บทที่ 3

วิธีดำเนินการวิจัย

3.1 การวางแผนและการเตรียมการ

การสร้างต้นแบบ (Prototyping) (ศรีไพร และเจษฎาพร, 2549)

การสร้างระบบต้นแบบเพื่อให้ผู้ใช้ทดลองใช้งาน ซึ่งนอกจากผู้ใช้จะได้แนวคิดเกี่ยวกับสารสนเทศที่ต้องการแล้ว ยังช่วยให้มองเห็นภาพของระบบที่จะพัฒนาได้ชัดเจนขึ้น หากต้นแบบที่สร้างขึ้นไม่เป็นไปตามความต้องการก็จะถูกนำมาแก้ไขปรับปรุงให้ดีขึ้นและเสนอต่อผู้ใช้ โดยจะทำกระบวนการเหล่านี้ซ้ำจนกระทั่งต้นแบบสามารถทำงานได้ตรงตามที่ผู้ใช้ต้องการจึงนำมาเป็นต้นแบบสำหรับการพัฒนาระบบสารสนเทศจริงต่อไป วิธีนี้จึงมีความยืดหยุ่นต่อการเปลี่ยนแปลงความต้องการมากกว่า รวมทั้งใช้เวลาและค่าใช้จ่ายน้อยกว่าวิธีการพัฒนาระบบงานแบบดั้งเดิม เนื่องจากต้นแบบที่สร้างขึ้นมาไม่ได้มีความสมบูรณ์ครบถ้วนทั้งหมด เช่น อาจยังไม่พิจารณาเรื่องประสิทธิภาพและเวลาตอบสนองของระบบกรณีที่มีผู้ใช้ระบบจำนวนมาก จึงทำให้การสร้าง ปรับปรุง และแก้ไขนั้นใช้เวลาและค่าใช้จ่ายน้อยกว่า

ขั้นที่ 1 ศึกษาภาพรวมของการทำระบบโดยการศึกษาระบบการรักษาความปลอดภัยของสำนักวิทยบริการและเทคโนโลยีสารสนเทศ (สวท.) ดังนี้

1.1 ศึกษาการทำ Network Segmented Architecture (Pfleeger and Pfleeger, 2007)

1.2 ศึกษาข้อมูลการรักษาความปลอดภัยของเครือข่ายเรื่องต่าง ๆ เช่น การเก็บข้อมูลจราจรทางคอมพิวเตอร์ (log file) ในส่วนของระบบจัดการความรู้กลุ่มงานวิจัย มหาวิทยาลัยเทคโนโลยีราชมงคลธัญบุรี ที่ สวท. ดำเนินการ

1.3 ศึกษาความต้องการใช้ Hardware และ Software สำหรับงาน Physical Environment Security, งานสร้างระบบต้นแบบ Prototype Production, Input/Output Control, งานสร้างระบบจริงทุกส่วน Practical Production และงานบำรุงรักษา Hardware and System Software Mantenance Controls and Integrity Controls รวมทั้งงาน Security Awareness and Training

1.4 ศึกษาการทำ Virtual Private Network (VPN) (Pfleeger and Pfleeger, 2007)

1.5 ศึกษาการทำ Ipsec หรือ seurity association

1.6 ศึกษาการทำการป้องกันเรียกว่า intrusion dectection system เป็น layerer network protection ที่เป็น logical view

1.7 ศึกษาการทำการควบคุมที่เรียกว่า Network Vulnerabilities and Controls

1.8 ศึกษาการทำการป้องกันแบบต่าง ๆ ของ Firewalls ได้แก่ packet filtering gateway หรือ screening routers, การทำ stateful inspection firewalls, การทำ application proxy

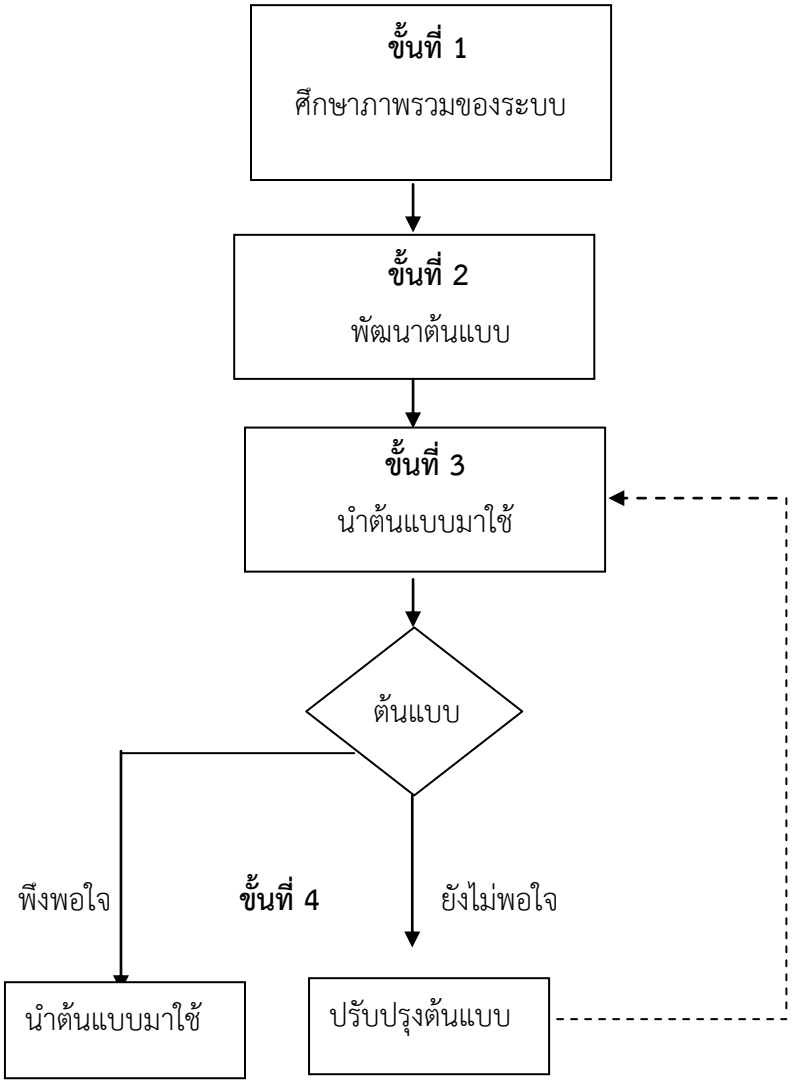
1.9 ศึกษาการทำการป้องกันแบบต่างๆ ของ Personal Firewalls ได้แก่ Norton Personal Firewalls, McAfee และ Zone Alarm

ผู้วิจัยได้ขอความอนุเคราะห์เรื่องมาตรการการรักษาความปลอดภัยของเว็บไซต์ และขอความอนุเคราะห์ Account ของผู้ใช้ที่มีสิทธิ์ดังนี้

- Upload หน้า Webpage ที่แก้ไขแล้วไปยัง www.ird.rmutt.ac.th/KMIS

- ขอชื่อผู้ใช้ที่มีสิทธิบริหารฐานข้อมูล MySQL เฉพาะฐานข้อมูลของระบบ KMIS ของสถาบันวิจัยและพัฒนา

การพัฒนาระบบโดยใช้ต้นแบบแบ่งออกเป็น 4 ขั้นตอน ดังรูปที่ 3.1



รูปที่ 3.1 ขั้นตอนการพัฒนาระบบโดยใช้ต้นแบบ

<u>ขั้นที่ 2</u> พัฒนาต้นแบบ เริ่มแรกผู้ออกแบบระบบทำการสร้างต้นแบบที่จะเป็นการรักษาความปลอดภัยแก่ระบบจัดการความรู้กลุ่มงานวิจัย โดยทดลองทำเป็นปฏิบัติการบนเครือข่ายจริง ใช้เพียงกลุ่มงานวิจัยบางกลุ่มและทำการจำลองบน Space ที่จัดขึ้นของเครือข่าย ดังนี้

2.1 ออกแบบและสร้างโปรแกรมเพื่อรองรับการโจมตีทางเว็บไซต์ประเภทการโจมตีเข้าสู่ระบบแบบ Injection ที่เป็นเทคนิค Brute-Force การโจมตีนี้เป็นการแก้รหัสผ่านแบบหนึ่งที่สามารถแก้ได้ทุกรหัสผ่าน ส่วนมากใช้วิธีนี้เมื่อไม่สามารถหาทางอื่นในการแก้รหัสผ่านได้ โดยวิธีนี้จะเกี่ยวกับการเช็ครหัสผ่านที่เป็นไปได้ทั้งหมดจนกว่าจะเจอรหัสที่ถูกต้องโดยมีการค้นหาจากความเป็นไปได้

ทั้งหมด การแก้รหัสผ่านโดยวิธี Brute Force จะใช้เวลามากหรือน้อยนั้นขึ้นอยู่กับความยาวของรหัสที่แก้ หากรหัสยาวจะเสียเวลามาก แต่หากรหัสสั้นจะใช้เวลาน้อย โดยในปัจจุบันการเข้ารหัสจะเป็นแบบ 128 บิต หรือ 256 บิต ซึ่งมีความซับซ้อนมากโดยจะใช้เวลานานกว่าแบบ 56 บิต โดยถ้า Brute Force ได้รหัสผ่านจากแบบ 56 บิต โดยใช้เวลา 1 วินาที การแก้รหัสแบบ 128 บิตจะใช้เวลา 128 149,745,258,842,898 ปี และถ้าเป็นการเข้ารหัสแบบ 256 บิตจะใช้เวลาเป็นอินฟินิตี้

แต่ทุกวันนี้เทคโนโลยีก็ก้าวหน้าขึ้นเรื่อยๆสำหรับปัจจุบันก็มีฮาร์ดแวร์ที่มีความสามารถในการคำนวณมาก เช่น GPU ที่อยู่ในการ์ดจอ ซึ่งมีการบรรจุหลายร้อยตัวประมวลผล ทำให้ส่งผลดีต่อการ Brute Force ทำให้มันเหมาะต่อการแก้รหัสผ่าน แต่ก็มีรหัสผ่านรูปแบบที่ไม่สามารถใช้วิธีการ Brute Force ได้ นั่นคือรหัสผ่านแบบใช้ครั้งเดียว โดยจะใช้การสุ่ม (Random) คือ ไม่มีรูปแบบที่แน่นอน และการเจาะรหัสผ่านแบบออฟไลน์จะป้องกันการเจาะไม่ได้แต่เมื่อเชื่อมต่อเข้ากับฐานข้อมูลแล้ว ผู้ดูแลระบบจะสามารถจำกัดการใส่รหัสผ่านหลาย ๆ ครั้งได้ หรืออาจจะต้องใช้การระบุอักษรตามภาพ และใช้การติดตาม IP ที่มีความผิดปกติ

ผู้วิจัยได้จัดทำระบบรักษาความปลอดภัยเพื่อป้องกันหรือสกัดการโจมตีแบบ Brute Force ดังนี้คือ การป้องกันโดยการนำข้อมูลรับเข้า (รหัสผ่านที่ผู้ใช้กรอก) มาผ่านกระบวนการเข้ารหัสอย่างนัยยะ (เฉพาะแบบทางเดียว) ดังรูปที่ 3.2

รูปที่ 3.2 การป้องกันการโจมตีแบบ Brute Force

2.2 ออกแบบและสร้างระบบป้องกันการโจมตีรูปแบบ SQL Injection และป้องกันการโจมตีที่ใช้เทคนิคการหลบหลีกการตรวจจับ SQL Injection เป็นการโจมตีที่ใช้มากที่สุดวิธีหนึ่งที่ผู้โจมตี (Attacker) ใช้ขโมยข้อมูลบุคคลหรือข้อมูลที่เป็นความลับของเว็บไซต์ใด ๆ โดยการใส่คำสั่งในการเข้าถึงฐานข้อมูล (Database) หรือที่รู้จักกันว่า “SQL Query” ไปยังช่องโหว่ของเว็บไซต์ใด ๆ ทำให้ผู้โจมตีนั้นมีสิทธิ์ในการเข้าถึงและจัดการข้อมูลทั้งหมดใน Database ในการนี้ Perimeter Firewall, IPS รวมทั้งเทคโนโลยี Web Application Firewall ที่จะพยายามวิเคราะห์และตรวจจับ SQL Injection โดยเทียบกับทางฐานข้อมูล signature ที่มีอยู่ซึ่งจะดูรูปแบบของข้อมูลภายใน Web Traffic Flows ว่าตรงกับ signature ใด ๆ ไหม เพื่อจะระบุว่าเป็น SQL Injection และบล็อกข้อมูลนั้น จากการทดสอบทราบว่า แค่ signature อย่างเดียวไม่เพียงพอที่จะป้องกันระบบได้

SQL Injection เป็นการโจมตีโดยการยิงคำสั่ง SQL เข้าไปยังเว็บใด ๆ ทำให้เห็นถึงข้อมูลที่เกี่ยวกับ Database วิธีการของมันคือใช้เพื่อตรวจสอบหาช่องโหว่ในการใส่ค่า Input ลงใน User Interface ของเว็บไซต์ ในทางทฤษฎีเว็บไซต์ควรมีการตรวจสอบทุกค่าที่ป้อนเข้ามาก่อนจะส่ง Query นั้นไปยัง Database แต่ถ้าการตรวจสอบ Input นั้นไม่ได้สมบูรณ์ทุกอันหรือทุก ๆ Input ที่เข้ามา (อาจมีเป็นร้อยเป็นพันเลยก็ได้) ผู้โจมตีอาจจะเปลี่ยนแปลงบางส่วนของ Web Request (ส่วนใหญ่ก็พวก URL Parameter) เพื่อให้สามารถ Query ข้อมูลจาก Database ได้ ผลของการ Query จะถูกแสดงในส่วนของ HTML response ซึ่งสร้างเว็บไซต์นั้น ตัวอย่าง SQL Injection โดยโจมตีเว็บระบบจัดการความรู้กลุ่มงานวิจัยในเว็บนี้จะมีการแสดงหมายเลขบัตรประชาชนของสมาชิกที่เป็นนักวิจัย เช่น อ้างอิงจากเพศด้วย URL

http://www.superhealth.com/show_members.asp?sex=m

หลังการส่ง Query นี้ไปยัง Database แล้ว Web Application จะมองว่าเป็นคำสั่งดังนี้

select SSN, NAME from PATIENTS where FAMILY = XXX and sex = ‘m’

โดย XXX จะหมายถึงชื่อของครอบครัวที่เอามาจากการล็อกอินฐานข้อมูล ถ้าเว็บไม่มีระบบรักษาความปลอดภัยต่อการ SQL Injection บนพารามิเตอร์ sex แล้วจะทำให้ผู้โจมตีสามารถจัดการเว็บนี้ได้อย่างง่ายได้ด้วยการ injection คำสั่งเข้ามา เช่น

http://www.superhealth.com/show_members.asp?sex=m’ or 1=1 or ‘1’=’1

เมื่อส่ง URL นี้ไปยัง Database แล้วจะเกิดผลอะไรบ้าง คือมันจะมีการ Query โดยคำสั่ง

select SSN, name from patients where family = xxxxxx and sex = ‘m’ or 1=1 or ‘1’=’1’

ซึ่งคำสั่งนี้จะทำให้มีการแสดงข้อมูลของคนไข้ทั้งหมดในฐานข้อมูลออกมา แล้วแสดงไปยัง Attacker SQL Injection นั้นสามารถทำให้ได้มาซึ่ง Username เป็นพัน ๆ ได้ในการโจมตีเพียงครั้งเดียวเลย ทำให้มันเป็นหนึ่งในภัยคุกคามที่อันตรายที่สุดเกี่ยวกับการขโมยข้อมูลของ Web application

ผู้วิจัยได้จัดทำระบบรักษาความปลอดภัยเพื่อการป้องกัน SQL Injection ด้วย Signature กระบวนการในป้องกันด้วย signature นั้น คือ ตัวอุปกรณ์ต่าง ๆ ไม่ว่าจะเป็น Firewall, IPS จะคอยตรวจดู Traffic ใน Network โดยจะมองหา Text String หรือข้อความใน Packet ที่เหมือนกับ Signature แล้วอุปกรณ์เหล่านั้นจะมองว่าเป็น ผู้โจมตีโดยส่วนใหญ่แล้ว String จะเป็นรูปแบบของ SQL Injection เช่น “or 1=1” แบบนี้เป็นแบบเบสิกที่ Attacker ใช้กัน ดังนั้นมันก็จะ match กับ Signature ที่มีอยู่ฐานข้อมูลของ Signature ส่วนใหญ่จะเป็นโจมตีอย่างง่าย ๆ รูปแบบทั่ว ๆ ไป เท่านั้นถึงมันจะระบุได้ว่าเป็นการโจมตี

การหลีกเลี่ยงเพื่อไม่ให้ Match กับ Signature อย่างง่าย ๆ ผู้โจมตีส่วนมากจะใช้เทคนิคอย่างง่าย ๆ เพื่อหลีกเลี่ยง Signature ดังนี้

1) การเข้ารหัส (Encoding) เหตุผลที่มีการใช้แบบนี้กันมากเพราะบางอุปกรณ์ล้มเหลวกับการถอดรหัส (Decoding), บางตัวอาจถอดรหัสได้แต่ไม่สามารถทำแบบ Real Time ได้ เทคนิคการเข้ารหัส เช่น URL Encoding, UTF-8 บ่อยครั้งที่เทคนิคพวกนี้จะใช้ซ่อน Attack จากการป้องกันแบบ Signature ได้

2) การใช้ช่องว่าง (White Spaces Diversity) Signature ที่ใช้ป้องกัน SQL Injection หลายรูปแบบจะใช้วิธีแยก Expression ด้วยช่องว่าง ทำให้เกิด False Positives จำนวนมากถ้ามันแยกไม่เจอคำอย่าง SELECT ที่ Match กับ Signature แต่ก็มีการที่จะหลีกเลี่ยงได้ โดยผู้โจมตีอาจหลบการตรวจเจอได้โดยแทนที่ 1 ช่องว่างหรือการกด spacebar 1 ครั้ง ด้วยการใส่ 2 ช่องว่าง หรือ 1 ช่องว่าง+กด tab 1 ครั้ง

3) การแบ่งเป็นส่วนย่อย ๆ ของ IP และ TCP เป็นเทคนิคการเลี่ยง Signature อีกวิธีหนึ่ง ที่จะซ่อนการโจมตีไม่ให้ Match กับ Signature โดยการแบ่งข้อความ (String) ใส่ลงไปในหลาย ๆ ส่วนย่อยของ Packet

เทคนิคการเลี่ยง Signature แบบขั้นสูง (Advance) ดังนี้

1) การเลี่ยง signature สำหรับ OR 1=1 SQL Injection เช่น “or 1=1” นั้น มีความหมายเช่นเดียวกับ (equivalent) “or 1=1” อย่าง OR ‘Unusual’ = ‘Unusual’
ระบบที่ตรวจจับได้ยังสามารถระบุรูปแบบที่เหมือนกันแบบง่าย ๆ นี้ได้ ดังนั้นผู้โจมตีจึงได้หาทางเพิ่มเป็น 2 expression เพื่อให้ดูต่างไปแต่ความหมายเหมือนเดิม อย่างการเพิ่ม N เข้าไป เช่น

OR ‘Simple’ = N’Simple’

คำสั่งนี้เป็นการบอก SQL Server ว่า Cast ค่า ‘Simple’หลัง N เป็นชนิด nvarchar แต่การเปรียบเทียบค่ายังเท่าเดิม แต่ยังมีวิธีที่ดีกว่านั้นโดยเราอาจแยก String ออกเป็น 2 ตัว ค่าก็ยังเท่าเดิม เช่น

OR ‘Simple’ = ‘Sim’ + ‘ple’

วิธีนี้อาจดูเหมือนว่าจะดีที่สุดมีการเลี่ยงไม่ให้ Signature จับได้ ผู้จ่ายระบบได้สร้าง regular expression มารับมือโดยจะมองหา “or” หรือ “=” ในข้อความแต่วิธีนี้ก็ทำให้เกิด False Positive จำนวนมากเลยทีเดียว แต่ผู้โจมตีทำการหา Expression ใหม่ที่ให้ค่าเป็น true ดังเช่นการใช้ “LIKE” ในคำสั่ง SQL ที่เป็นการเปรียบเทียบคำบางส่วน เช่น

OR ‘Simple LIKE ‘Sim%’

ตัวเลือกอื่นก็มีอย่างพวกตัวดำเนินการ “>”,”<”

OR ‘Simple’ > ‘S’

OR ‘Simple’ < ‘X’

OR 2 > 1

หรือผู้โจมตีอาจใช้ “IN” หรือ “BETWEEN” statements

OR ‘Simple’ IN (‘Simple’)

OR ‘Simple’ BETWEEN ‘R’ AND ‘T’

เมื่อเทคนิคหลบเลี่ยงใหม่ ๆ ถูกพัฒนาขึ้น Signature ที่รองรับการโจมตีนั้น ๆ ก็ถูกคิดค้นขึ้นตามด้วย แต่การพยายามเพิ่ม signature ให้ครอบคลุมทุก ๆ เทคนิคนั้นกลับทำให้ประสิทธิภาพของระบบลดลงและประมวลผลช้าอีกต่างหากและยังมีความเป็นได้หากให้ Match กับ SQL Keyword หรือ Meta Character จะเกิด False Positives เช่น URL ดังนี้

http://site/order.asp?ProdID=5&Quantity=4

เห็นได้ว่ากระบวนการตรวจจับด้วย Signature อาจไม่ได้แก้ปัญหาที่ตรงมากนัก

2) การเลี่ยง signature ด้วยช่องว่าง หลาย ๆ Signature จะรวมช่องว่างไปด้วย เพราะ String บางอย่าง เช่น “UNION SELECT” หรือ “EXEC SP_” จะต้องมีช่องว่างจึงจะทำงานได้ถูกต้อง ในตอนต้นได้มีการแนะนำการใช้ช่องว่างอย่างง่ายไปแล้ว แต่ควรเพิ่มเทคนิคเพื่อใช้กับ Signature ใหม่ ๆ โดยอาจไม่ใช้ช่องว่างเลยหรือเพิ่ม Character เข้าไป ตัวอย่างเช่น

...OrigText’ OR ‘Simple’ = ‘Simple’ อาจแทนด้วย ...OrigText’OR’Simple’='Simple’

จากที่แสดงข้างต้นทั้ง 2 แบบให้ค่าเท่ากันแต่แบบที่ล่างไม่มีช่องว่างทำให้หลบเลี่ยง Signature ได้ แต่วิธีนี้ใช้กับ Keyword อย่าง “UNION SELECT” ไม่ได้ วิธีที่เหมาะสมกับ Keyword แบบนี้ เช่นเดียวกับภาษาซี ถ้าต้องการให้ส่วนใดของโปรแกรมไม่แสดงผลการทำงานหรือใส่ comment ไว้ ก็ใช้สัญลักษณ์เริ่มด้วย “/*” และจบด้วย “*/” วิธีนี้ก็สามารถใช้ได้กับ SQL

SELECT *

FROM tblProducts /* List of Prods */

WHERE ProdID = 5

จากรูปแบบนี้เราก็สามารถ Injection code ได้ดังนี้

...&ProdID=2 UNION /**/ SELECT name ...

Signature จะพยายามตรวจจับ “UNION” ที่ตามด้วยช่องว่างแล้วตามด้วย “SELECT” ก็จะทำให้ไม่สามารถจับการโจมตีนี้ได้ ส่วนใหญ่กรณี “/**/” สามารถแทนที่ช่องว่างเพื่อหลีกเลี่ยง Signature ที่เรียกว่าอ่อนไหว (Sensitive) อย่าง “SELECT” หรือ “INSERT” SQL keyword ลักษณะนี้จะตามด้วย 1 ช่องว่าง จากตัวอย่างข้างบน ก็จะกลายเป็น

...&ProdID=2/**/UNION/**/SELECT/**/name...

เทคนิคนี้สามารถใช้กับ Oracle ได้ แม้ว่า Oracle นั้นจะไม่ยอมให้เว้นช่องว่างหลาย ๆ ช่องได้ แต่มันก็ยอมให้ใช้สัญลักษณ์ comment ดังนี้

...OrigText’/**/OR/**/’Simple’='Simple’

ส่วนเทคนิคต่อไปเป็นการหลบเลี่ยงเมื่อมีการส่งค่าพารามิเตอร์ 2 ค่าเข้าไปใน SQL เพื่อ Login ตัวอย่างเช่น

http://site/login.asp?User=X&Pass=Y

จาก Request นี้จะได้คำสั่ง Query ดังนี้
SELECT * FROM Users
WHERE User=’X’ AND Pass=’Y’

ในกรณีนี้เราสามารถใส่ comment เพื่อกำจัดไปพารามิเตอร์หนึ่งตัว แล้วใส่พารามิเตอร์อื่นเข้าไป ดังนี้

...login.asp?User=X'OR'1'/*&Pass=Y*/='1

Query ได้ผลลัพธ์ดังนี้

SELECT * FROM Users

WHERE User='X'OR'1'/* AND Pass='*/='1'

SQL Keyword อย่าง "SELECT" และ "INSERT" อาจนำมาทำเป็น Signature แต่หากพบปัญหาเช่นเดียวกับ "OR" ซึ่งก็คือ False Positive ถ้าอยู่ในหัวข้อ "Contact Us" ของเว็บไซต์ประเภท Ecommerce ลูกค้าอาจพิมพ์ว่า "I have selected the product, but then had a problem." ซึ่งไม่ใช่เป็นการโจมตีใด ๆ

3) การเลี่ยงด้วยรูปแบบของสตริง ส่วนอีกวิธีหนึ่งก็คือการแยก String ออกเป็นส่วน ๆ ซึ่งสามารถทำได้หลายรูปแบบเช่นกัน แบบแรกใช้วิธี Comment ของภาษาซี แม้ว่ามันจะไม่ทำงานเมื่อไปใช้แทนช่องว่างใน MySQL แต่มันก็สามารถแยกคำออกจากกันได้ เช่น

...UN/**/ION/**/ SE/**/LECT/**/...

ส่วนอีกวิธีที่ใช้การต่อ String (String Concatenation) ฐานใช้ข้อมูลส่วนใหญ่จะให้ User ใช้คำสั่ง Query โดยผ่านการส่ง String ไปยัง Server เราใช้วิธีนี้แฝงไปในสตริงที่ส่งออกไป Signature ตัวอย่าง ใน MS SQL ด้วยคำสั่ง EXEC ดังนี้

...; EXEC('INS'+'ERT INTO...')

โดย Keyword "INSERT" ถูกแยกเป็น 2 ส่วน ดังนั้น Signature ก็จะไม่สามารถจับได้ ยังมีตัวอย่างอื่น ๆ อีกที่คล้ายกัน คือ ใน MS SQL จะมี SP_EXECUTESQL เป็นรุ่นใหม่ของ SP_SQLEXEC ซึ่งทั้งสองนี้จะรับ String ที่มี SQL Query มาใช้ ทำให้ใช้วิธีนี้โจมตีได้ ในฐานข้อมูลอื่นก็พบปัญหาคล้าย ๆ กัน วิธีนี้ยังมีความน่าสนใจอีก คือ String ที่ใช้จะถูกเข้ารหัสเป็นฐาน 16 เพื่อไปประมวลผลอีกทีอย่าง "SELECT" จะได้ค่าเป็น "0X73656c656374" ซึ่งถ้าทำแบบนี้ signature ไม่มีทางตรวจเจอเลย อีกตัวอย่างของ MYSQL คือคำสั่ง OPENROWSET ที่อาจถูก String Concatenation Attack แฝงมาแม้การโจมตีนี้จะถูกพบแต่หลายอุปกรณ์ยังไม่สามารถใช้ signature ตรวจพบได้

เหตุผลที่การใช้ Signature ไม่มีประสิทธิภาพพอที่จะใช้กับ SQL Injection เนื่องจากการที่พยายามสร้าง Signature ให้ครอบคลุมทุกแบบของทำให้ประสิทธิภาพการโจมตีนั้น (Performance) ลดลงและเกิด False Positive

1) ประสิทธิภาพลดลง ภาษา SQL นั้นมีความยืดหยุ่นสูงผู้โจมตีสามารถเลี่ยง Signature ได้หลาย ๆ วิธี แม้ว่าเราจะมีฐานข้อมูล Signature เป็นจำนวนร้อย ๆ ต่อ 1 ฐานข้อมูลซึ่งทำให้มันตรวจจับการโจมตีได้ แต่องค์กรแต่ละแห่งไม่ได้มีแค่ Database ชนิดเดียวแน่ๆ คราวนี้ signature ที่ไว้ detect คงเกือบพัน แล้วทีนี้ performance (throughput & latency) ล่ะครับจะเป็นยังไง ลดลงอย่างฮวบฮาบแน่นอน คงจะไม่มีใครที่ยอมรับในเรื่องได้แน่เลย เพราะไม่งั้นก็ต้องมาเสียค่าใช้จ่ายในการเพิ่ม performance กันอีก

2) การเกิด False Positive อย่างใน MSSQL Server มี Keyword มากมาย เช่น SELECT, INSERT, CREATE, DELETE, FROM, WHERE, OR, AND, LIKE, EXEC, SP_, XP_, SQL, ROWSET, OPEN, BEGIN, END, DECLARE และ Meta Characters ได้แก่ ; — + ‘ ( ) = > < @ ถ้าคำเหล่านี้ถูกนำมาใช้เป็น Signature หมด ข้อมูลที่ User ส่งมาจะถูกบล็อกมากกว่าที่ผู้โจมตีโจมตีเข้ามาเสียอีก จึงจำเป็นต้องมีการขัดขวาง SQL Injection ด้วย SecureSphere Web Application Firewall โดยอุปกรณ์นี้รวมฐานข้อมูล Signature ของ IPS เข้ากับการเก็บข้อมูลเป็น Profile (Dynamic Profiling) และความรุนแรงของการโจมตีที่พบในแต่ละ Layer จะเชื่อมโยงกันโดยอัตโนมัติอย่างถูกต้องซึ่งการป้องกันโดย Signature อย่างเดียวไม่สามารถทำได้

SecureSphere Web Application Firewall นั้นถูกออกแบบมาเพื่อตรวจจับการรวมกันของ Character ที่จะสื่อถึง SQL Injection. ฐานข้อมูล signature เกี่ยวกับ SQL Injection ของ SecureSphere (ทุกชนิดฐานข้อมูล) จะมีการอัพเดตทุกสัปดาห์โดยทีมวิจัยของ Imperva (The Application Defense Center, ADC) ถ้าเพื่อน ๆ ยังจำได้จากตัวอย่างแรก ๆ เลย การโจมตีอย่างง่าย เช่น “or 1=1” SecureSphere IPS จะบล็อกทันที แต่ถ้าเป็นเทคนิคการเลี่ยง Signatureแบบต่าง ๆ นั้น SecureSphere จะอาศัยจากประสบการณ์ของมัน โดยขั้นแรก SecureSphere จะ Apply Signature ที่จำเป็นและไม่มากจนเกินไปนัก อย่างการเลี่ยงจาก Signature “or 1=1” ด้วย “Unusual = Unusual” ภัยคุกคามอันนี้จะถูกระบุได้โดย SecureSphere IPS signature จะมองหาการใช้ “or” และ “=” ร่วมกัน ภายใน URL parameter และ Signature Violation ก็จะแจ้งเตือนขึ้นมา แต่ถ้า “or” และ “=” เป็นคำทั่ว ๆ ไปใน parameter การ match บน signature นี้จะไม่ถูกบล็อกแต่อาจเกิด false positive เป็นบางครั้ง SecureSphere จะเก็บข้อมูลการใช้งานเพื่อเป็นหลักฐานไว้ใน Dynamic Profiling

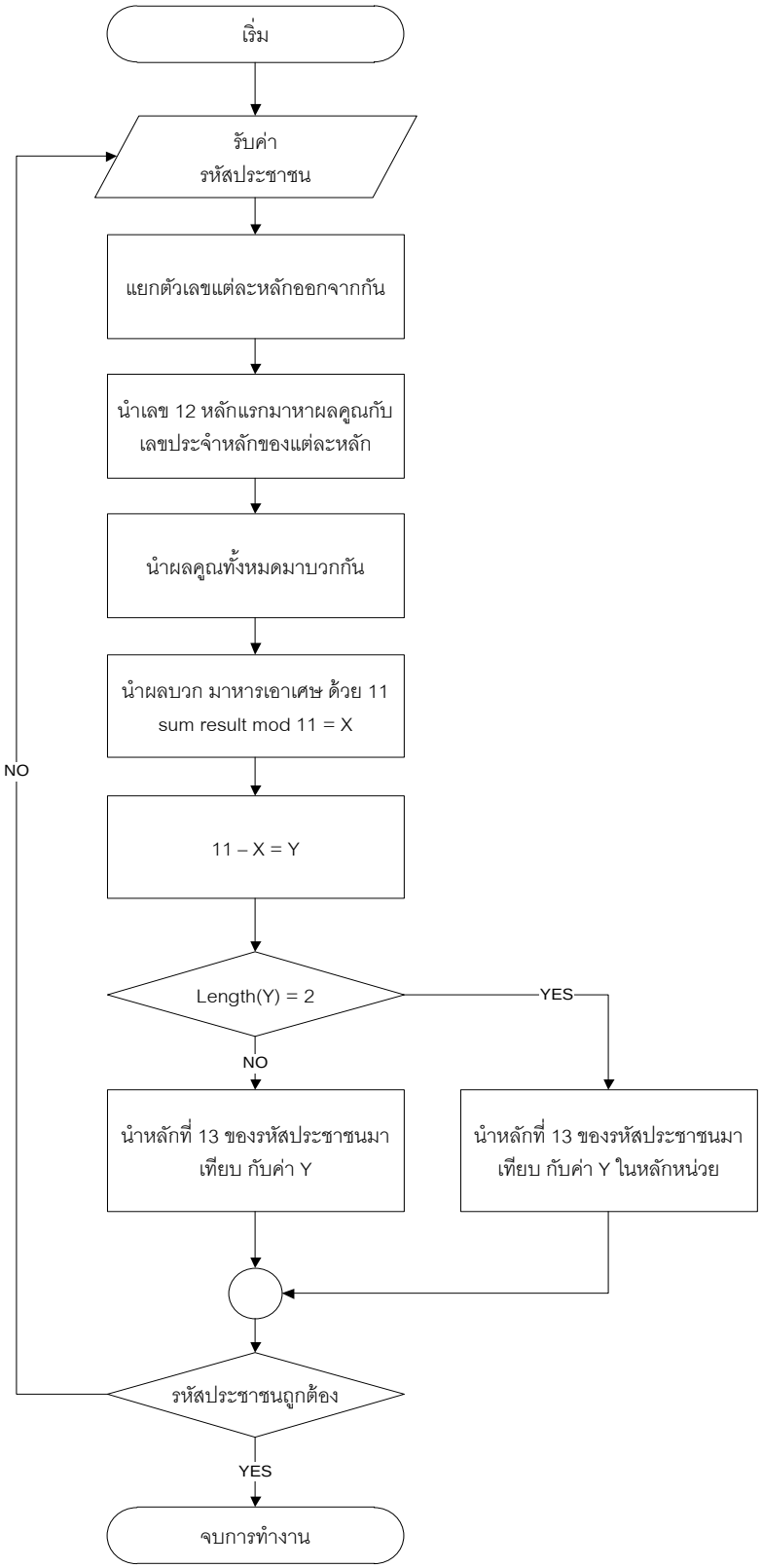
Dynamic Profiling ที่จะสามารถระบุพารามิเตอร์ที่ผิดปกติได้ โดยเทคโนโลยี Dynamic Profiling ของ SecureSphere จะพิจารณา Traffic อย่างอัตโนมัติแล้วสร้าง เป็น “Profile” ของเว็บไซต์นั้น ๆ โดยจะระบุแต่ละองค์ประกอบไว้ในโปรไฟล์ อาทิเช่น URLs, HTTP methods, Cookies, Parameter names, Parameter lengths, Parameter types และอื่น ๆ Profile จะทำการเก็บค่าเมื่อมีการใช้งาน Application และ SecureSphere นั้นสามารถตรวจจับพฤติกรรมการใช้งานเว็บไซต์และเข้าถึงฐานข้อมูลที่ผิดปกติได้ ดังนั้นเมื่อมีการใช้งานเว็บไซต์มากขึ้น SecureSphere จะมีอัลกอริทึมในการเรียนรู้และอัพเดตค่าที่เก็บในโปรไฟล์อัตโนมัติโดยเทียบจากพฤติกรรมส่วนใหญ่ที่ใช้เว็บไซต์นั้น ๆ และเรายังสามารถปรับค่าเองได้ด้วยถ้ามันเป็น False Positive ยกตัวอย่างเช่น เรื่องเกี่ยวกับเว็บเกี่ยวกับสุขภาพที่ต้องมีการส่งค่าพารามิเตอร์ sex ซึ่งอาจถูก SQL Injection ได้ เช่น “OR Unusual = Unusual” แต่เรารู้ว่า sex มีการส่งค่าเพียง 1 ตัวอักษร ถ้าเรากำหนดได้เราก็จะป้องกันการโจมตีนี้ได้

Dynamic Profile ยังใช้ในการตรวจสอบการโจมตีโดยถ้ามีเหตุการณ์ใดเกิดจาก User คนเดิมซ้ำ ๆ กัน จะทำให้อีกความสามารถของ SecureSphere คือ SecureSphere’s Correlated Attack Validation ทำงาน ซึ่ง Correlated Attack Validation (CAV) จะเชื่อมโยงการโจมตีหรือ Violation ต่าง ๆ ที่เกิดขึ้นจาก User คนเดียวกันที่ข้ามผ่านด่านความปลอดภัยเข้ามาได้ (IPS, Dynamic Profile ฯลฯ) ถ้า User คนหนึ่งโจมตีแบบเดิม ๆ หลาย ๆ ครั้ง เราก็จะมั่นใจได้เลยว่าเขาตั้งใจจะโจมตีจริง ๆ จากตัวอย่าง CAV เชื่อมโยง SQL Signature Violation กับ Parameter Length Violation เพื่อตรวจว่าเป็นการโจมตีแบบจริง ทำให้ CAV สามารถช่วยระบุการโจมตีได้อย่างแม่นยำเมื่อมีการใช้เทคนิคในการเลี่ยง Signature ขั้นสูง มันก็จะเชื่อมโยงทุก Violation ที่เกิดขึ้นแล้วระบุไปยัง User คนที่โจมตีมาได้

ระบบได้จัดทำส่วนการพิสูจน์สิทธิตนหลังจากการเข้าสู่ระบบอย่างถูกต้องของผู้ใช้งานแล้วจะมีการให้ค่า Session สำหรับผู้ใช้งานแต่ละคนโดยที่ค่าจะไม่ซ้ำกัน และค่าจะบ่งบอกถึงสิทธิในการใช้งานบริการและส่วนต่าง ๆ ของเว็บไซต์จำแนกตามประเภทของผู้ใช้คือ ผู้ใช้งานทั่วไป นักวิจัยหรือสมาชิก และ Admin โดยในเว็บไซต์ส่วนระบบสมาชิกนั้น หลังจากที่ผู้ใช้ล็อกอินเข้าสู่ระบบได้สำเร็จเว็บไซต์จะสร้าง Session ID ขึ้นมาเพื่อใช้เป็นตัวอ้างอิงถึงผู้ใช้คนนั้น Session ID จะเป็นค่า String ยาว ๆ ที่มักจะถูกสร้างขึ้นโดยใช้เทคนิคการสุ่มตัวอักษรผสมกับตัวเลข และทุกครั้งที่ผู้ใช้มีการส่ง HTTP Request ไปยังเว็บเซิร์ฟเวอร์เพื่อขอหน้าเว็บเพจ ใน HTTP Request จะมี Cookie และข้อมูลใน Cookie จะมี Session ID อยู่ภายใน เมื่อเว็บไซต์ปลายทางได้รับ HTTP Request ก็จะทราบว่าผู้ใช้คนใดเป็นร้องขอ โดยเว็บไซต์จะตรวจสอบจาก Session ID ที่อยู่ภายใน Cookie จากนั้นก็จะประมวลผลและตอบกลับด้วย HTTP Response ซึ่งมีข้อมูลที่ผู้ใช้คนนั้นต้องการ

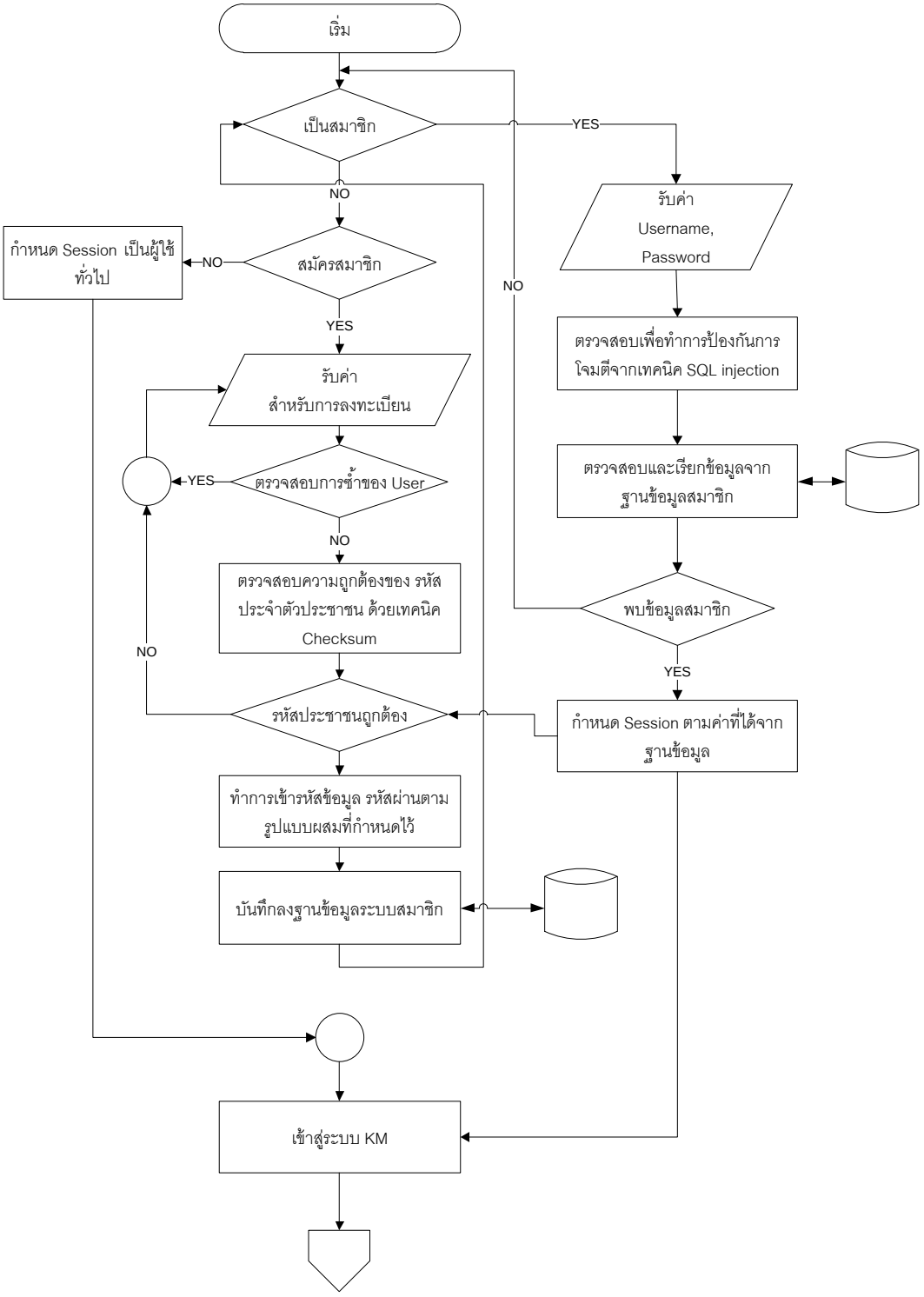
การส่ง Session ID จากบราวเซอร์ไปยังเว็บไซต์ นอกจากส่งโดยใช้ Cookie แล้ว ยังสามารถส่งทางอื่นได้อีก เช่น ส่งทางฟิลด์ที่ซ่อน (Hidden Field) หรือส่ง Session ID ทาง URL ก็มี การส่ง Session ID ทาง Hidden Field นั้น ผู้โจมตีสามารถเปลี่ยนค่าในฟิลด์ซ่อนได้โดยตรง

2.3 การดำเนินการวิจัยส่วนของการตรวจพิสูจน์ตัวตน (Authentication) ดังรูปที่ 3.3

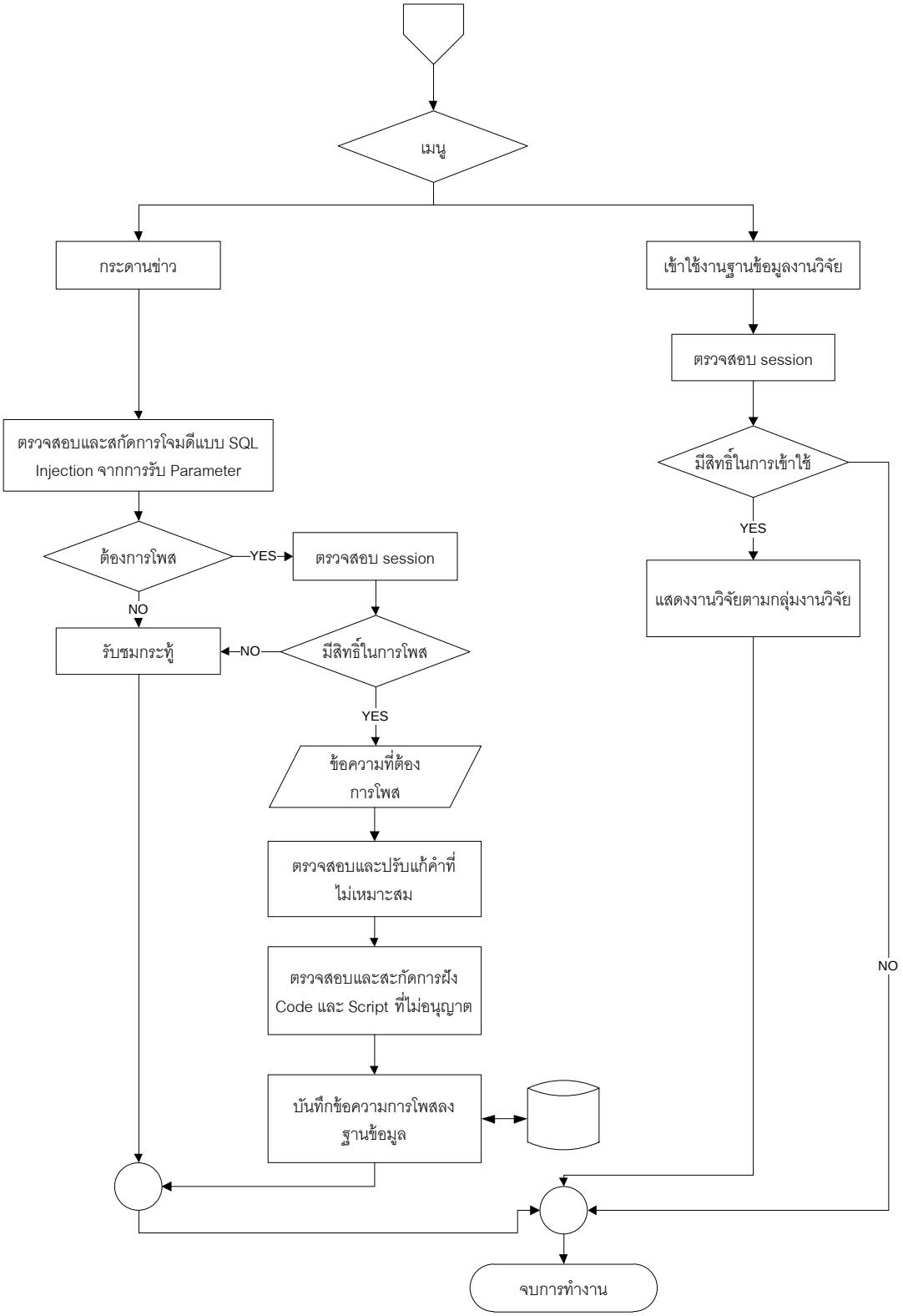


รูปที่ 3.3 การพิสูจน์ตัวตนโดยใช้เลขบัตรประชาชน

ภาพรวมเชิงเทคนิคระบบรักษาความปลอดภัย แสดงดังรูปที่ 3.4



รูปที่ 3.4 ผังโปรแกรมภาครวมเชิงเทคนิคของระบบรักษาความปลอดภัย



รูปที่ 3.4 ผังโปรแกรมภาครวมเชิงเทคนิคของระบบรักษาความปลอดภัย (ต่อ)

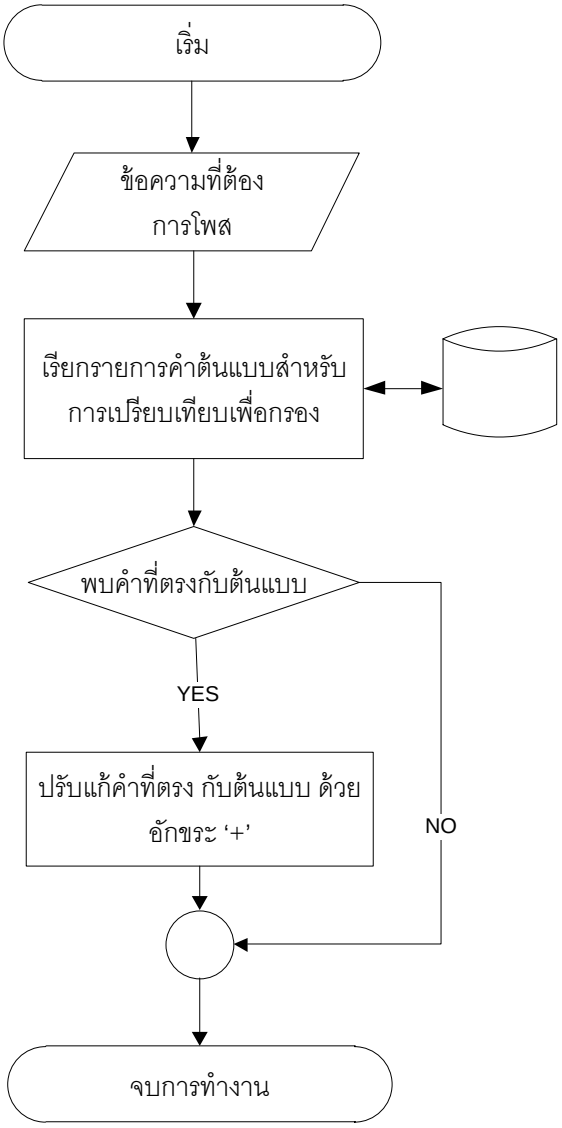
จากรูปที่ 3.4 มีการออกแบบและสร้างรูปแบบการเข้ารหัสผ่านให้ยากต่อการโจมตี ซึ่งมีการโจมตีแบบต่าง ๆ คือ การโจมตีโดยแอบอ้างสิทธิและเข้ามาล้วงข้อมูล รวมถึงกรณีที่ผู้โจมตีเข้ามาแล้วและล้วงข้อมูลของระบบจัดการความรู้กลุ่มงานวิจัยไปแล้วก็ตาม ซึ่งการรักษาความปลอดภัยยังจัดทำวิธีการที่ผู้โจมตีจะไม่สามารถนำข้อมูลไปใช้ประโยชน์ได้ เนื่องจากข้อมูลที่ได้มีการเข้ารหัสไว้

ขั้นที่ 3 นำต้นแบบมาใช้ โดยจัดทำ Penetration Test คือ การทดลองเจาะระบบเว็บไซต์ การจัดการความรู้กลุ่มงานวิจัย เพื่อตรวจสอบการบุกรุกทุกรูปแบบที่ผู้วิจัยได้ใส่ระบบรักษาความปลอดภัยไว้ให้กับกลุ่มงานวิจัยกลุ่มพลังงานทดแทน

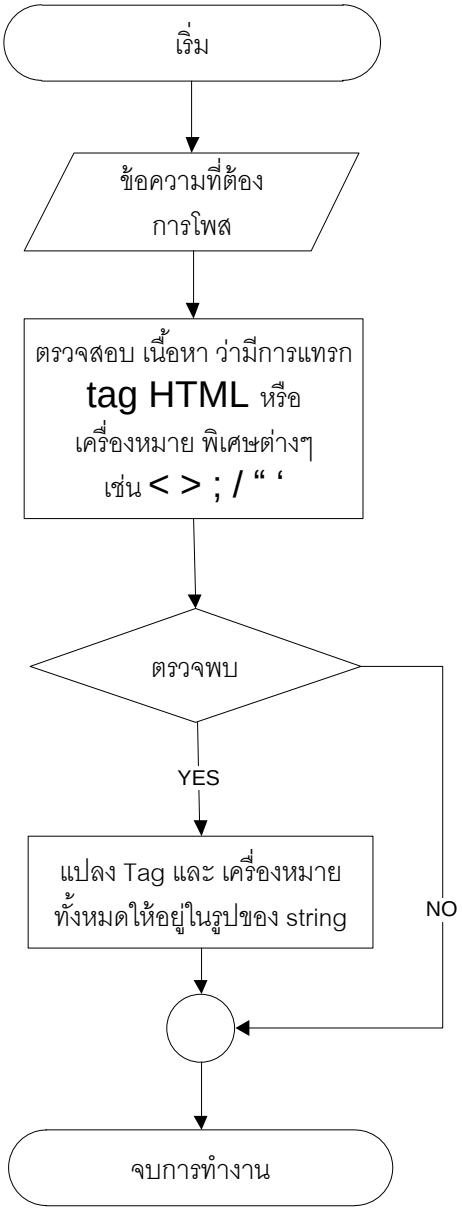
ขั้นที่ 4 ปรับปรุงแก้ไขต้นแบบในขั้นตอนที่ 3 แล้ววนกลับไปในขั้นตอนที่ 3 จนกระทั่งได้รับการยอมรับ จากนั้นดำเนินการนำต้นแบบไปใช้กับทุก ๆ กลุ่มของงานวิจัย

3.2 การดำเนินการรักษาความปลอดภัยทุกกลุ่มงานวิจัยของระบบจัดการความรู้กลุ่มงานวิจัย มหาวิทยาลัยเทคโนโลยีราชมงคลธัญบุรี

ผู้วิจัยได้ดำเนินการนำโปรแกรมเทคนิคการตรวจพิสูจน์ตัวตนด้วยหมายเลขบัตรประชาชนก่อนเข้าสู่ระบบของกลุ่มงานวิจัยแต่ละกลุ่ม และดำเนินการนำโปรแกรมเทคนิคการตรวจสอบสิทธิรวมทั้งการป้องกันทุกรูปแบบใส่ในระบบจัดการความรู้กลุ่มงานวิจัยมหาวิทยาลัยเทคโนโลยีราชมงคลธัญบุรี



รูปที่ 3.5 ลำดับการทำงานในส่วนของการกรองคำไม่เหมาะสม



รูปที่ 3.6 ลำดับการทำงานในส่วนการสกัดการฝัง script และ code ที่ไม่ได้รับอนุญาต