่ติดตั้งเพื่อที่จะตอบสนองต่อการเปลี่ยนแปลงในสภาพแวดล้อมที่ทำการติดตั้ง

7) การถอนการติดตั้ง (Deinstallation) เป็นการถอดซอฟต์แวร์ที่ใช้งานออกจากเครื่อง ซึ่งตรงข้ามกับการติดตั้ง

2.1.4. **แผนภาพมิสยูสเคส** (Misuse Case Diagram)

แผนภาพมิสยูสเคส เป็นเครื่องมือแบบจำลองกระบวนการทางธุรกิจที่ถูกใช้ในธุรกิจการพัฒนาซอฟต์แวร์ เป็นการอธิบายถึงการกระทำที่ไม่เหมาะสมต่อระบบ ซึ่งระบบไม่ควรจะอนุญาตให้เกิดขึ้น [8]

แผนภาพมิสยูสเคส ประกอบด้วย 4 ส่วน [9] คือ

1) มิสยูสเคส (Misuse Case) เป็นลำดับของการกระทำที่ถูกปฏิบัติโดยบุคคลใดๆ ที่ไม่เหมาะสมต่อระบบ

2) มิสยูสเซอร์ (Misuser) คือ ผู้กระทำที่เป็นคนเริ่มกระทำมิสยูสเคส ทั้งที่เจตนา หรือไม่เจตนา

3) ความสัมพันธ์ (Relationship) แสดงความสัมพันธ์ที่เกิดขึ้นในแผนภาพ โดยแผนภาพมิสยูสเคส จะมีความสัมพันธ์เช่นเดียวกับแผนภาพยูสเคส แต่ได้เพิ่มความสัมพันธ์อีก 2 แบบคือ ความสัมพันธ์แบบบรรเทา (Mitigates) และความสัมพันธ์แบบคุกคาม (Threatens)

4) คำอธิบาย (Descriptions) ประกอบด้วย ชื่อมิสยูสเคส (Misuse Case Name) ผู้เขียน (Author) การรับประกันการบรรเทา (Mitigation Guarantee) ประวัติมิสยูสเซอร์ที่เป็นไปได้ (Potential Misuser Profile) และผู้มีส่วนเกี่ยวข้องและการคุกคาม (Stakeholders and Threats) เป็นต้น

2.1.5. **แบบรูปความมั่นคงปลอดภัย** (Security Pattern)

แบบรูปความมั่นคงปลอดภัย คือ หลักและวิธีการแก้ไขปัญหาชนิดใดชนิดหนึ่ง ที่สามารถนำไปใช้กับปัญหาชนิดเดียวกันที่เกิดซ้ำได้ โดยแบบรูปในการแก้ปัญหานั้น จะต้องอธิบายโครงสร้างของความมั่นคงปลอดภัยไว้อย่างละเอียด ไม่ว่าจะเป็นชื่อแบบรูป ปัญหาของแบบรูป วิธีการแก้ปัญหา และผลที่ตามมา การใช้แบบรูปจะช่วยให้งานผลิตซอฟต์แวร์ดำเนินไปได้ด้วยความรวดเร็ว

แบบรูปความมั่นคงปลอดภัย ประกอบด้วย 10 ส่วน [10] คือ

1) บริบท (Context) คือ สถานการณ์ที่ประยุกต์ใช้แบบรูปความมั่นคงปลอดภัย

2) ปัญหา (Problem) คือ ส่วนที่แสดงช่องโหว่ของระบบที่ต้องการแก้ไข ระบุสถานการณ์ที่เกิดขึ้นเป็นประจำที่ทำให้เกิดปัญหาขึ้นได้

3) การแก้ไขปัญหา (Solution) คือ การกำหนดรูปแบบการดำเนินงานเพื่อใช้แก้ไขปัญหาทางด้านความมั่นคงปลอดภัย

4) โครงสร้าง (Structure) คือ การอธิบายองค์ประกอบโครงสร้างของแบบรูปความมั่นคงปลอดภัย

5) ไดนามิก (Dynamics) คือ การอธิบายลักษณะการทำงานของแบบรูปความมั่นคงปลอดภัยที่เกิดขึ้นขณะเวลาทำงาน

6) การทำให้เกิดผล (Implementation) คือ แนวทางสำหรับการดำเนินการแบบรูปความมั่นคงปลอดภัย

7) ตัวอย่างการแก้ปัญหา (Example Resolved) คือ การอธิบายหลักสำคัญในการแก้ไขปัญหา

8) ข้อจำกัด (Variants) คือ การอธิบายข้อจำกัดที่เกี่ยวข้องกับแบบรูปความมั่นคงปลอดภัย

9) การใช้ประโยชน์หรือเป็นที่รู้จัก (Known Uses) คือ ตัวอย่างของการใช้แบบรูปความมั่นคงปลอดภัย

10) ผลที่ตามมา (Consequences) คือ ผลที่เกิดขึ้นจากการแก้ไขปัญหาด้วยการใช้แบบรูปความมั่นคงปลอดภัย

2.2. เอกสารและงานวิจัยที่เกี่ยวข้อง

ลีนุกซ์ ยังคงมีจุดอ่อนในปัจจุบัน ดังนั้นจึงมีหลายงานวิจัยที่เน้นพัฒนาลีนุกซ์ให้มีความปลอดภัยมากยิ่งขึ้น โดยในปี ค.ศ. 2001 ปีเตอร์ เอ ลอซคอคโค และสตีเฟน ดี สมอลเลย์ (Peter A. Loscocco and Stephen D. Smalley) [11] ได้เสนอแนวคิดในการเพิ่มความปลอดภัยให้กับลีนุกซ์ในส่วนสถาปัตยกรรม Mandatory Access Control (MAC) โดยการใช้ SELinux ปี ค.ศ. 2004 อาร์จาน แวน ดี เวน (Arjan van de Ven) [12] ได้เสนอ Exec Shield มาใช้ในการแบ่งแยกสิทธิ์การอ่านและการเขียนหน่วยความจำออกจากกัน ปี ค.ศ. 2007 ทัมมี ฟอกซ์ (Tammy Fox) [13] ได้เสนอวิธีการป้องกันผู้บุกรุกด้วยการใช้ SELinux และ Exec Shield รวมถึงการตั้งค่าไฟร์วอลล์ด้วยการเปิดการใช้งาน iptables

ซอฟต์แวร์ที่ถูกพัฒนาขึ้นมานั้นจะต้องถูกนำไปให้ลูกค้าใช้งานได้ ดังนั้นกระบวนการการติดตั้งใช้งานซอฟต์แวร์จึงมีความสำคัญ โดยในปี ค.ศ. 2011 ไมกา วี แมนทีลา และจาริ แวนฮาเนน (Mika V. Mantyla and Jari Vanhanen) [14] ได้ค้นคว้าเกี่ยวกับเรื่องการติดตั้งใช้งานซอฟต์แวร์ โดยได้กำหนดกิจกรรมในการติดตั้งใช้งานซอฟต์แวร์เพื่อให้การติดตั้งใช้งานซอฟต์แวร์ประสบความสำเร็จ และเสนอกรณีตัวอย่างที่เป็นปัญหาที่เกิดขึ้นได้ในกระบวนการติดตั้งใช้งานซอฟต์แวร์

แนวคิดในการเก็บรวบรวมและวิเคราะห์ความต้องการความมั่นคงปลอดภัยของซอฟต์แวร์ให้สามารถทำได้ตั้งแต่ขั้นตอนความต้องการ โดยได้นำเสนอแผนภาพมิสยูสเคสขึ้นเพื่อช่วยในการอธิบายสิ่งที่ไม่ควรเกิดขึ้นในระบบและใช้จำลองการโจมตีระบบ เพื่อที่จะได้สามารถหาวิธีการป้องกันล่วงหน้าได้ โดยแนวคิดแรกได้เกิดขึ้นมาเมื่อประมาณสิบปีเศษ ในปี ค.ศ. 2000 [9] ค.ศ. 2001 [15] และค.ศ. 2005 [16] กัททอม ซินดรี และแอนเดรีย แอล ออปดาล (Guttorm Sindre and Andreas L. Opdahl) ได้เพิ่มแนวคิดการออกแบบแผนภาพยูสเคสให้คำนึงถึงความมั่นคงปลอดภัย โดยมีการกำหนดมิสยูสเซอร์และมิสยูสเคสในแผนภาพยูสเคส อีกทั้งยังกำหนดรูปแบบสำหรับอธิบายมิสยูสเคส และกำหนดขั้นตอนในการออกแบบมิสยูสเคส

ในปี ค.ศ. 2008 รายมานดัส มาทูลีวิเชียส นิโคลัส มายเออร์ และแพทริก เฮย์มานส์ (Raimundas Matulevicius, Nicolus Mayer and Patrick Heymans) [17] ได้เสนอแนวคิดมิสยูสเคสควบคู่ไปกับการจัดการความเสี่ยงด้านความมั่นคงปลอดภัย ในปีเดียวกัน ฟาบริซิโอ เอ บราซ เอดัวอาร์โด บี เฟอร์นานเดส และมิเชล แวนฮิลส์ (Fabricio A. Braz, Eduardo B. Fernandez and Michael VanHilst [18] ได้เสนอแนวคิดการเก็บรวบรวมความต้องการทางด้าน

ความมั่นคงปลอดภัยในรูปแบบมิสยูส โดยมีนโยบายช่วยในการตรวจสอบกิจกรรมของมิสยูสที่ได้ออกแบบไว้ ทั้งนี้ได้กำหนดไว้ว่าในแต่ละการคุกคามที่อยู่ในมิสยูสสัมพันธ์กับนโยบายในรูปแบบหลีกเลี่ยง และ/หรือ การบรรเทา รวมทั้งแสดงว่า แต่ละกิจกรรมมีใครเกี่ยวข้องบ้าง ทรัพยากรที่ถูกโจมตีสัมพันธ์กับการคุกคามแบบใด

ต่อมาในปี ค.ศ. 2009 ทาคาโอ โอคุบุ เคนจิ ทากุชิ และโนบุคากะ โยชิโอกะ (Takao Okubu, Kenji Taguchi, Nobukazu Yoshioka) [19] ได้เสนอแผนภาพมิสยูสเคสแบบใหม่ในการวิเคราะห์และรวบรวมความต้องการความมั่นคงปลอดภัยที่เน้นไปที่ทรัพยากรและเป้าหมายด้านความมั่นคงปลอดภัย มีการกำหนดกระบวนการในการออกแบบ พร้อมยกตัวอย่างกรณีศึกษาเกี่ยวกับระบบบัญชี

ปี ค.ศ.2006 เอ็ม ชูมาเชอร์ อี เฟอร์นานเดซ-บูกลีออนนิ ดี ไฮเบอร์สัน เอฟ บูซมาน และพี ซัมเมอร์ลัด (M. Schumacher, E. Fernandez-Buglioni, D. Hybertson, F. Buschmann, and P. Sommerlad) [20] และปี ค.ศ.2009 แองเจล คับวาส และพอล อี เคอรี่ (Angel Cubvas and Paul E. Khoury) [21] ได้เสนอแนวคิดเกี่ยวกับแบบรูปความมั่นคงปลอดภัย โดยสร้างเป็นแบบรูปที่เกี่ยวกับการป้องกันและรักษาความมั่นคงปลอดภัยเพื่อแก้ไขปัญหาชนิดใดชนิดหนึ่งที่สามารถนำไปใช้กับปัญหาชนิดเดียวกันที่เกิดซ้ำได้

ที่กล่าวมาในบทที่ 2 นี้ เป็นการกล่าวถึงแนวคิด ทฤษฎี เอกสาร และงานวิจัยที่เกี่ยวข้องที่ใช้เป็นแนวทางในการทำวิจัย โดยเรื่องวิศวกรรมด้านการรักษาความมั่นคงปลอดภัยฮาร์ดเดนนิ่ง และการติดตั้งใช้งานซอฟต์แวร์ เป็นแนวทางเพื่อใช้ในการศึกษาข้อมูลเกี่ยวกับความมั่นคงปลอดภัยของระบบ และเรื่องแผนภาพมิสยูสเคส เป็นแนวทางเพื่อใช้ในการวิเคราะห์และรวบรวมความต้องการด้านความมั่นคงปลอดภัย และสุดท้ายเรื่องแบบรูปความมั่นคงปลอดภัยเป็นแนวทางเพื่อใช้ในการแก้ปัญหาอย่างมีแบบแผน สะดวกต่อการนำไปประยุกต์ใช้งาน ส่วนในบทถัดไป จะกล่าวถึงระเบียบวิธีวิจัย

บทที่ 3
ระเบียบวิธีวิจัย

เนื้อหาในบทนี้ กล่าวถึง ระเบียบวิธีวิจัย โดยสามารถแบ่งออกเป็น 2 ส่วน ดังนี้

1) แบบจำลองเชิงแนวคิด

2) ระเบียบวิธีวิจัย ประกอบด้วย

2.1) การวิเคราะห์การโจมตี และวิธีการป้องกันการโจมตี กล่าวถึงในบทที่ 3 หัวข้อ 3.3

2.2) การกำหนดแผนภาพมิสยูสเคสจากการประเมินการโจมตีความมั่นคงปลอดภัย กล่าวถึงในบทที่ 3 หัวข้อ 3.4

2.3) แบบรูปความมั่นคงปลอดภัย กล่าวถึงในบทที่ 4

2.4) การประยุกต์ใช้แบบรูปความมั่นคงปลอดภัย กล่าวถึงในบทที่ 5

2.5) การทดสอบและอภิปรายผลการวิจัย กล่าวถึงในบทที่ 6

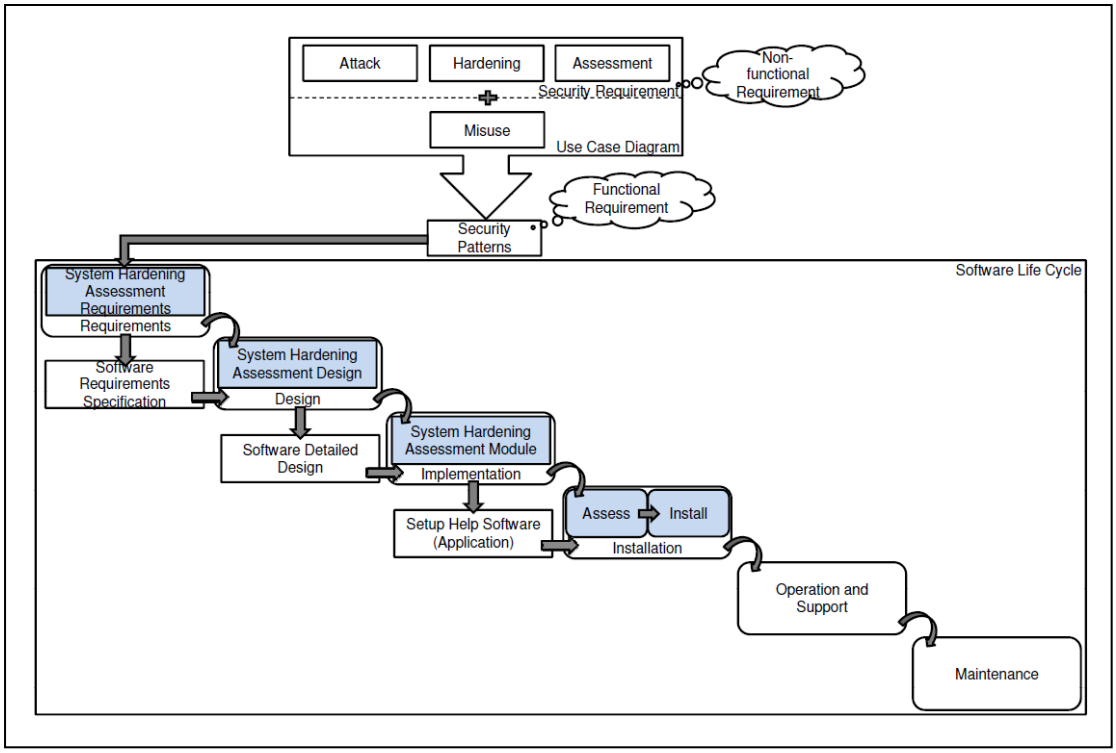
2.6) การสรุปผลการวิจัย กล่าวถึงในบทที่ 7

ซึ่งในแต่ละส่วน มีรายละเอียดของระเบียบวิธีวิจัย ดังต่อไปนี้

3.1. แบบจำลองเชิงแนวคิด

ฮาร์ดเดนนิ่ง เป็นกระบวนการหนึ่งในการรักษาความมั่นคงปลอดภัยให้กับระบบ แต่การประเมินสถานะฮาร์ดเดนนิ่งของระบบเป็นเรื่องที่ทำได้ยาก และส่วนใหญ่ต้องอาศัยผู้เชี่ยวชาญด้านความมั่นคงปลอดภัย ซึ่งแตกต่างกันตามทักษะและประสบการณ์ในแต่ละบุคคล อีกทั้ง ความต้องการด้านฮาร์ดเดนนิ่งของระบบ เป็นความต้องการด้านความมั่นคงปลอดภัย ซึ่งโดยปกติจัดเป็นความต้องการที่ไม่ใช่เชิงหน้าที่ ดังนั้น อาจถูกละเลยไปได้ จึงเป็นสิ่งสำคัญในการหาแนวทางในการทำให้ความต้องการที่ไม่ใช่เชิงหน้าที่เป็นความต้องการเชิงหน้าที่

งานวิจัยนี้ มุ่งเน้นการใช้มิสยูสมาทำการวิเคราะห์และศึกษาการประเมินสถานะฮาร์ดเดนนิ่งของระบบให้เกิดขึ้นโดยอัตโนมัติในขณะติดตั้งซอฟต์แวร์ประยุกต์ โดยไม่ต้องอาศัยผู้เชี่ยวชาญด้านความมั่นคงปลอดภัย ทั้งนี้เพื่อเป็นการสร้างความมั่นใจว่า จะไม่ติดตั้งซอฟต์แวร์ประยุกต์ในระบบที่ไม่มั่นคงปลอดภัย โดยนำเสนอแบบจำลองเชิงแนวคิด ดังภาพที่ 3.1



ภาพที่ 3.1 แบบจำลองเชิงแนวคิดของ System Hardening Assessment

จากภาพที่ 3.1 ทำการศึกษาข้อมูลและรวบรวมความต้องการด้านความมั่นคงปลอดภัยเกี่ยวกับการโจมตี ฮาร์ดเดนนิ่ง และการประเมินผล โดยทั่วไปจัดเป็นความต้องการที่ไม่ใช่เชิงหน้าที่ (Non-functional Requirement) จากนั้น นำหลักการมิสยูสมาช่วยวิเคราะห์และรวบรวมความต้องการด้านความมั่นคงปลอดภัย ทำให้เปลี่ยนจากความต้องการที่ไม่ใช่เชิงหน้าที่เป็นความต้องการเชิงหน้าที่ (Functional Requirement) โดยให้ปรากฏอย่างเป็นทางการในแผนภาพยูสเคสในรูปแบบของแผนภาพมิสยูสเคส เพื่อใช้เป็นข้อกำหนดความต้องการเชิงหน้าที่ในส่วนของความมั่นคงปลอดภัยของซอฟต์แวร์ประยุกต์ โดยงานวิจัยนี้ มุ่งเน้นไปในส่วนการประเมินความมั่นคงปลอดภัยของระบบในขณะติดตั้งซอฟต์แวร์ประยุกต์อย่างอัตโนมัติ โดยไม่ต้องพึ่งพาผู้เชี่ยวชาญด้านความมั่นคงปลอดภัย ทั้งนี้ได้สร้างเป็นแบบรูปความมั่นคงปลอดภัยขึ้นมาเพื่อสะดวกต่อการประยุกต์ใช้กับความต้องการเชิงหน้าที่ด้านอื่นๆ ของซอฟต์แวร์ประยุกต์ ซึ่งเป็นอิสระจากซอฟต์แวร์ประยุกต์เหล่านั้น

งานวิจัยนี้ ได้นำแบบรูปความมั่นคงปลอดภัยมาใช้ในกระบวนการพัฒนาซอฟต์แวร์ [22] ในขั้นตอนความต้องการ (Requirements Process) โดยทำให้เป็นความต้องการเชิงหน้าที่ด้วยการใช้แผนภาพมิสยูสเคสในการวิเคราะห์และรวบรวมความต้องการ โดยนำเอาแบบ

รูปความมั่นคงปลอดภัย ซึ่งระบุถึงความสามารถหรือหน้าที่ของซอฟต์แวร์ในส่วนของการประเมินฮาร์ดเดนนิ่งของระบบที่จะนำซอฟต์แวร์ประยุกต์นั้นไปติดตั้ง และเป็นส่วนหนึ่งของโมดูลการติดตั้งของซอฟต์แวร์ และความต้องการด้านอื่นของซอฟต์แวร์ประยุกต์มาจัดทำเป็นรายละเอียดข้อกำหนดความต้องการของซอฟต์แวร์

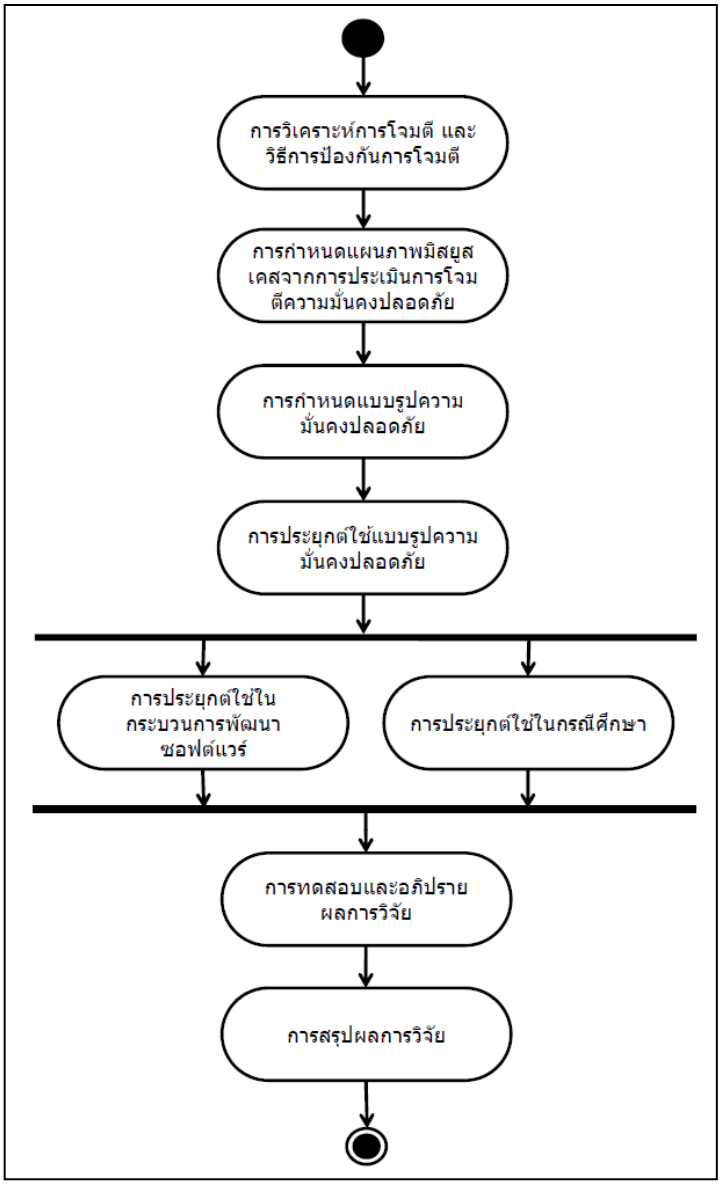
จากนั้นเข้าสู่ขั้นตอนการออกแบบ (Design Process) โดยนำเอารายละเอียดข้อกำหนดความต้องการของซอฟต์แวร์ที่ได้จากขั้นตอนความต้องการมาทำการออกแบบระบบในส่วนของการประเมินความมั่นคงปลอดภัยของระบบ และส่วนของซอฟต์แวร์ประยุกต์ โดยจัดทำเป็นรายละเอียดการออกแบบซอฟต์แวร์ และจากนั้นเข้าสู่ขั้นตอนการพัฒนา (Implementation Process) โดยนำเอารายละเอียดการออกแบบซอฟต์แวร์ที่ได้จากขั้นตอนการออกแบบมาพัฒนาเป็นซอฟต์แวร์ช่วยเหลือการติดตั้ง โดยเป็นการรวมโมดูลการประเมินฮาร์ดเดนนิ่งของระบบเข้ากับซอฟต์แวร์ประยุกต์

เมื่อพัฒนาซอฟต์แวร์ประยุกต์เสร็จสิ้น จะเข้าสู่ขั้นตอนการติดตั้ง (Installation Process) ในขั้นตอนนี้จะเป็นการนำซอฟต์แวร์ช่วยเหลือการติดตั้งที่ได้จากขั้นตอนการพัฒนามาทำการติดตั้งบนระบบ โดยเข้าทำการประเมินความมั่นคงปลอดภัยของระบบ ก่อนที่จะติดตั้งซอฟต์แวร์ประยุกต์ต่อไป

จากนั้นจะเข้าสู่ขั้นตอนการดำเนินการและสนับสนุน (Operation and Support Process) และขั้นตอนการบำรุงรักษา (Maintenance Process) ตามลำดับ ซึ่งเป็นขั้นตอนปกติของกระบวนการพัฒนาซอฟต์แวร์

3.2. ระเบียบวิธีวิจัย

จากแบบจำลองเชิงแนวคิด ได้นำมากำหนดระเบียบวิธีวิจัย โดยทำการศึกษาและวิเคราะห์การโจมตี พร้อมทั้งหาวิธีการป้องกันการโจมตี ซึ่งได้มุ่งเน้นในส่วนการประเมินสถานะฮาร์ดเดนนิ่งของระบบ จากนั้นนำข้อมูลการวิเคราะห์ที่ได้มากำหนดเป็นแผนภาพมิสยูสเคส โดยนำเสนอเป็นแบบรูปความมั่นคงปลอดภัยเพื่อให้สะดวกต่อการนำไปประยุกต์ใช้งาน โดยสามารถนำไปประยุกต์ใช้งานได้ในกระบวนการพัฒนาซอฟต์แวร์ โดยแสดงให้เห็นการประยุกต์ใช้งานจริงในกรณีศึกษา จากนั้นทำการทดสอบการใช้โมดูลการประเมินฮาร์ดเดนนิ่งของระบบพร้อมทั้งอภิปรายผลการวิจัย และทำการสรุปผลการวิจัย ดังภาพที่ 3.2



ภาพที่ 3.2 แผนภาพกิจกรรมของระเบียบวิธีวิจัย

3.3. การวิเคราะห์การโจมตี และวิธีการป้องกันการโจมตี

โดยปกติ ผู้โจมตีพยายามที่จะลักลอบเข้าถึงระบบโดยไม่ได้รับอนุญาต และอาจสร้างความเสียหายให้กับระบบได้ ดังนั้นเพื่อป้องกันการบุกรุกหรือปัญหาต่างๆ ที่อาจจะเกิดขึ้น จึงจำเป็นต้องระบุการโจมตีที่สามารถเกิดขึ้นได้ เพื่อที่จะนำมาใช้วิเคราะห์และหาวิธีป้องกันการโจมตีเหล่านั้น

ทั้งนี้ได้กำหนดฮาร์ดเดนนิ่ง 4 แบบ เพื่อเป็นตัวอย่างในการวิเคราะห์การโจมตีดังนี้

1) การแยกหน่วยความจำชุดคำสั่งและหน่วยความจำข้อมูล
2) การกำหนดสิทธิ์ของผู้ใช้
3) การตั้งค่าไฟร์วอลล์
4) การเก็บบันทึกการใช้งานระบบ

โดยฮาร์ดเดนนิ่งที่เลือกมานี้เป็นข้อมูลที่บุคคลทั่วไปในวงการคอมพิวเตอร์รู้จักกันเป็นอย่างดี โดยไม่จำเป็นต้องเป็นผู้เชี่ยวชาญทางด้านความมั่นคงปลอดภัย อีกทั้งในแต่ละฮาร์ดเดนนิ่งสะดวกต่อการตรวจสอบ ไม่มีขั้นตอนที่ยุ่งยากซับซ้อน ซึ่งได้ระบุวิธีการประเมินในแต่ละฮาร์ดเดนนิ่ง ดังตารางที่ 3.1

ตารางที่ 3.1 การประเมินฮาร์ดเดนนิ่งของระบบ

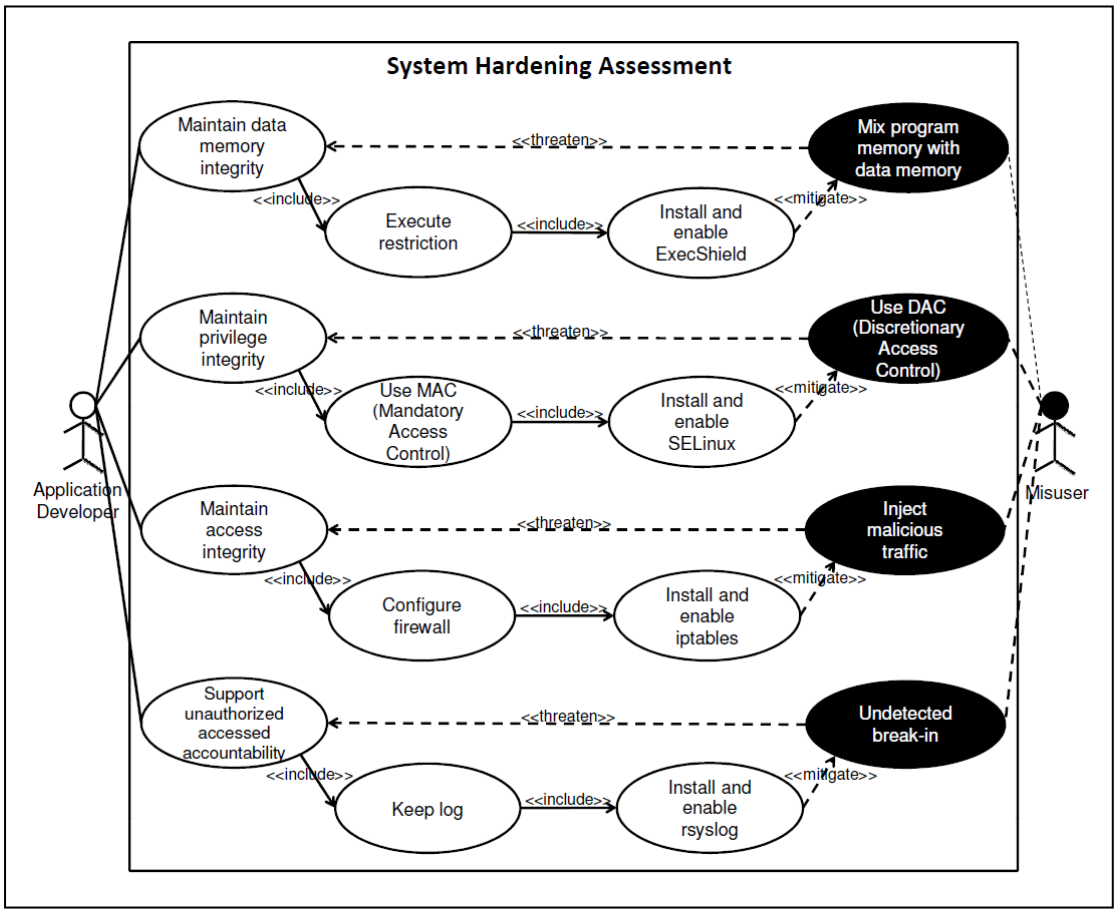
Attack	Misuse	Protection	Assessment
Memory	Mix program memory with data memory	Execute restriction	ExecSheild
Access Control	Use DAC (Discretionary Access Control)	Use MAC (Mandatory Access Control)	SELinux
Access	Inject malicious traffic	Configure firewall	iptables
Log	Undetected break-in	Keep log	rsyslog

จากตารางที่ 3.1 การโจมตีหน่วยความจำสามารถนำเสนอมิสยูสเกี่ยวกับการรวมหน่วยความจำชุดคำสั่งและหน่วยความจำข้อมูล จากปัญหานี้ กลยุทธ์ที่ใช้ในการหากลไกการป้องกันคือ การจำกัดการประมวลผล ซึ่งสามารถใช้คุณสมบัติ Exec Shield เป็นพื้นฐานในการประเมิน ส่วนการโจมตีการควบคุมการเข้าถึงสามารถนำเสนอมิสยูสเกี่ยวกับการใช้ DAC จากปัญหานี้ กลยุทธ์ที่ใช้ในการหากลไกการป้องกันคือ การใช้ MAC ซึ่งสามารถใช้คุณสมบัติ SELinux เป็นพื้นฐานในการประเมิน ส่วนการโจมตีการเข้าถึงระบบสามารถนำเสนอมิสยูสเกี่ยวกับการสร้างทราฟฟิกที่ประสงค์ร้ายเข้าไปในระบบ จากปัญหานี้ กลยุทธ์ที่ใช้ในการหากลไกการป้องกันคือ การตั้งค่าการใช้งานไฟร์วอลล์ ซึ่งสามารถใช้คุณสมบัติ iptables เป็นพื้นฐานในการประเมิน และส่วนสุดท้าย การโจมตีบันทึกการใช้งานสามารถนำเสนอมิสยูสเกี่ยวกับการไม่สามารถตรวจจับการบุกรุก จากปัญหานี้ กลยุทธ์ที่ใช้ในการหากลไกการป้องกันคือ การเก็บบันทึกการใช้งานระบบ ซึ่งสามารถใช้คุณสมบัติ rsyslog เป็นพื้นฐานในการประเมิน

ทั้งนี้สามารถใช้คุณสมบัติอื่นๆ ในการตรวจสอบฮาร์ดเดนนิ่งของระบบได้ เช่น การใช้คุณสมบัติของ PAX และ NXBit แทนคุณสมบัติของ Exec Shield การใช้คุณสมบัติของ Mandatory Integrity Control แทนคุณสมบัติของ SELinux การใช้คุณสมบัติของ ipchains แทนคุณสมบัติของ iptables และการใช้คุณสมบัติของ syslog-ng แทนคุณสมบัติของ rsyslog เป็นต้น

3.4. การกำหนดแผนภาพมิสยูสเคสจากการประเมินการโจมตีความมั่นคงปลอดภัย

การวิจัยนี้ได้นำหลักการมิสยูสมาช่วยวิเคราะห์และพิจารณาข้อมูลที่ได้จากตารางที่ 1 โดยแสดงออกมาในรูปแผนภาพมิสยูสเคส ดังภาพที่ 3.3 โดยทำการวิเคราะห์และรวบรวมความต้องการด้านความมั่นคงปลอดภัยที่เกี่ยวกับฮาร์ดเดนนิ่งในสี่กรณี และกำหนดเป็นมิสยูสได้สี่แบบ โดยปรากฏอย่างเป็นทางการในแผนภาพยูสเคสในรูปแบบแผนภาพมิสยูสเคส ซึ่งสามารถดูคำอธิบายยูสเคสได้ในภาคผนวก ก และคำอธิบายมิสยูสเคสได้ในภาคผนวก ข



ภาพที่ 3.3 แผนภาพมิสยูสเคสของระบบ System Hardening Assessment

จากภาพที่ 3.3 ระบบการประเมินฮาร์ดเดนนิ่งของระบบ ประกอบไปด้วยฟังก์ชันการทำงานต่างๆ โดยระบบมีผู้กระทำที่เกี่ยวข้องทั้งหมดสองผู้กระทำ คือ มิสยูสเซอร์ (Misuser) ซึ่งเป็นผู้ที่กระทำในสิ่งที่ไม่เหมาะสมต่อระบบ แสดงอยู่ในรูปคนหัวสีดำ และนักพัฒนาซอฟต์แวร์ (Application Developer) ซึ่งเป็นผู้พัฒนาระบบ แสดงอยู่ในรูปคนหัวสีขาว

ทั้งนี้ในส่วนฟังก์ชันการทำงานของระบบจะมีการทำงานที่แตกต่างกันไป โดยยูสเคส ซึ่งเป็นฟังก์ชันการทำงานปกติของระบบ จะแสดงเป็นวงรีสีขาว ส่วนมิสยูสเคส ซึ่งเป็นฟังก์ชันการทำงานที่ไม่เหมาะสมต่อระบบ จะแสดงเป็นวงรีสีดำ

สำหรับความสัมพันธ์ที่เกิดขึ้นในแผนภาพมิสยูสเคสจะมีความหมายแตกต่างกัน โดยเส้นทึบ แสดงถึงความสัมพันธ์ที่เกิดขึ้นระหว่างยูสเคสกับยูสเคส และเส้นประ แสดงถึงความสัมพันธ์ที่เกิดขึ้นระหว่างยูสเคสกับมิสยูสเคส

ฟังก์ชัน Mix program memory with data memory มีความสัมพันธ์แบบคุกคามกับ ฟังก์ชัน Maintain data memory integrity นั่นคือ มิสยูสเซอร์โจมตีหน่วยความจำผ่านทางการรวมหน่วยความจำชุดคำสั่งและหน่วยความจำข้อมูล ฟังก์ชัน Maintain data memory integrity มีความสัมพันธ์แบบรวมกับ ฟังก์ชัน Execute restriction และ ฟังก์ชัน Execute restriction มีความสัมพันธ์แบบรวมกับ ฟังก์ชัน Install and enable ExecShield นั่นคือ วิธีการหนึ่งที่ช่วยปกป้องหน่วยความจำได้ คือ การจำกัดการประมวลผล โดยการใช้ Exec Shield ทั้งนี้ ฟังก์ชัน Install and enable ExecShield มีความสัมพันธ์แบบบรรเทากับ ฟังก์ชัน Mix program memory with data memory กล่าวคือ การใช้ Exec Shield จะช่วยลดการเกิดการรวมหน่วยความจำชุดคำสั่งและหน่วยความจำข้อมูล

ฟังก์ชัน Use DAC (Discretionary Access Control) มีความสัมพันธ์แบบคุกคามกับ ฟังก์ชัน Maintain privilege integrity นั่นคือ มิสยูสเซอร์โจมตีการควบคุมการเข้าถึงผ่านทางการใช้ DAC ฟังก์ชัน Maintain privilege integrity มีความสัมพันธ์แบบรวมกับ ฟังก์ชัน Use MAC (Mandatory Access Control) และ ฟังก์ชัน Use MAC (Mandatory Access Control) มีความสัมพันธ์แบบรวมกับ ฟังก์ชัน Install and enable SELinux นั่นคือ วิธีการหนึ่งที่ช่วยปกป้องการควบคุมการเข้าถึงได้ คือ การใช้ MAC โดยการใช้ SELinux ทั้งนี้ ฟังก์ชัน Install and enable SELinux มีความสัมพันธ์แบบบรรเทากับ ฟังก์ชัน Use DAC (Discretionary Access Control) กล่าวคือ การใช้ SELinux จะช่วยลดการเกิดการใช้ DAC

ฟังก์ชัน Inject malicious traffic มีความสัมพันธ์แบบคุกคามกับ ฟังก์ชัน Maintain access integrity นั่นคือ มิสยูสเซอร์โจมตีการเข้าถึงระบบผ่านทางการสร้าง ทราฟฟิกที่ประสงค์ร้ายเข้าไปในระบบ ฟังก์ชัน Maintain access integrity มีความสัมพันธ์แบบรวมกับ ฟังก์ชัน Configure firewall และ ฟังก์ชัน Configure firewall มีความสัมพันธ์แบบรวมกับ ฟังก์ชัน Install and enable iptables นั่นคือ วิธีการหนึ่งที่ช่วยปกป้องการเข้าถึงระบบได้ คือ การตั้งค่าการใช้งานไฟร์วอลล์ โดยการใช้ iptables ทั้งนี้ ฟังก์ชัน Install and enable iptables มีความสัมพันธ์แบบบรรเทากับ ฟังก์ชัน Inject malicious traffic กล่าวคือ การใช้ iptables จะช่วยลดการเกิดการสร้างทราฟฟิกที่ประสงค์ร้ายเข้าไปในระบบ

ฟังก์ชัน Undetected break-in มีความสัมพันธ์แบบคุกคามกับ ฟังก์ชัน Support unauthorized accessed accountability นั่นคือ มิสยูสเซอร์โจมตีบันทึกการใช้งานผ่านทางการไม่สามารถตรวจจับการบุกรุก ฟังก์ชัน Support unauthorized accessed accountability มีความสัมพันธ์แบบรวมกับ ฟังก์ชัน Keep log และ ฟังก์ชัน Keep log มีความสัมพันธ์แบบรวมกับ ฟังก์ชัน Install and enable rsyslog นั่นคือ วิธีการหนึ่งที่ช่วยปกป้องการตรวจสอบการบุกรุกได้ คือ การเก็บบันทึกการใช้งานระบบ โดยการใช้ rsyslog ทั้งนี้ ฟังก์ชัน Install and enable rsyslog มีความสัมพันธ์แบบบรรเทากับ ฟังก์ชัน Undetected break-in กล่าวคือ การใช้ rsyslog จะช่วยลดการเกิดการไม่สามารถตรวจจับการบุกรุก

ที่กล่าวมาในบทที่ 3 นี้ เป็นการกล่าวถึงระเบียบวิธีวิจัย ที่ใช้เป็นแนวคิดในการทำวิจัย โดยนำเสนอแบบจำลองแนวคิดของงานวิจัยนี้ พร้อมทั้งวิเคราะห์และรวบรวมการโจมตี และการหากลยุทธ์ในการป้องกัน ซึ่งถูกนำเสนอในแผนภาพมิสยูสเคส ส่วนในบทถัดไป จะกล่าวถึงแบบรูปความมั่นคงปลอดภัย

บทที่ 4
แบบรูปความมั่นคงปลอดภัย

เนื้อหาในบทนี้ กล่าวถึง แบบรูปความมั่นคงปลอดภัย โดยสามารถแบ่งออกเป็น 4 กรณี ดังนี้

1) แบบรูปการแยกหน่วยความจำชุดคำสั่งและหน่วยความจำข้อมูล
2) แบบรูปการกำหนดสิทธิ์ของผู้ใช้
3) แบบรูปการตั้งค่าไฟร์วอลล์
4) แบบรูปการเก็บบันทึกการใช้งานระบบ

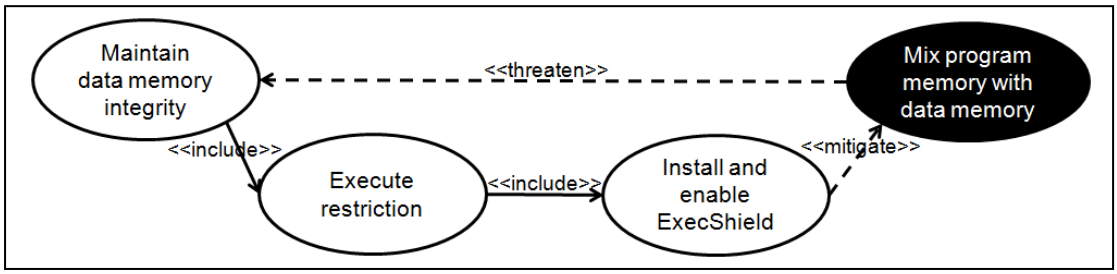
ซึ่งในแต่ละกรณี มีรายละเอียดของแบบรูปความมั่นคงปลอดภัย ดังต่อไปนี้

แบบรูปความมั่นคงปลอดภัยนี้ เป็นแบบรูปที่มีไว้เพื่อช่วยเหลือนักออกแบบซอฟต์แวร์ ซึ่งอาจไม่ได้เป็นผู้เชี่ยวชาญทางด้านความมั่นคงปลอดภัยมากนัก โดยเฉพาะเรื่องฮาร์ดเดนนิ่งของระบบ

4.1. **แบบรูปการแยกหน่วยความจำชุดคำสั่งและหน่วยความจำข้อมูล** (Pattern for separation of program and data memory)

บริบท: แบบรูปนี้กำหนดว่า ระบบจะปลอดภัยจากปฏิบัติการของรหัสคำสั่งที่เป็นอันตรายในหน่วยความจำข้อมูล

ปัญหา: ผู้โจมตีสามารถโจมตีระบบโดยทำการป้อนรหัสคำสั่งผ่านพารามิเตอร์ที่ให้บริการหรือทำให้เกิดการล้นของบัฟเฟอร์ (Buffer Overflow) ผ่านทางช่องโหว่นี้ ซึ่งสามารถแสดงมิสยูสเคสได้ ดังภาพที่ 4.1

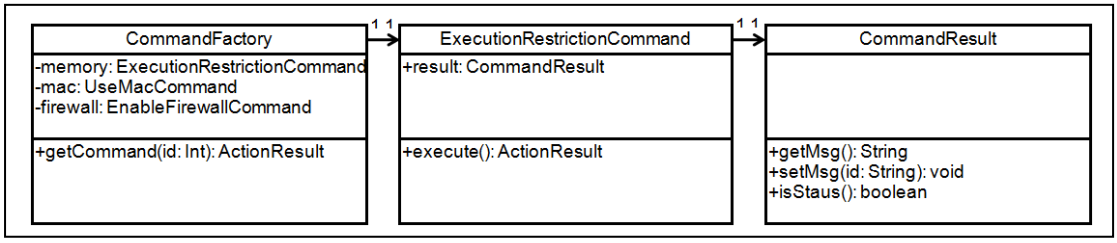


ภาพที่ 4.1 มิสยูสเคสสำหรับการประเมินผลหน่วยความจำ

จากภาพที่ 4.1 การที่ระบบเปิดช่องโหว่ให้เกิดการโจมตีหน่วยความจำผ่านทางการที่ระบบมีการรวมหน่วยความจำชุดคำสั่งและหน่วยความจำข้อมูลเข้าไว้ด้วยกัน ทำให้สะดวกต่อการป้อนคำสั่งที่เป็นอันตรายเข้าไปแฝงตัวในระบบ ดังนั้นระบบจำเป็นต้องหาวิธีในการปกป้องหน่วยความจำ ซึ่งวิธีหนึ่งในการปกป้องคือ การที่ระบบต้องจำกัดการประมวลผล โดยการเปิดใช้งาน Exec Shield ทำให้เกิดการแยกหน่วยความจำชุดคำสั่งและหน่วยความจำข้อมูลออกจากกัน

การแก้ไข: ทำการตรวจสอบว่า ระบบมีการเปิดใช้งาน Exec Shield

โครงสร้าง: จากมิสยูสเคสในภาพที่ 4.1 สามารถนำมาออกแบบระบบในรูปแบบของแผนภาพคลาสได้ ดังภาพที่ 4.2



ภาพที่ 4.2 แผนภาพคลาสสำหรับการประเมินผลหน่วยความจำ

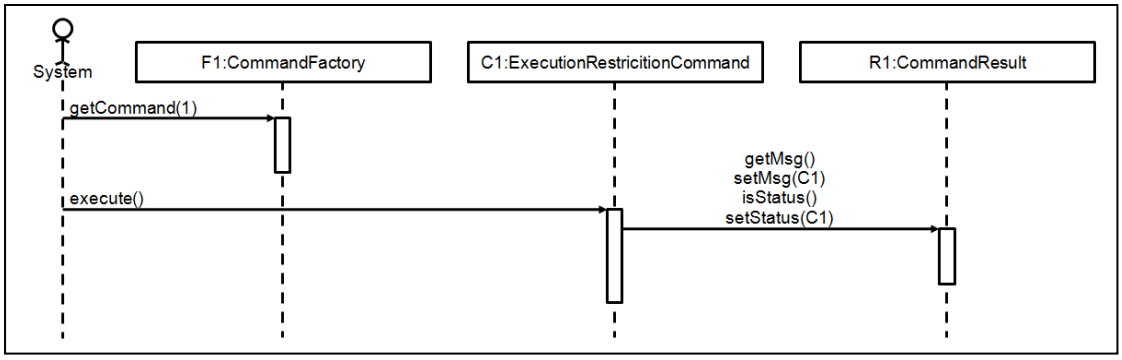
จากภาพที่ 4.2 คลาส CommandFactory ใช้ในการรับคำสั่งในการตรวจสอบฮาร์ดเดนนิ่งของระบบว่า ระบบมีการจำกัดการประมวลผลหรือไม่ โดยเรียกคลาส ExecutionRestrictionCommand เพื่อเรียกคำสั่งในการตรวจสอบว่า ระบบมีการเปิดใช้งาน Exec Shield หรือไม่ จากนั้นแสดงผลลัพธ์จากการตรวจสอบฮาร์ดเดนนิ่งของระบบผ่านทางคลาส CommandResult

ไดนามิก: ใช้คำสั่ง:

cat /proc/sys/kernel/exec-shield

ถ้าผลลัพธ์คือ 1 แสดงว่า ระบบมีการเปิดใช้งาน Exec Shield นั่นคือระบบได้มีการแยกส่วนของหน่วยความจำชุดคำสั่งออกจากหน่วยความจำข้อมูล

การทำให้เกิดผล: ทำการประเมินความมั่นคงปลอดภัยของระบบ โดยนำแผนภาพคลาสในภาพที่ 4.2 มาออกแบบลำดับการทำงานของระบบในรูปแบบของแผนภาพลำดับได้ ดังภาพที่ 4.3



ภาพที่ 4.3 แผนภาพลำดับสำหรับการประเมินผลหน่วยความจำ

จากภาพที่ 4.3 ระบบส่งข้อความ getCommand(1) ไปยังอ็อบเจกต์ F1:CommandFactory เพื่อแสดงว่า ระบบต้องการตรวจสอบการจำกัดการประมวลผล และส่งข้อความ execute() ไปยังอ็อบเจกต์ C1:ExecutionRestrictionCommand เพื่อเรียกคำสั่งในการตรวจสอบการเปิดใช้งาน Exec Shield จากนั้นอ็อบเจกต์ C1:ExecutionRestrictionCommand ส่งข้อความ getMSG() setMSG(C1) isStatus() และ setStatus(C1) ไปยังอ็อบเจกต์ R1:CommandResult เพื่อต้องการทราบว่า ระบบมีการจำกัดการประมวลผลหรือไม่

ตัวอย่างการแก้ปัญหา: ใช้ Exec Shield ในการตรวจสอบว่า ระบบมีการแยกหน่วยความจำชุดคำสั่งออกจากหน่วยความจำข้อมูล

ข้อจำกัด: ใช้สำหรับลีนุกซ์เท่านั้น

การใช้ประโยชน์หรือเป็นที่รู้จัก: การประเมินระบบว่า มีการเปิดใช้งาน Exec Shield หรือไม่ โดยดำเนินการในขั้นตอนการติดตั้งของกระบวนการพัฒนาซอฟต์แวร์ โดยเข้าทำการประเมินความมั่นคงปลอดภัยของระบบโดยอัตโนมัติในขณะติดตั้งซอฟต์แวร์ประยุกต์

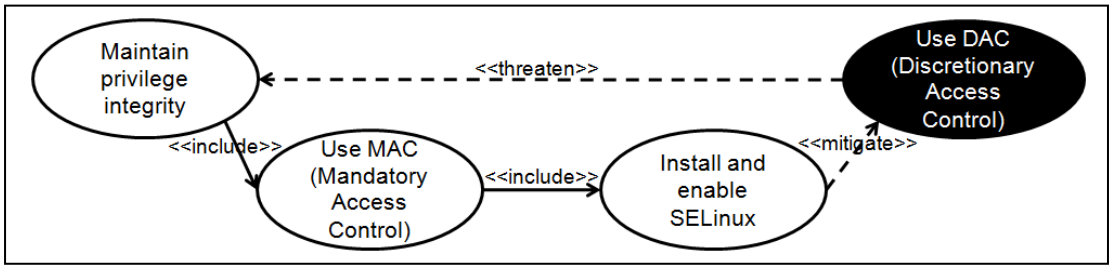
ผลที่ตามมา: ปกป้องภัยคุกคามต่างๆ ที่อาจเกิดขึ้นได้

4.2. **แบบรูปการกำหนดสิทธิ์ของผู้ใช้** (Pattern for limiting user privileges)

บริบท: แบบรูปนี้กำหนดว่า ระบบใช้ MAC ในการตั้งค่าสิทธิ์การใช้งานของผู้ใช้

ปัญหา: ผู้ใช้สามารถเขียนทับหรือปรับเปลี่ยนนโยบายความมั่นคงปลอดภัยเนื่องจากระบบใช้ DAC ซึ่งอนุญาตให้ผู้ใช้สามารถกำหนดนโยบายและคุณสมบัติด้านความมั่นคง

ปลอดภัยได้เอง เช่น ผู้ใช้ กลุ่ม และสิทธิ์การอ่าน (Read) – เขียน (Write) – ปฏิบัติ (Execute) ซึ่งสามารถแสดงมิสยูสเคสได้ ดังภาพที่ 4.4

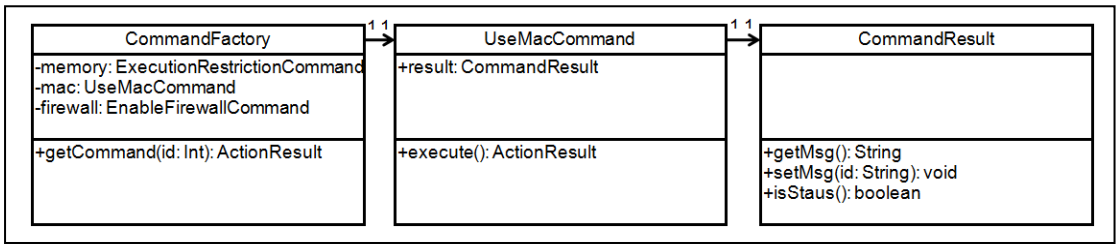


ภาพที่ 4.4 มิสยูสเคสสำหรับการประเมินผลการควบคุมการเข้าถึง

จากภาพที่ 4.4 การที่ระบบเปิดช่องโหว่ให้เกิดการโจมตีการควบคุมการเข้าถึงผ่านทางการที่ระบบมีการใช้ DAC ทำให้ง่ายต่อการกำหนดสิทธิ์ที่ไม่เหมาะสมต่อระบบ ดังนั้นระบบจำเป็นต้องหาวิธีในการปกป้องการควบคุมการเข้าถึง ซึ่งวิธีหนึ่งในการปกป้องคือ การที่ระบบต้องใช้ MAC โดยการเปิดใช้งาน SELinux ทำให้เกิดการกำหนดสิทธิ์ของผู้ใช้

การแก้ไข: ทำการตรวจสอบว่า ระบบมีการเปิดใช้งาน SELinux

โครงสร้าง: จากมิสยูสเคสในภาพที่ 4.4 สามารถนำมาออกแบบระบบในรูปแบบของแผนภาพคลาสได้ ดังภาพที่ 4.5



ภาพที่ 4.5 แผนภาพคลาสสำหรับการประเมินผลการควบคุมการเข้าถึง

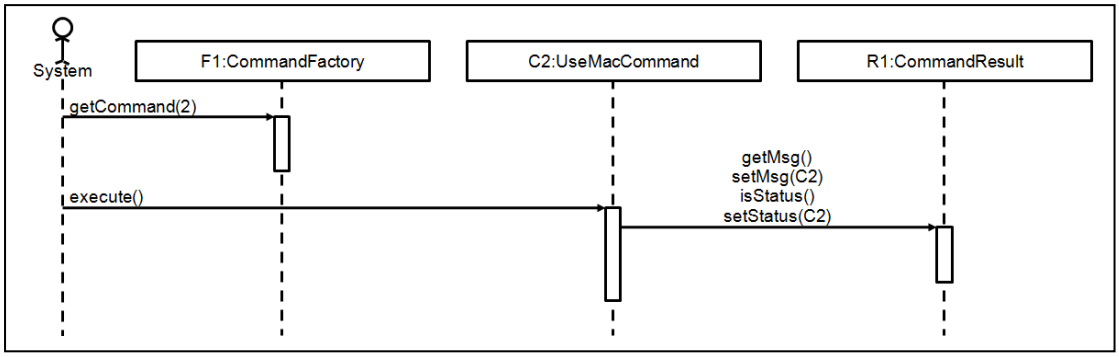
จากภาพที่ 4.5 คลาส CommandFactory ใช้ในการรับคำสั่งในการตรวจสอบฮาร์ดเดนนิ่งของระบบว่า ระบบมีการใช้ MAC หรือไม่ โดยเรียกคลาส UseMacCommand เพื่อเรียกคำสั่งในการตรวจสอบว่า ระบบมีการเปิดใช้งาน SELinux หรือไม่ จากนั้นแสดงผลลัพธ์จากการตรวจสอบฮาร์ดเดนนิ่งของระบบผ่านทางคลาส CommandResult

ไดนามิก: ใช้คำสั่ง:

cat /selinux/enforce

ถ้าผลลัพธ์คือ 1 แสดงว่า ระบบมีการเปิดใช้งาน SELinux นั่นคือ ระบบได้มีการใช้ MAC

การทำให้เกิดผล: ทำการประเมินความมั่นคงปลอดภัยของระบบ โดยนำแผนภาพคลาสในภาพที่ 4.5 มาออกแบบลำดับการทำงานของระบบในรูปแบบของแผนภาพลำดับได้ ดังภาพที่ 4.6



ภาพที่ 4.6 แผนภาพลำดับสำหรับการประเมินผลการควบคุมการเข้าถึง

จากภาพที่ 4.6 ระบบส่งข้อความ getCommand(2) ไปยังอ็อบเจกต์ F1:CommandFactory เพื่อแสดงว่า ระบบต้องการตรวจสอบการใช้ MAC และส่งข้อความ execute() ไปยังอ็อบเจกต์ C2: UseMacCommand เพื่อเรียกคำสั่งในการตรวจสอบการเปิดใช้งาน SELinux จากนั้นอ็อบเจกต์ C2: UseMacCommand ส่งข้อความ getMSG() setMSG(C2) isStatus() และ setStatus(C2) ไปยังอ็อบเจกต์ R1:CommandResult เพื่อต้องการทราบว่า ระบบมีการใช้ MAC หรือไม่

ตัวอย่างการแก้ปัญหา: ใช้ SELinux ในการตรวจสอบว่า ระบบมีการใช้ MAC

ข้อจำกัด: ใช้สำหรับลีนุกซ์เท่านั้น

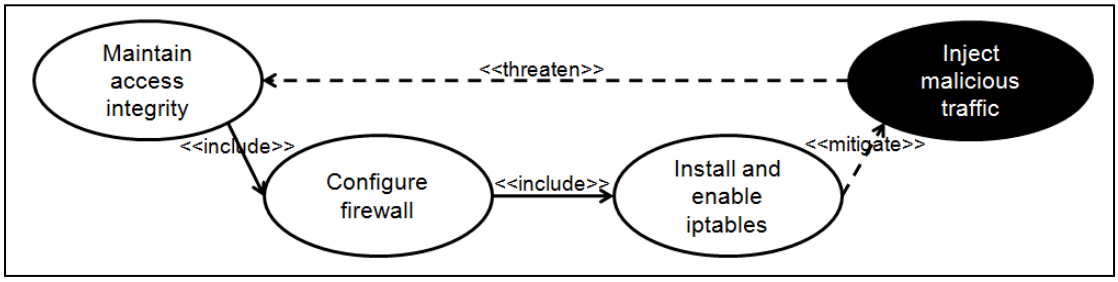
การใช้ประโยชน์หรือเป็นที่รู้จัก: การประเมินระบบว่า มีการเปิดใช้งาน SELinux หรือไม่ โดยดำเนินการในขั้นตอนการติดตั้งของกระบวนการพัฒนาซอฟต์แวร์ โดยเข้าทำการประเมินความมั่นคงปลอดภัยของระบบโดยอัตโนมัติในขณะติดตั้งซอฟต์แวร์ประยุกต์

ผลที่ตามมา: ป้องกันผู้ใช้กำหนดสิทธิ์ที่ไม่เหมาะสม

4.3. **แบบรูปการตั้งค่าไฟร์วอลล์** (Pattern for configuring firewall)

บริบท: แบบรูปนี้กำหนดว่า ระบบต้องมีการเปิดใช้งานไฟร์วอลล์

ปัญหา: ระบบที่มีการเชื่อมต่อกับระบบอื่น หรือเครือข่ายอื่น โดยที่ไม่มีไฟร์วอลล์เท่ากับเป็นการเปิดช่องโหว่ให้ระบบสามารถถูกโจมตีหรือบุกรุกได้อย่างง่ายดาย เนื่องจากไฟร์วอลล์ทำหน้าที่ในการกรองทราฟฟิกที่มาจากภายนอกและควบคุมการใช้งานของระบบเมื่อต้องติดต่อกับเครือข่ายอื่น ซึ่งสามารถแสดงมิสยูสเคสได้ ดังภาพที่ 4.7

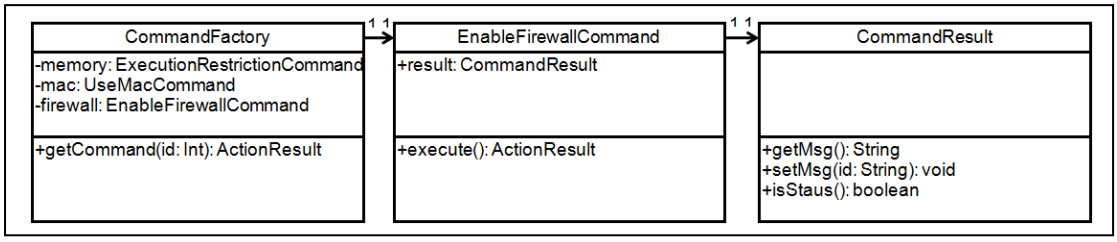


ภาพที่ 4.7 มิสยูสเคสสำหรับการประเมินผลไฟร์วอลล์

จากภาพที่ 4.7 การที่ระบบเปิดช่องโหว่ให้เกิดการโจมตีการเข้าถึงระบบผ่านทางการที่ระบบไม่มีการใช้งานไฟร์วอลล์ ทำให้สะดวกต่อการสร้างทราฟฟิกที่ประสงค์ร้ายเข้าไปในระบบ ดังนั้น ระบบจำเป็นต้องหาวิธีในการปกป้องช่องทางการเข้าถึงระบบ ซึ่งวิธีหนึ่งในการปกป้องคือ การที่ระบบต้องตั้งค่าการใช้งานไฟร์วอลล์ โดยการเปิดใช้งาน iptables ทำให้เกิดการตั้งค่าไฟร์วอลล์

การแก้ไข: ทำการตรวจสอบว่า ระบบมีการเปิดใช้งาน iptables

โครงสร้าง: จากมิสยูสเคสในภาพที่ 4.7 สามารถนำมาออกแบบระบบในรูปแบบของแผนภาพคลาสได้ ดังภาพที่ 4.8



ภาพที่ 4.8 แผนภาพคลาสสำหรับการประเมินผลไฟร์วอลล์

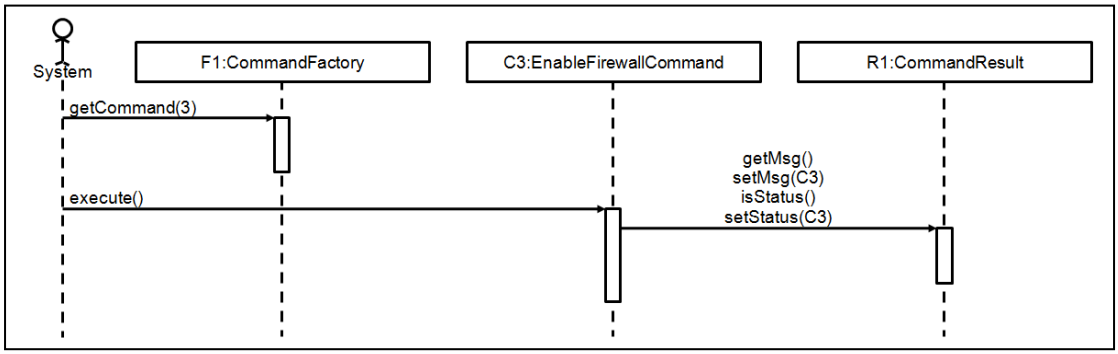
จากภาพที่ 4.8 คลาส CommandFactory ใช้ในการรับคำสั่งในการตรวจสอบฮาร์ดเดนนิ่งของระบบว่า ระบบมีการตั้งค่าการใช้งานไฟร์วอลล์หรือไม่ โดยเรียกคลาส EnableFirewallCommand เพื่อเรียกคำสั่งในการตรวจสอบว่า ระบบมีการเปิดใช้งาน iptables หรือไม่ จากนั้นแสดงผลลัพธ์จากการตรวจสอบฮาร์ดเดนนิ่งของระบบผ่านทางคลาส CommandResult

ไดนามิก: ใช้คำสั่ง:

service iptables status

ถ้าผลลัพธ์คือ Start แสดงว่า ระบบมีการเปิดใช้งาน iptables นั่นคือระบบได้มีการเปิดใช้งานไฟร์วอลล์

การทำให้เกิดผล: ทำการประเมินความมั่นคงปลอดภัยของระบบ โดยนำแผนภาพคลาสในภาพที่ 4.8 มาออกแบบลำดับการทำงานของระบบในรูปแบบของแผนภาพลำดับได้ ดังภาพที่ 4.9



ภาพที่ 4.9 แผนภาพลำดับสำหรับการประเมินผลไฟร์วอลล์

จากภาพที่ 4.9 ระบบส่งข้อความ getCommand(3) ไปยังอ็อบเจกต์ F1:CommandFactory เพื่อแสดงว่า ระบบต้องการตรวจสอบการตั้งค่าการใช้งานไฟร์วอลล์ และส่งข้อความ execute() ไปยังอ็อบเจกต์ C3: EnableFirewallCommand เพื่อเรียกคำสั่งในการตรวจสอบการเปิดใช้งาน iptables จากนั้นอ็อบเจกต์ C3: EnableFirewallCommand ส่งข้อความ getMSG() setMSG(C3) isStatus() และ setStatus(C3) ไปยังอ็อบเจกต์ R1:CommandResult เพื่อต้องการทราบว่า ระบบมีการเปิดใช้งานไฟร์วอลล์หรือไม่

ตัวอย่างการแก้ปัญหา: ใช้ iptables ในการตรวจสอบว่า ระบบมีการเปิดใช้งานไฟร์วอลล์

ข้อจำกัด: ใช้สำหรับลีนุกซ์เท่านั้น

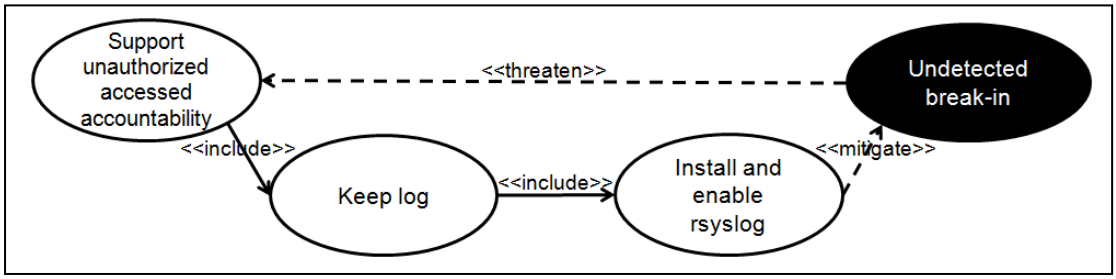
การใช้ประโยชน์หรือเป็นที่รู้จัก: การประเมินระบบว่า มีการเปิดใช้งาน iptables หรือไม่ โดยดำเนินการในขั้นตอนการติดตั้งของกระบวนการพัฒนาซอฟต์แวร์ โดยเข้าทำการประเมินความมั่นคงปลอดภัยของระบบโดยอัตโนมัติในขณะติดตั้งซอฟต์แวร์ประยุกต์

ผลที่ตามมา: ป้องกันทราฟฟิกที่เป็นอันตราย

4.4. แบบรูปการเก็บบันทึกการใช้งานระบบ (Pattern for recording accesses to system)

บริบท: แบบรูปนี้กำหนดว่า ระบบมีการเก็บบันทึกกิจกรรมต่างๆที่เกิดขึ้นในระบบ

ปัญหา: ในกรณีของ break-in จำเป็นต้องใช้บันทึกการใช้งานระบบในการวิเคราะห์ ซึ่งสามารถแสดงมิสยูสเคสได้ ดังภาพที่ 4.10

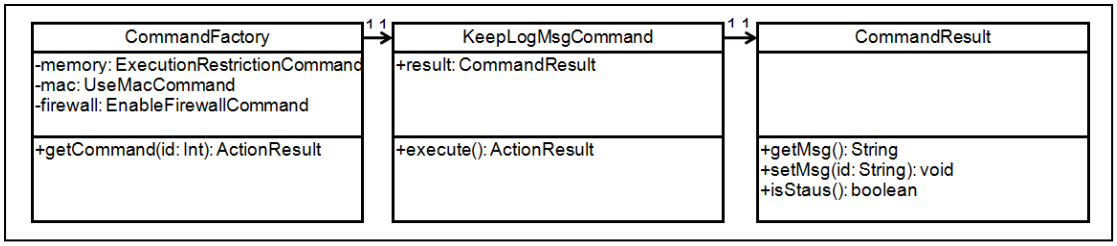


ภาพที่ 4.10 มิสยูสเคสสำหรับการประเมินผลบันทึกการใช้งาน

จากภาพที่ 4.10 การที่ระบบเปิดช่องโหว่ให้เกิดการโจมตีบันทึกการใช้งานผ่านทางการที่ระบบไม่มีการตรวจจับการบุกรุก ทำให้ยากต่อการตรวจจับการกระทำที่ไม่เหมาะสมที่เกิดขึ้นในระบบ ดังนั้น ระบบจำเป็นต้องหาวิธีในการปกป้องบันทึกการใช้งาน ซึ่งวิธีหนึ่งในการปกป้องคือ การที่ระบบต้องเก็บบันทึกการใช้งานระบบ โดยการเปิดใช้งาน rsyslog ทำให้เกิดการเก็บบันทึกกิจกรรมที่เกิดขึ้นในระบบ

การแก้ไข: ทำการตรวจสอบว่า ระบบมีการเปิดใช้งาน rsyslog

โครงสร้าง: จากมิสยูสเคสในภาพที่ 4.10 สามารถนำมาออกแบบระบบในรูปแบบของแผนภาพคลาสได้ ดังภาพที่ 4.11



ภาพที่ 4.11 แผนภาพคลาสสำหรับการประเมินผลบันทึกการใช้งาน

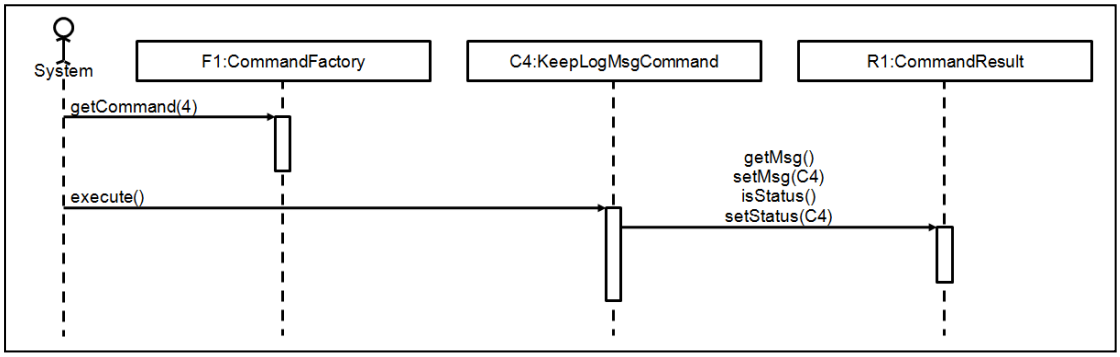
จากภาพที่ 4.11 คลาส CommandFactory ใช้ในการรับคำสั่งในการตรวจสอบฮาร์ดเดนนิ่งของระบบว่า ระบบมีการเก็บบันทึกการใช้งานระบบหรือไม่ โดยเรียกคลาส KeepLogMsgCommand เพื่อเรียกคำสั่งในการตรวจสอบว่า ระบบมีการเปิดใช้งาน rsyslog หรือไม่ จากนั้นแสดงผลลัพธ์จากการตรวจสอบฮาร์ดเดนนิ่งของระบบผ่านทางคลาส CommandResult

ไดนามิก: ใช้คำสั่ง:

service rsyslog status

ถ้าผลลัพธ์คือ Start แสดงว่า ระบบมีการเปิดใช้งาน rsyslog นั่นคือระบบได้มีการเก็บบันทึกการใช้งาน

การทำให้เกิดผล: ทำการประเมินความมั่นคงปลอดภัยของระบบ โดยนำแผนภาพคลาสในภาพที่ 4.11 มาออกแบบลำดับการทำงานของระบบในรูปแบบของแผนภาพลำดับได้ ดังภาพที่ 4.12



ภาพที่ 4.12 แผนภาพลำดับสำหรับการประเมินผลบันทึกการใช้งาน

จากภาพที่ 4.12 ระบบส่งข้อความ getCommand(4) ไปยังอ็อบเจกต์ F1:CommandFactory เพื่อแสดงว่า ระบบต้องการตรวจสอบการเก็บบันทึกการใช้งานระบบ และ

ส่งข้อความ execute() ไปยังอ็อบเจกต์ C4: KeepLogMsgCommand เพื่อเรียกคำสั่งในการตรวจสอบการเปิดใช้งาน rsyslog จากนั้นอ็อบเจกต์ C4: KeepLogMsgCommand ส่งข้อความ getMSG() setMSG(C4) isStatus() และ setStatus(C4) ไปยังอ็อบเจกต์ R1:CommandResult เพื่อต้องการทราบว่า ระบบมีการเก็บบันทึกการใช้งานหรือไม่

ตัวอย่างการแก้ปัญหา: ใช้ rsyslog ในการตรวจสอบว่า ระบบมีการเก็บประวัติกิจกรรมต่างๆ ที่เกิดขึ้นในระบบ

ข้อจำกัด: ใช้สำหรับลีนุกซ์เท่านั้น

การใช้ประโยชน์หรือเป็นที่รู้จัก: การประเมินระบบว่า มีการเปิดใช้งาน rsyslog หรือไม่ โดยดำเนินการในขั้นตอนการติดตั้งของกระบวนการพัฒนาซอฟต์แวร์ โดยเข้าทำการประเมินความมั่นคงปลอดภัยของระบบโดยอัตโนมัติในขณะติดตั้งซอฟต์แวร์ประยุกต์

ผลที่ตามมา: ใช้เป็นข้อมูลอ้างอิงสำคัญสำหรับการตรวจสอบเส้นทางในกรณี break-in

สำหรับคำอธิบายของแผนภาพต่างๆ ที่อยู่ในแบบรูปความมั่นคงปลอดภัย สามารถดูได้ที่ภาคผนวก โดยคำอธิบายยูสเคสดูได้ที่ภาคผนวก ก คำอธิบายมิสยูสเคสดูได้ที่ภาคผนวก ข และบัตรซีอาร์ซีดูได้ที่ภาคผนวก ค

ที่กล่าวมาในบทที่ 4 นี้ เป็นการกล่าวถึงแบบรูปความมั่นคงปลอดภัย ที่ใช้เป็นตัวอย่างในการทำวิจัย โดยนำเสนอสี่กรณี ได้แก่ แบบรูปการแยกหน่วยความจำชุดคำสั่งและหน่วยความจำข้อมูล แบบรูปการกำหนดสิทธิ์ของผู้ใช้ แบบรูปการตั้งค่าไฟร์วอลล์ และแบบรูปการเก็บบันทึกการใช้งานระบบ ส่วนในบทถัดไป จะเป็นการประยุกต์ใช้แบบรูปความมั่นคงปลอดภัย

บทที่ 5
การประยุกต์ใช้แบบรูปความมั่นคงปลอดภัย

เนื้อหาในบทนี้ กล่าวถึง การประยุกต์ใช้แบบรูปความมั่นคงปลอดภัย โดยสามารถแบ่งออกเป็น 2 ส่วน ดังนี้

1) การประยุกต์ใช้แบบรูปความมั่นคงปลอดภัยในกระบวนการพัฒนาซอฟต์แวร์

2) การประยุกต์ใช้แบบรูปความมั่นคงปลอดภัยในกรณีศึกษา

2.1) กรณีศึกษา 1 คือ ซอฟต์แวร์ประยุกต์ GoMapGen

2.2) กรณีศึกษา 2 คือ ซอฟต์แวร์ประยุกต์ JavaPoint

2.3) กรณีศึกษา 3 คือ ซอฟต์แวร์ประยุกต์ JCalculator

ซึ่งในแต่ละส่วน มีรายละเอียดของการประยุกต์ใช้แบบรูปความมั่นคงปลอดภัยดังต่อไปนี้

5.1. การประยุกต์ใช้แบบรูปความมั่นคงปลอดภัยในกระบวนการพัฒนาซอฟต์แวร์

จากแบบรูปความมั่นคงปลอดภัยทั้ง 4 แบบรูป ประกอบด้วย

1) แบบรูปการแยกหน่วยความจำชุดคำสั่งและหน่วยความจำข้อมูล

2) แบบรูปการกำหนดสิทธิ์ของผู้ใช้

3) แบบรูปการตั้งค่าไฟร์วอลล์

4) แบบรูปการเก็บบันทึกการใช้งานระบบ

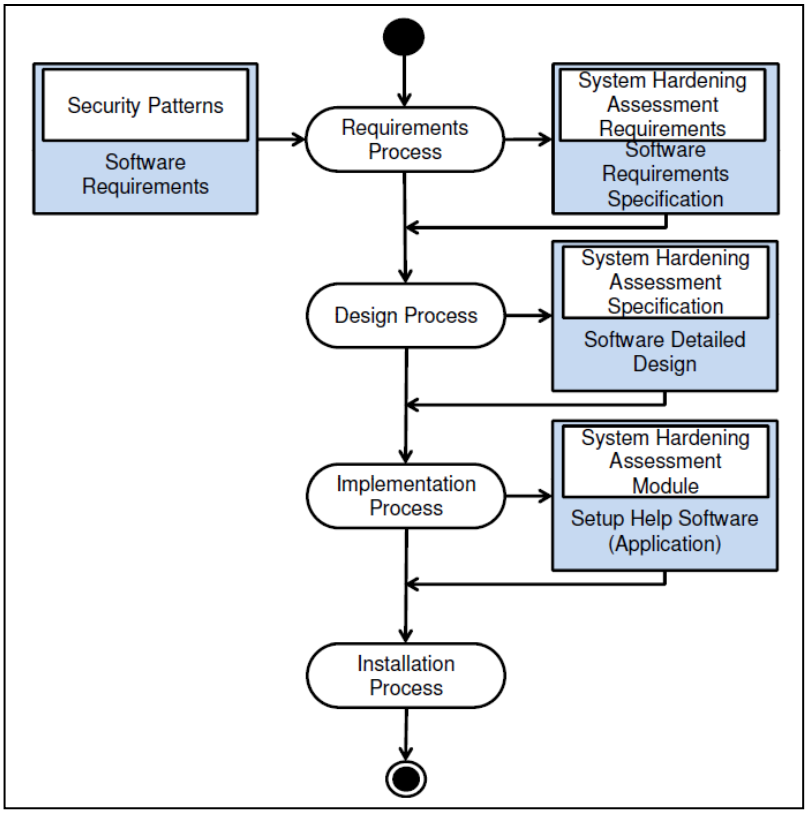
สามารถนำมาประยุกต์ใช้ในกระบวนการพัฒนาซอฟต์แวร์ ดังภาพที่ 5.1 ซึ่งแบ่งกระบวนการพัฒนาซอฟต์แวร์เป็น 4 ขั้นตอน ประกอบด้วย

1) ขั้นตอนความต้องการ

2) ขั้นตอนการออกแบบ

3) ขั้นตอนการพัฒนา

4) ขั้นตอนการติดตั้ง



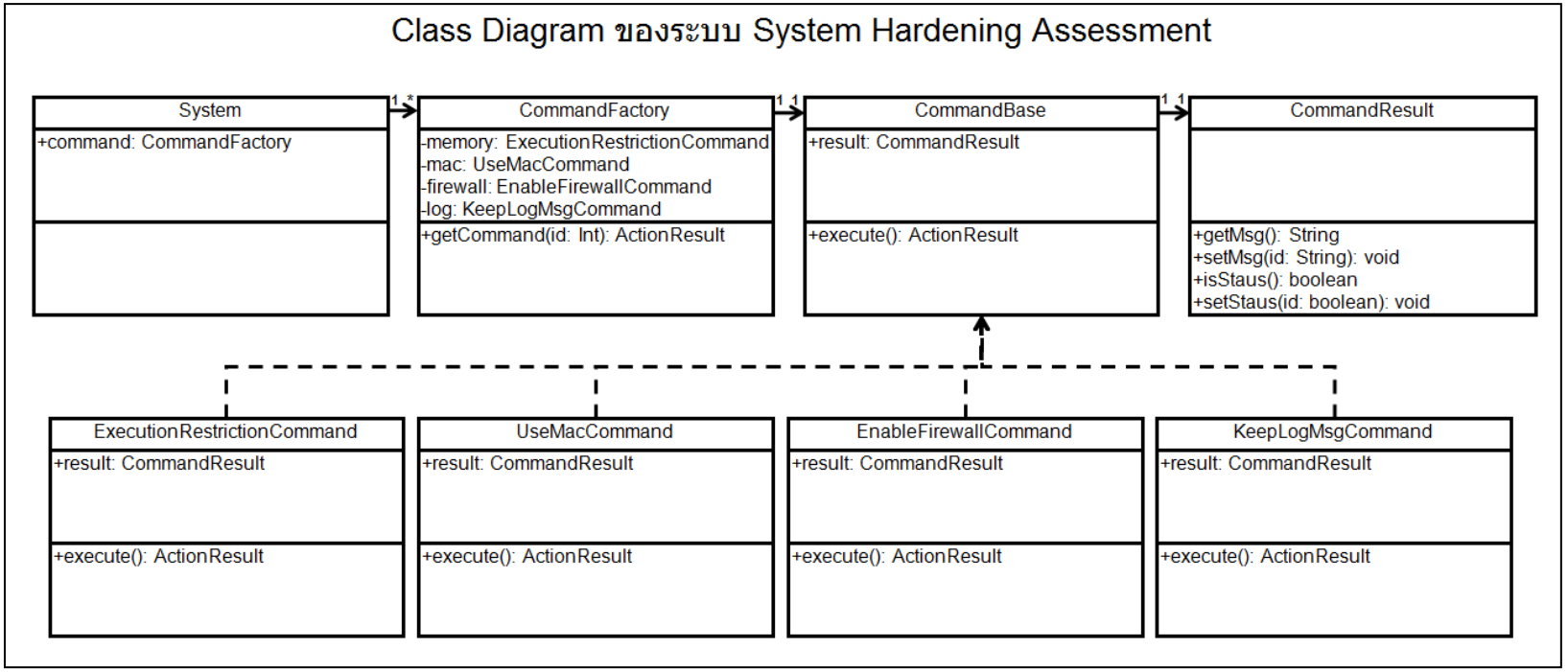
ภาพที่ 5.1 แผนภาพกิจกรรมของการประยุกต์ใช้แบบรูปความมั่นคงปลอดภัยในกระบวนการพัฒนาซอฟต์แวร์

5.1.1. ขั้นตอนความต้องการ

ขั้นตอนนี้ เป็นการนำเอาแบบรูปความมั่นคงปลอดภัยทั้ง 4 แบบรูป ซึ่งเป็นความต้องการด้านความมั่นคงปลอดภัยในส่วนของการประเมินฮาร์ดเดนนิ่งของระบบในขณะติดตั้งซอฟต์แวร์ประยุกต์ มาผสมรวมกับความต้องการเชิงหน้าที่ด้านอื่นๆ ของซอฟต์แวร์ประยุกต์ โดยจัดทำเป็นรายละเอียดข้อกำหนดความต้องการของซอฟต์แวร์

5.1.2. ขั้นตอนการออกแบบ

ขั้นตอนนี้ เป็นการนำเอารายละเอียดข้อกำหนดความต้องการซอฟต์แวร์ที่ได้จากขั้นตอนความต้องการมาทำการออกแบบระบบ เพื่อนำไปใช้ในการพัฒนาระบบต่อไป โดยในส่วนของการประเมินความมั่นคงปลอดภัยของระบบ สามารถนำแผนภาพมิสยูสเคส แผนภาพคลาส และแผนภาพลำดับของแบบรูปความมั่นคงปลอดภัยไปใช้ในการออกแบบร่วมกันกับการออกแบบซอฟต์แวร์ประยุกต์ได้ ซึ่งสามารถออกแบบแผนภาพคลาสในส่วนของการประเมินความมั่นคงปลอดภัยของระบบ ดังภาพที่ 5.2



ภาพที่ 5.2 การออกแบบแผนภาพคลาสในส่วนของการประเมินฮาร์ดเดนนิ่งของระบบ

จากภาพที่ 5.2 เป็นการนำแผนภาพคลาสที่ได้จากแต่ละแบบรูปมาออกแบบเป็นแผนภาพคลาสในส่วนของการประเมินฮาร์ดเดนนิ่งของระบบ โดยคลาส System เรียกใช้คำสั่งในการตรวจสอบฮาร์ดเดนนิ่งของระบบ ผ่านทางคลาส CommandFactory โดยคำสั่งที่ใช้ในการตรวจสอบฮาร์ดเดนนิ่งของระบบ ประกอบด้วยสี่คำสั่ง ได้แก่ คำสั่งในการตรวจสอบการจำกัดการประมวลผล คำสั่งในการตรวจสอบการใช้ MAC คำสั่งในการตรวจสอบการตั้งค่าการใช้งานไฟร์วอลล์ และคำสั่งในการตรวจสอบการเก็บบันทึกการใช้งานระบบ ซึ่งคลาส CommandFactory จะเรียกใช้คำสั่งในการตรวจสอบฮาร์ดเดนนิ่งของระบบ ผ่านทางคลาส CommandBase ซึ่งประกอบด้วย คลาส ExecutionRestrictionCommand คลาส UseMacCommand คลาส EnableFirewallCommand และคลาส KeepLogMsgCommand ทั้งนี้สามารถแสดงผลลัพธ์จากการตรวจสอบฮาร์ดเดนนิ่งของระบบผ่านทางคลาส CommandResult

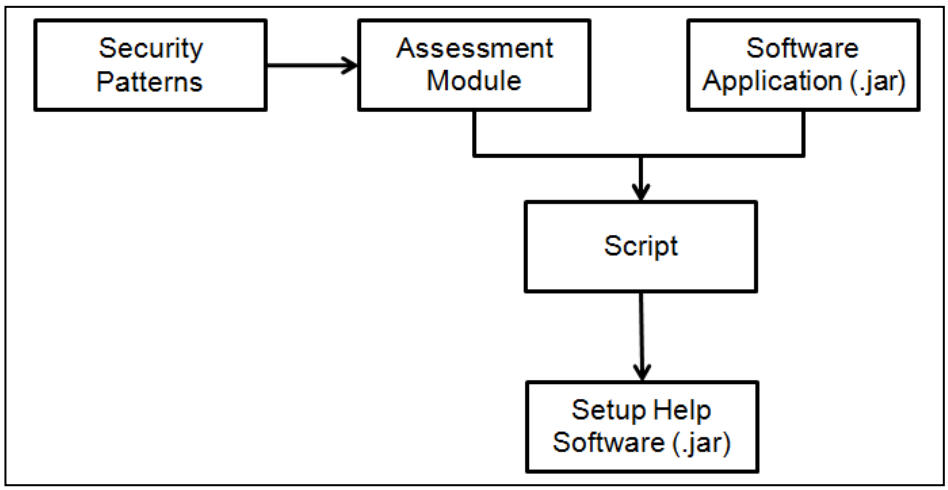
5.1.3. ขั้นตอนการพัฒนา

ขั้นตอนนี้ เป็นการนำเอารายละเอียดการออกแบบระบบที่ได้จากขั้นตอนการออกแบบมาพัฒนา โดยในส่วนของการประเมินความมั่นคงปลอดภัยของระบบสามารถนำแบบรูปความมั่นคงปลอดภัยที่ได้มาพัฒนาเป็นโมดูลการประเมินฮาร์ดเดนนิ่งของระบบ ซึ่งแสดงหน้าจอการประเมินฮาร์ดเดนนิ่งของระบบได้ ดังภาพที่ 5.3 ทั้งนี้สามารถดูซอร์สโคสของโมดูลการประเมินฮาร์ดเดนนิ่งของระบบได้ในภาคผนวก ง

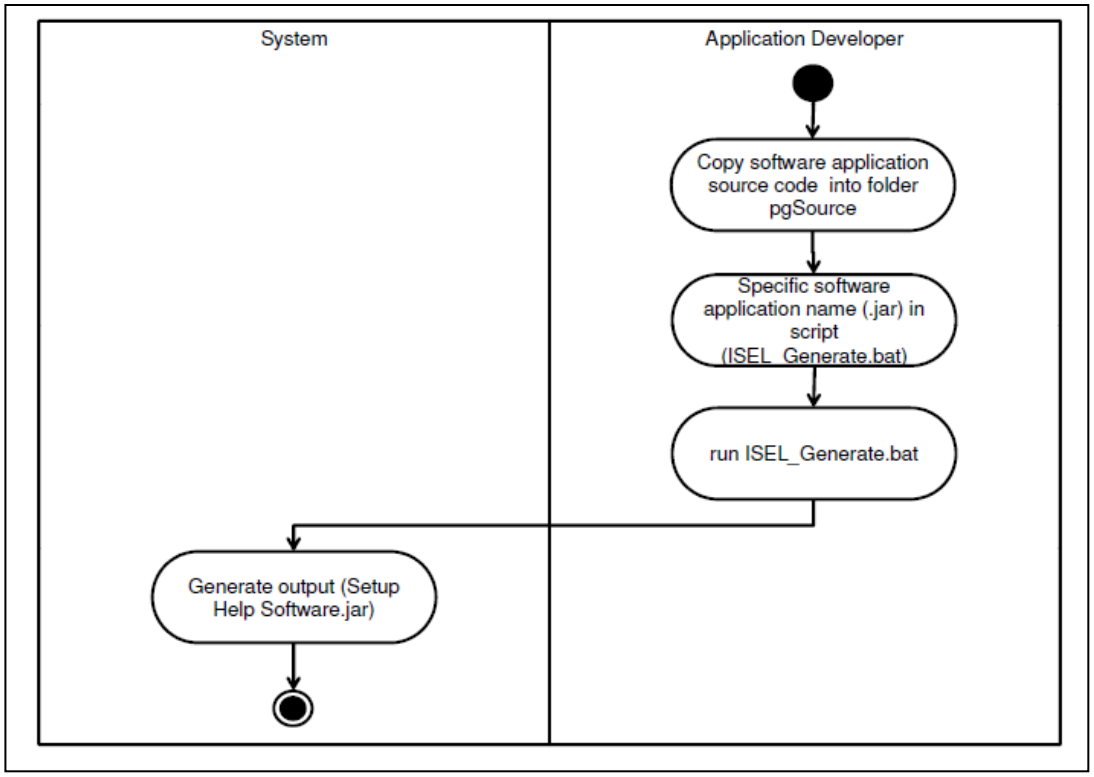
ภาพที่ 5.3 หน้าจอการประเมินฮาร์ดเดนนิ่งของระบบ

จากภาพที่ 5.3 เป็นหน้าจอการประเมินฮาร์ดเดนนิ่งของระบบ โดยทำการประเมินฮาร์ดเดนนิ่งในสี่กรณีตัวอย่าง ได้แก่ การจำกัดการประมวลผล (Execution Restriction) แมค (MAC) ไฟร์วอลล์ (Firewall) และบันทึกการใช้งาน (Log) ซึ่งในแต่ละฮาร์ดเดนนิ่ง จะแสดงผลการประเมินเป็น มีการใช้งานฮาร์ดเดนนิ่ง (Active) หรือ ไม่มีการใช้งานฮาร์ดเดนนิ่ง (Not Active) ซึ่งจากภาพที่ 5.3 จะเห็นว่า เป็นแอคทีฟทั้งหมด นั่นแสดงว่า มีการใช้งานฮาร์ดเดนนิ่งทุกตัว

เมื่อนำแบบรูปความมั่นคงปลอดภัยมาพัฒนาเป็นโมดูลการประเมินฮาร์ดเดนนิ่งของระบบแล้ว จากนั้นนำมาพัฒนาต่อเป็นโมดูลการติดตั้งของซอฟต์แวร์ (Setup Help Software) เพื่อสะดวกต่อการนำไปใช้งานกับซอฟต์แวร์ประยุกต์ โดยเขียนสคริปต์ในการรวมโมดูลการประเมินฮาร์ดเดนนิ่งของระบบ และซอฟต์แวร์ประยุกต์ (.jar) เพื่อสร้างเป็นซอฟต์แวร์ช่วยเหลือการติดตั้ง (.jar) ดังภาพที่ 5.4 และภาพที่ 5.5 ทั้งนี้สามารถดูสคริปต์ในการรวมโมดูลการประเมินฮาร์ดเดนนิ่งของระบบเข้ากับซอฟต์แวร์ประยุกต์ได้ในภาคผนวก จ



ภาพที่ 5.4 ขั้นตอนการสร้างซอฟต์แวร์ช่วยเหลือการติดตั้ง

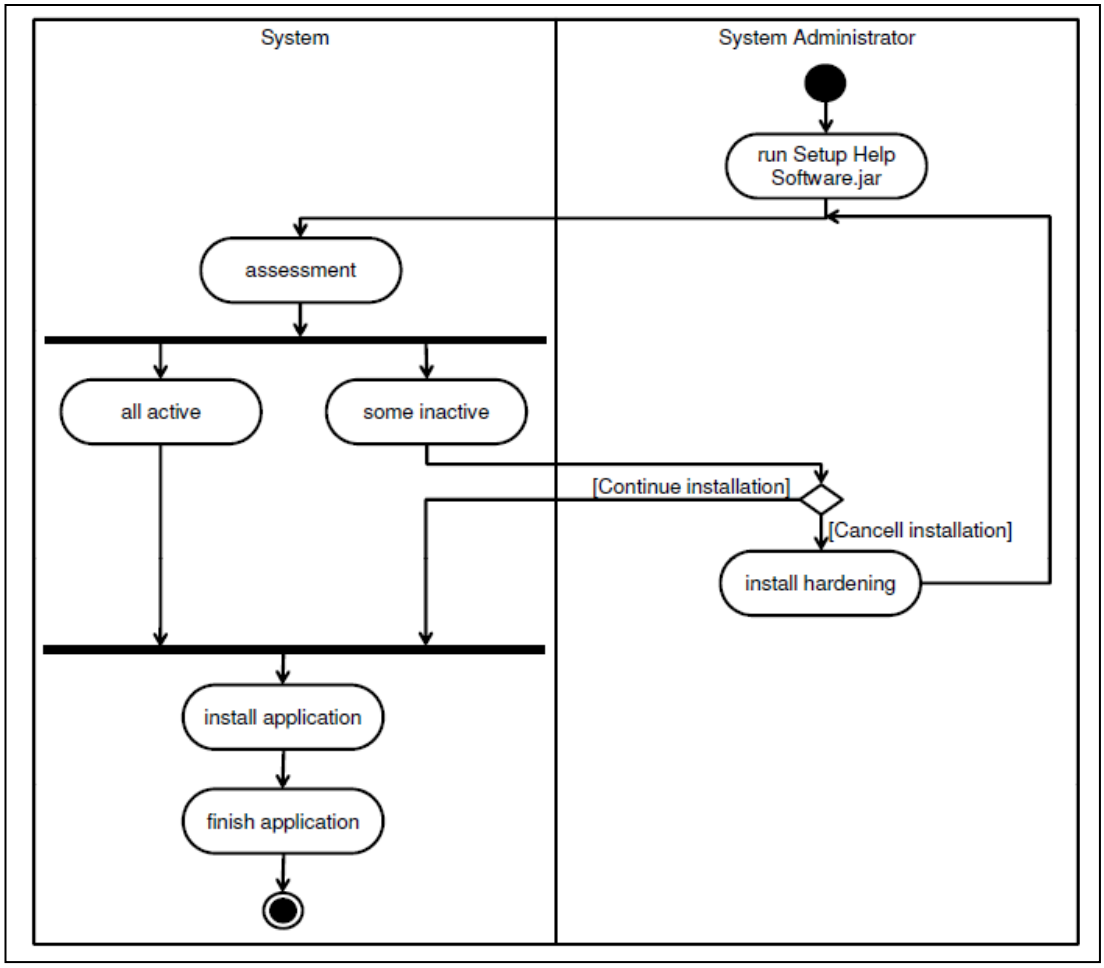


ภาพที่ 5.5 แผนภาพกิจกรรมของกระบวนการสร้างซอฟต์แวร์ช่วยเหลือการติดตั้ง

จากภาพที่ 5.5 เป็นการสร้างซอฟต์แวร์ช่วยเหลือการติดตั้ง โดยนำซอร์สโคดของซอฟต์แวร์ประยุกต์ที่ต้องการไปวางไว้ใน โฟลเดอร์ pgSource และระบุชื่อซอฟต์แวร์ประยุกต์ (.jar) ในไฟล์ ISEL_Generate.bat จากนั้นรันไฟล์ ISEL_Generate.bat จึงได้ผลลัพธ์เป็นซอฟต์แวร์ช่วยเหลือการติดตั้ง (Setup Help Software.jar)

5.1.4. ขั้นตอนการติดตั้ง

ขั้นตอนนี้ เป็นการนำเอาซอฟต์แวร์ช่วยเหลือการติดตั้งที่ได้จากขั้นตอนการพัฒนามาทำการติดตั้งลงบนระบบปฏิบัติการเพื่อใช้งาน โดยซอฟต์แวร์ที่พัฒนาขึ้นมานั้นจะเข้าทำการประเมินความมั่นคงปลอดภัยของระบบก่อนที่จะติดตั้งซอฟต์แวร์ประยุกต์ต่อไป ดังภาพที่ 5.6



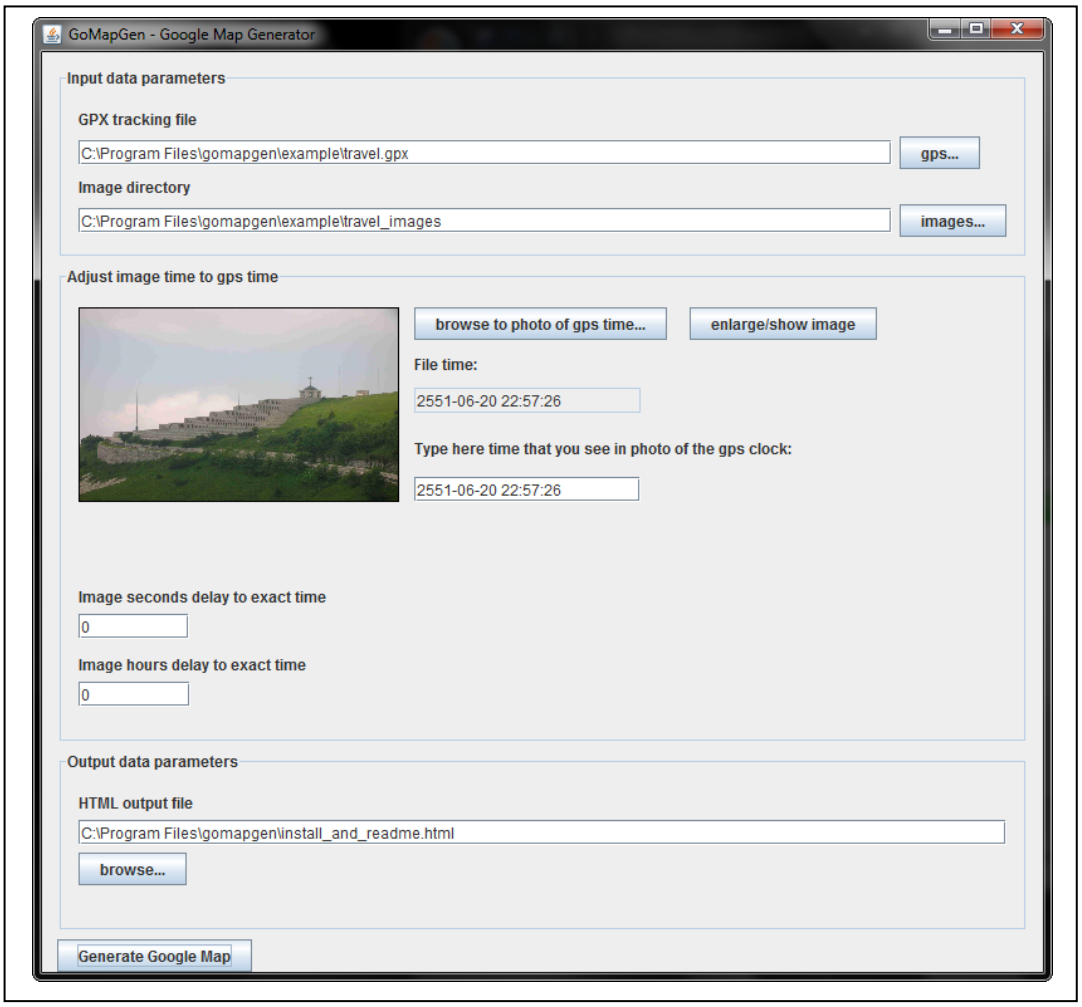
ภาพที่ 5.6 แผนภาพกิจกรรมของกระบวนการติดตั้งซอฟต์แวร์ประยุกต์

จากภาพที่ 5.6 เป็นการรันซอฟต์แวร์ช่วยเหลือการติดตั้ง (Setup Help Software.jar) จากนั้น ซอฟต์แวร์จะทำการประเมินความมั่นคงปลอดภัยของระบบ ถ้าผลการประเมินฮาร์ดเดนนิ่งทั้ง 4 ข้อที่ระบุไว้ในแบบรูปความมั่นคงปลอดภัยอยู่ในสถานะแอคทีฟทั้งหมด ก็จะเข้าสู่การติดตั้งซอฟต์แวร์ประยุกต์ต่อไปโดยอัตโนมัติ แต่หากมีบางตัวอยู่ในสถานะไม่แอคทีฟ ก็เป็นดุลพินิจของผู้ทำการติดตั้งใช้งานว่าจะทำอย่างไร โดยจะยกเลิกการติดตั้งซอฟต์แวร์ประยุกต์แล้วกลับไปทำฮาร์ดเดนนิ่งก่อน หรือจะติดตั้งซอฟต์แวร์ประยุกต์ต่อไป โดยไม่สนใจว่า ระบบมีความเสี่ยงในเรื่องความมั่นคงปลอดภัยหรือไม่

5.2. กรณีศึกษา 1 ซอฟต์แวร์ประยุกต์ GoMapGen

5.2.1. ลักษณะซอฟต์แวร์ประยุกต์ GoMapGen

ซอฟต์แวร์ประยุกต์ที่ใช้ในกรณีศึกษาที่ 1 คือ GoMapGen ซึ่งถูกพัฒนาโดย Fabio Ferronato [23] โดยเป็นซอฟต์แวร์ประยุกต์ที่มีลักษณะการทำงานเป็นตัวสร้างแผนที่กูเกิล (Google Map Generator) ดังภาพที่ 5.7



ภาพที่ 5.7 ซอฟต์แวร์ประยุกต์ GoMapGen

5.2.2. สภาพแวดล้อมการประยุกต์ใช้โมดูลการประเมินฮาร์ดเดนนิ่งของระบบกับซอฟต์แวร์ประยุกต์ GoMapGen

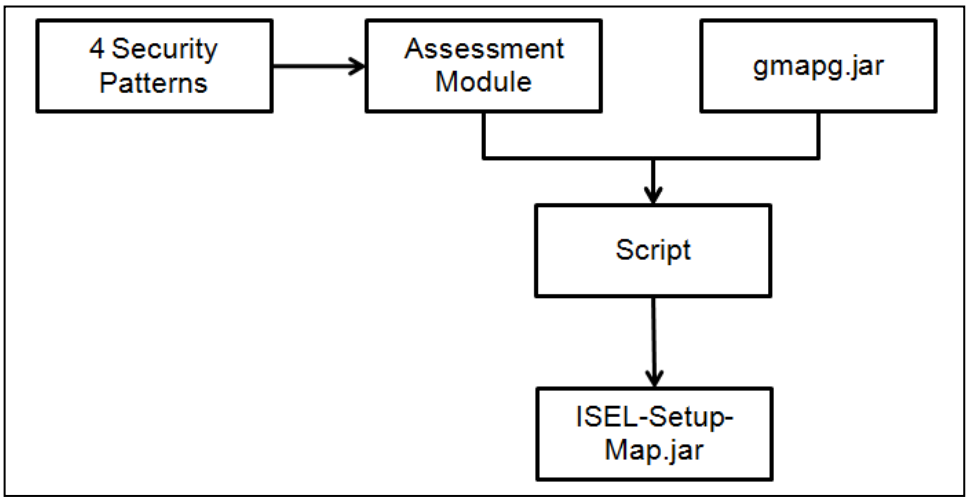
กำหนดสภาพแวดล้อมที่ใช้ในการประยุกต์ใช้โมดูลการประเมินฮาร์ดเดนนิ่งของระบบกับซอฟต์แวร์ประยุกต์ GoMapGen ได้ ดังตารางที่ 5.1

ตารางที่ 5.1 สภาพแวดล้อมการประยุกต์ใช้กับซอฟต์แวร์ประยุกต์ GoMapGen

ฮาร์ดแวร์	• ซีพียู 1.5 GHz หรือดีกว่า • หน่วยความจำ 1 GB หรือดีกว่า • พื้นที่ดิสก์ว่าง 7.5 GB หรือมากกว่า
ซอฟต์แวร์	• VMWare Player • Red Hat Enterprise Linux 6.1 • Java runtime 1.6 updated 30
กรณีศึกษา	• GoMapGen – Google Map Generator

5.2.3. การพัฒนาซอฟต์แวร์ประยุกต์ GoMapGen

จากแบบรูปความมั่นคงปลอดภัยทั้ง 4 แบบรูป พัฒนาเป็นโมดูลการประเมินฮาร์ดเดนนิ่งของระบบ จากนั้นเขียนสคริปต์ในการรวมโมดูลการประเมินฮาร์ดเดนนิ่งของระบบเข้ากับซอฟต์แวร์ประยุกต์ gmapg.jar เพื่อสร้างเป็นซอฟต์แวร์ช่วยเหลือการติดตั้ง ISEL-Setup-Map.jar ดังภาพที่ 5.8

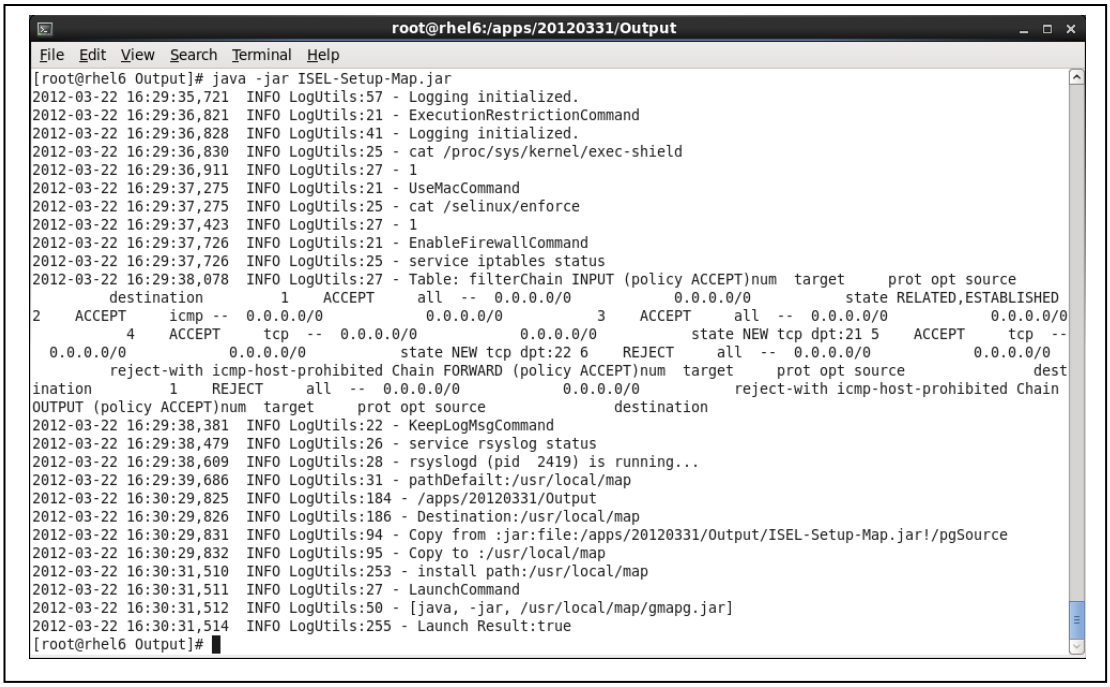


ภาพที่ 5.8 ขั้นตอนการสร้างซอฟต์แวร์ช่วยเหลือการติดตั้ง ISEL-Setup-Map.jar

5.2.4. การติดตั้งซอฟต์แวร์ประยุกต์ GoMapGen

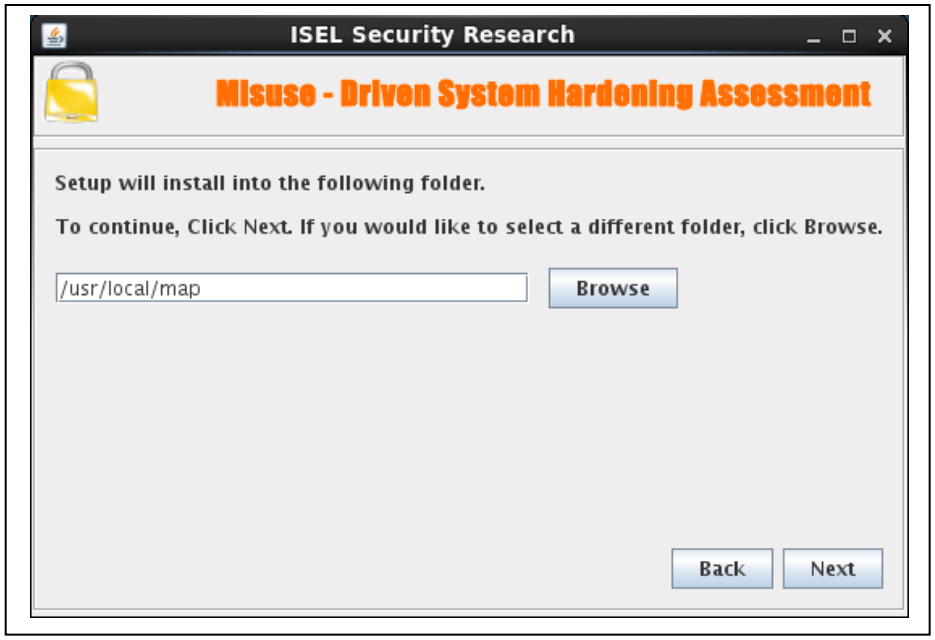
นำซอฟต์แวร์ช่วยเหลือการติดตั้ง ISEL-Setup-Map.jar มาทำการติดตั้งลงบนลีนุกซ์เพื่อใช้งาน ดังภาพที่ 5.9 โดยจะเข้าทำการประเมินความมั่นคงปลอดภัยของระบบ ดังภาพที่

5.10 แล้วจึงทำการเลือก path ที่ต้องการติดตั้ง ดังภาพที่ 5.11 จากนั้นจึงทำการติดตั้งซอฟต์แวร์ประยุกต์ GoMapGen ตาม path ที่ระบุ ดังภาพที่ 5.12 สุดท้ายเป็นการเสร็จสิ้นการติดตั้งจะได้ซอฟต์แวร์ประยุกต์ GoMapGen – Google Map Generator ดังภาพที่ 5.13

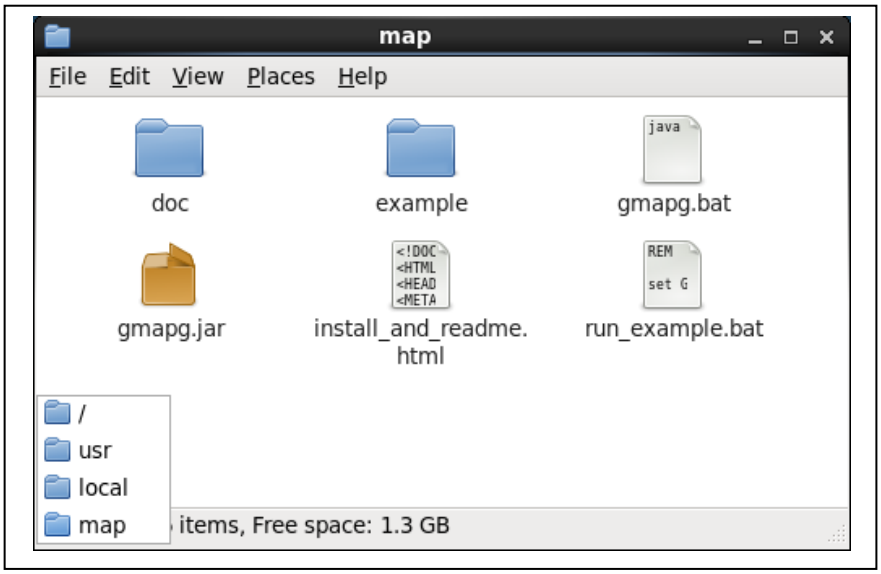


ภาพที่ 5.9 การรันซอฟต์แวร์ช่วยเหลือการติดตั้ง ISEL-Setup-Map.jar

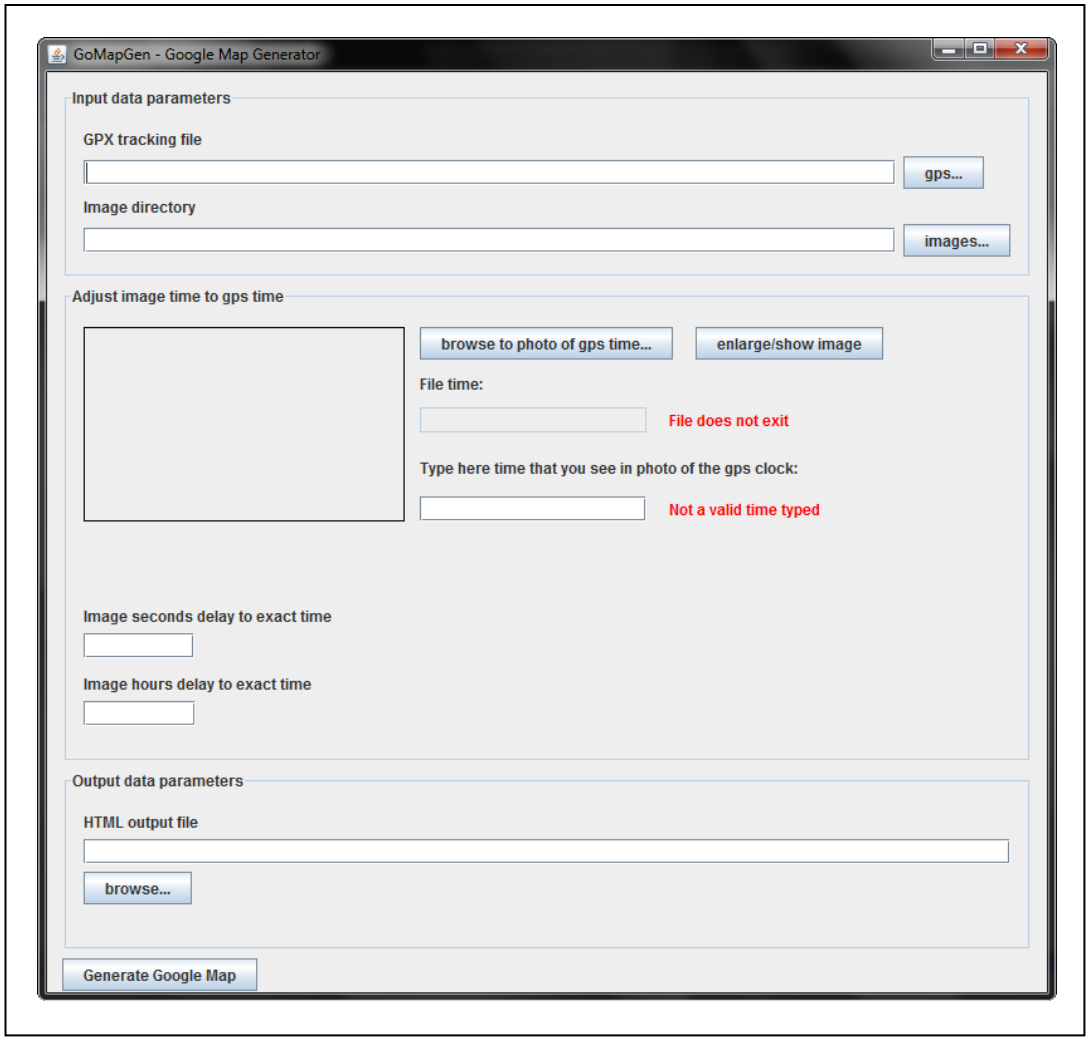
ภาพที่ 5.10 ผลลัพธ์ที่ได้จากการประเมินความมั่นคงปลอดภัยของระบบ



ภาพที่ 5.11 การเลือก path ที่ต้องการติดตั้งซอฟต์แวร์ประยุกต์ GoMapGen



ภาพที่ 5.12 การติดตั้งซอฟต์แวร์ประยุกต์ GoMapGen ตาม path ที่ระบุไว้

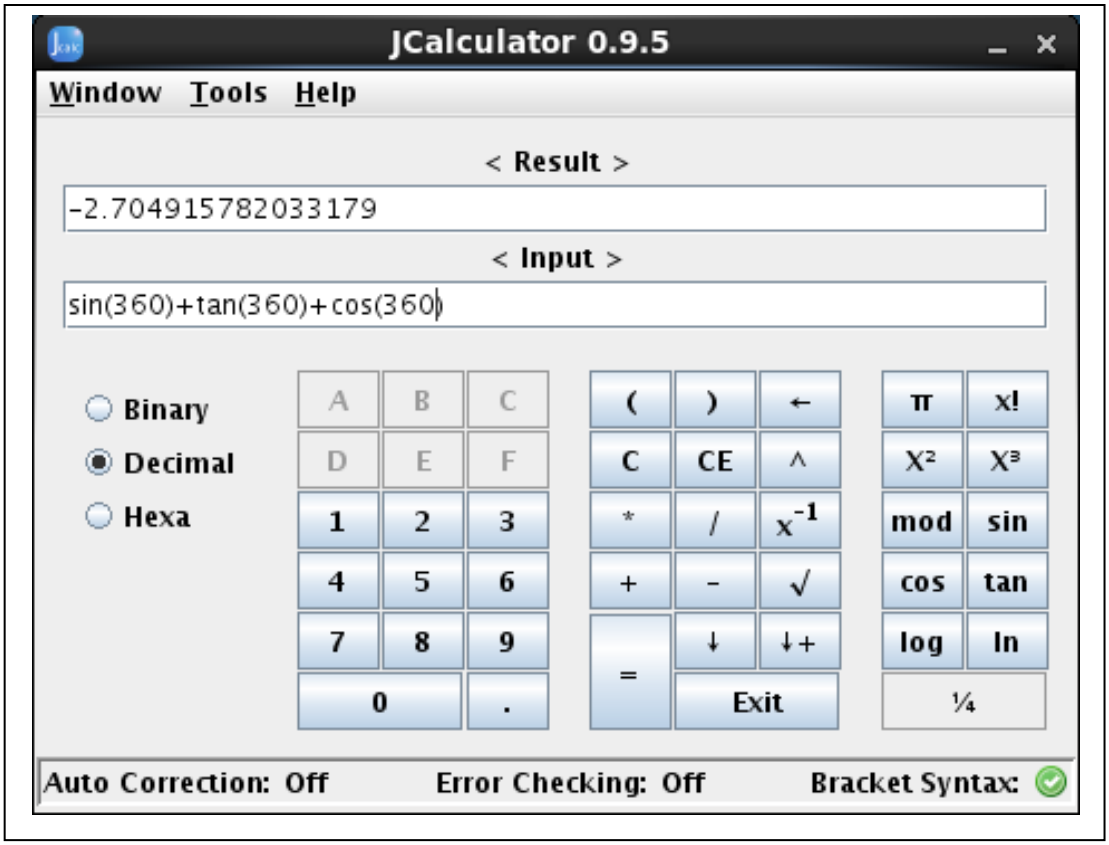


ภาพที่ 5.13 การใช้งานซอฟต์แวร์ประยุกต์ GoMapGen – Google Map Generator

5.3. กรณีศึกษา 2 ซอฟต์แวร์ประยุกต์ JCalculator

5.3.1. ลักษณะซอฟต์แวร์ประยุกต์ JCalculator

ซอฟต์แวร์ประยุกต์ที่ใช้ในกรณีศึกษาที่ 2 คือ JCalculator ซึ่งถูกพัฒนาโดย Adrian B. [24] โดยเป็นซอฟต์แวร์ประยุกต์ที่มีลักษณะการทำงานเป็นเครื่องคิดเลข ซึ่งสามารถคำนวณได้ทั้งเลขฐานสอง เลขฐานสิบ และเลขฐานสิบหก ดังภาพที่ 5.14



ภาพที่ 5.14 ซอฟต์แวร์ประยุกต์ JCalculator 0.9.5

5.3.2. สภาพแวดล้อมการประยุกต์ใช้โมดูลการประเมินฮาร์ดเดนนิ่งของระบบกับซอฟต์แวร์ประยุกต์ JCalculator

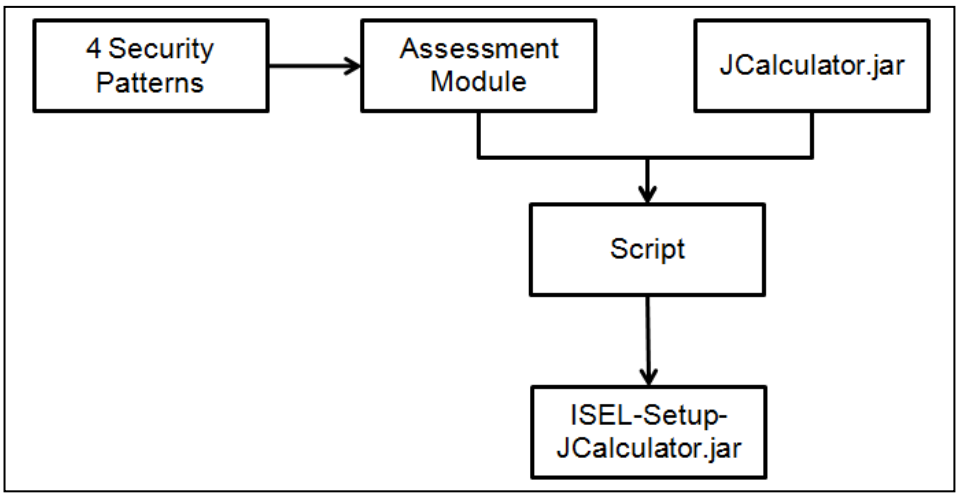
กำหนดสภาพแวดล้อมที่ใช้ในการประยุกต์ใช้โมดูลการประเมินฮาร์ดเดนนิ่งของระบบกับซอฟต์แวร์ประยุกต์ JCalculator ได้ ดังตารางที่ 5.2

ตารางที่ 5.2 สภาพแวดล้อมการประยุกต์ใช้กับซอฟต์แวร์ประยุกต์ JCalculator

ฮาร์ดแวร์	• ซีพียู 1.5 GHz หรือดีกว่า • หน่วยความจำ 1 GB หรือดีกว่า • พื้นที่ดิสก์ว่าง 7.5 GB หรือมากกว่า
ซอฟต์แวร์	• VMWare Player • Red Hat Enterprise Linux 6.1 • Java runtime 1.6 updated 30
กรณีศึกษา	• JCalculator 0.9.5

5.3.3. การพัฒนาซอฟต์แวร์ประยุกต์ JCalculator

จากแบบรูปความมั่นคงปลอดภัยทั้ง 4 แบบรูป พัฒนาเป็นโมดูลการประเมินฮาร์ดเดนนิ่งของระบบ จากนั้นเขียนสคริปต์ในการรวมโมดูลการประเมินฮาร์ดเดนนิ่งของระบบเข้ากับซอฟต์แวร์ประยุกต์ JCalculator.jar เพื่อสร้างเป็นซอฟต์แวร์ช่วยเหลือการติดตั้ง ISEL-Setup-JCalculator.jar ดังภาพที่ 5.15



ภาพที่ 5.15 ขั้นตอนการสร้างซอฟต์แวร์ช่วยเหลือการติดตั้ง ISEL-Setup-JCalculator.jar

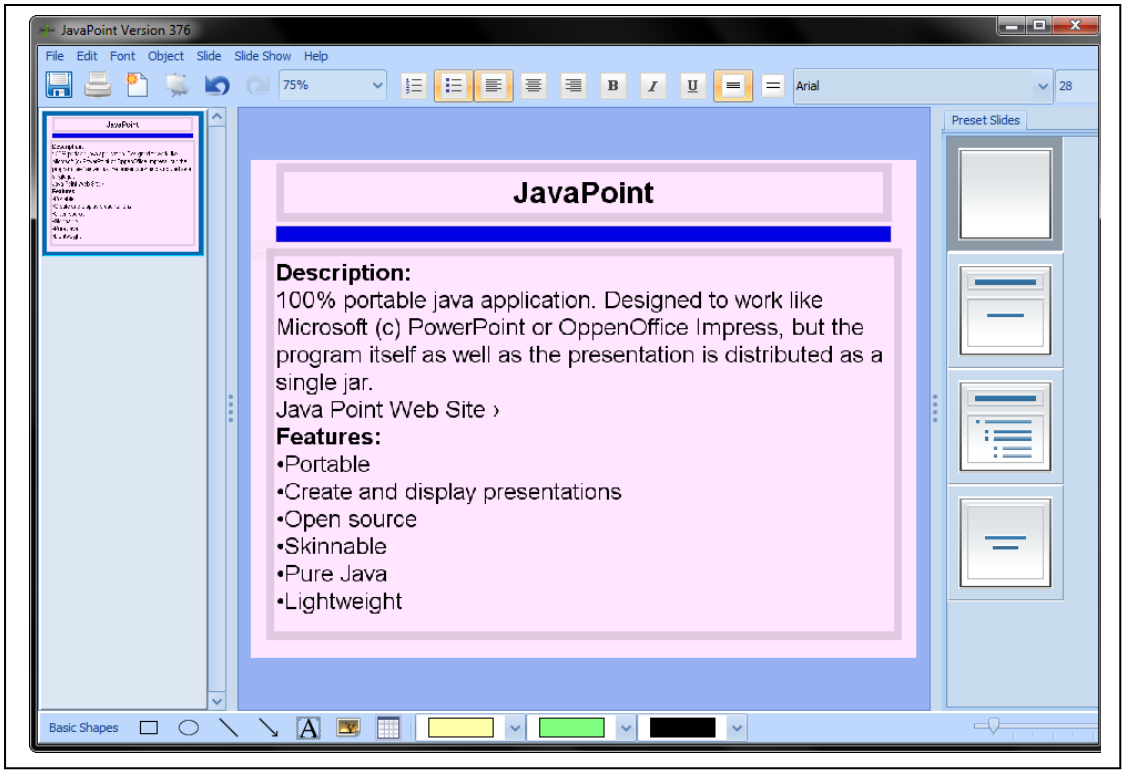
5.3.4. การติดตั้งซอฟต์แวร์ประยุกต์ JCalculator

นำซอฟต์แวร์ช่วยเหลือการติดตั้ง ISEL-Setup-JCalculator.jar มาทำการติดตั้งลงบนลีนุกซ์เพื่อใช้งาน เช่นเดียวกับการติดตั้งซอฟต์แวร์ประยุกต์ GoMapGen ในขั้นตอน 5.2.4

5.4. กรณีศึกษา 3 ซอฟต์แวร์ประยุกต์ JavaPoint

5.4.1. ลักษณะซอฟต์แวร์ประยุกต์ JavaPoint

ซอฟต์แวร์ประยุกต์ที่ใช้ในกรณีศึกษาที่ 3 คือ JavaPoint ซึ่งถูกพัฒนาโดย Kyle Flanigan [25] โดยเป็นซอฟต์แวร์ประยุกต์ที่มีลักษณะการทำงานคล้าย Microsoft Office PowerPoint ดังภาพที่ 5.16



ภาพที่ 5.16 ซอฟต์แวร์ประยุกต์ JavaPoint Version 376

5.4.2. สภาพแวดล้อมการประยุกต์ใช้โมดูลการประเมินฮาร์ดเดนนิ่งของระบบกับซอฟต์แวร์ประยุกต์ JavaPoint

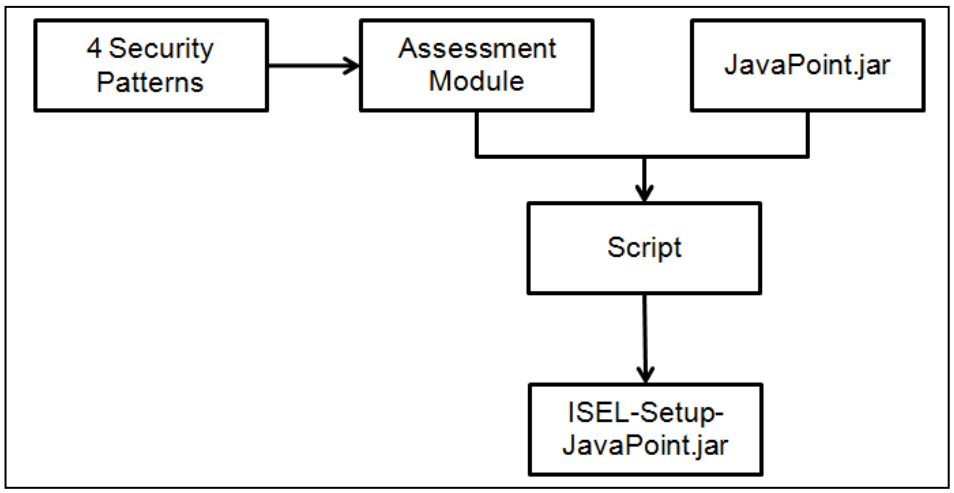
กำหนดสภาพแวดล้อมที่ใช้ในการประยุกต์ใช้โมดูลการประเมินฮาร์ดเดนนิ่งของระบบกับซอฟต์แวร์ประยุกต์ JavaPoint ได้ ดังตารางที่ 5.3

ตารางที่ 5.3 สภาพแวดล้อมการประยุกต์ใช้กับซอฟต์แวร์ประยุกต์ JavaPoint

ฮาร์ดแวร์	• ซีพียู 1.5 GHz หรือดีกว่า • หน่วยความจำ 1 GB หรือดีกว่า • พื้นที่ดิสก์ว่าง 7.5 GB หรือมากกว่า
ซอฟต์แวร์	• VMWare Player • Red Hat Enterprise Linux 6.1 • Java runtime 1.6 updated 30
กรณีศึกษา	• JavaPoint Version 376

5.4.3. การพัฒนาซอฟต์แวร์ประยุกต์ JavaPoint

จากแบบรูปความมั่นคงปลอดภัยทั้ง 4 แบบรูป พัฒนาเป็นโมดูลการประเมินฮาร์ดเดนนิ่งของระบบ จากนั้นเขียนสคริปต์ในการรวมโมดูลการประเมินฮาร์ดเดนนิ่งของระบบเข้ากับซอฟต์แวร์ประยุกต์ JavaPoint.jar เพื่อสร้างเป็นซอฟต์แวร์ช่วยเหลือการติดตั้ง ISEL-Setup-JavaPoint.jar ดังภาพที่ 5.17



ภาพที่ 5.17 การสร้างซอฟต์แวร์ช่วยเหลือการติดตั้ง ISEL-Setup-JavaPoint.jar

5.4.4. การติดตั้งซอฟต์แวร์ประยุกต์ JavaPoint

นำซอฟต์แวร์ช่วยเหลือการติดตั้ง ISEL-Setup- JavaPoint.jar มาทำการติดตั้งลงบนลีนุกซ์เพื่อใช้งาน เช่นเดียวกับการติดตั้งซอฟต์แวร์ประยุกต์ GoMapGen ในขั้นตอน 5.2.4

ที่กล่าวมาในบทที่ 5 นี้ เป็นการกล่าวถึงการประยุกต์ใช้แบบรูปความมั่นคงปลอดภัยที่ใช้เป็นตัวอย่างในการนำไปใช้งานจริง โดยนำเสนอการประยุกต์ใช้ในขั้นตอนต่างๆ ของกระบวนการพัฒนาซอฟต์แวร์ ซึ่งแสดงกิจกรรมที่เกิดขึ้นในแต่ละขั้นตอนของกระบวนการพัฒนาซอฟต์แวร์ทั้งสี่ขั้นตอน ได้แก่ ขั้นตอนความต้องการ ขั้นตอนการออกแบบ ขั้นตอนการพัฒนา และขั้นตอนการติดตั้ง พร้อมทั้งการประยุกต์ใช้งานจริงกับกรณีศึกษาทั้งสามกรณี ได้แก่ ซอฟต์แวร์ประยุกต์ GoMapGen ซอฟต์แวร์ประยุกต์ JCalculator และซอฟต์แวร์ประยุกต์ JavaPoint ส่วนในบทถัดไป จะเป็นการทดสอบและอภิปรายผลการวิจัย

บทที่ 6
การทดสอบและอภิปรายผลการวิจัย

เนื้อหาในบทนี้ กล่าวถึง การทดสอบและอภิปรายผลการวิจัย โดยสามารถแบ่งออกเป็น 2 ส่วน ดังนี้

1) การทดสอบการทำงานของโมดูลการประเมินฮาร์ดเดนนิ่งของระบบ
2) อภิปรายผลการวิจัย

ซึ่งในแต่ละส่วน มีรายละเอียดของการทดสอบและอภิปรายผลการวิจัย ดังต่อไปนี้

6.1. การทดสอบการทำงานของโมดูลการประเมินฮาร์ดเดนนิ่งของระบบ

การประเมินความมั่นคงปลอดภัยของระบบในขั้นตอนการติดตั้งสามารถกำหนดเงื่อนไขการประเมินฮาร์ดเดนนิ่งของระบบ โดยใช้คุณสมบัติ Exec Shield SELinux iptables และ rsyslog ในการทำการทดสอบได้ดังตารางที่ 6.1

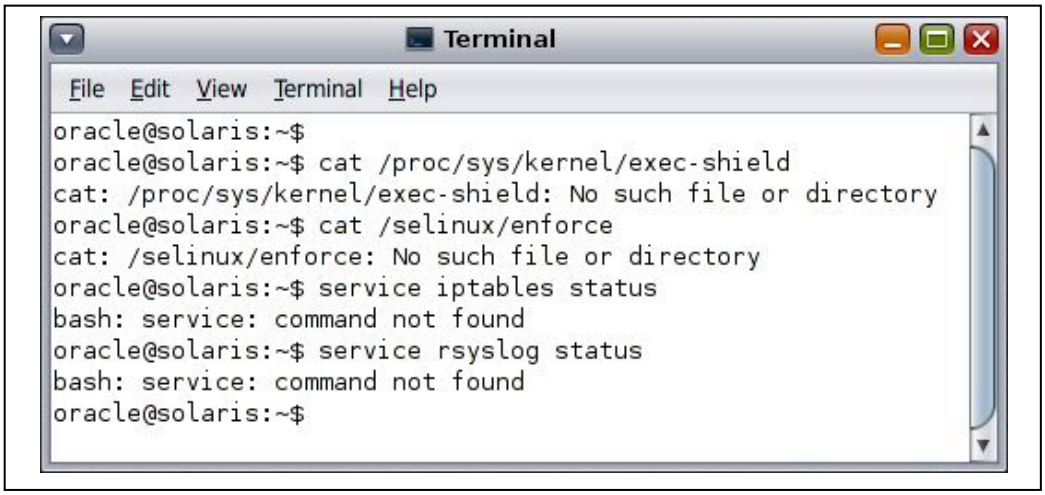
ตารางที่ 6.1 การกำหนดเงื่อนไขการทดสอบฮาร์ดเดนนิ่งของระบบ

กรณี	การติดตั้ง				การเปิดการใช้งาน			
	Exec Shield	SELinux	iptables	rsyslog	Exec Shield	SELinux	iptables	rsyslog
1	X	X	X	X	X	X	X	X
2	/	/	/	/	X	X	X	X
3	/	/	/	/	X	/	/	/
4	/	/	/	/	/	X	/	/
5	/	/	/	/	/	/	X	/
6	/	/	/	/	/	/	/	X
7	/	/	/	/	/	/	/	/

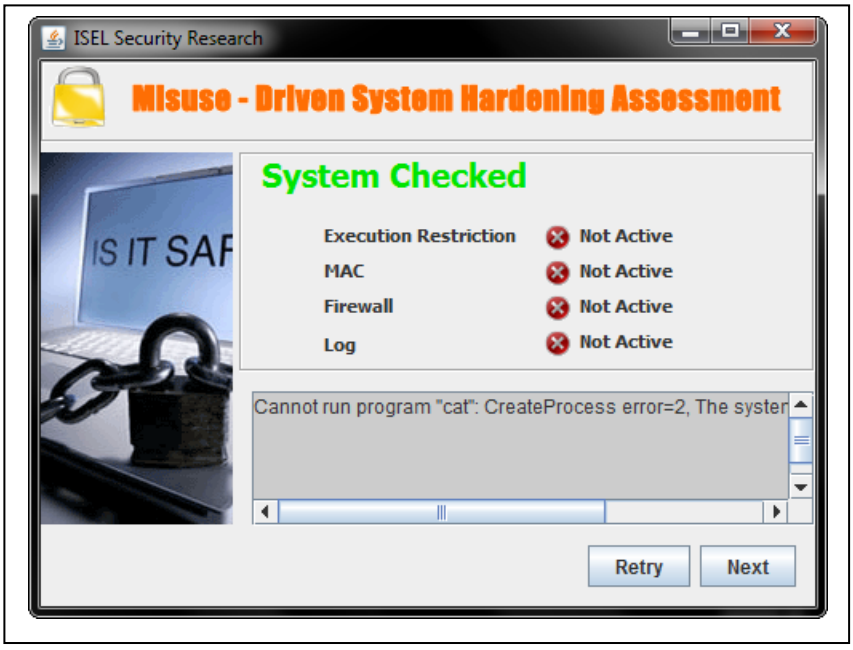
โดย / หมายถึง มีการติดตั้ง หรือการเปิดการใช้งานฮาร์ดเดนนิ่งในระบบ และ X หมายถึง ไม่มีการติดตั้ง หรือไม่มีการเปิดใช้งานฮาร์ดเดนนิ่งในระบบ

จากเงื่อนไขฮาร์ดเดนนิ่งของระบบในตารางที่ 6.1 สามารถสร้างกลุ่มการทดสอบได้ 3 รูปแบบ ดังนี้

1) รูปแบบที่ 1 - กรณีที่ 1 คือ ระบบไม่มีการติดตั้ง Exec Shield SELinux iptables และ rsyslog ซึ่งแสดงสภาพแวดล้อมจริงของระบบ ดังภาพที่ 6.1 และแสดงผลการประเมินเป็น Not Active ทุกตัว ดังภาพที่ 6.2

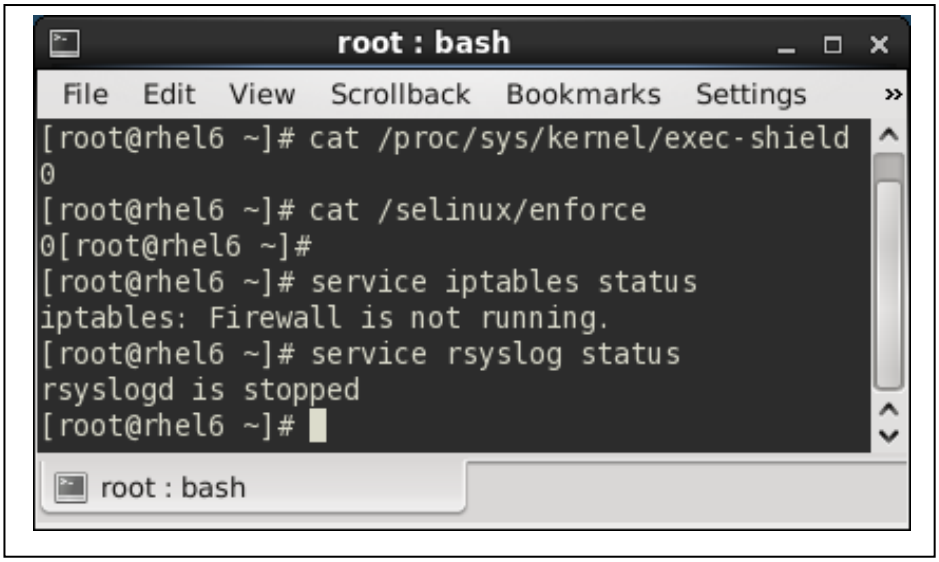


ภาพที่ 6.1 สภาพแวดล้อมจริงของระบบที่ไม่มีการติดตั้ง Exec Shield SELinux iptables และ rsyslog



ภาพที่ 6.2 ผลการประเมินระบบที่ไม่มีการติดตั้ง Exec Shield SELinux iptables และ rsyslog

2) รูปแบบที่ 2 - กรณีที่ 2 คือ ระบบมีการติดตั้ง Exec Shield SELinux iptables และ rsyslog แต่ไม่เปิดใช้งาน ซึ่งแสดงสภาพแวดล้อมจริงของระบบ ดังภาพที่ 6.3 และแสดงผลการประเมินเป็น Not Active ทุกตัว ดังภาพที่ 6.4



ภาพที่ 6.3 สภาพแวดล้อมจริงของระบบที่มีการติดตั้ง Exec Shield SELinux iptables และ rsyslog แต่ไม่เปิดใช้งาน

ภาพที่ 6.4 ผลการประเมินระบบที่มีการติดตั้ง Exec Shield SELinux iptables และ rsyslog แต่ไม่เปิดใช้งาน

สำหรับระบบใดๆ ที่มีการติดตั้ง Exec Shield SELinux iptables และ rsyslog แต่ไม่เปิดใช้งาน ผลการประเมินความมั่นคงปลอดภัยของระบบ จะแสดงค่าเป็น Not Active ในแต่ละตัวที่ไม่เปิดการใช้งาน และแสดงค่าเป็น Active ในแต่ละตัวที่เปิดการใช้งาน และในกรณีที่ผลการประเมินความมั่นคงปลอดภัยเป็น Not Active โมดูลการประเมินฮาร์ดเดนนิ่งของระบบจะระบุถึงปัญหา และวิธีการแก้ไขปัญหาในแต่ละฮาร์ดเดนนิ่ง ดังต่อไปนี้

กรณีที่ 3 คือ ระบบมีการติดตั้ง Exec Shield แต่ไม่เปิดใช้งาน ดังภาพที่ 6.5

ภาพที่ 6.5 ผลการประเมินระบบที่มีการติดตั้ง Exec Shield แต่ไม่เปิดใช้งาน

กรณีที่ 4 คือ ระบบมีการติดตั้ง SELinux แต่ไม่เปิดใช้งาน ดังภาพที่ 6.6

ภาพที่ 6.6 ผลการประเมินระบบที่มีการติดตั้ง SELinux แต่ไม่เปิดใช้งาน

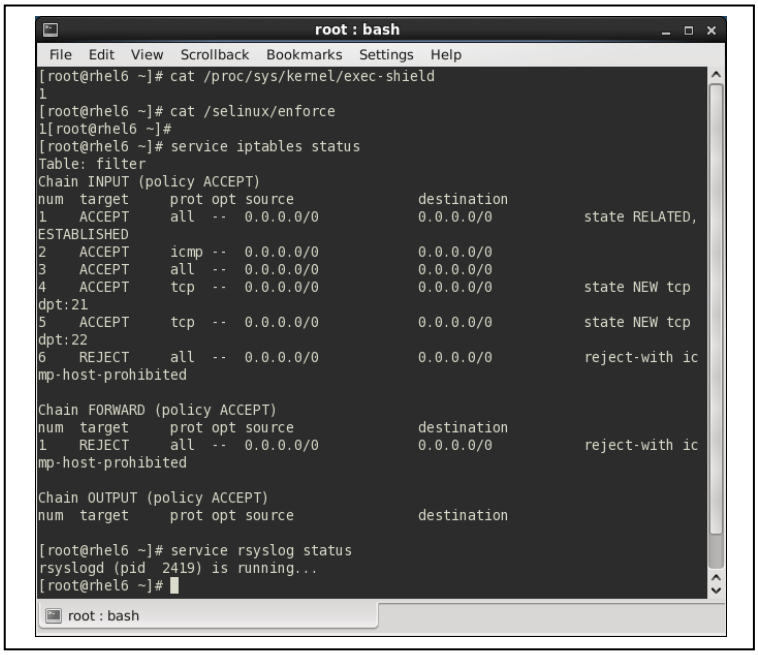
กรณีที่ 5 คือ ระบบมีการติดตั้ง iptables แต่ไม่เปิดใช้งาน ดังภาพที่ 6.7

ภาพที่ 6.7 ผลการประเมินระบบที่มีการติดตั้ง iptables แต่ไม่เปิดใช้งาน

กรณีที่ 6 คือ ระบบมีการติดตั้ง rsyslog แต่ไม่เปิดใช้งาน ดังภาพที่ 6.8

ภาพที่ 6.8 ผลการประเมินระบบที่มีการติดตั้ง rsyslog แต่ไม่เปิดใช้งาน

3) รูปแบบที่ 3 - กรณีที่ 7 คือ ระบบมีการติดตั้งและเปิดใช้งาน Exec Shield SELinux iptables และ rsyslog ซึ่งแสดงสภาพแวดล้อมจริงของระบบ ดังภาพที่ 6.9 และแสดงผลการประเมินเป็น Active ทุกตัว ดังภาพที่ 6.10



ภาพที่ 6.9 สภาพแวดล้อมจริงของระบบที่มีการติดตั้งและเปิดใช้งาน Exec Shield SELinux iptables และ rsyslog

ภาพที่ 6.10 ผลการประเมินระบบที่มีการติดตั้งและเปิดใช้งาน Exec Shield SELinux iptables และ rsyslog

6.2. อภิปรายผลการวิจัย

ในงานวิจัยนี้ได้ทำการทดสอบโมดูลการประเมินฮาร์ดเดนนิ่งของระบบที่พัฒนามาจากแบบรูปความมั่นคงปลอดภัย โดยใช้ Exec Shield SELinux iptables และ rsyslog โดยทำการเปรียบเทียบผลการประเมินที่ได้จากโมดูลการประเมินฮาร์ดเดนนิ่งของระบบกับสภาพแวดล้อมจริงในการติดตั้งและเปิดใช้งาน Exec Shield SELinux iptables และ rsyslog ของระบบ ว่าได้ผลตรงกันหรือไม่ ซึ่งพิจารณาเงื่อนไขการทดสอบ 7 กรณี และสามารถสรุปผลการทดสอบเป็น 3 รูปแบบ ได้ดังนี้

1) กรณีที่ระบบไม่มีการติดตั้ง Exec Shield SELinux iptables และ rsyslog ผลการประเมินความมั่นคงปลอดภัยของระบบ จะแสดงค่าเป็น Not Active

2) กรณีที่ระบบมีการติดตั้ง Exec Shield SELinux iptables และ rsyslog แต่ไม่เปิดใช้งาน ผลการประเมินความมั่นคงปลอดภัยของระบบ จะแสดงค่าเป็น Not Active

3) กรณีที่ระบบมีการติดตั้งและเปิดใช้งาน Exec Shield SELinux iptables และ rsyslog ผลการประเมินความมั่นคงปลอดภัยของระบบ จะแสดงค่าเป็น Active

ในการทำการทดสอบได้ทดสอบบนลีนุกซ์ ประเภท Redhat ซึ่งมีคุณสมบัติของฮาร์ดเดนนิ่งครบทั้ง 4 ประเภท ได้แก่ Exec Shield SELinux iptables และ rsyslog แต่ในลีนุกซ์

ประเภทอื่น เช่น UBuntu CentOS และ Solaris เป็นต้น อาจจะต้องยุ่งยากในการศึกษาค้นคว้าว่าลีนุกซ์ประเภทนั้นๆ สามารถติดตั้งและเปิดใช้งานฮาร์ดเดนนิ่ง Exec Shield SELinux iptables และ rsyslog ได้หรือไม่ เพราะในลีนุกซ์บางประเภทอาจจะไม่มีคุณสมบัติของฮาร์ดเดนนิ่งบางตัวได้

ที่กล่าวมาในบทที่ 6 นี้ เป็นการกล่าวถึงการทดสอบและอภิปรายผลการวิจัย ที่ใช้โมดูลการประเมินฮาร์ดเดนนิ่งของระบบ โดยนำเสนอการทดสอบคุณสมบัติฮาร์ดเดนนิ่งทั้ง 4 ประเภท ได้แก่ Exec Shield SELinux iptables และ rsyslog ซึ่งได้ทดสอบบนลีนุกซ์ประเภท RedHat และเปรียบเทียบผลการประเมินกับสภาพแวดล้อมจริงของการติดตั้งและเปิดใช้งาน Exec Shield SELinux iptables และ rsyslog ของระบบ ส่วนในบทถัดไป จะเป็นการสรุปผลการวิจัย

บทที่ 7
การสรุปผลการวิจัย

เนื้อหาในบทนี้ กล่าวถึง การสรุปผลการวิจัย โดยสามารถแบ่งออกเป็น 3 ส่วน ดังนี้

1) สรุปผลที่ได้รับจากงานวิจัย
2) ข้อจำกัดของงานวิจัย
3) แนวทางการทำวิจัยในอนาคต

ซึ่งในแต่ละส่วน มีรายละเอียดของการสรุปผลการวิจัย ดังต่อไปนี้

7.1. สรุปผลที่ได้รับจากงานวิจัย

งานวิจัยนี้นำเสนอระเบียบวิธีในการประยุกต์ใช้แผนภาพมิสยูสเคสในการประเมินฮาร์ดเดนนิ่งของระบบ สำหรับนักออกแบบซอฟต์แวร์นำไปใช้แสดงความต้องการด้านฮาร์ดเดนนิ่งของระบบ ซึ่งเป็นความต้องการที่ไม่ใช่เชิงหน้าที่ ให้เป็นความต้องการเชิงหน้าที่ กล่าวคือ การใช้มิสยูสแสดงการโจมตี การนำเสนอวิธีการป้องกัน และการประเมินช่วยในการแก้ไขปัญหา โดยได้วิเคราะห์ฮาร์ดเดนนิ่งในสี่กรณีเป็นตัวอย่าง ได้แก่ การแยกหน่วยความจำชุดคำสั่งและหน่วยความจำข้อมูล การกำหนดสิทธิ์ของผู้ใช้ การตั้งค่าไฟร์วอลล์ และการเก็บบันทึกการใช้งานระบบ มิสยูสจะถูกพัฒนาเป็นการประเมินระบบที่จะนำซอฟต์แวร์ประยุกต์นั้นไปติดตั้ง และเป็นส่วนหนึ่งของโมดูลการติดตั้งของซอฟต์แวร์ นอกจากนี้ได้นำแบบรูปความมั่นคงปลอดภัยมาใช้ลดความซับซ้อนในการรวมการประเมินฮาร์ดเดนนิ่งเข้าสู่การออกแบบ การแสดงการประเมินฮาร์ดเดนนิ่งเป็นความต้องการเชิงหน้าที่สำหรับช่วยให้การประเมินเกิดขึ้นโดยอัตโนมัติในขณะติดตั้งซอฟต์แวร์ประยุกต์ โดยไม่ต้องอาศัยความเชี่ยวชาญของผู้ดูแลระบบ

จากตัวอย่างได้พัฒนาเป็นสี่แบบรูปความมั่นคงปลอดภัย แบบรูปแรก เป็นการแยกหน่วยความจำชุดคำสั่งและหน่วยความจำข้อมูล โดยใช้ Exec Shield เป็นพื้นฐานในการประเมิน แบบรูปที่สอง เป็นการตรวจสอบการใช้ SELinux ในการบังคับใช้ MAC และการกำหนดสิทธิ์ของผู้ใช้ แบบรูปที่สาม เป็นการตรวจสอบการติดตั้งและใช้งานไฟร์วอลล์ โดยใช้ iptables และแบบรูปสุดท้าย เป็นการตรวจสอบบันทึกการเข้าใช้งานระบบ โดยใช้ rsyslog ในการตรวจสอบการบุกรุก

แม้ว่าคำสั่งที่ใช้ทำการประเมินฮาร์ดเดนนิ่งของระบบมีเพียงบรรทัดเดียว แต่ก็แสดงให้เห็นได้ว่า หากไม่มีการประเมินความมั่นคงปลอดภัยแบบอัตโนมัติที่เกิดจากระเบียบวิธีนี้ คำสั่งบรรทัดเดียวนี้จะต้องกำหนดและกระทำโดยผู้ดูแลระบบขณะทำการติดตั้งซอฟต์แวร์ประยุกต์ และตัวอย่างฮาร์ดเดนนิ่งทั้งสี่ตัวอย่างที่เลือกมานี้ตั้งใจเลือกเฉพาะที่สามารถทำการประเมินได้ง่าย ในความเป็นจริงฮาร์ดเดนนิ่งอื่นๆ อาจต้องใช้วิธีการประเมินที่ซับซ้อนกว่านี้ ดังนั้นระเบียบวิธีนี้จึงเกิดประโยชน์ได้มาก ทำให้ลดการพึ่งพาผู้ดูแลระบบ

7.2. ข้อจำกัดของงานวิจัย

1) งานวิจัยนี้ มุ่งเน้นฮาร์ดเดนนิ่ง ซึ่งเป็นความต้องการเกี่ยวกับเรื่องทั่วไปของระบบ ไม่ได้เป็นความต้องการเฉพาะเจาะจงของซอฟต์แวร์ประยุกต์

2) งานวิจัยนี้ อาจทำให้การพัฒนาซอฟต์แวร์มีความซับซ้อนมากยิ่งขึ้น และจำเป็นต้องทำการอบรมให้นักพัฒนาซอฟต์แวร์มีความรู้ในเรื่องฮาร์ดเดนนิ่งมากยิ่งขึ้น

7.3. แนวทางการทำวิจัยในอนาคต

1) การใช้มิสยูสสำหรับแสดงการโจมตี การป้องกัน และการแก้ไขปัญหา ที่ใช้ในการประเมินอย่างอัตโนมัติ เพื่อรวมเป็นความต้องการและการออกแบบซอฟต์แวร์ ในแต่ละการประเมินขึ้นอยู่กับประเภทของฮาร์ดเดนนิ่งและระบบมากกว่าลักษณะของซอฟต์แวร์ ดังนั้นรูปแบบของฮาร์ดเดนนิ่งสามารถถูกสำรวจและสร้างเป็นไลบรารีของแบบรูปความมั่นคงปลอดภัยเพื่อใช้รองรับการรวมการประเมินเข้าสู่การออกแบบ การพัฒนา และการติดตั้งซอฟต์แวร์อย่างปลอดภัย ไลบรารี่นี้สามารถเพิ่มขยายปรับเปลี่ยนตามเทคโนโลยีของการฮาร์ดเดนนิ่งได้

2) จากตัวอย่างฮาร์ดเดนนิ่งทั้งสี่กรณีที่ถูกนำเสนอในงานวิจัยนี้ เป็นเพียงส่วนหนึ่งของมุมมองด้านความมั่นคงปลอดภัย ซึ่งสามารถขยายการใช้งานมิสยูสในด้านอื่นๆ ได้ เช่น การตรวจสอบอินพุท และสามารถวิจัยวิธีการอื่นๆ นอกเหนือไปจากการประเมินระบบ ทั้งนี้วิธีการใหม่เหล่านี้อาจนำไปสู่แบบรูปใหม่ๆ ได้

3) จากแบบรูปความมั่นคงปลอดภัยทั้งสี่แบบรูปสามารถหาแนวทางในการป้องกันมิสยูสให้มีความซับซ้อนและยืดหยุ่นมากกว่านี้ได้ เช่น การใช้คุณสมบัติของ PAX และ NXBit แทนคุณสมบัติของ Exec Shield การใช้คุณสมบัติของ Mandatory Integrity Control แทนคุณสมบัติของ SELinux การใช้คุณสมบัติของ ipchains แทนคุณสมบัติของ iptables และการใช้คุณสมบัติของ syslog-ng แทนคุณสมบัติของ rsyslog เป็นต้น รวมถึงการเสริมวิธีการใหม่เมื่อมีแนวทางการโจมตีและการป้องกันใหม่ๆ ถูกพัฒนาขึ้น

รายการอ้างอิง

[1] J.D. Meier, Alex Mackman, Michael Dunner, Srinath Vasireddy, Ray Escamilla, and Anandha Murukan. Improving Web Application Security: Threats and Countermeasures. [Online]. 2003. Available from : http://msdn.microsoft.com/en-us/library/ff648644.aspx [2011, December 1]

[2] Charles P. Pfleege. Security in Computing. 4th. NJ: Pearson Education, 2007.

[3] Carlo Kopp. Hardening Your Computing Assets. Computer Magazine Group, 1997.

[4] Syngress. Hardening the Operation System. [Online]. 2007. Available from : http://www.syngress.com [2011, September 15]

[5] Ulrich Drepper. Security Enhancements in Red Hat Enterprise Linux (beside SELinux). Version 1.6. Red Hat Inc, 2005.

[6] Vivek Gite. 20 Linux Server Hardening Security Tips. [Online]. 2009. Available from : http://www.cyberciti.biz/tips/linux-security.html [2011, October 20]

[7] Alan Dearle. Software Deployment, Past, Present and Future. Future of Software Engineering (FOSE'07). pp. 269-284, 2007.

[8] Jon Whittle, Duminda Wijesekera, and Mark Hartong. Executable Misuse Cases for Modeling Security Concerns. 30th International Conference on Software Engineering (ICSE'08). pp. 121–130, 2008.

[9] Guttorm Sindre and Andreas L. Opdahl. Eliciting Security Requirements by Misuse Cases. 37th International Conference on Technology of Object-Oriented Languages and Systems. pp. 120–131, 2000.

[10] Nobukazu Yoshioka, Hironori Washizaki, and Katsuhisa Maruyama. A servey on security patterns. National Instiute of Informatics. pp. 35-47, 2008.

[11] Peter A. Loscocco and Stephen D. Smalley. Meeting Critical Security Objectives with Security-Enhanced Linux. Ottawa Linux Symposium, 2001.

[12] Arjan van de Ven. New Security Enhancements in Red Hat Enterprise. Version 3. Red Hat Inc. WHP0006US 8/04, 2004.

[13] Tammy Fox. Red Hat Enterprise Linux Administration Unleashed. Sams Indianapolis, 2007.

[14] Mika V. Mantyla and Jari Vanhanen. Software Deployment Activities and Challenges – A Case Study of Four Software Product Companies. 15th European Conference on Software Maintenance and Reengineering. pp. 131-139, 2011.

[15] Guttorm Sindre and Andreas L. Opdahl. Templates for Misuse Case Description. 7th International Workshop on Requirements Engineering Foundation for Software Quality (REFSQ'01), 2001.

[16] Guttorm Sindre and Andreas L. Opdahl. Eliciting security requirements with misuse cases. Requirements Engineering. Volumn 10. Number 1. pp. 34-44, 2005.

[17] R. Matulevicius, N. Mayer, and P. Heymans. Alignment of misuse cases with security risk management. 3th International Conference on Availability, Reliability and Security (ARES'08). pp. 1397–1404, 2008.

[18] Fabricio A. Braz, Eduardo B. Fernandez, and Michael VanHilst. Eliciting Security Requirements through Misuse Activities. 19th International Conference on Database and Expert Systems Applications. pp. 328-333, 2008.

[19] Okubo Takao, Taguchi Kenji, and Yoshioka Nobukazu. Misuse Cases + Assets + Security Goals. International Conference on Computational Science and Engineering. pp. 129-144, 2009.

[20] M. Schumacher, E. Fernandez-Buglioni, D. Hybertson, F. Buschmann, and P. Sommerlad. Security Patterns : Integrating Security and Systems Engineering (Wiley Software Patterns Series). John Wiley & Sons, 2006.

[21] Angel Cubvas and Paul E. Khoury. A Security Pattern for Untraceable Secret Handshakes. 3rd International Conference on Emerging Security

Information, Systems and Technologies (SECURWARE'09). pp. 8–14, 2009.

[22] David J. Schultz et al. IEEE Std 1074-1997, IEEE Standard for Developing Software Life Cycle Processes. The Institute of Electrical and Electronics Engineers Inc, 1997.

[23] Fabio Ferronato. GOogle MAP GEnerator. [Online]. 2011. Available from : http://sourceforge.net/projects/gomapgen/?source=directory [2012, March 1]

[24] Adrian B. Scientifical Java Calculator. [Online]. 2012. Available from : http://sourceforge.net/projects/jcalclator/?source=directory [2012, March 1]

[25] Kyle Flanigan. Java Point. [Online]. 2011. Available from : http://sourceforge.net/projects/javapoint/?source=directory [2012, March 1]

[26] OMG. Unified Modeling 2.3 Specification. [Online]. 2010. Available from : http://www.omg.org/spec/UML/2.3/ [2012, January 15]

[27] Alan Dennis, Barbara H. Wixom, and David Tegarde. Systems Analysis and Design with UML An Object-Oriented Approach. 3rd. NJ: Wiley, 2010.

ภาคผนวก

ภาคผนวก ก
คำอธิบายยูสเคสของระบบการประเมินฮาร์ดเดนนิ่งของระบบ

1) ฟังก์ชัน Maintain data memory integrity

ชื่อยูสเคส:	Maintain data memory integrity	รหัส: UDC-001	ระดับความสำคัญ: มาก
ผู้กระทำหลัก:	นักพัฒนาซอฟต์แวร์	ประเภทยูสเคส:	เชิงละเอียด
ผู้มีส่วนเกี่ยวข้องและการใช้ประโยชน์:	นักพัฒนาซอฟต์แวร์: ต้องการปกป้องหน่วยความจำ		
คำอธิบาย:	เป็นยูสเคสที่ใช้ในการคงไว้ซึ่งบูรณภาพข้อมูลของหน่วยความจำข้อมูล		
สิ่งกระตุ้น:	ระบบต้องคงไว้ซึ่งบูรณภาพข้อมูลของหน่วยความจำข้อมูลอยู่เสมอ		
ประเภทของสิ่งกระตุ้น:	ภายใน		
ความสัมพันธ์:			
ความเกี่ยงเนื่อง:	นักพัฒนาซอฟต์แวร์		
การรวม:	Execute restriction		
การขยาย:			
การรับทอดคุณสมบัติ:			
ขั้นตอนการทำงานปกติ:	1. ระบบมีการใช้งานหน่วยความจำ 2. ระบบต้องคงไว้ซึ่งบูรณภาพข้อมูลของหน่วยความจำข้อมูล		
ขั้นตอนการทำงานย่อย:			
ขั้นตอนการทำงานทางเลือก / พิเศษ (ถ้ามี):			

ภาพที่ ก.1 คำอธิบายยูสเคส Maintain data memory integrity

2) ฟังก์ชัน Execute restriction

ชื่อยูสเคส:	Execute restriction	รหัส: UDC-002	ระดับความสำคัญ: มาก
ผู้กระทำหลัก:	Maintain data memory integrity	ประเภทยูสเคส:	เชิงละเอียด
ผู้มีส่วนเกี่ยวข้องและการใช้ประโยชน์:	Maintain data memory integrity: ต้องการจำกัดการประมวลผล		
คำอธิบาย:	เป็นยูสเคสที่ใช้ในการจำกัดการประมวลผล		
สิ่งกระตุ้น:	ระบบต้องจำกัดการประมวลผลอยู่เสมอ		
ประเภทของสิ่งกระตุ้น:	ตามเวลา		
ความสัมพันธ์:			
ความเกี่ยงเนื่อง:	Maintain data memory integrity		
การรวม:	Install and enable ExecShield		
การขยาย:			
การรับทอดคุณสมบัติ:			
ขั้นตอนการทำงานปกติ:	1. ระบบมีการใช้งานหน่วยความจำ 2. ระบบต้องจำกัดการประมวลผล		
ขั้นตอนการทำงานย่อย:			
ขั้นตอนการทำงานทางเลือก / พิเศษ (ถ้ามี):			

ภาพที่ ก.2 คำอธิบายยูสเคส Execute restriction

3) ฟังก์ชัน Install and enable ExecShield

ชื่อยูสเคส:	Install and enable ExecShield	รหัส: UDC-003	ระดับความสำคัญ: มาก
ผู้กระทำหลัก:	Execute restriction	ประเภทยูสเคส:	เชิงละเอียด
ผู้มีส่วนเกี่ยวข้องและการใช้ประโยชน์:	Execute restriction: ต้องการลดการเกิดการรวมหน่วยความจำชุดคำสั่งและหน่วยความจำข้อมูล		
คำอธิบาย:	เป็นยูสเคสที่ใช้ในการเปิดใช้งาน Exec Shield		
สิ่งกระตุ้น:	ระบบต้องเปิดใช้งาน Exec Shield อยู่เสมอ		
ประเภทของสิ่งกระตุ้น:	ตามเวลา		
ความสัมพันธ์:			
ความเกี่ยงเนื่อง:	Execute restriction		
การรวม:			
การขยาย:			
การรับทอดคุณสมบัติ:			
ขั้นตอนการทำงานปกติ:	1. ระบบมีการใช้งานหน่วยความจำ 2. ระบบต้องเปิดใช้งาน Exec Shield		
ขั้นตอนการทำงานย่อย:			
ขั้นตอนการทำงานทางเลือก / พิเศษ (ถ้ามี):			

ภาพที่ ก.3 คำอธิบายยูสเคส Install and enable ExecShield

4) ฟังก์ชัน Maintain privilege integrity

ชื่อยูสเคส:	Maintain privilege integrity	รหัส: UDC-004	ระดับความสำคัญ: มาก
ผู้กระทำหลัก:	นักพัฒนาซอฟต์แวร์	ประเภทยูสเคส:	เชิงละเอียด
ผู้มีส่วนเกี่ยวข้องและการใช้ประโยชน์:	นักพัฒนาซอฟต์แวร์: ต้องการปกป้องการควบคุมการเข้าถึง		
คำอธิบาย:	เป็นยูสเคสที่ใช้ในการคงไว้ซึ่งบูรณภาพข้อมูลของสิทธิ์		
สิ่งกระตุ้น:	ระบบต้องคงไว้ซึ่งบูรณภาพข้อมูลของสิทธิ์อยู่เสมอ		
ประเภทของสิ่งกระตุ้น:	ตามเวลา		
ความสัมพันธ์:			
ความเกี่ยงเนื่อง:	นักพัฒนาซอฟต์แวร์		
การรวม:	Use Mac (Mandatory Access Control)		
การขยาย:			
การรับทอดคุณสมบัติ:			
ขั้นตอนการทำงานปกติ:	1. ระบบมีการใช้งานสิทธิ์ 2. ระบบต้องคงไว้ซึ่งบูรณภาพข้อมูลของสิทธิ์		
ขั้นตอนการทำงานย่อย:			
ขั้นตอนการทำงานทางเลือก / พิเศษ (ถ้ามี):			

ภาพที่ ก.4 คำอธิบายยูสเคส Maintain privilege integrity

5) ฟังก์ชัน Use MAC (Mandatory Access Control)

ชื่อยูสเคส:	Use MAC (Mandatory Access Control)	รหัส: UDC-005	ระดับความสำคัญ: มาก
ผู้กระทำหลัก:	Maintain data privilege integrity	ประเภทยูสเคส:	เชิงละเอียด
ผู้มีส่วนเกี่ยวข้องและการใช้ประโยชน์:	Maintain data privilege integrity: ต้องการใช้ MAC		
คำอธิบาย:	เป็นยูสเคสที่ใช้ในการใช้ MAC		
สิ่งกระตุ้น:	ระบบต้องใช้ MAC อยู่เสมอ		
ประเภทของสิ่งกระตุ้น:	ตามเวลา		
ความสัมพันธ์:			
ความเกี่ยงเนื่อง:	Maintain data privilege integrity		
การรวม:	Install and enable SELinux		
การขยาย:			
การรับทอดคุณสมบัติ:			
ขั้นตอนการทำงานปกติ:	1. ระบบมีการใช้งานสิทธิ์ 2. ระบบต้องใช้ MAC		
ขั้นตอนการทำงานย่อย:			
ขั้นตอนการทำงานทางเลือก / พิเศษ (ถ้ามี):			

ภาพที่ ก.5 คำอธิบายยูสเคส Use MAC (Mandatory Access Control)

6) ฟังก์ชัน Install and enable SELinux

ชื่อยูสเคส:	Install and enable SELinux	รหัส: UDC-006	ระดับความสำคัญ: มาก
ผู้กระทำหลัก:	Use MAC (Mandatory Access Control)	ประเภทยูสเคส:	เชิงละเอียด
ผู้มีส่วนเกี่ยวข้องและการใช้ประโยชน์:	Use MAC (Mandatory Access Control): ต้องการลดการเกิดการใช้ DAC		
คำอธิบาย:	เป็นยูสเคสที่ใช้ในการเปิดใช้งาน SELinux		
สิ่งกระตุ้น:	ระบบต้องเปิดใช้งาน SELinux อยู่เสมอ		
ประเภทของสิ่งกระตุ้น:	ตามเวลา		
ความสัมพันธ์:			
ความเกี่ยงเนื่อง:	Use MAC (Mandatory Access Control)		
การรวม:			
การขยาย:			
การรับทอดคุณสมบัติ:			
ขั้นตอนการทำงานปกติ:	1. ระบบมีการใช้งานสิทธิ์ 2. ระบบต้องเปิดใช้งาน SELinux		
ขั้นตอนการทำงานย่อย:			
ขั้นตอนการทำงานทางเลือก / พิเศษ (ถ้ามี):			

ภาพที่ ก.6 คำอธิบายยูสเคส Install and enable SELinux

7) ฟังก์ชัน Maintain data access integrity

ชื่อยูสเคส:	Maintain data access integrity	รหัส: UDC-007	ระดับความสำคัญ: มาก
ผู้กระทำหลัก:	นักพัฒนาซอฟต์แวร์	ประเภทยูสเคส:	เชิงละเอียด
ผู้มีส่วนเกี่ยวข้องและการใช้ประโยชน์:	นักพัฒนาซอฟต์แวร์: ต้องการปกป้องการเข้าถึงระบบ		
คำอธิบาย:	เป็นยูสเคสที่ใช้ในการคงไว้ซึ่งบูรณภาพข้อมูลของการเข้าถึงระบบ		
สิ่งกระตุ้น:	ระบบต้องคงไว้ซึ่งบูรณภาพข้อมูลของการเข้าถึงระบบอยู่เสมอ		
ประเภทของสิ่งกระตุ้น:	ตามเวลา		
ความสัมพันธ์:			
ความเกี่ยงเนื่อง:	นักพัฒนาซอฟต์แวร์		
การรวม:	Configure firewall		
การขยาย:			
การรับทอดคุณสมบัติ:			
ขั้นตอนการทำงานปกติ:	1. ระบบมีการใช้งานไฟร์วอลล์ 2. ระบบต้องคงไว้ซึ่งบูรณภาพข้อมูลของไฟร์วอลล์		
ขั้นตอนการทำงานย่อย:			
ขั้นตอนการทำงานทางเลือก / พิเศษ (ถ้ามี):			

ภาพที่ ก.7 คำอธิบายยูสเคส Maintain access integrity

8) ฟังก์ชัน Configure firewall

ชื่อยูสเคส:	Configure firewall	รหัส: UDC-008	ระดับความสำคัญ: มาก
ผู้กระทำหลัก:	Maintain data access integrity	ประเภทยูสเคส:	เชิงละเอียด
ผู้มีส่วนเกี่ยวข้องและการใช้ประโยชน์:	Maintain data access integrity: ต้องการตั้งค่าการใช้งานไฟร์วอลล์		
คำอธิบาย:	เป็นยูสเคสที่ใช้ในการตั้งค่าการใช้งานไฟร์วอลล์		
สิ่งกระตุ้น:	ระบบต้องตั้งค่าการใช้งานไฟร์วอลล์อยู่เสมอ		
ประเภทของสิ่งกระตุ้น:	ตามเวลา		
ความสัมพันธ์:			
ความเกี่ยงเนื่อง:	Maintain data access integrity		
การรวม:	Install and enable iptables		
การขยาย:			
การรับทอดคุณสมบัติ:			
ขั้นตอนการทำงานปกติ:	1. ระบบมีการใช้งานไฟร์วอลล์ 2. ระบบต้องตั้งค่าการใช้งานไฟร์วอลล์		
ขั้นตอนการทำงานย่อย:			
ขั้นตอนการทำงานทางเลือก / พิเศษ (ถ้ามี):			

ภาพที่ ก.8 คำอธิบายยูสเคส Configure firewall

9) ฟังก์ชัน Install and enable iptables

ชื่อยูสเคส:	Install and enable iptables	รหัส: UDC-009	ระดับความสำคัญ: มาก
ผู้กระทำหลัก:	Configure firewall	ประเภทยูสเคส:	เชิงละเอียด
ผู้มีส่วนเกี่ยวข้องและการใช้ประโยชน์:	Configure firewall: ต้องการลดการเกิดการสร้างทราฟฟิกที่ประสงค์ร้ายเข้าไปในระบบ		
คำอธิบาย:	เป็นยูสเคสที่ใช้ในการเปิดใช้งาน iptables		
สิ่งกระตุ้น:	ระบบต้องเปิดใช้งาน iptables อยู่เสมอ		
ประเภทของสิ่งกระตุ้น:	ตามเวลา		
ความสัมพันธ์:			
ความเกี่ยงเนื่อง:	Configure firewall		
การรวม:			
การขยาย:			
การรับทอดคุณสมบัติ:			
ขั้นตอนการทำงานปกติ:	1. ระบบมีการใช้งานไฟร์วอลล์ 2. ระบบต้องเปิดใช้งาน iptables		
ขั้นตอนการทำงานย่อย:			
ขั้นตอนการทำงานทางเลือก / พิเศษ (ถ้ามี):			

ภาพที่ ก.9 คำอธิบายยูสเคส Install and enable iptables

10) ฟังก์ชัน Support unauthorized accessed accountability

ชื่อยูสเคส:	Support unauthorized accessed accountability	รหัส: UDC-010	ระดับความสำคัญ: มาก
ผู้กระทำหลัก:	นักพัฒนาซอฟต์แวร์	ประเภทยูสเคส:	เชิงละเอียด
ผู้มีส่วนเกี่ยวข้องและการใช้ประโยชน์:	นักพัฒนาซอฟต์แวร์: ต้องการปกป้องบันทึกการใช้งาน		
คำอธิบาย:	เป็นยูสเคสที่ใช้ในการคงไว้ซึ่งบูรณภาพข้อมูลของบันทึกการใช้งาน		
สิ่งกระตุ้น:	ระบบต้องคงไว้ซึ่งบูรณภาพข้อมูลของบันทึกการใช้งานอยู่เสมอ		
ประเภทของสิ่งกระตุ้น:	ตามเวลา		
ความสัมพันธ์:			
ความเกี่ยงเนื่อง:	นักพัฒนาซอฟต์แวร์		
การรวม:	Keep log		
การขยาย:			
การรับทอดคุณสมบัติ:			
ขั้นตอนการทำงานปกติ:	1. ระบบมีการใช้งานบันทึกการใช้งาน 2. ระบบต้องคงไว้ซึ่งบูรณภาพข้อมูลของบันทึกการใช้งาน		
ขั้นตอนการทำงานย่อย:			
ขั้นตอนการทำงานทางเลือก / พิเศษ (ถ้ามี):			

ภาพที่ ก.10 คำอธิบายยูสเคส Support unauthorized accessed accountability

11) ฟังก์ชัน Keep log

ชื่อยูสเคส:	Keep log	รหัส: UDC-011	ระดับความสำคัญ: มาก
ผู้กระทำหลัก:	Support unauthorized accessed accountability	ประเภทยูสเคส:	เชิงละเอียด
ผู้มีส่วนเกี่ยวข้องและการใช้ประโยชน์:	Support unauthorized accessed accountability: ต้องการเก็บบันทึกการใช้งานระบบ		
คำอธิบาย:	เป็นยูสเคสที่ใช้ในการเก็บบันทึกการใช้งานระบบ		
สิ่งกระตุ้น:	ระบบต้องเก็บบันทึกการใช้งานระบบอยู่เสมอ		
ประเภทของสิ่งกระตุ้น:	ตามเวลา		
ความสัมพันธ์:			
ความเกี่ยงเนื่อง:	Support unauthorized accessed accountability		
การรวม:	Install and enable rsyslog		
การขยาย:			
การรับทอดคุณสมบัติ:			
ขั้นตอนการทำงานปกติ:	1. ระบบมีการใช้งานบันทึกการใช้งาน 2. ระบบต้องเก็บบันทึกการใช้งาน		
ขั้นตอนการทำงานย่อย:			
ขั้นตอนการทำงานทางเลือก / พิเศษ (ถ้ามี):			

ภาพที่ ก.11 คำอธิบายยูสเคส Keep log

12) ฟังก์ชัน Install and enable rsyslog

ชื่อยูสเคส:	Install and enable rsyslog	รหัส: UDC-012	ระดับความสำคัญ: มาก
ผู้กระทำหลัก:	Keep log	ประเภทยูสเคส:	เชิงละเอียด
ผู้มีส่วนเกี่ยวข้องและการใช้ประโยชน์:	Keep log: ต้องการลดการเกิดการไม่สามารถตรวจจับการบุกรุก		
คำอธิบาย:	เป็นยูสเคสที่ใช้ในการเปิดใช้งาน rsyslog		
สิ่งกระตุ้น:	ระบบต้องเปิดใช้งาน rsyslog อยู่เสมอ		
ประเภทของสิ่งกระตุ้น:	ตามเวลา		
ความสัมพันธ์:			
ความเกี่ยงเนื่อง:	Keep log		
การรวม:			
การขยาย:			
การรับทอดคุณสมบัติ:			
ขั้นตอนการทำงานปกติ:	1. ระบบมีการใช้งานบันทึกการใช้งาน 2. ระบบต้องเปิดใช้งาน rsyslog		
ขั้นตอนการทำงานย่อย:			
ขั้นตอนการทำงานทางเลือก / พิเศษ (ถ้ามี):			

ภาพที่ ก.12 คำอธิบายยูสเคส Install and enable rsyslog

ภาคผนวก ข
คำอธิบายมิสยูสเคสของระบบการประเมินฮาร์ดเดนนิ่งของระบบ

1) ฟังก์ชัน Mix program memory with data memory

ชื่อมิสยูสเคส:	Mix program memory with data memory	รหัส: MDC-001	ระดับความสำคัญ: มาก
ผู้กระทำหลัก:	มิสยูสเซอร์	ประเภทมิสยูสเคส:	เชิงละเอียด
ผู้มีส่วนเกี่ยวข้องและการคุกคาม:	มิสยูสเซอร์: โจมตีหน่วยความจำผ่านทางการรวมหน่วยความจำชุดคำสั่งและหน่วยความจำข้อมูล		
สรุป:	เป็นมิสยูสเคสที่ใช้ในการรวมหน่วยความจำชุดคำสั่งและหน่วยความจำข้อมูล		
สิ่งกระตุ้น:	มิสยูสเซอร์ต้องการโจมตีระบบ		
ประเภทของสิ่งกระตุ้น:	ภายนอก		
เงื่อนไขก่อน:	ระบบมีการใช้งานหน่วยความจำ		
สมมติฐาน:	ระบบอนุญาตให้ป้อนรหัสคำสั่งเข้าไปในระบบ		
การรับประกันการบรรเทา:	ระบบมีการเปิดใช้งาน Exec Shield ดังนั้นทำให้เกิดการแยกหน่วยความจำชุดคำสั่งและหน่วยความจำข้อมูล		
กฎทางธุรกิจที่เกี่ยวข้อง:			
ประวัติมิสยูสเซอร์ที่เป็นไปได้:	นักพัฒนาซอฟต์แวร์ที่มีความชำนาญสูง ผู้ดูแลระบบที่มีเจตนาประสงค์ร้ายต่อระบบ		
ขอบเขต:	สภาพแวดล้อมของระบบ		
ระดับนามธรรม:	บรรลุเป้าหมายการโจมตี		
ความสัมพันธ์:			
ความเกี่ยวเนื่อง:	มิสยูสเซอร์		
การรวม:			
การขยาย:			
การรับทอดคุณสมบัติ:			
การคุกคาม:	Maintain data memory integrity		
การบรรเทา:	Install and enable ExecShield		
เส้นทางเดินปกติ:	1. ระบบมีการใช้งานหน่วยความจำ 2. มิสยูสเซอร์ป้อนรหัสคำสั่งผ่านพารามิเตอร์ที่ให้บริการหรือทำให้เกิดการล้นของบัฟเฟอร์		
เส้นทางเดินทางเลือก:			

ภาพที่ ข.1 คำอธิบายมิสยูสเคส Mix program memory with data memory

2) ฟังก์ชัน Use DAC (Discretionary Access Control)

ชื่อมิสยูสเคส:	Use DAC (Discretionary Access Control)	รหัส: MDC-002	ระดับความสำคัญ: มาก
ผู้กระทำหลัก:	มิสยูสเซอร์	ประเภทมิสยูสเคส:	เชิงละเอียด
ผู้มีส่วนเกี่ยวข้องและการคุกคาม:	มิสยูสเซอร์: โจมตีการควบคุมการเข้าถึงผ่านทางการใช้ DAC		
สรุป:	เป็นมิสยูสเคสที่ใช้ในการใช้ DAC		
สิ่งกระตุ้น:	มิสยูสเซอร์ต้องการโจมตีระบบ		
ประเภทของสิ่งกระตุ้น:	ภายนอก		
เงื่อนไขก่อน:	ระบบมีการใช้งานสิทธิ์		
สมมติฐาน:	ระบบอนุญาตให้เขียนทับหรือปรับเปลี่ยนนโยบายความมั่นคงปลอดภัยได้เอง		
การรับประกันการบรรเทา:	ระบบมีการเปิดใช้งาน SELinux ดังนั้นทำให้เกิดการกำหนดสิทธิ์ของผู้ใช้		
กฎทางธุรกิจที่เกี่ยวข้อง:			
ประวัติมิสยูสเซอร์ที่เป็นไปได้:	นักพัฒนาซอฟต์แวร์ที่มีความชำนาญสูง ผู้ดูแลระบบที่มีเจตนาประสงค์ร้ายต่อระบบ		
ขอบเขต:	สภาพแวดล้อมของระบบ		
ระดับนามธรรม:	บรรลุเป้าหมายการโจมตี		
ความสัมพันธ์:			
ความเกี่ยวเนื่อง:	มิสยูสเซอร์		
การรวม:			
การขยาย:			
การรับทอดคุณสมบัติ:			
การคุกคาม:	Maintain privilege integrity		
การบรรเทา:	Install and enable SELinux		
เส้นทางเดินปกติ:	1. ระบบมีการใช้งานสิทธิ์ 2. มิสยูสเซอร์เขียนทับหรือปรับเปลี่ยนนโยบายความมั่นคงปลอดภัย		
เส้นทางเดินทางเลือก:			

ภาพที่ ข.2 คำอธิบายมิสยูสเคส Use DAC (Discretionary Access Control)

3) ฟังก์ชัน Inject malicious traffic

ชื่อมิสยูสเคส:	Inject malicious traffic	รหัส: MDC-003	ระดับความสำคัญ: มาก
ผู้กระทำหลัก:	มิสยูสเซอร์	ประเภทมิสยูสเคส:	เชิงละเอียด
ผู้มีส่วนเกี่ยวข้องและการคุกคาม:	มิสยูสเซอร์: โจมตีช่องการเข้าถึงระบบผ่านทางการสร้างทราฟฟิกที่ประสงค์ร้ายเข้าไปในระบบ		
สรุป:	เป็นมิสยูสเคสที่ใช้ในการสร้างทราฟฟิกที่ประสงค์ร้ายเข้าไปในระบบ		
สิ่งกระตุ้น:	มิสยูสเซอร์ต้องการโจมตีระบบ		
ประเภทของสิ่งกระตุ้น:	ภายนอก		
เงื่อนไขก่อน:	ระบบไม่มีการใช้งานไฟร์วอลล์		
สมมติฐาน:	ระบบมีการเชื่อมต่อกับระบบอื่นหรือเครือข่ายอื่น โดยที่ไม่มีไฟร์วอลล์		
การรับประกันการบรรเทา:	ระบบมีการเปิดใช้งาน iptables ดังนั้นทำให้เกิดการกำหนดค่าไฟร์วอลล์		
กฎทางธุรกิจที่เกี่ยวข้อง:			
ประวัติมิสยูสเซอร์ที่เป็นไปได้:	นักพัฒนาซอฟต์แวร์ที่มีความชำนาญสูง ผู้ดูแลระบบที่มีเจตนาประสงค์ร้ายต่อระบบ		
ขอบเขต:	สภาพแวดล้อมของระบบ		
ระดับนามธรรม:	บรรลุเป้าหมายการโจมตี		
ความสัมพันธ์:			
ความเกี่ยวเนื่อง:	มิสยูสเซอร์		
การรวม:			
การขยาย:			
การรับทอดคุณสมบัติ:			
การคุกคาม:	Maintain access integrity		
การบรรเทา:	Install and enable iptables		
เส้นทางเดินปกติ:	1. ระบบไม่มีการใช้งานไฟร์วอลล์ 2. มิสยูสเซอร์สร้างทราฟฟิกที่ประสงค์ร้ายเข้าไปในระบบ		
เส้นทางเดินทางเลือก:			

ภาพที่ ข.3 คำอธิบายมิสยูสเคส Inject malicious traffic

4) ฟังก์ชัน Undetected break-in

ชื่อมิสยูสเคส:	Undetected break-in	รหัส: MDC-004	ระดับความสำคัญ: มาก
ผู้กระทำหลัก:	มิสยูสเซอร์	ประเภทมิสยูสเคส:	เชิงละเอียด
ผู้มีส่วนเกี่ยวข้องและการคุกคาม:	มิสยูสเซอร์: โจมตีบันทึกการใช้งานผ่านทางการไม่สามารถตรวจจับการบุกรุก		
สรุป:	เป็นมิสยูสเคสที่ใช้ในการไม่สามารถตรวจจับการบุกรุก		
สิ่งกระตุ้น:	มิสยูสเซอร์ต้องการโจมตีระบบ		
ประเภทของสิ่งกระตุ้น:	ภายนอก		
เงื่อนไขก่อน:	ระบบไม่มีการใช้งานบันทึกการใช้งาน		
สมมติฐาน:	ระบบอนุญาตให้เข้าใช้งานในระบบ		
การรับประกันการบรรเทา:	ระบบมีการเปิดใช้งาน rsyslog ดังนั้นทำให้เกิดการเก็บบันทึกการใช้งานระบบ		
กฎทางธุรกิจที่เกี่ยวข้อง:			
ประวัติมิสยูสเซอร์ที่เป็นไปได้:	นักพัฒนาซอฟต์แวร์ที่มีความชำนาญสูง ผู้ดูแลระบบที่มีเจตนาประสงค์ร้ายต่อระบบ		
ขอบเขต:	สภาพแวดล้อมของระบบ		
ระดับนามธรรม:	บรรลุเป้าหมายการโจมตี		
ความสัมพันธ์:			
ความเกี่ยวเนื่อง:	มิสยูสเซอร์		
การรวม:			
การขยาย:			
การรับทอดคุณสมบัติ:			
การคุกคาม:	Support unauthorized accessed accountability		
การบรรเทา:	Install and enable rsyslog		
เส้นทางเดินปกติ:	1. ระบบไม่มีการใช้งานบันทึกการใช้งาน 2. มิสยูสเซอร์เข้าใช้งานระบบ โดยที่ระบบไม่สามารถตรวจจับกิจกรรมที่เกิดขึ้นได้		
เส้นทางเดินทางเลือก:			

ภาพที่ ข.4 คำอธิบายมิสยูสเคส Undetected break-in

ภาคผนวก ค
บัตรซีอาร์ซีของระบบการประเมินฮาร์ดเดนนิ่งของระบบ

1) คลาส CommandFactory

ด้านหน้า:

ชื่อคลาส: CommandFactory	รหัส: CRC-001	ประเภท: Concrete, Domain
คำอธิบาย: เป็นโครงสร้างข้อมูลที่ใช้ในการรับคำสั่งในการตรวจสอบฮาร์ดเดนนิ่งของระบบ เช่น ตรวจสอบการจำกัดการประมวลผล ตรวจสอบการใช้ MAC ตรวจสอบการตั้งค่าการใช้งานไฟร์วอลล์ และตรวจสอบการเก็บบันทึกการใช้งานระบบ		ยูสเคสที่เกี่ยวข้อง: - Execute restriction - Use MAC (Mandatory Access Control) - Configure firewall - Keep log
หน้าที่ - รับคำสั่งในการตรวจสอบฮาร์ดเดนนิ่งของระบบ		ผู้ร่วมมือ -

ด้านหลัง:

ลักษณะประจำ: - getCommand
ความสัมพันธ์:
แบบลักษณะทั่วไป (เป็นประเภทหนึ่งของ):
แบบการรวมกลุ่ม (ประกอบด้วย):
แบบอื่นๆ: - ExecutionRestrictionCommand - UseMacCommand - EnableFirewallCommand - KeepLogMsgCommand

ภาพที่ ค.1 บัตรซีอาร์ซี CommandFactory

2) คลาส ExecutionRestrictionCommand

ด้านหน้า:

ชื่อคลาส: ExecutionRestrictionCommand	**รหัส:** CRC-002	**ประเภท:** Concrete, Domain
คำอธิบาย: เป็นโครงสร้างข้อมูลที่ใช้ในการตรวจสอบการจำกัดการประมวลผล		**ยูสเคสที่เกี่ยวข้อง:** - Install and enable ExecSheild
หน้าที่ - ตรวจสอบว่า ระบบมีการเปิดใช้งาน Exec Shield หรือไม่ โดยใช้คำสั่ง cat /proc/sys/kernel/exec-shield ถ้าได้ผลลัพท์เป็น 1 แสดงว่า ระบบมีการเปิดใช้งาน Exec Shield		**ผู้ร่วมมือ** -

ด้านหลัง:

ลักษณะประจำ: - execute
ความสัมพันธ์:
แบบลักษณะทั่วไป (เป็นประเภทหนึ่งของ):
แบบการรวมกลุ่ม (ประกอบด้วย):
แบบอื่นๆ: - CommandFactory - CommandResult

ภาพที่ ค.2 บัตรซีอาร์ซี ExecutionRestrictionCommand

3) คลาส UseMacCommand

ด้านหน้า:

ชื่อคลาส: UseMacCommand	**รหัส:** CRC-003	**ประเภท:** Concrete, Domain
คำอธิบาย: เป็นโครงสร้างข้อมูลที่ใช้ในการตรวจสอบการใช้ MAC		**ยูสเคสที่เกี่ยวข้อง:** - Install and enable SELinux
หน้าที่ - ตรวจสอบว่า ระบบมีการเปิดใช้งาน SELinux หรือไม่ โดยใช้คำสั่ง cat /selinux/enforce ถ้าได้ผลลัพท์เป็น 1 แสดงว่า ระบบมีการเปิดใช้งาน SELinux		**ผู้ร่วมมือ** -

ด้านหลัง:

ลักษณะประจำ: - execute
ความสัมพันธ์:
แบบลักษณะทั่วไป (เป็นประเภทหนึ่งของ):
แบบการรวมกลุ่ม (ประกอบด้วย):
แบบอื่นๆ: - CommandFactory - CommandResult

ภาพที่ ค.3 บัตรซีอาร์ซี UseMacCommand

4) คลาส EnableFirewallCommand

ด้านหน้า:

ชื่อคลาส: EnableFirewallCommand	**รหัส:** CRC-004	**ประเภท:** Concrete, Domain
คำอธิบาย: เป็นโครงสร้างข้อมูลที่ใช้ในการตรวจสอบการตั้งค่าการใช้งานไฟร์วอลล์		**ยูสเคสที่เกี่ยวข้อง:** - Install and enable iptables
หน้าที่ - ตรวจสอบว่า ระบบมีการเปิดใช้งาน iptables หรือไม่ โดยใช้คำสั่ง service iptables status ถ้าได้ผลลัพท์เป็น Start แสดงว่าระบบมีการเปิดใช้งาน iptables		**ผู้ร่วมมือ** -

ด้านหลัง:

ลักษณะประจำ: - execute
ความสัมพันธ์:
แบบลักษณะทั่วไป (เป็นประเภทหนึ่งของ):
แบบการรวมกลุ่ม (ประกอบด้วย):
แบบอื่นๆ: - CommandFactory - CommandResult

ภาพที่ ค.4 บัตรซีอาร์ซี EnableFirewallCommand

5) คลาส KeepLogMsgCommand

ด้านหน้า:

ชื่อคลาส: KeepLogMsgCommand	**รหัส:** CRC-005	**ประเภท:** Concrete, Domain
คำอธิบาย: เป็นโครงสร้างข้อมูลที่ใช้ในการตรวจสอบการเก็บบันทึกการใช้งานระบบ		**ยูสเคสที่เกี่ยวข้อง:** - Install and enable rsyslog
หน้าที่ - ตรวจสอบว่า ระบบมีการเปิดใช้งาน rsyslog หรือไม่ โดยใช้คำสั่ง service rsyslog status ถ้าได้ผลลัพท์เป็น Start แสดงว่าระบบมีการเปิดใช้งาน rsyslog		**ผู้ร่วมมือ** -

ด้านหลัง:

ลักษณะประจำ: - execute
ความสัมพันธ์:
แบบลักษณะทั่วไป (เป็นประเภทหนึ่งของ):
แบบการรวมกลุ่ม (ประกอบด้วย):
แบบอื่นๆ: - CommandFactory - CommandResult

ภาพที่ ค.5 บัตรซีอาร์ซี KeepLogMsgCommand

6) คลาส CommandResult

ด้านหน้า:

ชื่อคลาส: CommandResult	รหัส: CRC-006	ประเภท: Concrete, Domain
คำอธิบาย: เป็นโครงสร้างข้อมูลที่ใช้ในการแสดงผลลัพธ์จากการตรวจสอบฮาร์ดเดนนิ่งของระบบ		ยูสเคสที่เกี่ยวข้อง: - Install and enable ExecSheild - Install and enable SELinux - Install and enable iptables - Install and enable rsyslog
หน้าที่ - แสดงผลลัพธ์จากการตรวจสอบฮาร์ดเดนนิ่งของระบบ		ผู้ร่วมมือ -

ด้านหลัง:

ลักษณะประจำ: - getMsg - setMsg - isStatus - setStatus
ความสัมพันธ์:
แบบลักษณะทั่วไป (เป็นประเภทหนึ่งของ):
แบบการรวมกลุ่ม (ประกอบด้วย):
แบบอื่นๆ: - ExecutionRestrictionCommand - UseMacCommand - EnableFirewallCommand - KeepLogMsgCommand

ภาพที่ ค.6 บัตรซีอาร์ซี CommandResult

ภาคผนวก ง
ซอร์สโคดของโมดูลการประเมินฮาร์ดเดนนิ่งของระบบ

1) ชุดคำสั่งในการเลือกแต่ละการโจมตี

```
public class CommandFactory {
  public static Command getCommand(int command) {
     if(command==1){
        return new ExecutionRestrictionCommand();
     }else if (command==2){
        return new UseMacCommand();
     }else if (command==3){
        return new EnableFirewallCommand();
     }else if (command==4){
        return new KeepLogMsgCommand();
     }
     return null;
  }
}
```

2) ชุดคำสั่งในการประเมินระบบของ หน่วยความจำ

```
public class ExecutionRestrictionCommand extends CommandBase {
    public CommandResult execute() {
       CommandResult commandResult = new CommandResult();
       System.out.println("ExecutionRestrictionCommand");
       String result = null;
       try {
          result = run(CommandConstant.EXECUTION_RESTRICTIONACTIVE);
          System.out.println(result);
       } catch (Exception e) {
          commandResult.setStatus(false);
```

```
            commandResult.setMsg(e.getMessage());
            return commandResult;
        }
        if (result.equalsIgnoreCase("0")) {
            commandResult.setStatus(false);
commandResult.setMsg(MessageConstant.EXECUTION_RESTRICTION_NOT_ACTIVE_
DESC);
        } else {
            commandResult.setStatus(true);
commandResult.setMsg(MessageConstant.EXECUTION_RESTRICTION_ACTIVE_DESC)
;
        }
        return commandResult;
    }
}
```

3) ชุดคำสั่งในการประเมินระบบของ การควบคุมการเข้าถึง

```
public class UseMacCommand extends CommandBase {
    @Override
    public CommandResult execute() {
        CommandResult commandResult = new CommandResult();
        System.out.println("UseMacCommand");
        String result = null;
        try {
            result = run(CommandConstant.USEMAC);
            System.out.println(result);
        } catch (Exception e) {
            commandResult.setStatus(false);
            commandResult.setMsg(e.getMessage());
            return commandResult;
```

```
        }
        if (result.equalsIgnoreCase("0")) {
            commandResult.setStatus(false);
            commandResult.setMsg(MessageConstant.USEMAC_NOT_ACTIVE_DESC);
        } else {
            commandResult.setStatus(true);
            commandResult.setMsg(MessageConstant.USEMAC_ACTIVE_DESC);
        }
        return commandResult;
    }
}
```

4) ชุดคำสั่งในการประเมินระบบของ ไฟร์วอลล์

```
public class EnableFirewallCommand extends CommandBase {
    @Override
    public CommandResult execute() {
        CommandResult commandResult = new CommandResult();
        System.out.println("EnableFirewallCommand");
        String result = null;
        try {
            result = run(CommandConstant.ENABLE_FIREWALL);
            System.out.println(result);
        } catch (Exception e) {
            commandResult.setStatus(false);
            commandResult.setMsg(e.getMessage());
            return commandResult;
        }
        if (result.equalsIgnoreCase("iptables: Firewall is not running.")) {
            commandResult.setStatus(false);
commandResult.setMsg(MessageConstant.ENABLE_FIREWALL_NOT_ACTIVE_DESC);
```

```
        } else {
            commandResult.setStatus(true);
commandResult.setMsg(MessageConstant.ENABLE_FIREWALL_ACTIVE_DESC);
        }
        return commandResult;
    }
}
```

5) ชุดคำสั่งในการประเมินระบบของ บันทึกการใช้งาน

```
public class KeepLogMsgCommand extends CommandBase {
    @Override
    public CommandResult execute() {
        CommandResult commandResult = new CommandResult();
        System.out.println("KeepLogMsgCommand");
        String result = null;
        try {
            result = run(CommandConstant.KEEP_LOG_MESSAGE);
            System.out.println(result);
        } catch (Exception e) {
            commandResult.setStatus(false);
            commandResult.setMsg(e.getMessage());
            return commandResult;
        }
        if (result.equalsIgnoreCase("rsyslogd is stopped")) {
            commandResult.setStatus(false);
commandResult.setMsg(MessageConstant.KEEP_ON_MESSAGE_NOT_ACTIVE_DESC);
        } else {
            commandResult.setStatus(true);
commandResult.setMsg(MessageConstant.KEEP_ON_MESSAGE_ACTIVE_DESC);
        }
```

```
        return commandResult;
    }
}
```

6) ชุดคำสั่งในการแสดงผลลัพธ์การประเมินระบบ

```
public class CommandResult {
    private boolean status;
    private String msg;
    public String getMsg() {
        return msg;
    }

    public void setMsg(String msg) {
        this.msg = msg;
    }

    public boolean isStatus() {
        return status;
    }

    public void setStatus(boolean status) {
        this.status = status;
    }
}
```

ภาคผนวก จ
สคริปต์การสร้างซอฟต์แวร์ช่วยเหลือการติดตั้ง

```
REM ************Start Setting**************
set APP_PATH_DEFAULT_FIND=APP_PATH_DEFAULT
set APP_LAUNCH_FIND=APP_LAUNCH
REM define default program destination path
set APP_PATH_DEFAULT_VALUE=/usr/local/calculator
REM define program source execute file
set APP_LAUNCH_VALUE=jCalculator.jar
REM define program source path
set PROGRAM_SRC=pgSource_win_jar
REM define output name
set OUTPUT_NAME=ISEL-Setup-Calculator
REM **************End Setting**************
del Output\*.* /S /Q
cd ISEL-Template
rd pgSource /S /Q
mkdir pgSource
cd ..
copy .\properties\template-application.properties .\ISEL-Template\application.properties
strfind -r -y %APP_PATH_DEFAULT_FIND% %APP_PATH_DEFAULT_VALUE% .\ISEL-
Template\application.properties
strfind -r -y %APP_LAUNCH_FIND% %APP_LAUNCH_VALUE% .\ISEL-
Template\application.properties
xcopy /S /Y %PROGRAM_SRC% ISEL-Template\pgSource
cd ISEL-Template
zip.exe -r ../Output/%OUTPUT_NAME%.jar *
rd pgSource /S /Q
mkdir pgSource
```

ประวัติผู้เขียนวิทยานิพนธ์

นางสาวขวัญชนก ลิ่มบัณฑิต เกิดเมื่อวันที่ 4 กุมภาพันธ์ พ.ศ. 2524 ที่ อ.เมือง จ.ระนอง สำเร็จการศึกษาปริญญาตรี วิศวกรรมศาสตรบัณฑิต สาขา วิศวกรรมคอมพิวเตอร์ จาก ภาควิชาวิศวกรรมคอมพิวเตอร์ คณะ วิศวกรรมศาสตร์ มหาวิทยาลัยสงขลานครินทร์ เมื่อปีการศึกษา 2547 จากนั้น เข้าศึกษาต่อในหลักสูตรวิทยาศาสตรมหาบัณฑิต สาขา วิศวกรรมซอฟต์แวร์ ที่ ภาควิชาวิศวกรรมคอมพิวเตอร์ คณะ วิศวกรรมศาสตร์ จุฬาลงกรณ์มหาวิทยาลัย ใน ภาคการศึกษา ปลาย ปีการศึกษา 2552

บทความเรื่อง Misuse for Security Hardening Assessment in Application Software Deployment ซึ่งเป็นส่วนหนึ่งของวิทยานิพนธ์เรื่องนี้ได้รับการตีพิมพ์ในวารสารระดับนานาชาติ The International Journal of Future Computer and Communication (IJFCC, ISSN: 2010-3751) ปีที่พิมพ์ 2012 และเข้าร่วมประชุมวิชาการระดับนานาชาติ The International Conference on Advancements in Information Technology (ICAIT 2012) ณ ฮ่องกง ระหว่างวันที่ 2 – 3 มิถุนายน พ.ศ. 2555