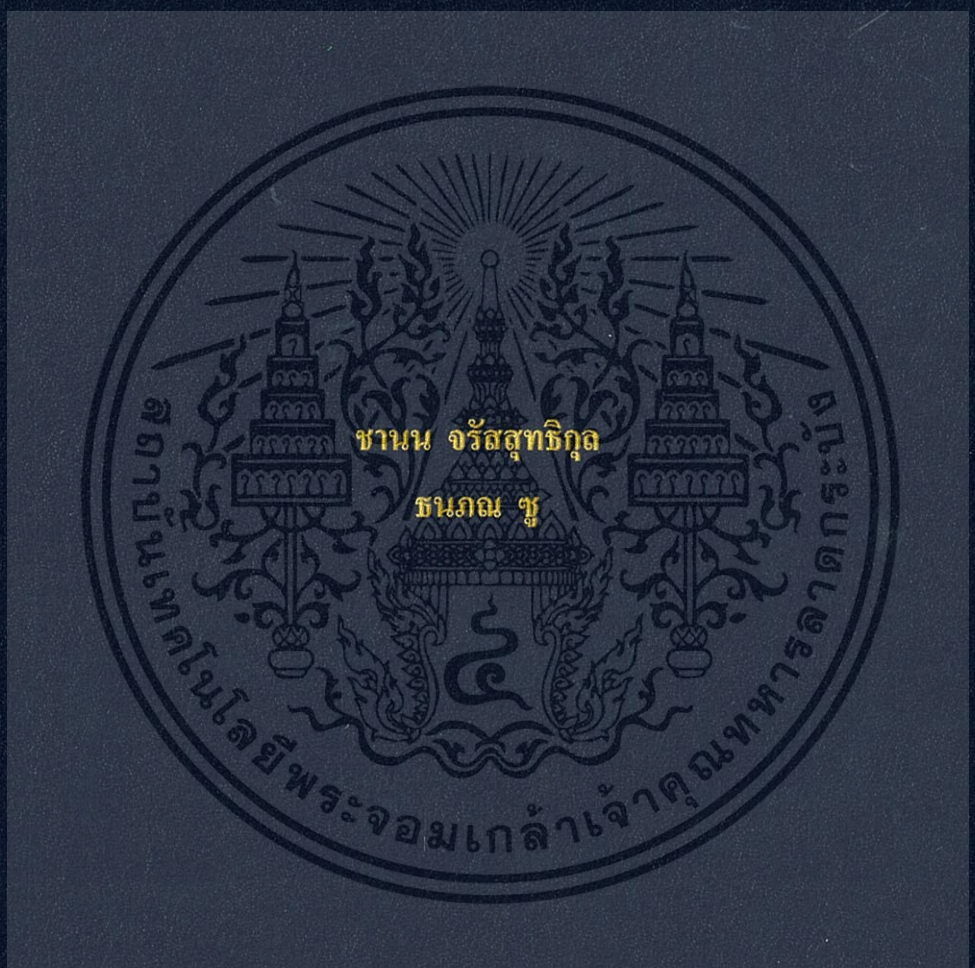
ระบบวิเคราะห์และตรวจจับการบุกรุกสำหรับศูนย์ข้อมูล

LIFE: SCALABLE LOG ANALYSIS AND INTRUSION DETECTION SYSTEM



ชานน จรัสสุทธิกุล

ธนภณ ซู

ปริญญานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิศวกรรมศาสตรบัณฑิต

ภาควิชาวิศวกรรมคอมพิวเตอร์ คณะวิศวกรรมศาสตร์

สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

ปีการศึกษา 2558

ระบบวิเคราะห์และตรวจจับการบุกรุกสำหรับศูนย์ข้อมูล

LIFE: SCALABLE LOG ANALYSIS AND INTRUSION DETECION SYSTEM

ชานน จรัสสุทธิกุล

ธนภณ ซู

ปริญญานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิศวกรรมศาสตรบัณฑิต
ภาควิชาวิศวกรรมคอมพิวเตอร์ คณะวิศวกรรมศาสตร์
สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง
ปีการศึกษา 2558

ปริญญานิพนธ์ปีการศึกษา 2558

ภาควิชาวิศวกรรมคอมพิวเตอร์

คณะวิศวกรรมศาสตร์ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

เรื่อง ระบบวิเคราะห์และตรวจจับการบุกรุกสำหรับศูนย์ข้อมูล

LIFE: SCALABLE LOG ANALYSIS AND INTRUSION DETECTION SYSTEM

ผู้จัดทำ

1.นายชานน จรัสสุทธิกุล รหัสนักศึกษา 55010280

2. นายธนภณ ซู รหัสนักศึกษา 55010496

อาจารย์ที่ปรึกษา

(ดร.อรทัย สังข์เพ็ชร)

อาจารย์ที่ปรึกษา

(ดร.อักฤทธิ์ สังข์เพ็ชร)

ระบบวิเคราะห์และตรวจจับการบุกรุกสำหรับศูนย์ข้อมูล

นายชานน จรัสสุทธิกุล 55010280
นายธนภณ ซู 55010496
ดร.อรทัย สังข์เพ็ชร อาจารย์ที่ปรึกษา
ดร.อักฤทธิ์ สังข์เพ็ชร อาจารย์ที่ปรึกษาร่วม
ปีการศึกษา 2558

บทคัดย่อ

ในการตรวจจับการบุกรุกระบบด้วยการวิเคราะห์แฟ้มบันทึกข้อมูลการใช้งาน ผู้ดูแลระบบจะต้องเข้าใจความหมายและวิเคราะห์ข้อมูลจากหลายแฟ้มข้อมูล เช่น แฟ้มข้อมูลการเข้าใช้งานระบบ แฟ้มข้อมูลสถานะระบบ เป็นต้น ซึ่งวิธีนี้ไม่สามารถทำได้อย่างทันท่วงทีเวลาที่มีเครื่องคอมพิวเตอร์จำนวนมาก โครงงานนี้จึงได้พัฒนาระบบวิเคราะห์แฟ้มบันทึกข้อมูลอัตโนมัติเพื่อค้นหาหลักฐานการโจมตีระบบโดยใช้วิธีทางสถิติ หรือการทำเหมืองข้อมูล หรือการเรียนรู้ของเครื่อง โดยโครงงานนี้จะมุ่งเน้นการตรวจจับการโจมตีแบบบ SSH brute-force และการโจมตีแบบ SQL injection นอกจากนี้ระบบถูกออกแบบให้ปรับขยายปริมาณการประมวลผลให้รองรับกับจำนวนข้อมูลที่จะเติบโตในอนาคตได้

LIFE: Scalable Log Analysis and Intrusion Detection System

Mr. Chanon	Jaratsuttikul	55010280
Mr. Thanaphon	Soo	55010496
Dr. Orathai	Sangpetch	Advisor
Dr. Akkarit	Sangpetch	Co-Advisor

ABSTRACT

In order to identify potential threats in a data center, operators or administrators need to manually review and analyze a variety of log files, such as access logs and message logs. This approach does not scale well with hundreds or thousands of computers. The longer it takes to perform log analysis, the more damages an attack can cause. Therefore, we propose LIFE, an intrusion detection system, which automatically analyzes log files using various analytical methods, such as statistical analysis, data mining or machine learning algorithms. In this project, we will focus on brute-force attack and SQL injection attack, which are prevalent in current applications. Moreover, LIFE needs to scale with the growth of log data in the data center.

กิตติกรรมประกาศ

รายงานระบบวิเคราะห์และตรวจจับการบุกรุกสำหรับศูนย์ข้อมูลฉบับนี้สำเร็จได้ด้วยความอนุเคราะห์ของบุคคลเหล่านี้

ดร.อรทัย สังข์เพ็ชร(อาจารย์ที่ปรึกษา) และ ดร.อักฤทธิ์ สังเพ็ชร(อาจารย์ที่ปรึกษาร่วม) เป็นผู้ให้คำแนะนำ คำปรึกษา แนวทางการปฏิบัติงาน และให้ความช่วยเหลือตลอดการทำงาน

รศ.ดร.เกียรติกูล เจียรนัยธนะกิจ และ ผศ.ดร.ชุติเมษฎ์ ศรีนิลทา อาจารย์ประจำภาควิชาที่ให้คำปรึกษาและความรู้อันเป็นประโยชน์ในการทำงาน

นายวิษุวัต ซันเฮม เพื่อนสนิทที่ให้คำแนะนำ และกำลังใจในการทำงาน

กลุ่มนักศึกษาสังกัดสำนักบริการคอมพิวเตอร์ CSAG (Computer System Administrator Group) ที่ให้โอกาสเก็บเกี่ยวความรู้ และประสบการณ์ชีวิต เพื่อเป็นพื้นฐานในการทำโครงงาน

ห้องวิจัย SOUP (System, Operation, Usability and Parallel computing Laboratory) ที่ได้เอื้อเฟื้อสถานที่และทรัพยากรต่างๆในการทำวิจัยและพัฒนาโครงงาน

บิดา มารดา และครอบครัวที่ได้เลี้ยงดูและสั่งสอน ให้กำลังใจ พร้อมทั้งให้โอกาสทางการศึกษา

คณะผู้จัดทำขอขอบพระคุณเป็นอย่างสูง และหวังว่ารายงานฉบับนี้จะเป็นประโยชน์ต่อทุกท่าน

ชานน จรัสสุทธิกุล

ธนภณ ซู

สารบัญ

หน้า

สารบัญ (ต่อ)

หน้า

สารบัญตาราง

สารบัญตาราง (ต่อ)

ตาราง **หน้า**

สารบัญรูป

บทที่ 1
บทนำ

1.1 ความเป็นมาของปัญหา

แฟ้มบันทึกข้อมูลของเครื่องคอมพิวเตอร์ จะเก็บบันทึกข้อมูลการทำงานของระบบ ซึ่งแฟ้มบันทึกข้อมูลถูกบันทึกเก็บไว้ตลอดเวลาบนเครื่องคอมพิวเตอร์ เป็นเหตุให้แฟ้มข้อมูลในศูนย์ข้อมูลมีจำนวนมากและผู้ดูแลระบบไม่สามารถวิเคราะห์การบุกรุกได้อย่างทันท่วงที ยิ่งใช้เวลาในการแก้ปัญหานานมากเท่าไร ความสูญเสียที่เกิดขึ้นและผลกระทบด้านความปลอดภัยก็จะยิ่งมากขึ้นเท่านั้น อีกทั้งผู้ดูแลระบบต้องมีความเข้าใจและความสามารถในการวิเคราะห์แฟ้มบันทึกข้อมูลทาง คณะผู้จัดทำจึงได้พัฒนาระบบตรวจจับการบุกรุก 2 ประเภท คือ การตรวจจับการโจมตี SSH brute-force และการโจมตีแบบ SQL injection

1.1.1 SSH Brute Force

การโจมตีที่เกิดขึ้นบนพอร์ต 22 ซึ่งเป็นพอร์ตค่าเริ่มต้นของบริการ SSH ถูกรายงานว่าเกิดขึ้นมากเป็นอันดับสอง รองจากการโจมตีที่เกิดขึ้นบนพอร์ต 80 รูปแบบการโจมตีที่พบเห็นได้บ่อยที่สุดคือการตรวจจับการโจมตีแบบ brute-force [1] [2]

1.1.2 SQL injection

จากรายงาน The Open Web Application Security Project (OWASP) Top 10 -2013 พบว่าการโจมตีประเภท injection คำสั่งมีความร้ายแรงมากที่สุด (A1 OWASP Top-10) โดยพิจารณาจากโอกาสสำเร็จในการโจมตี, ความเป็นไปได้ในการตรวจพบช่องโหว่และผลกระทบที่เกิดขึ้นจากการโจมตี [3]

แม้จะมีเครื่องมือในการตรวจจับการโจมตีแบบ SSH brute-force และการโจมตีแบบ SQL injection แต่ก็มีความยุ่งยากในการกำหนดค่าต่างๆให้เหมาะสม และหากผู้ดูแลระบบกำหนดค่าเครื่องมือที่ไม่เหมาะสมจะทำให้เครื่องมือไม่มีประสิทธิภาพในการตรวจจับและผู้บุกรุกสามารถหาช่องทางหลีกเลี่ยงการตรวจจับได้

ด้วยเหตุนี้ทางคณะผู้จัดทำจึงได้เล็งเห็นถึงปัญหาการโจมตี SSH brute-force และการโจมตีเว็บแอปพลิเคชันด้วย SQL injection โดยใช้วิธีทำเหมืองข้อมูล (Data Mining), การเรียนรู้ของเครื่อง (Machine Learning) หรือ ข้อมูลทางสถิติ

1.2 วัตถุประสงค์ของโครงงาน

1) สร้างระบบตรวจจับการบุกรุกประเภท SSH Brute Force โดยใช้การเรียนรู้ของเครื่อง เช่น Support Vector Machine
2) สร้างระบบตรวจจับการบุกรุกประเภท SQL Injection โดยใช้ วิธีการทางสถิติ หรือ การเรียนรู้ของเครื่อง
3) สร้างส่วนแสดงผลลัพธ์ที่เกิดจากการทำงานของระบบที่ได้จากข้อ 1และ ข้อ 2
4) สร้างระบบจากข้อ 1 และ ข้อ 2 ให้สามารถปรับขยายปริมาณการประมวลผลให้รองรับจำนวนข้อมูลที่จะเติบโตในอนาคตได้

1.3 ขอบเขตของโครงงาน

1) ระบบสามารถช่วยผู้ดูแลระบบตรวจจับการโจมตีประเภท SSH Brute Force จากแฟ้มบันทึกข้อมูล
2) ระบบสามารถช่วยผู้ดูแลระบบตรวจจับการโจมตีเว็บแอปพลิเคชันประเภท SQL Injection จากแฟ้มบันทึกข้อมูล
3) ระบบในข้อ 1 ประกอบด้วยแบบจำลองที่ได้มาจากการทำ Data Mining, Machine Learning หรือ ข้อมูลทางสถิติ
4) ระบบในข้อ 1 ทางคณะผู้จัดทำจะศึกษาอัลกอริทึมดังนี้
   4.1) Logistic Regression
   4.2) Neural Network
   4.3) Support Vector Machine
5) ระบบในข้อ 1 และข้อ 2 สามารถประมวลผลแฟ้มบันทึกข้อมูลจากเครื่องหลายๆเครื่องได้
6) ระบบในข้อ 1 และข้อ 2 สามารถปรับขยายปริมาณการประมวลผลให้รองรับจำนวนข้อมูลที่อาจจะเติบโตในอนาคตได้
7) ระบบในข้อ 1 และข้อ 2 สามารถแสดงผลผ่าน Web browser ได้

1.4 ประโยชน์ที่คาดว่าจะได้รับ

โครงงานนี้สามารถนำไปใช้ในการตรวจจับการบุกรุกประเภท Brute Force SSH และสามารถตรวจจับการบุกรุกเว็บแอปพลิเคชันประเภท SQL Injection โดยผู้ดูแลระบบไม่ต้องกำหนดค่าต่างๆ ก่อนการใช้งาน อีกทั้งโครงงานนี้ยังสามารถรองรับปริมาณแฟ้มข้อมูลจำนวนมากจากเครื่องคอมพิวเตอร์หลายๆเครื่อง และสามารถนำแฟ้มข้อมูลมาประมวลผลเพื่อตรวจจับการบุกรุกได้ทันท่วงที

บทที่ 2
ทฤษฎีที่เกี่ยวข้อง

2.1 SSH Brute-Force Attack

SSH (Secure Shell) คือ วิธีที่ใช้ในการติดต่อสื่อสารระหว่างเครื่องคอมพิวเตอร์อย่างปลอดภัย เนื่องจากทุกๆการเชื่อมต่อจะถูกเข้ารหัสข้อมูล

SSH Brute-Force เป็นการโจมตีโดยสุ่มข้อมูลที่ใช้ในการเชื่อมต่อ SSH [4]

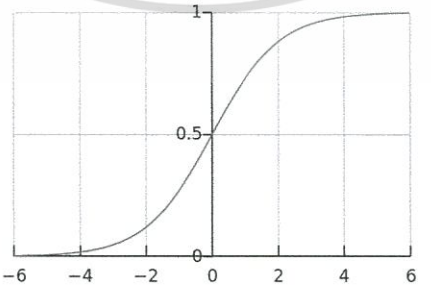
2.2 การทำเหมืองข้อมูล (Data Mining)

การเหมืองข้อมูลเป็นกระบวนการของการวิเคราะห์ข้อมูลจากแง่มุมต่างๆและสรุปข้อมูลเหล่านั้นออกมาในรูปแบบที่สามารถนำไปใช้ประโยชน์ได้

Data Mining Software เป็นหนึ่งในเครื่องมือที่ช่วยในการวิเคราะห์ข้อมูล โดย Data Mining Software ช่วยให้ผู้ใช้งานวิเคราะห์ข้อมูลจากหลายมิติ หรือหลายมุม แล้วจัดจำแนกข้อมูล สรุปออกมาในรูปแบบของความสัมพันธ์ตามที่ผู้ใช้กำหนด ในทางเทคนิค Data mining เป็นกระบวนการในการหาความสอดคล้องหรือรูปแบบของคุณลักษณะต่างๆ ในฐานข้อมูลขนาดใหญ่ [5]

2.2.1 การวิเคราะห์การถดถอยแบบโลจิสติก (Logistic Regression)

เป็นแบบจำลองการถดถอยชนิดหนึ่ง ที่มีตัวแปรตามเป็นข้อมูลที่มีลักษณะเป็นหมวดหมู่ ต่างจากการวิเคราะห์การถดถอยแบบอื่นๆที่มีตัวแปรตามเป็นค่าแบบต่อเนื่อง โดยความสัมพันธ์ระหว่างตัวแปรต้นและตัวแปรตามในการวิเคราะห์การถดถอยแบบโลจิสติกจะเป็นฟังก์ชันแบบโลจิสติก



รูป 2.1 ความสัมพันธ์แบบฟังก์ชันโลจิสติก

ในกรณีที่มีตัวแปรอิสระเพียงตัวเดียว เขียนสมการของการถดถอยโลจิสติกได้ดังนี้

$$p(y) = \frac{e^{\beta_0+\beta_1 x}}{1+e^{\beta_0+\beta_1 x}} = \frac{1}{1+e^{-(\beta_0+\beta_1 x)}} \quad (2.1)$$

เมื่อ $p(y)$ เป็น ความน่าจะเป็นของการเกิดเหตุการณ์ y

β_0และ β_1 เป็นสัมประสิทธิ์ที่ประมาณได้จากข้อมูล

จากสมการข้างต้น เมื่อมีตัวแปรอิสระมากกว่าหนึ่งตัว สามารถเขียนสมการใหม่ได้เป็น

$$p(y) = \frac{1}{1+e^{-z}} \quad (2.2)$$

เมื่อ $z = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \cdots \beta_n x_n$

การถดถอยโลจิสติกสามารถนำมาใช้ในการจำแนกข้อมูลออกเป็นสองกลุ่มได้เช่น กลุ่ม A และ B ด้วยการกำหนดค่า t โดย

$$\text{ข้อมูล } x \text{ อยู่ในกลุ่ม } \begin{cases} A, & p(y) \geq t \\ B, & p(y) < t \end{cases} \quad (2.3)$$

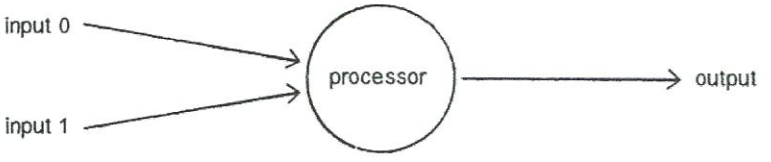
[6]

2.2.2 โครงข่ายประสาทเทียม (Artificial Neural Network)

โครงข่ายประสาทเทียม ประกอบด้วย เซลล์ประสาท(Neuron) เป็นเซลล์หนึ่งเซลล์ที่อยู่ในโครงข่ายประสาทที่รับข้อมูลนำเข้า จากนั้นก็จะประมวลผลข้อมูลนำเข้านั้น แล้วส่งผลลัพธ์ออกมา

แบบจำลองที่ง่ายที่สุดของโครงข่ายประสาทเทียม คือ 1 Perceptron ซึ่งเป็นแบบจำลองของ 1 Neuron Perceptron นั้นประกอบไปด้วย

1) ส่วนของข้อมูลนำเข้า
2) ส่วนของการประมวลผล
3) ส่วนของผลลัพธ์ของการประมวลผล ดังรูป



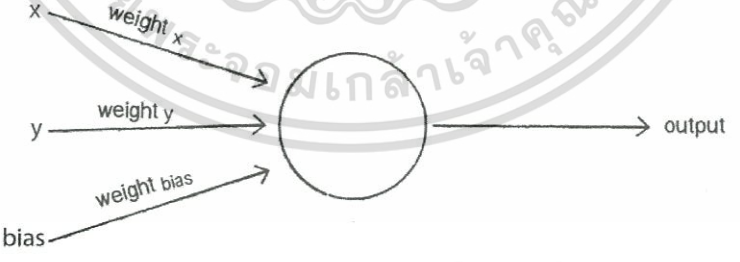
รูป 2.2 Perceptron ซึ่งเป็นแบบจำลองที่ง่ายที่สุดของโครงข่ายประสาทเทียม

Perceptron จะใช้รูปแบบที่เรียกว่า feed-forward กล่าวคือ ส่วนของข้อมูลนำเข้าจะส่งข้อมูลที่รับแล้วส่งไปให้ส่วนของการประมวลผลข้อมูล เพื่อทำการประมวลผลข้อมูล แล้วส่งผลลัพธ์ออกมา

สรุปขั้นตอนการทำงานของ perceptron

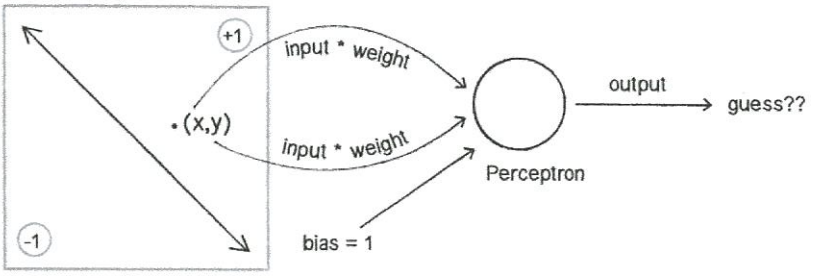
1) ทุกๆข้อมูลนำเข้าจะคูณด้วยน้ำหนักซึ่งส่วนใหญ่น้ำหนักจะมีค่าระหว่าง -1 ถึง 1
2) รวมผลลัพธ์ที่ได้จากการคูณในข้อ 1)
3) ประมวลผลที่ได้จากข้อ 2.) โดยนำมาผ่านฟังก์ชันชนิดหนึ่งที่เรียกว่า activation function
4) ผลลัพธ์จาก activation function มีค่า -1 หรือ -1

เนื่องจากข้อมูลนำเข้าที่รับเข้ามามีโอกาสที่จะเป็น 0 ทั้งหมด เช่น การนำค่าข้อมูลนำเข้าจากจุด (0,0) ซึ่งจะได้ข้อมูลนำเข้าที่เป็น 0 เมื่อนำมาคูณด้วยน้ำหนักแล้วหาผลรวมย่อมได้ 0 เพราะฉะนั้นน้ำหนักที่นำมาคูณจะไม่มีผลกระทบแต่อย่างใด ด้วยเหตุนี้จึงมีการกำหนดค่า Bias ขึ้นมาเพื่อป้องกันปัญหาที่อาจจะเกิดขึ้นดังกล่าว โดยทั่วไปค่า Bias ค่าเท่ากับ 1 นำมาคูณกับค่าน้ำหนักของ Bias แล้วนำผลลัพธ์มารวมกับผลรวมของการคูณน้ำหนักกับข้อมูลนำเข้า ดังแสดงในรูป



รูป 2.3 การนำค่า Bias มาคำนวณกับข้อมูลนำเข้าเพื่อแก้ปัญหากรณีข้อมูลนำเข้าเป็น 0

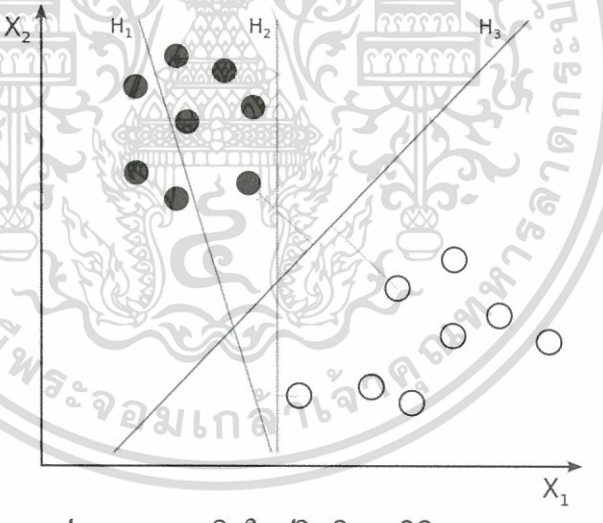
เพื่อให้ได้ค่าน้ำหนักที่เหมาะสมกับค่าข้อมูลนำเข้า Perceptron จึงใช้หลักการ backpropagation โดยจะหาค่าความผิดพลาดจากการเรียนรู้ แล้วส่งค่าความผิดพลาดนั้นย้อนกลับไปยังเครือข่ายเพื่อปรับค่า ในท้ายที่สุดก็จะได้ Perceptron ที่สามารถรับข้อมูลนำเข้าและประมวลผลลัพธ์ได้ ดังรูป



รูป 2.4 การทำงานของ Perceptron ตั้งแต่การรับข้อมูลนำเข้าจนกระทั่งประมวลผลลัพธ์ [7]

2.2.3 Support Vector Machine (SVM)

SVM เป็นอัลกอริทึมการเรียนรู้ของเครื่องแบบมีผู้สอน สามารถนำมาใช้ในการจำแนกประเภทข้อมูล เมื่อกำหนดชุดข้อมูลเรียนรู้ที่ข้อมูลแต่ละตัวอย่างถูกกำหนดให้อยู่ในกลุ่มใดกลุ่มหนึ่งจากกลุ่มจำนวนสองกลุ่ม อัลกอริทึมในการเรียนรู้ของ SVM จะทำการสร้างแบบจำลองที่สามารถทำการจำแนกกลุ่มให้กับข้อมูลชุดใหม่ที่ไม่เคยเห็นมาก่อน โดยการหาระนาบเกิน (hyperplane) ที่มีระยะห่างของการแบ่งแยกระหว่างกลุ่มสองกลุ่มมากที่สุดจากชุดข้อมูลเรียนรู้



รูป 2.5 ระนาบเกินในปริภูมิสองมิติ

จากตัวอย่างในรูปที่ 2.5 ระนาบเกิน H_1 ไม่สามารถแยกกลุ่มข้อมูลได้ ระนาบเกิน H_2 สามารถแบ่งกลุ่มข้อมูลได้ แต่มีระยะห่างจากกลุ่มข้อมูลทั้งสองที่น้อย ในขณะที่ระนาบเกิน H_3 สามารถแบ่งกลุ่มข้อมูลได้เช่นกันและมีระยะห่างจากกลุ่มข้อมูลทั้งสองมากที่สุดด้วย [8]

2.2.4 Hidden Markov Model

เป็นแบบจำลองที่เหมาะสำหรับจำลองข้อมูลที่มีลักษณะต่อเนื่องกันเป็นลำดับ ตัวอย่างงานที่นิยมนำ hidden Markov model มาใช้ได้แก่ การรู้จำเสียง การรู้จำลายมือ เป็นต้น

Hidden Markov model สามารถอธิบายด้วยองค์ประกอบ 5 อย่าง ได้แก่

Q = เซ็ตของ state = $\{q_1, q_2, \dots, q_n\}$ (2.4)

V = เซ็ตของ $output\ alphabet = \{v_1, v_2, \dots, v_n\}$ (2.5)

$\pi(i)$ = ความน่าจะเป็นที่จะอยู่ใน *state* q_i เมื่อเวลา $t = 0$ (2.6)

$A = \{a_{ij}\}$ ความน่าจะเป็นขแงการเปลี่ยนแปลง state = $\{a_{ij}\}$ (2.7)

โดย $a_{ij} = \Pr(state = q_j, t+1 \mid state = q_i, t)$ (2.8)

B = ความน่าจะเป็นของ output alphabe ในแต่ละ state = $\{b_j(k)\}$ (2.9)

โดย $b_j(k) = \Pr(output\ alphabet = v_k | state = , t)$ (2.10)

Hidden Markov Model สามารถหาความน่าจะเป็นที่จะเกิดผลลัพธ์จากแบบจำลองด้วยการหาผลรวมของความน่าจะเป็นจากเส้นทางที่เป็นไปได้ทั้งหมด ตัวอย่างเช่น เราสามารถหาความน่าจะเป็นของการเกิดผลลัพธ์ ab จากแบบจำลองในรูป 2.7 ดังนี้

$\Pr(a\|q_1)$	0.5
$\Pr(b\|q_1)$	0.5

$\Pr(a\|q_2)$	0.7
$\Pr(b\|q_2)$	0.3

รูป 2.6 แบบจำลอง Hidden Markov

$$\begin{aligned} Pr(ab) = &(0.7 * 0.5 * 0.8 * 0.5) + \\ &(0.7 * 0.5 * 0.2 * 0.3) + \\ &(0.3 * 0.7 * 0.6 * 0.3) + \\ &(0.3 * 0.7 * 0.4 * 0.5) \\ = &\ 0.2408 \end{aligned} \quad (2.11)$$

ในการนำ Hidden Markov Model มาทำการจำแนกประเภทของข้อมูล จะต้องทำการสร้างแบบจำลองตามจำนวนของประเภทของข้อมูล แบบจำลองไหนที่ให้ผลลัพธ์ความน่าจะเป็นของข้อมูลมากที่สุด จะถือว่าข้อมูลเป็นข้อมูลประเภทนั้น [9]

2.3 แฟ้มบันทึกข้อมูล

แฟ้มบันทึกข้อมูล หรือ Log file เป็นไฟล์ที่เก็บเหตุการณ์ต่างๆที่เกิดขึ้น ในระบบคอมพิวเตอร์ แฟ้มบันทึกข้อมูลจะถูกสร้างขึ้นอัตโนมัติ ซึ่งแฟ้มบันทึกข้อมูลส่วนใหญ่จะถูกบันทึกไว้ในรูปแบบข้อความ (Plain text format) ทำให้ช่วยลดขนาดไฟล์และสามารถเปิดอ่านได้ด้วยโปรแกรมอ่านไฟล์ทั่วไป

2.3.1 Secure Log

เป็นแฟ้มบันทึกข้อมูลที่บันทึกข้อมูลของระบบพิสูจน์ตัวตนและระบบในการให้สิทธิ์ผู้ใช้งาน เช่น ระบบ Pluggable Authentication Module (PAM) คำสั่ง sudo และการเข้าสู่ระบบจากระยะไกลผ่าน SSH เป็นต้น [10]

2.3.2 General Query Log

general query log คือ การบันทึกข้อมูลสิ่งที่ mysql กระทำอยู่ เครื่องคอมพิวเตอร์จะเขียนข้อมูลเมื่อเครื่องฝั่งผู้ใช้งานมีการเชื่อมต่อหรือขาดการเชื่อมต่อ และจะบันทึกคำสั่งภาษา SQL ที่ได้รับจากเครื่องฝั่งผู้ใช้งาน general query log มีประโยชน์มากเมื่อคาดว่ามีความผิดพลาดที่ฝั่งผู้ใช้งานระบบและต้องการทราบสิ่งที่ฝั่งผู้ใช้งานระบบส่งมา [11]

2.4 ระบบตรวจจับการบุกรุก (Intrusion Detection System)

ระบบตรวจจับการบุกรุก หรือ Intrusion Detection System คือ ซอฟต์แวร์ หรือ ฮาร์ดแวร์ ที่ได้รับการออกแบบมาเพื่อตรวจสอบการเชื่อมต่อที่ไม่พึงประสงค์ หรือความพยายามที่จะเข้ามาทำอันตรายต่อเครือข่าย โดยผ่านระบบต่างๆ เช่น อินเตอร์เน็ต เป็นต้น

2.4.1 ระบบตรวจจับการบุกรุกในระบบเครือข่าย หรือ Network Based (Network IDS)

ระบบตรวจจับการบุกรุกในระบบเครือข่ายหรือ Network based intrusion detection จะตรวจสอบหาพฤติกรรมที่ไม่พึงประสงค์ในระบบเครือข่าย โดยใช้วิธีการต่างๆ เช่น การดักจับข้อมูลระบบเครือข่าย (network tap) หรือการกระจายข้อมูลของพอร์ต เป็นต้น

2.4.2 ระบบตรวจจับการบุกรุกภายในเครื่อง หรือ Host Based (HIDS)

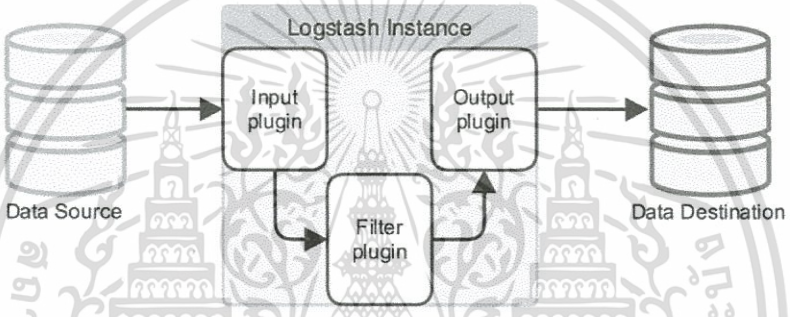
ระบบตรวจจับการบุกรุกภายในเครื่อง หรือ HIDS นั้นจะตรวจจับการบุกรุกในแต่ละระบบ โดยจะเฝ้าระวังและแจ้งเตือนเมื่อระบบปฏิบัติการหรือแอปพลิเคชันภายในเครื่อง มีเหตุการณ์บางอย่าง ไม่ถูกต้องตามลักษณะ, สมมติฐาน หรือกฎ ที่กำหนดไว้

2.5 Logstash

Logstash คือ ซอฟต์แวร์โอเพนซอร์ซ (Open source software) สำหรับรวมข้อมูลจากแหล่งข้อมูลหลายๆแหล่งและปรับแต่งข้อมูลให้อยู่ในรูปแบบที่ผู้ใช้งานต้องการ เพื่อนำข้อมูลไปวิเคราะห์และแสดงข้อมูลให้อยู่ในรูปแบบที่สามารถเข้าใจได้ง่าย

Logstash pipeline เป็นคุณสมบัติที่สำคัญของ Logstash มีส่วนประกอบหลักในการทำงาน 3 ส่วน

ส่วนที่ 1 มีหน้าที่รับข้อมูลนำเข้า(Input plugin) แล้วนำข้อมูลนำเข้าผ่านส่วนที่ 2 คือส่วนของการกรองข้อมูล(Filter plugin) จากนั้นนำข้อมูลที่ผ่านการกรองมาเข้าส่วนที่ 3 คือส่วนของการส่งออกข้อมูล (Output plugin)ดังรูปที่ 2.6



รูป 2.7 การทำงานของ logstash pipeline

โดยส่วนของ Input plugin สามารถกำหนดค่าตำแหน่งของแฟ้มข้อมูลในเครื่องที่จะรับเข้ามาเป็นข้อมูลนำเข้า ดังตัวอย่าง

ตัวอย่าง 2.1 การตั้งค่าส่วนนำเข้าของมูลของ Logstash

```
Input {
    File {
        path => "/path/to/groksample.log"
        start_position => beginning
    }
}
```

ส่วนของ Filter plugin สามารถทำงานได้ โดยใช้ grok filter plugin ซึ่งเป็นค่าเริ่มต้นใน Logstash

```
83.149.9.216 - - [04/Jan/2015:05:13:42 +0000] "GET /presentations/logstash-monitorama-2013/images/kibana-search.png
HTTP/1.1" 200 203023 "http://semicomplete.com/presentations/logstash-monitorama-2013/" "Mozilla/5.0 (Macintosh; Intel
Mac OS X 10_9_1) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/32.0.1700.77 Safari/537.36"
```

รูป 2.8 ตัวอย่างข้อมูลในแฟ้มบันทึกข้อมูลก่อนที่จะผ่านส่วนของ Filter plugin

grok filter plugin นั้นจะตรวจสอบรูปแบบของข้อมูลในแฟ้มบันทึกข้อมูล แล้วจะแปลงข้อมูลตามรูปแบบที่ระบุไว้ในไฟล์ตั้งค่า

ตัวอย่าง 2.2 การตั้งค่า grok filter

```
filter {
    grok {
        match => { "message" =>
                   "%{COMBINEDAPACHELOG}"}
    }
}
```

แล้วจะนำข้อมูลที่ผ่านการแปลงรูปมาจัดให้อยู่ในลักษณะของ JSON (JavaScript Object Notation)

```
{
"clientip" : "83.149.9.216",
"ident" : ,
"auth" : ,
"timestamp" : "04/Jan/2015:05:13:42 +0000",
"verb" : "GET",
"request" : "/presentations/logstash-monitorama-2013/images/kibana-search.png",
"httpversion" : "HTTP/1.1",
"response" : "200",
"bytes" : "203023",
"referrer" : "http://semicomplete.com/presentations/logstash-monitorama-2013/",
"agent" : "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_9_1) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/32.0.1700.77 Safari/
537.36"
}
```

รูป 2.9 ผลลัพธ์หลังจากข้อมูลผ่าน filter plugin แล้ว ในรูปแบบของ JSON

ส่วนที่ 3 ส่วนของการส่งออกข้อมูล Output plugin สามารถกำหนดค่าส่งออกข้อมูลไปยัง elasticsearch ได้ ดังตัวอย่าง

ตัวอย่าง 2.3 การตั้งค่า output plugin เพื่อส่งข้อมูลไปยัง Elasticsearch

```
filter {
    grok {
        match => { "message" =>
                   "%{COMBINEDAPACHELOG}"}
    }
}
```

2.7 Elasticsearch

Elasticsearch เป็น open source software ที่พัฒนาจาก Apache Lucene มีความสามารถในการค้นคืนข้อความแบบเต็มรูปแบบ (Full-text searching) และประมวลผลแบบทันที โดยถูกออกแบบมาให้รองรับการปรับขยายจำนวนเครื่องคอมพิวเตอร์ที่ใช้ในการค้นหาและประมวลผลได้

Elasticsearch มีการเก็บข้อมูลแบบกระจาย โดยจะเก็บข้อมูลในกลุ่มของข้อมูลที่เรียกว่า index แต่ละ index จะแบ่งข้อมูลที่เก็บออกเป็นส่วนๆเรียกว่า shard เพื่อแบ่งภาระในการค้นหาและประมวลผลและแต่ละ shard สามารถมีสำเนาได้ตั้งแต่หนึ่งสำเนาขึ้นไป เรียกว่า replica [12]

2.8 งานที่เกี่ยวข้อง

การตรวจจับการบุกรุกเป็นโครงงานที่ได้รับความสนใจกันอย่างแพร่หลาย ทางคณะผู้จัดทำได้เล็งเห็นถึงความสำคัญของการตรวจจับการบุกรุกทาง SSH Brute Force ทางคณะผู้จัดทำได้ศึกษางานวิจัย [13] งานชิ้นนี้กล่าวถึงการตรวจจับผู้บุกรุก โดยใช้ข้อมูลซึ่งมาจากการจราจรทางเครือข่ายแล้วนำไปสร้างเป็นชุดข้อมูลเรียนรู้ของเครื่องทั้งแบบ Supervised และ Unsupervised โครงงาน LIFE จะใช้ข้อมูลที่มาจากแฟ้มบันทึกข้อมูลในการตรวจจับผู้บุกรุก และจัดเตรียมองค์ประกอบและคุณลักษณะที่เหมาะสมมาสร้างแบบจำลองโดยใช้ Support Vector Machine

นอกเหนือจากนี้ทางคณะผู้จัดทำได้เล็งเห็นถึงความสำคัญของการโจมตีประเภท SQL Injection และได้ศึกษางานวิจัย [14] งานชิ้นนี้กล่าวถึงการตรวจจับการบุกรุกบนเว็บแอปพลิเคชันด้วยวิธี anomaly detection ซึ่งจะต้องมีการสร้างแบบจำลองของการใช้งานที่ปกติขึ้นมาก่อน ในงานชินนี้ได้ทำการสร้างแบบจำลองขึ้นมาจากคุณลักษณะต่างๆของพารามิเตอร์ใน query string เช่น ความยาวของพารามิเตอร์ การกระจายตัวของตัวอักษรในพารามิเตอร์ เป็นต้น

ด้วยวิธีการข้างต้น ยังมีข้อจำกัดตรงที่ไม่สามารถตรวจจับการโจมตีที่เกิดขึ้นจากพารามิเตอร์ที่อยู่ใน HTTP request body รวมถึงปัญหาจากการที่คุณสมบัติของพารามิเตอร์ในแต่ละเว็บแอปพลิเคชันมีความแตกต่างกัน ทำให้ไม่สามารถประยุกต์ใช้แบบจำลองกับแอปพลิเคชันอื่นๆได้แต่ทาง

โครงงาน LIFE นั้นจะมุ่งเน้นตรวจจับการบุกรุกประเภท SQL Injection ด้วยการวิเคราะห์แฟ้มบันทึกข้อมูล general query log ซึ่งแตกต่างกับการตรวจจับความผิดปกติจาก query string

ทางคณะผู้จัดทำยังได้ทำการศึกษางานวิจัยเพิ่มเติม จากการศึกษางานวิจัย [15] เป็นงานวิจัยสำหรับการใช้ข้อมูลที่ได้จาก SQL query มาทำการวิเคราะห์ เพื่อสร้างแบบจำลองสำหรับตรวจจับการโจมตีประเภท SQL Injection โดยใช้อัลกอริทึม apriori แต่โครงงาน LIFE จะสร้างแบบจำลองโดยใช้อัลกอริทึม Machine Learning เช่น Hidden Markov Model เป็นต้น

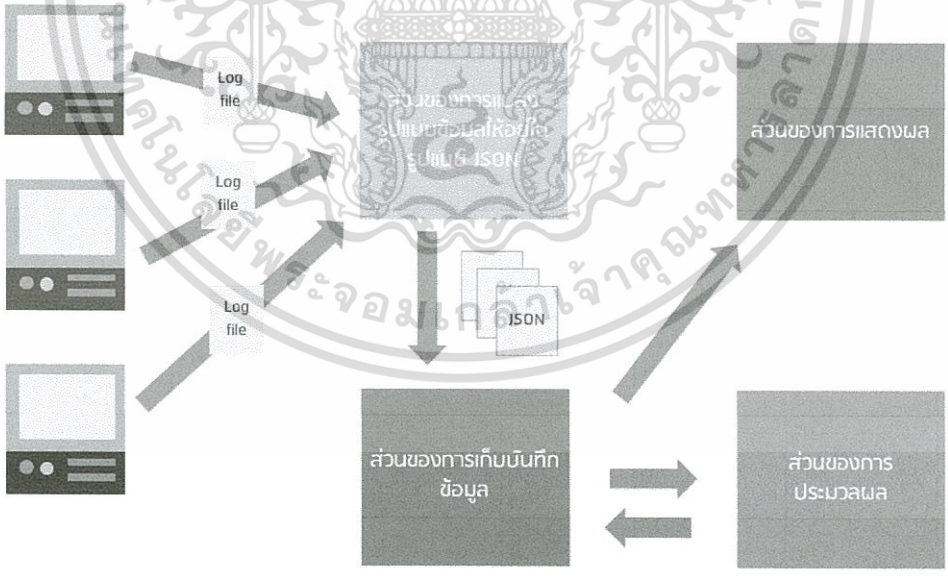
บทที่ 3
การออกแบบและการพัฒนา

3.1 ภาพรวมของระบบ

ในการตรวจจับการบุกรุกโดยใช้แฟ้มบันทึกข้อมูล เครื่อง ร์ปลายทางในศูนย์ข้อมูล จะต้องส่งข้อมูลในแฟ้มบันทึกข้อมูลมายังส่วนแปลงรูปแบบข้อมูล แล้วส่วนแปลงรูปแบบข้อมูลจะทำการแปลงข้อมูลให้อยู่ในรูปแบบที่เหมาะสม เพื่อเก็บบันทึกข้อมูลที่ได้รับการแปลงรูปแบบข้อมูลแล้วลงในฐานข้อมูล หลังจากนั้นส่วนประมวลผลจะนำข้อมูลในฐานข้อมูลไปทำการประมวลผลเพื่อตรวจจับการบุกรุก

ระบบที่นำเสนอแบ่งออกเป็น 5 ส่วนหลักๆ ดังนี้

1) ส่วนการรวบรวมแฟ้มบันทึกข้อมูลจากเครื่องคอมพิวเตอร์เพื่อใช้ในการตรวจจับการบุกรุก
2) ส่วนของการแปลงรูปแบบของข้อมูลให้อยู่ในรูปแบบ JSON
3) ส่วนของการเก็บบันทึกข้อมูล
4) ส่วนของการประมวลผลข้อมูล
5) ส่วนของการแสดงผล



รูป 3.1 ภาพรวมของระบบที่แบ่งออกเป็น 5 ส่วนหลัก

3.1.1 ส่วนของการรวบรวมแฟ้มบันทึกข้อมูลจากเครื่องคอมพิวเตอร์(Shipper)

ส่วนนี้ใช้ในการนำข้อมูลจากเครื่องคอมพิวเตอร์ปลายทาง ส่งมาให้ส่วนของการแปลงรูปแบบของข้อมูลให้อยู่ในรูป JSON โดยใช้ Logstash Forwarder ซึ่งเป็นเครื่องมือที่ใช้ในการส่งแฟ้มบันทึกข้อมูล Logstash Forwarder ใช้ network protocol ที่ชื่อว่า lumberjack สามารถลด bandwidth ในการส่งข้อมูล และ ทำหน้าที่ encryption ข้อมูลที่ส่งอีกด้วย

3.1.2 ส่วนของการแปลงรูปแบบของข้อมูลให้อยู่ในรูปแบบ JSON

ส่วนนี้ใช้ในการแปลงรูปแบบของข้อมูลที่ได้รับจาก Logstash Forwarder ให้อยู่ในรูปแบบของ JSON โดยใช้ Logstash

3.1.3 ส่วนของการเก็บบันทึกข้อมูล

ส่วนนี้ใช้ในการเก็บบันทึกข้อมูลที่ได้รับจาก Logstash และยังเก็บข้อมูลที่ได้รับจากการประมวลผลของส่วนการประมวลผลข้อมูล โดยใช้ Elasticsearch เป็นฐานข้อมูล

3.1.4 ส่วนการประมวลผลข้อมูล

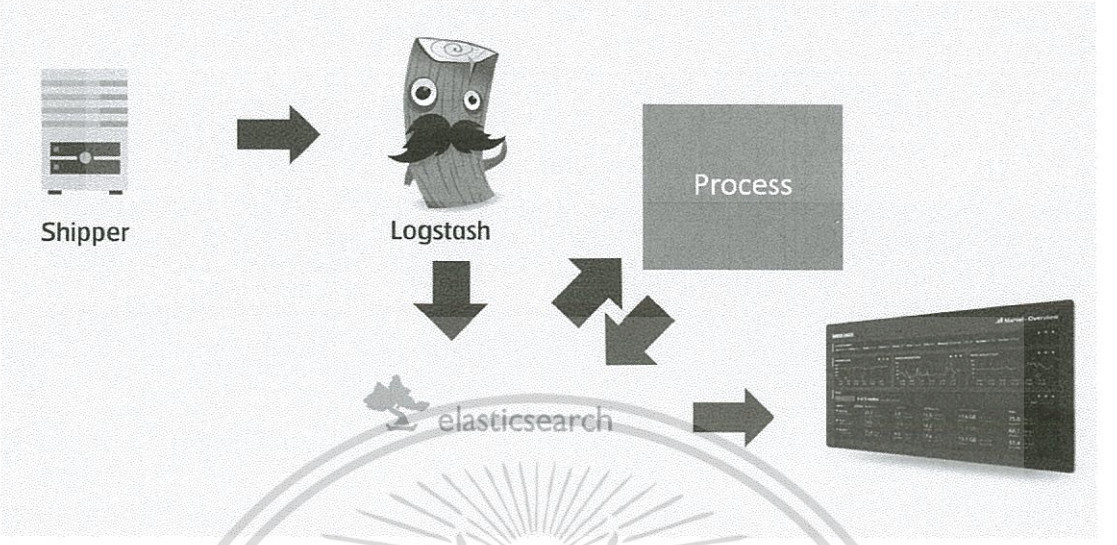
ส่วนนี้ใช้ในการนำข้อมูลที่อยู่ในฐานข้อมูลมาทำการจัดองค์ประกอบและคุณลักษณะของข้อมูล เพื่อนำไปสร้างแบบจำลองในการตรวจจับข้อมูล โดยใช้กระบวนการทำเหมืองข้อมูล (Data Mining), กระบวนการเรียนรู้ของเครื่อง (Machine Learning) หรือกระบวนการทางสถิติ แบบจำลองที่สร้างมานั้นสามารถนำไปใช้ในการตรวจจับการบุกรุกได้ สามารถช่วยให้ผู้ดูแลระบบทราบถึงภัยคุกคามและเร่งดำเนินมาตรการป้องกันได้อย่างทันท่วงที การทำงานในส่วนนี้คณะผู้จัดทำใช้ภาษา Python ในการจัดการองค์ประกอบและคุณลักษณะของข้อมูล ส่วนการประมวลผลข้อมูลในโครงงานชิ้นนี้ได้ตรวจจับการโจมตี 2 ประเภท ดังนี้

1) ส่วนสร้างแบบจำลองตรวจจับการโจมตีประเภท SSH Brute-force
2) ส่วนสร้างแบบจำลองตรวจจับการโจมตีประเภท SQL Injection

3.1.5 ส่วนของการแสดงผล

ส่วนนี้ใช้ในการแสดงผลของข้อมูล โดยจะนำข้อมูลที่ผ่านส่วนการประมวลผลข้อมูล มาแสดงให้อยู่ในรูปแบบที่ผู้ใช้งานเข้าใจได้ง่าย

3.2 โครงสร้างในการพัฒนาระบบ



รูป 3.2 โครงสร้างของระบบ

3.2.1 ส่วนของเครื่องคอมพิวเตอร์ปลายทางที่จะส่งข้อมูลในแฟ้มบันทึกข้อมูลเพื่อใช้ในการตรวจจับการบุกรุก

คณะผู้จัดทำได้ใช้ Logstash Forwarder เป็นเครื่องมือในการส่งข้อมูลของแฟ้มบันทึกข้อมูลผ่าน lumberjack protocol

ตัวอย่าง 3.1 การตั้งค่า Logstash Forwarder

```
{
  "network":{
    "server" : ["logstash_server"],
    "ssl ca": "path_to_ca",
    "timeout": 15
  },
  "files":[
    {
      "paths":[
        "path_to_logfile"
      ],
      "fields": {"type":"secure"}
    }
  ]
}
```

3.2.2 ส่วนของการแปลงรูปแบบของข้อมูลให้อยู่ในรูปแบบ JSON

คณะผู้จัดทำได้ใช้ Logstash ในการแปลงรูปแบบของข้อมูลที่ได้รับจาก Logstash Forwarder ให้อยู่ในรูปของ JSON format มีการกำหนดค่า Logstash ดังนี้

ตัวอย่าง 3.2 การตั้งค่า Logstash

```
Input {
  lumberjack{
    port => 5000
    ssl_certificate=>"path_to_crt"
    ssl_key => "path_to_key"
    type => "secure"
  }
  filter{
    if[type] == "secure"{
      grok{
        match => {"message" => "format_message"}
        date{
          match => ["format_date"]
        }
      }
    }
  }
}
```

ข้อมูลที่ได้รับจากส่วนของ Input plugin จะเป็นข้อมูลที่ยังไม่ได้จัดรูปแบบข้อมูลให้อยู่ในรูปแบบ JSON ดังที่แสดงในตัวอย่าง 3.3

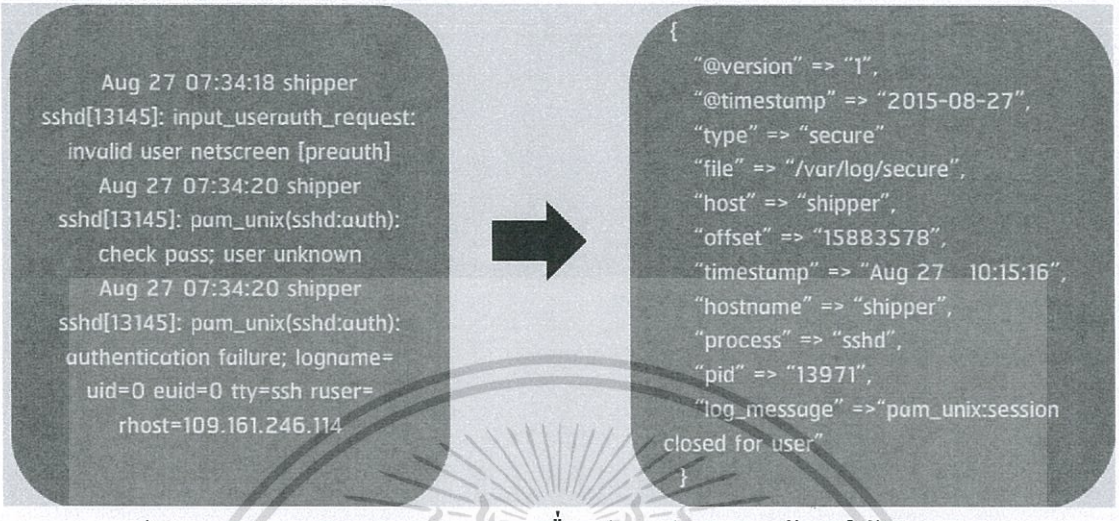
ตัวอย่าง 3.3 ข้อมูลที่เข้าสู่ input plugin ของ Logstash

```
Aug 27 07:34:18 shipper sshd[13145]:
input_userauth_request: invalid user netscreen
[preauth]

Aug 27 07:34:20 shipper sshd[13145]:
pam_unix(sshd:auth): check pass; user unknown

Aug 27 07:34:20 shipper sshd[13145]:
pam_unix(sshd:auth): authentication failure; logname=
uid=0 euid=0 tty=ssh ruser= rhost=109.161.246.114
```

หลังจากที่ข้อมูลนำเข้าผ่าน Input plugin ของ Logstash แล้ว ต่อไปจะเป็นการทำงานของ FIlter plugin เพื่อเปลี่ยนรูปแบบข้อมูลของข้อมูลนำเข้าให้อยู่ในรูปแบบของ JSON ดังรูป



รูป 3.3 การทำงานของ Filter plugin เพื่อแปลงรูปแบบของข้อมูลให้เป็น JSON

หลังจากที่ผ่าน Filter plugin เรียบร้อยแล้ว ข้อมูลที่อยู่ในรูปแบบของ JSON จะถูกส่งไปยัง Elasticsearch ผ่าน output plugin

3.2.3 ส่วนของการเก็บบันทึกข้อมูล

ส่วนนี้ทำหน้าที่นำผลลัพธ์ที่ได้จาก Logstash มาเก็บลงฐานข้อมูล โดยใช้ Elasticsearch

3.2.4 ส่วนของการประมวลผลข้อมูล

นำแบบจำลองที่ได้จากการเตรียมองค์ประกอบและคุณลักษณะมาผ่านอัลกอริทึมการเรียนรู้ ส่วนการประมวลผลข้อมูลทางคณะผู้จัดทำได้จัดทำการตรวจจับการโจมตี 2 ประเภท ดังนี้

3.2.4.1 ส่วนการประมวลผลข้อมูลโดยใช้แบบจำลอง SSH Brute Force

ในส่วนนี้จะนำข้อมูลที่ Query จาก Elasticsearch มาทำการจัดลักษณะตาม Attribute ต่างๆ ดังนี้

1) จำนวนครั้งที่มีการใส่ password ผิด

2) จำนวนครั้งที่มีการใส่ username ผิด

3) ช่วงระยะเวลาระหว่าง log file ที่น้อยที่สุด

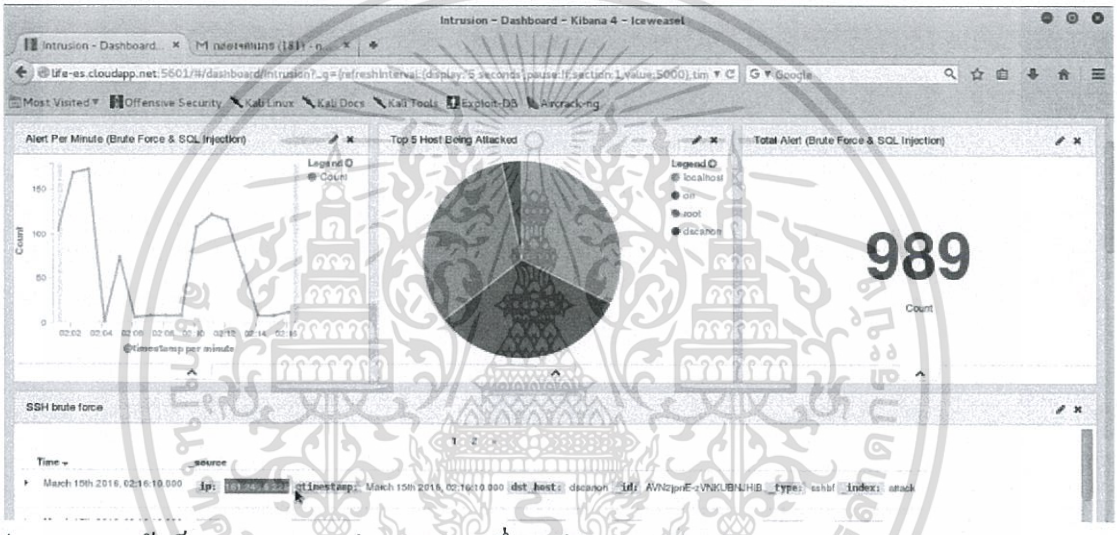
ซึ่งในหน่วยประมวลผลจะนำ Attribute ทั้ง 3 ค่านี้นำไปตรวจจับการบุกรุกโดยผ่านแบบจำลองที่ได้จากการเรียนรู้ด้วย Algorithm Support Vector Machine

3.2.4.2 ส่วนการประมวลผลข้อมูลโดยใช้แบบจำลองตรวจจับการโจมตีประเภท SQL Injection

ในส่วนนี้จะนำ Query Statement ที่เก็บอยู่ใน Elasticsearch มาทำการแบ่ง Token เพื่อนำไปวิเคราะห์ด้วยแบบจำลอง Hidden Markov โดยจะคำนวณ transition probability เพื่อใช้ในการเปรียบเทียบ หากค่า transition probability น้อยกว่าค่าปกติที่ได้จากการ training model จะถูกตรวจจับว่าเป็นผู้บุกรุก

3.2.5 ส่วนของการแสดงผล

หน่วยแสดงผลจะทำหน้าที่ดึงข้อมูลที่ผ่านระบบตรวจจับการบุกรุก มาแสดงผลในรูปแบบของหน้าเว็บ Dashboard ผ่าน Kibana ซึ่งจะมีลักษณะดังรูป



รูป 3.4 หน้า Dashboard ของโครงงาน LIFE ผ่าน Kibana

หน้า Dashboard จะแบ่งออกเป็น 4 ส่วนหลักๆ ดังนี้

1) ส่วนแสดงกราฟการแจ้งเตือนของการบุกรุกในช่วงระยะเวลา 15 นาทีที่ผ่านมา
2) ส่วนของการแสดงจำนวนเครื่อง Host ที่ถูกโจมตีมากที่สุด 5 อันดับแรก
3) ส่วนของการแสดงจำนวนการยุกรุกที่เกิดขึ้นทั้งหมดในช่วงระยะเวลา 15 นาที
4) ส่วนของการแสดงรายละเอียดของ log file ของ IP Address ล่าสุด ที่ถูกตรวจจับว่าเป็นผู้บุกรุก ประกอบด้วยรายละเอียดสำคัญ ดังนี้
    a) Time Stamp บอกรายละเอียดเกี่ยวกับวันและเวลา
    b) IP บอกเลข IP Address ที่ถูกตรวจจับว่าเป็นผู้บุกรุก
    c) Destination host คือชื่อเครื่องที่ถูกโจมตี

บทที่ 4

การทดลองและผลการทดลอง

4.1 การทดลองวัดความแม่นยำของอัลกอริทึมในการตรวจสอบผู้บุกรุก SSH Brute Force

4.1.1 วัตถุประสงค์

เพื่อทดสอบหาความแม่นยำของ 3 อัลกอริทึมคือ Artificial Neural Network, Support Vector Machine และ Logistic Regression เพื่อนำไปพิจารณาหาอัลกอริทึมที่เหมาะสมในการตรวจสอบหาการโจมตีประเภท SSH Brute Force ต่อไป

4.1.2 ข้อมูลที่ใช้ในการทดลอง

1) จำนวน Instance ทั้งหมด 1432 Instance โดยเป็นคน (letigimate user) 716 คน และเป็นบอท (Malicious user) 716 คน
2) Attribute ที่ใช้มีดังนี้
   2.1) จำนวนครั้งที่มีการใส่ password ผิด
   2.2) จำนวนครั้งที่มีการใส่ username ผิด
   2.3) ช่วงระยะเวลาระหว่าง log file น้อยที่สุด
   2.4) จำนวนครั้งที่มีการ log in ถูกต้อง
3) อัลกอริทึมที่ใช้ คือ Artifiial Neural Network, Support Vector Machine และ Logistic Regression
4) ใช้ข้อมูล log file จากการ query ทุกๆ 15 นาที (time windows = 15 mins)

4.1.3 วิธีการดำเนินการ

1) นำคุณสมบัติที่ได้จากการการดำเนินการเก็บคุณสมบัติมาสร้างแบบจำลองของ ทั้ง 3 algorithm
2) บันทึกผลการทดลอง
3) สรุปผลการทดลอง

4.1.4 ผลการทดลอง

4.1.4.1 Artificial Neural Network

ตัวแปรต่างๆของ Artificial Neural Network มีดังนี้

1) hidden layers : a
2) learning rate: 0.3
3) momentum : 0.2
4) seed : 0
5) traning time : 500
6) validation setsize : 0
7) validation Threshold : 20

ตาราง 4.1 Confusion matrix ของ Artificial Neural Network

ทายว่าเป็นคน	ทายว่าเป็นบอท	หน่วย
704	12	คน
1	715	บอท

False Positive = (12/716)x100 = 1.7%

True Positive = (704/716)x100 = 98.3%

False Negative = (1/716)x100 = 0.1%

True Negative = (715/716)x100 = 99.8%

4.1.4.2 Support Vector Machine

ตาราง 4.2 Confusion matric ของ Support Vector Machine

ทายว่าเป็นคน	ทายว่าเป็นบอท	หน่วย
704	12	คน
0	716	บอท

False Positive = (12/716)x100 = 1.7%

True Positive = (704/716)x100 = 98.3%

False Negative = (0/716)x100 = 0%

True Negative = (716/716)x100 = 100%

4.1.4.3 Logistic Regression

ตาราง 4.3 Confusion matric ของ Logistic Regression

ทายว่าเป็นคน	ทายว่าเป็นบอท	หน่วย
708	8	คน
42	675	บอท

False Positive = (8/716)x100 = 1%

True Positive = (708/716)x100 = 99%

False Negative = (42/716)x100 = 5.86%

True Negative = (675/716)x100 = 94.14%

4.1.5 สรุปผลการทดลอง

จากการทดลองสามารถสรุปได้ว่าอัลกอริทึม Support Vector Machine มีความแม่นยำมากที่สุด รองลงมาคือ Artificial Neural Network และ Logistic Regression ซึ่งในลำดับต่อไปเราจึงทำการทดสอบความแม่นยำของอัลกอริทึม Support Vector Machine เมื่อตัดแต่ละ attribute ออก

4.2 การทดลองหาน้ำหนักความสำคัญของแต่ละ attribute เมื่อใช้อัลกอริทึม Support Vector Machine

4.2.1 วัตถุประสงค์

เพื่อทดสอบว่าแต่ละ attribute มีความสำคัญต่อการเรียนรู้ของอัลกอริทึมมากน้อยเพียงใด ทางคณะผู้จัดทำจึงได้ทำการทดสอบวัดความแม่นยำของอัลกอริทึม Support Vector Machine เมื่อตัดแต่ละ attribute ออก

4.2.2 ข้อมูลที่ใช้ในการทดลอง

1) จำนวน Instance ทั้งหมด 1432 Instance โดยเป็นคน (legitimate user) 716 คน และเป็นบอท (Malicious user) 716 คน
2) Attribute ที่ใช้มีดังนี้
   2.1) จำนวนครั้งที่มีการใส่ password ผิด
   2.2) จำนวนครั้งที่มีการใส่ username ผิด
   2.3) ช่วงระยะเวลาระหว่าง log file น้อยที่สุด
   2.4) จำนวนครั้งที่มีการ log in ถูกต้อง

3) อัลกอริทึมที่ใช้ คือ Support Vector Machine

4) ใช้ข้อมูล log file จากการ query ทุกๆ 15 นาที (time windows = 15 mins)

4.2.3 วิธีการดำเนินการ

นำคุณสมบัติที่ได้จากการการดำเนินการเก็บคุณสมบัติมาสร้างแบบจำลองของ ทั้ง 3 algorithm โดยตัด attribute "จำนวนครั้งที่มีการใส่ password ผิด" ออก

1) บันทึกผลการทดลอง

2) ทำการทดลองตามข้อ 1) อีกครั้ง โดยเปลี่ยน attribute ที่ตัดออก เป็น "จำนวนครั้งที่มีการใส่ username ผิด"

3) บันทึกผลการทดลอง

4) ทำการทดลองตามข้อ 1) อีกครั้ง โดยเปลี่ยน attribute ที่ตัดออก เป็น "ช่วงระยะเวลาระหว่าง log file น้อยที่สุด"

5) บันทึกผลการทดลอง

6) ทำการทดลองตามข้อ 1) อีกครั้ง โดยเปลี่ยน attribute ที่ตัดออก เป็น "จำนวนครั้งที่มีการ log in ถูกต้อง"

7) บันทึกผลการทดลอง

8) สรุปผลการทดลอง

4.2.4 บันทึกผลการทดลอง

ใช้ Support Vector Machine โดยมี ตัวแปรต่างๆ ดังนี้

1) ค่า c =1.0

2) epsilon = 1.0E-12

3) numFolds = -1

4) randomSeed =1

5) toleranceParameter =1

4.2.4.1 ตัด attribute "จำนวนครั้งที่มีการใส่ password ผิด" ออก

ตาราง 4.4 Confusion matrix ของ Support Vector Machine

ทายว่าเป็นคน	ทายว่าเป็นบอท	หน่วย
704	12	คน
0	716	บอท

ความแม่นยำ : 99.162%

False Positive = (12/716)x100 = 1.67%

True Positive = (704/716)x100 = 98.32%

False Negative = (0/716)x100 = 0%

True Negative = (716/716)x100 = 100.00%

4.2.4.2 ตัด attribute "จำนวนครั้งที่มีการใส่ username ผิด" ออก

ตาราง 4.5 Confusion matrix ของ Support Vector Machine

ทายว่าเป็นคน	ทายว่าเป็นบอท	หน่วย
704	12	คน
0	716	บอท

ความแม่นยำ : 99.162%

False Positive = (12/716)x100 = 1.67%

True Positive = (704/716)x100 = 98.32%

False Negative = (0/716)x100 = 0%

True Negative = (716/716)x100 = 100.00%

4.2.4.3 ตัด attribute "ช่วงระยะเวลาระหว่าง log file น้อยที่สุด" ออก

ตาราง 4.6 Confusion matrix ของ Support Vector Machine

ทายว่าเป็นคน	ทายว่าเป็นบอท	หน่วย
704	12	คน
0	716	บอท

ความแม่นยำ : 99.162%

False Positive = (12/716)x100 = 1.67%

True Positive = (704/716)x100 = 98.32%

False Negative = (0/716)x100 = 0%

True Negative = (716/716)x100 = 100.00%

4.2.4.4 ตัด attribute "จำนวนครั้งที่มีการ log in ถูกต้อง" ออก

ตาราง 4.7 Confusion matrix ของ Support Vector Machine

ทายว่าเป็นคน	ทายว่าเป็นบอท	หน่วย
714	2	คน
481	235	บอท

ความแม่นยำ : 66.270%

False Positive = (2/716)x100 = 0.27%

True Positive = (714/716)x100 = 99.72%

False Negative = (235/716)x100 = 32.82%

True Negative = (481/716)x100 = 67.17%

4.2.5 สรุปผลการทดลอง

จากการทดลองสรุปได้ว่า attribute "จำนวนครั้งที่มีการ log in ถูกต้อง" มีน้ำหนักความสำคัญมากที่สุดและ attribute อื่นๆ ไม่มีความสำคัญ ทางคณะผู้จัดทำจึงมาพิจารณาสาเหตุถึงปัญหาจึงทราบว่า หากผู้ใช้งานสามารถ log in ได้ แบบจำลองจะตัดสินว่าเป็นบุคคลทั่วไป (legitimate user) เนื่องจากข้อมูลที่นำมาสร้างแบบจำลองนั้น ไม่มีข้อมูลของผู้ไม่ประสงค์ดี (malicious user) ที่สามารถเข้าสู่ระบบได้ ทางคณะผู้จัดทำจึงได้ทำการทดสอบวัดความแม่นยำของอัลกอริทึมใหม่อีกครั้งในการทดลองที่ 4.3

4.3 การทดลองวัดความแม่นยำของอัลกอริทึมในการตรวจสอบผู้บุกรุก SSH Brute Force โดยไม่ใช้ attribute "จำนวนครั้งที่มีการ log in ถูกต้อง"

4.3.1 วัตถุประสงค์

เพื่อทดสอบหาความแม่นยำของ 3 อัลกอริทึมคือ Artificial Neural Network, Support Vector Machine และ Logistic Regression เพื่อนำไปพิจารณาหาอัลกอริทึมที่เหมาะสมในการตรวจสอบหาการโจมตีประเภท SSH Brute Force ต่อไป

4.3.2 ข้อมูลที่ใช้ในการทดลอง

1) จำนวน Instance ทั้งหมด 1432 Instance โดยเป็นคน (legitimate user) 716 คน และเป็นบอท (Malicious user) 716 คน

2) Attribute ที่ใช้มีดังนี้

2.1) จำนวนครั้งที่มีการใส่ password ผิด

2.2) จำนวนครั้งที่มีการใส่ username ผิด

2.3) ช่วงระยะเวลาระหว่าง log file น้อยที่สุด

3) อัลกอริทึมที่ใช้ คือ Artificial Neural Network, Support Vector Machine และ Logistic Regression

4) ใช้ข้อมูล log file จากการ query ทุกๆ 15 นาที (time windows = 15 mins)

4.3.3 วิธีการดำเนินการ

1) นำคุณสมบัติที่ได้จากการการดำเนินการเก็บคุณสมบัติมาสร้างแบบจำลองของ ทั้ง 3 algorithm

2) ทำการปรับค่าตัวแปรต่างๆ

3) ทำการทดลองขั้นที่ 1 ใหม่อีกครั้ง

4) บันทึกผลการทดลอง

5) สรุปผลการทดลอง

4.3.4 ผลการทดลอง

4.3.4.1 Artificial Neural Network

ตัวแปรต่างๆของ Artificial Neural Network มีดังนี้

1) hidden layers : a

2) learning rate: 0.3

3) momentum : 0.2

4) seed : 0

5) training time : 500

6) validation set size : 0

7) validation Threshold : 20

ตาราง 4.8 Confusion matrix ของ Artificial Neural Network

ทายว่าเป็นคน	ทายว่าเป็นบอท	หน่วย
706	10	คน
164	552	บอท

ความแม่นยำ : 87.849%

False Positive = (10/716)x100 = 1.3966%

True Positive = (706/716)x100 = 98.32%

False Negative = (164/716)x100 = 22.90%

True Negative = (552/716)x100 = 77.09%

4.3.4.2 Support Vector Machine

ตัวแปรต่างๆ ของ Support Vector Machine มีดังนี้

1) ค่า c =1.0
2) epsilon = 1.0E-12
3) numFolds = -1
4) randomSeed =1
5) toleranceParameter =1

ตาราง 4.9 Confusion matrix SVM - Kernel Function : Poly Kernel

ทายว่าเป็นคน	ทายว่าเป็นบอท	หน่วย
714	2	คน
481	235	บอท

ความแม่นยำ : 66.2709%

False Positive = (2/716)x100 = 0.27%

True Positive = (714/716)x100 = 99.72%

False Negative = (481/716)x100 = 67.17%

True Negative = (235/716)x100 = 32.82%

ตาราง 4.10 Confusion matrix ของ SVM - Kernel Function : Normalize Kernel

ทายว่าเป็นคน	ทายว่าเป็นบอท	หน่วย
664	52	คน
0	716	บอท

ความแม่นยำ : 96.3687%

False Positive = (52/716)x100 = 7.26%

True Positive = (664/716)x100 = 92.73%

False Negative = (0/716)x100 = 0%

True Negative = (716/716)x100 = 100%

ตาราง 4.11 Confusion matrix ของ SVM - Kernel Function : Puk Kernel

ทายว่าเป็นคน	ทายว่าเป็นบอท	หน่วย
714	2	คน
393	323	บอท

ความแม่นยำ : 72.416%

False Positive = (2/716)x100 = 0.27%

True Positive = (714/716)x100 = 99.72%

False Negative = (393/716)x100 = 54.81%

True Negative = (323/716)x100 = 45.05%

ตาราง 4.12 Confusion matrix ของ SVM - Kernel Function : RBF Kerne

ทายว่าเป็นคน	ทายว่าเป็นบอท	หน่วย
644	72	คน
575	141	บอท

ความแม่นยำ : 54.8184%

False Positive = (72/716) x 100 = 10.05%

True Positive = (644/716)x100 = 89.94%

False Negative = (575/716)x100 = 80.30%

True Negative = (141/716)x100 = 19.69%

4.3.4.3 Logistic Regression

ตาราง 4.13 Confusion matrix ของ Logistic Regression

ทายว่าเป็นคน	ทายว่าเป็นบอท	หน่วย
697	19	คน
118	598	บอท

ความแม่นยำ : 90.433%

False Positive = (19/716)x100 = 2.65%

True Positive = (697/716)x100 = 97.34%

False Negative = (118/716)x100 = 16.48%

True Negative = (598/716)x100 = 83.51%

4.3.5 สรุปผลการทดลอง

จากการทดลองสามารถสรุปได้ว่าแบบจำลองที่ใช้ Support Vector Machine มีความแม่นยำมากที่สุด โดยใช้ Normalize Kernel (96.3712%) รองลงมาคือ Logistic Regression (90.3699%) และ Artificial Neural Network (84.508%)

4.4 การทดลองวัดความแม่นยำของอัลกอริทึมในการตรวจสอบผู้บุกรุก SSH Brute Force ในแต่ละ attribute

4.4.1 วัตถุประสงค์

เพื่อตรวจสอบความสำคัญของคุณลักษณะต่างๆที่ใช้ในการสร้างแบบจำลองตรวจจับการโจมตี SSH Brute Force ด้วยอัลกอริทึม Support Vector Machine โดยใช้ Normalize Kernel

4.4.2 ข้อมูลที่ใช้ในการทดลอง

1) จำนวน Instance ทั้งหมด 1432 Instance โดยเป็นคน (letigimate user) 716 คน และเป็นบอท (Malicious user) 716 คน
2) Attribute ที่ใช้มีดังนี้
   2.1) จำนวนครั้งที่มีการใส่ password ผิด
   2.2) จำนวนครั้งที่มีการใส่ username ผิด
   2.3) ช่วงระยะเวลาระหว่าง log file น้อยที่สุด
3) อัลกอริทึมที่ใช้ คือ Support Vector Machine

4) ใช้ข้อมูล log file จากการ query ทุกๆ 15 นาที (time windows = 15 mins)

4.4.3 วิธีการดำเนินการ

1) นำคุณสมบัติที่ได้จากการการดำเนินการเก็บคุณสมบัติมาสร้างแบบจำลองของอัลกอริทึม Support Vector Machine โดยใช้ Normalize Kernel

2) ทำการทดสอบนำ attribute "จำนวนครั้งที่มีการใส่ password ผิด" ออก เหลือ 2 attribute

3) ทำการทดสอบนำ attribute "จำนวนครั้งที่มีการใส่ username ผิด" ออก เหลือ 2 attribute

4) ทำการทดสอบนำ attribute "ช่วงระยะเวลาระหว่าง log file น้อยที่สุด" ออก เหลือ 2 attribute

5) ทำการทดสอบเฉพาะ attribute "จำนวนครั้งที่มีการใส่ password ผิด"

6) ทำการทดสอบเฉพาะ attribute "จำนวนครั้งที่มีการใส่ username ผิด"

7) ทำการทดสอบเฉพาะ attribute "ช่วงระยะเวลาระหว่าง log file น้อยที่สุด"

8) บันทึกผลการทดลอง

9) สรุปผลการทดลอง

4.4.4 ผลการทดลอง

ตาราง 4.14 การทดสอบนำ "จำนวนครั้งที่มีการใส่ password ผิด" ออก เหลือ 2 attribute

ทายว่าเป็นคน	ทายว่าเป็นบอท	หน่วย
674	42	คน
13	703	บอท

ความแม่นยำ : 96.1592%

False Positive = (42/716)x100 = 5.86%

True Positive = (674/716)x100 = 94.13%

False Negative = (13/716)x100 = 1.81%

True Negative = (703/716)x100 = 98.18%

ตาราง 4.15 การทดสอบนำ "จำนวนครั้งที่มีการใส่ username ผิด" ออก เหลือ 2 attribute

ทายว่าเป็นคน	ทายว่าเป็นบอท	หน่วย
638	33	คน
57	659	บอท

ความแม่นยำ : 93.7151%

False Positive = (33/716)x100 = 4.60%

True Positive = (638/716)x100 = 89.10%

False Negative = (57/716)x100 = 7.96%

True Negative = (659/716)x100 = 92.03%

ตาราง 4.16 การทดสอบนำ "ช่วงระยะเวลาระหว่าง log file น้อยที่สุด" ออกเหลือ 2 attribute

ทายว่าเป็นคน	ทายว่าเป็นบอท	หน่วย
664	52	คน
0	716	บอท

ความแม่นยำ : 96.3687%

False Positive = (52/716)x100 = 7.26%

True Positive = (664/716)x100 = 92.73%

False Negative = (0/716)x100 = 0%

True Negative = (716/716)x100 = 100.00%

ตาราง 4.17 การทดสอบเฉพาะ "จำนวนครั้งที่มีการใส่ password ผิด"

ทายว่าเป็นคน	ทายว่าเป็นบอท	หน่วย
685	31	คน
135	581	บอท

ความแม่นยำ : 88.4078%

False Positive = (31/716)x100 = 4.32%

True Positive = (685/716)x100 = 95.67%

False Negative = (135/716)x100 = 18.85%

True Negative = (581/716)x100 = 81.14%

ตาราง 4.18 การทดสอบเฉพาะ "จำนวนครั้งที่มีการใส่ username ผิด"

ทายว่าเป็นคน	ทายว่าเป็นบอท	หน่วย
682	34	คน
293	423	บอท

ความแม่นยำ : 77.1648%

False Positive = (34/716)x100 = 4.74%

True Positive = (682/716)x100 = 95.25%

False Negative = (293/716)x100 = 40.92%

True Negative = (423/716)x100 = 59.07%

ตาราง 4.19 การทดสอบเฉพาะ "ช่วงระยะเวลาระหว่าง log file น้อยที่สุด"

ทายว่าเป็นคน	ทายว่าเป็นบอท	หน่วย
693	23	คน
71	645	บอท

ความแม่นยำ : 93.4358%

False Positive = (23/716)x100 = 3.21%

True Positive = (693/716)x100 = 96.78%

False Negative = (71/716)x100 = 9.91%

True Negative = (645/716)x100 = 90.08%

4.4.5 สรุปผลการทดลอง

ทุก attribute มีความสำคัญต่อการเรียนรู้ของ Support Vector Machine

4.5 การทดลองหาค่า time windows ที่เหมาะสมสำหรับการนำ log file มาวิเคราะห์

4.5.1 จุดประสงค์การทดลอง

เพื่อหาช่วงเวลาที่เหมาะสมสำหรับการเตรียม log file มาสร้างแบบจำลองการเรียนรู้ของเครื่องต่อไป

โดยทางคณะผู้จัดทำจะเลือกช่วงเวลาที่นำมาทำการทดสอบวัดความแม่นยำ 1 นาที, 15 นาที และ 30 นาที ตามลำดับ

4.5.2 ข้อมูลที่ใช้ในการทดลอง

1) จำนวน Instance ทั้งหมด 3074 Instance โดยเป็นคน (letigimate user) 1537 คน และเป็นบอท (Malicious user) 1537 คน สำหรับการทดลองที่มีช่วงเวลาในการเตรียม log file 1 นาที
2) จำนวน Instance ทั้งหมด 1432 Instance โดยเป็นคน (letigimate user) 716 คน และเป็นบอท (Malicious user) 716 คน สำหรับการทดลองที่มีช่วงเวลาในการเตรียม log file 15 นาที
3) จำนวน Instance ทั้งหมด 1274 Instance โดยเป็นคน (letigimate user) 637 คน และเป็นบอท (Malicious user) 637 คน สำหรับการทดลองที่มีช่วงเวลาในการเตรียม log file 30 นาที
4) Attribute ที่ใช้มีดังนี้
   4.1) จำนวนครั้งที่มีการใส่ password ผิด
   4.2) จำนวนครั้งที่มีการใส่ username ผิด
   4.3) ช่วงระยะเวลาระหว่าง log file น้อยที่สุด
5) อัลกอริทึมที่ใช้ คือ Support Vector Machine โดยใช้ Normailize Kernel

4.5.3 วิธีการดำเนินการ

1) เตรียม attribute ของ log file โดยใช้ time windows 1 นาที
2) นำ attribute ที่ได้จากข้อ 1 มาสร้างแบบจำลองโดยใช้อัลกอริทึม Support Vector Machine
3) บันทึกผลการทดลอง
4) ทำการทดลองตามข้อ 1-3 โดยเปลี่ยน time windoes เป็น 15 นาที
5) ทำการทดลองตามข้อ 1-3 โดยเปลี่ยน time windows เป็น 30 นาที

4.5.4 ผลการทดลอง

ตาราง 4.20 การทดลองหา time windows ที่เหมาะสม เมื่อกำหนด time windows มีค่า 1 นาที

ทายว่าเป็นคน	ทายว่าเป็นบอท	หน่วย
1414	123	คน
121	1416	บอท

ความแม่นยำ : 92.0625%

False Positive = (123/716)x100 = 17.17%

True Positive = (704/1537)x100 = 45.80%

False Negative = (121/1537)x100 = 7.87%

True Negative = (1416/1537)x100 = 92.12%

ตาราง 4.21 การทดลองหา time windows ที่เหมาะสม เมื่อกำหนด time windows มีค่า 15 นาที

ทายว่าเป็นคน	ทายว่าเป็นบอท	หน่วย
664	52	คน
0	716	บอท

ความแม่นยำ : 96.3687%

False Positive = (52/716)x100 = 7.26%

True Positive = (664/716)x100 = 92.73%

False Negative = (0/716)x100 = 0%

True Negative = (716/716)x100 = 100.00%

ตาราง 4.22 การทดลองหา time windows ที่เหมาะสม เมื่อกำหนด time windows มีค่า 30 นาที

ทายว่าเป็นคน	ทายว่าเป็นบอท	หน่วย
507	130	คน
0	637	บอท

ความแม่นยำ : 89.7959%

False Positive = (130/637)x100 = 20.40%

True Positive = (507/637)x100 = 79.59%

False Negative = (0/637)x100 = 0%

True Negative = (637/637)x100 = 100%

4.5.5 สรุปผลการทดลอง

การทดลองสามารถสรุปได้ว่า เมื่อกำหนด time windows มีค่า 15 นาที จะทำให้แบบจำลองมีความแม่นยำมากที่สุด ด้วยความแม่นยำ 96.3687% รองลงมาคือ time windows มีค่า 1 นาที จะทำให้แบบจำลองมีความแม่นยำ 92.0625%

4.6 ผลจากการสังเกตกลุ่มข้อมูลที่ทำการทดลองว่าเหมาะสมกับอัลกอริทึมที่เลือกใช้หรือไม่

4.6.1 วัตถุประสงค์

เพื่อสังเกตลักษณะของกลุ่มข้อมูลที่นำมาทำการทดลองว่ามีความเหมาะสมกับ Algorithm Support Vector Machine มากน้อยเพียงใด

4.6.2 สรุปผลการสังเกต

ลักษณะข้อมูลที่ได้จากการ plot จุดต่างๆ ของ data set ตาม attribute ที่ใช้ในการสร้างแบบจำลองพบว่ามีลักษณะดังรูป

รูป 4.1 ลักษณะ Data set ที่ plot ในกราฟตาม Attribute จำนวนครั้งที่ใส่ password ผิด และระยะห่างระหว่าง log file ที่น้อยที่สุด

จากลักษณะของ Data set ดังกล่าว สามารถสรุปได้ว่าข้อมูลตาม Attribute ที่ปรากฏในรูปข้างต้นไม่สามารถแบ่งได้ด้วย Linear Classifier จึงมีความเหมาะสมที่ทางคณะผู้จัดทำจะใช้ Support Vector Machine เป็น Algorithm ในการตรวจจับการบุกรุก เนื่องจาก Support Vector Machine นั้นสามารถแบ่งแยกลักษณะกลุ่มข้อมูลดังรูปข้างต้นได้ เพราะ SVM จะสร้าง Dimension space ขึ้นมาใหม่ ตาม Kernel Function

4.7 การทดลองวิเคราะห์พฤติกรรมของผู้บุกรุก (SSH Brute-force)

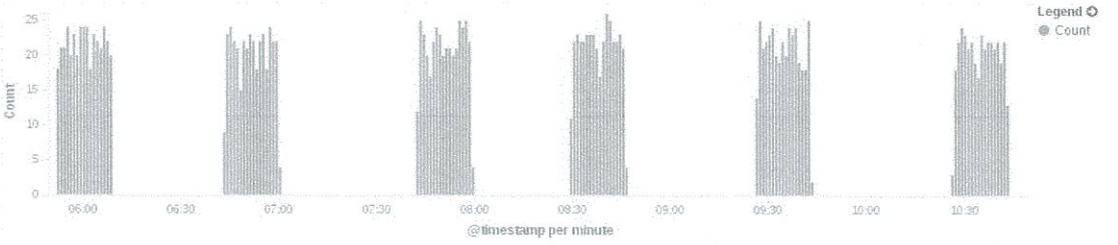
4.7.1 วัตถุประสงค์

1) เพื่อศึกษาลักษณะพฤติกรรมของการโจมตีแบบ SSH brute-force ที่เกิดขึ้น
2) เพื่อศึกษาช่วงเวลาในการโจมตีแบบ SSH brute-force ที่เกิดขึ้น

4.7.2 ออกแบบการทดลอง

1) คณะผู้จัดทำจะนำ log file มาวิเคราะห์พฤติกรรม โดยใช้ system log (secure log) มาทำการจัดรูปแบบและเก็บใน Elasticsearch
2) นำข้อมูลที่จัดรูปแบบแล้วที่ได้จากข้อ 1 มา Query เพื่อหาผู้บุกรุก โดยทางคณะผู้จัดทำจะ Query หาผู้บุกรุก และตรวจสอบให้แน่ใจว่าเป็นผู้บุกรุกที่เกิดขึ้นจริง โดยนำไปตรวจสอบกับข้อมูลใน Blacklist ของ Dshield (Internet Storm Center)
3) นำข้อมูลที่ Query ได้จากข้อ 2 มาแสดงในรูปแบบของกราฟผ่าน Kibana
4) คณะผู้จัดทำสังเกต วิเคราะห์ และบันทึกผลจากผลการทดลองดังกล่าว

4.7.3 ผลการทดลอง



รูป 4.2 ลักษณะของพฤติกรรมของการโจมตีประเภท SSH Brute-force ที่เกิดขึ้น

จากรูปที่ 4.1 แสดงลักษณะการโจมตี SSH Brute-force ของ IP 183.3.202.112 ซึ่งเป็น IP Address ที่มีการโจมตีเข้ามามากที่สุด และลักษณะการโจมตี 5 อันดับแรกของกลุ่มข้อมูลมีลักษณะคล้ายคลึงกับกราฟข้างต้น โดย แกน X แสดงถึงช่วงเวลาที่เกิดการโจมตีเกิดขึ้น ส่วนแกน Y แสดงถึงจำนวนครั้งที่มีการโจมตีเกิดขึ้น

4.7.4 สรุปผลการทดลอง

ลักษณะการโจมตีที่เกิดขึ้นมีลักษณะเป็น Phase ตามงานวิจัยของ Sperrotto [16] นอกจากนี้การโจมตีที่เกิดขึ้นมีช่วงของ Time windows ประมาณ 15 นาที ซึ่งสอดคล้องกับการทดลองของคณะผู้จัดทำ

4.8 การทดลองวัดประสิทธิภาพของเว็บแอปพลิเคชัน เมื่อ Disable general query log

4.8.1 วัตถุประสงค์

1) เพื่อศึกษาค่า Throughput ของแอปพลิเคชัน เมื่อ Disable general query log
2) เพื่อศึกษาค่า Throughput ของแอปพลิเคชัน เมื่อเพิ่มขนาดฐานข้อมูล เป็น 50,000, 100,000, 300,000, 1,000,000, 9,000,000 และ 100,000,000 rows ตามลำดับ

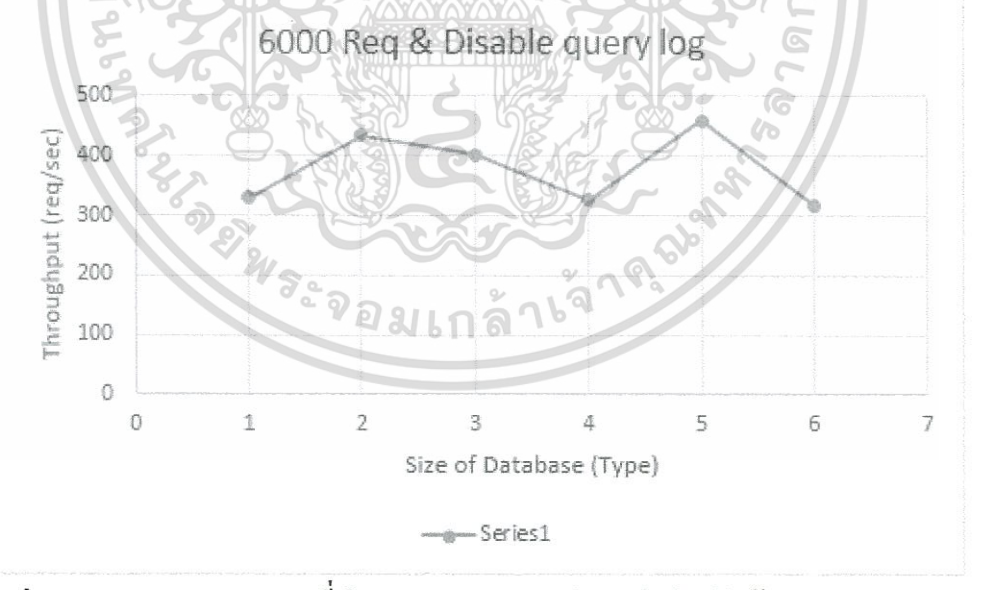
4.8.2 ออกแบบการทดลอง

1) เครื่องคอมพิวเตอร์ที่ใช้ทำการทดลองอยู่ในเครือข่ายเดียวกัน
2) เครื่องมือที่ใช้ทดสอบประสิทธิภาพคือ Apache Jmeter
3) เว็บแอปพลิเคชันที่ใช้ทดสอบ คือ Damn Vulnerable Web Application (DVWA)
4) ทำการส่ง 6000 requests พร้อมๆ กัน
5) ผลการทดสอบทางกลุ่มจะสังเกตและบันทึกค่า Throughput(req/sec)
6) Disable general query log

4.8.3 ผลการทดลอง (Disable general query log)

ตาราง 4.23 ค่า Throughput ของแอปพลิเคชันเมื่อเพิ่มขนาดฐานข้อมูลให้มีขนาด 50,000, 100,000, 300,000, 1,000,000, 9,000,000 และ 100,000,000 rows ตามลำดับและ Disable general query log

Type	Size of Database (rows)	Throughput(Req/sec)
1	50,000	329.4
2	100,000	432.8
3	300,000	400.9
4	1,000,000	325.9
5	9,000,000	459
6	100,000,000	318.2



รูป 4.3 ค่า Throughput (req/sec) ที่เกิดจากการทดสอบเว็บแอปพลิเคชันด้วย Apache Jmeter เมื่อ Disable general query log

4.8.4 สรุปผลการทดลอง

จากรูป 4.3 แสดงกราฟของค่า Throughput เมื่อเพิ่มขนาดฐานข้อมูลให้มีขนาดใหญ่ขึ้น โดยค่า Throughput ของแอปพลิเคชันมีค่า 329.4 req/sec ที่ขนาดฐานข้อมูล 50,000 rows Throughput มีอัตราการเพิ่มขึ้นโดยมีค่า 432.8 req/sec เมื่อวัด Throughput ที่ขนาดฐานข้อมูล 100,000 rows Throughput มีค่าลดลงเมื่อวัดที่ขนาดฐานข้อมูล 300,000 และ 1,000,000 rows โดยมีค่า 400.0 req/sec และ 325.9 req/sec ตามลำดับ ค่า Throughput ได้เพิ่มขึ้นอีกครั้งเมื่อวัดที่ขนาดฐานข้อมูล 9,000,000 rows โดยมีค่าอยู่ที่ 459 และ ค่า Throughput มีค่าลดลงเมื่อวัดที่ขนาดฐานข้อมูล 100,000,000 rows

และสามารถสรุปได้ว่า

1) ค่า Throughput ของแอปพลิเคชันเมื่อ Disable general query log มีค่าอยู่ที่ช่วง 318.2 req/sec ถึง 432.8 req/sec
2) ขนาดฐานข้อมูลไม่มีผลต่อ Throughput ของแอปพลิเคชัน

4.9 วัดประสิทธิภาพของเว็บแอปพลิเคชัน เมื่อ Enable general query log

4.9.1 วัตถุประสงค์

1) เพื่อศึกษาค่า Throughput ของแอปพลิเคชัน เมื่อ Enable general query log
2) เพื่อศึกษาค่า Throughput ของแอปพลิเคชัน เมื่อเพิ่มขนาดฐานข้อมูล เป็น 50,000, 100,000, 300,000, 1,000,000, 9,000,000 และ 100,000,000 rows ตามลำดับ

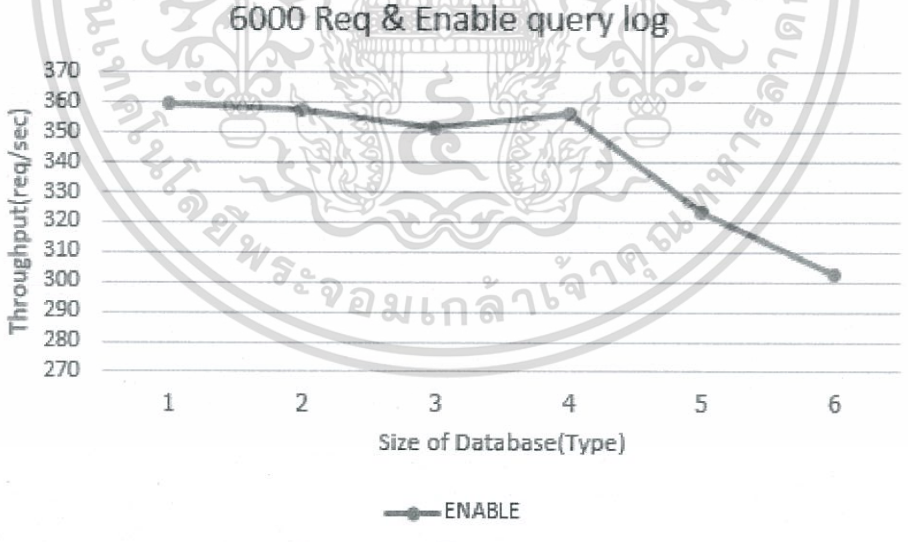
4.9.2 ออกแบบการทดลอง

1) เครื่องคอมพิวเตอร์ที่ใช้ทำการทดลองอยู่ในเครือข่ายเดียวกัน
2) เครื่องมือที่ใช้ทดสอบประสิทธิภาพคือ Apache Jmeter
3) เว็บแอปพลิเคชันที่ใช้ทดสอบ คือ Damn Vulnerable Web Application (DVWA)
4) ทำการส่ง 6000 requests พร้อมๆ กัน
5) ผลการทดสอบทางกลุ่มจะสังเกตและบันทึกค่า Throughput(req/sec)
6) Enable general query log

4.9.3 ผลการทดลอง (Enable query log)

ตาราง 4.24 ค่า Throughput ของแอปพลิเคชันเมื่อเพิ่มขนาดฐานข้อมูลให้มีขนาด 50,000, 100,000, 300,000, 1,000,000, 9,000,000 และ 100,000,000 rows ตามลำดับและ enable general query log

Type	Size of Database (rows)	Throughput(Req/sec)
1	50,000	359.4
2	100,000	357.8
3	300,000	351.5
4	1,000,000	356.4
5	9,000,000	323.4
6	100,000,000	302.8



รูป 4.4 ค่า Throughput (req/sec) ที่เกิดจากการทดสอบเว็บแอปพลิเคชันด้วย Apache Jmeter เมื่อ Enable general query log

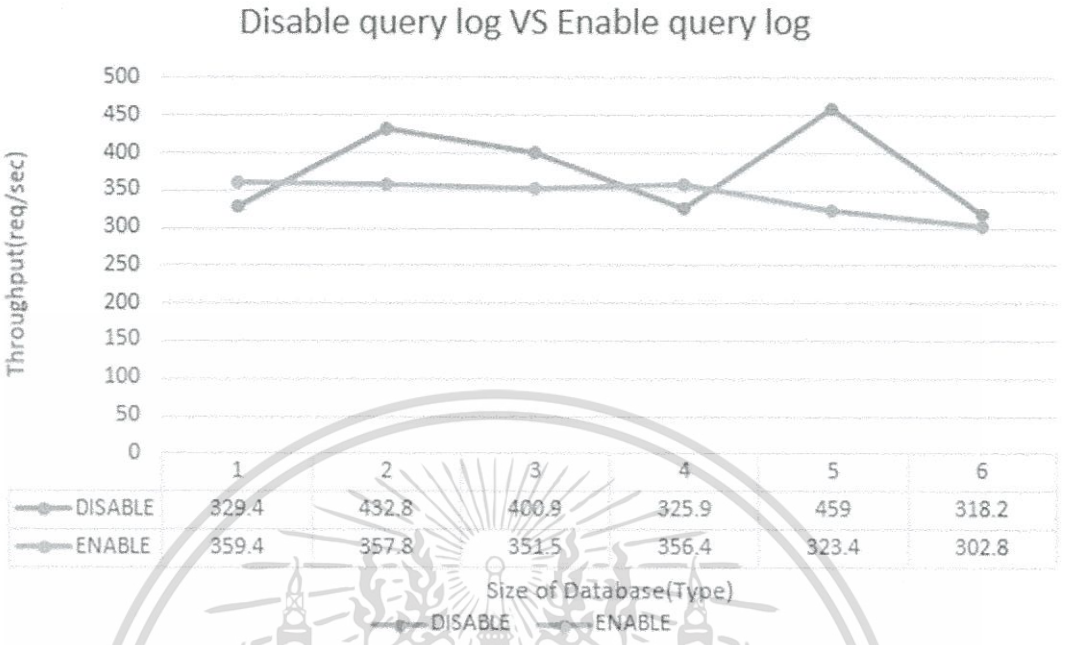
4.9.4 สรุปผลการทดลอง

จากรูป 4.4 แสดงกราฟของค่า Throughput เมื่อเพิ่มขนาดฐานข้อมูลให้มีขนาดใหญ่ขึ้น โดยค่า Throughput ของแอปพลิเคชันมีค่า 359.4 req/sec ที่ขนาดฐานข้อมูล 50,000 rows Throughput มีค่าลดลงโดยมีค่า 357.8 req/sec, 351.5 req/sec เมื่อวัด Throughput ที่ขนาดฐานข้อมูล 100,000 rows และ 300,000 rows ตามลำดับ Throughput มีค่าเพิ่มขึ้นอีกครั้งเมื่อวัดที่ขนาดฐานข้อมูล 1,000,000 rows โดยมีค่า 356.4 req/sec หลังจากนั้นค่า Throughput มีแนวโน้มลดลง โดยมีค่า 323.4 req/sec , 302.8 req/sec เมื่อขนาดฐานข้อมูล 9,000,000 และ 100,000,000 rows

และสามารถสรุปได้ว่า

1) ค่า Throughput ของแอปพลิเคชันเมื่อ Enable general query log มีค่าอยู่ที่ช่วง 302.8 req/sec ถึง 359.4 req/sec
2) ขนาดฐานข้อมูลไม่มีผลต่อ Throughput ของแอปพลิเคชัน

4.9.5 อภิปรายและสรุปผลการทดลองที่ 4.6 และ การทดลองที่ 4.7



รูป 4.5 ค่า Throughput (req/sec) ที่เกิดจากการทดสอบเว็บแอปพลิเคชันด้วย Apache Jmeter เมื่อ enable query log เปรียบเทียบกับ disable general query log

ข้อสรุปที่ 1 ขนาดของฐานข้อมูลไม่มีผลต่อ ค่า Throughput

ข้อสรุปที่ 2 throughput เฉลี่ยของการทดสอบประสิทธิภาพเมื่อ disable general query log คือ 377.7 req/sec และ throughput เฉลี่ยของการทดสอบประสิทธิภาพเมื่อ enable general query log คือ 341.88 req/sec

เมื่อเปรียบเทียบประสิทธิภาพของ throughput ระหว่าง enable general query log และ disable general query log พบว่า เมื่อ enable general query log จะทำให้ค่า throughput ของแอปพลิเคชันลดลง ดังนี้

% ค่า throughput ของแอปพลิเคชันที่ลดลง = (TPEN-TPDIS)/TPDIS x 100 เมื่อ

TPEN คือ throughput เฉลี่ยของกรณี enable general query log

TPDIS คือ throughput เฉลี่ยของกรณี disable general query log

สรุปได้ว่า เมื่อ enable general query log ค่า throughput ของแอปพลิเคชันจะลดลง 9.4%

4.10 การทดลองวัดประสิทธิภาพของแอปพลิเคชันโดยวัดค่า Latency ของแอปพลิเคชัน

4.10.1 วัตถุประสงค์

เพื่อศึกษาประสิทธิภาพของแอปพลิเคชันเมื่อ Disable general query log และ Disable general query log

4.10.2 ออกแบบการทดลอง

1) เครื่องคอมพิวเตอร์ที่ใช้ทำการทดลองอยู่ในเครือข่ายเดียวกัน
2) เครื่องมือที่ใช้ทดสอบประสิทธิภาพคือ Apache Jmeter
3) เว็บแอปพลิเคชันที่ใช้ทดสอบ คือ Damn Vulnerable Web Application (DVWA)
4) ทางกลุ่มทำการทดสอบที่ฐานข้อมูลขนาด 100,000,000 rows
5) ทำการส่ง 6000 requests พร้อมๆ กัน
6) ทำการหา Latency เฉลี่ยจากทั้ง 6000 requests
7) ทางกลุ่มทำการทดสอบ ตามข้อ 6 จำนวน 5 ครั้ง เมื่อ disable general query log และ enable general query log ตามลำดับ
8) หาค่า latency ของแต่ละกรณีโดยการหาค่าเฉลี่ย latency ของการทดลองทั้ง 5 ครั้ง
9) ผลการทดสอบทางกลุ่มจะสังเกตและบันทึกค่า Latency(ms) เฉลี่ย

4.10.3 ผลการทดลอง

ตาราง 4.25 ค่า Latency ของแอปพลิเคชันเมื่อ enable general query log

ครั้งที่	Latency ของแต่ละครั้ง (ms)
1	589
2	662
3	771
4	631
5	641

ค่าเฉลี่ย Latency เมื่อ Enable general query log = 658.8 ms

ตาราง 4.26 ค่า Latency ของแอปพลิเคชันเมื่อ disable general query log

ครั้งที่	Latency ของแต่ละครั้ง (ms)
1	643
2	489
3	623
4	548
5	550

ค่าเฉลี่ย Latency เมื่อ Disable general query log =570.6 ms

4.10.4 สรุปผลการทดลองที่ 3

เมื่อเปรียบเทียบประสิทธิภาพของ latency ระหว่าง enable general query log และ disable general query log พบว่า เมื่อ enable general query log จะทำให้ค่า latency ของแอปพลิเคชันเพิ่มขึ้น ดังนี

% ค่า latency ของแอปพลิเคชันที่เพิ่มขึ้น = (LEN-LDIS)/LDIS x 100 เมื่อ

LEN คือ latency ของกรณี enable general query log

LDIS คือ latency ของกรณี disable general query log

สรุปได้ว่า เมื่อ enable general query log ค่า latency ของแอปพลิเคชันจะเพิ่มขึ้น 15.45%

4.11 การทดลองทดสอบ Algorithm Hidden Markov Model เพื่อตรวจจับการโจมตีประเภท SQL Injection

4.11.1 วัตถุประสงค์

เพื่อศึกษาว่า Algorithm Hidden Markov Model สามารถตรวจจับการโจมตีประเภท SQL Injection ได้จริงหรือไม่

4.11.2 ออกแบบการทดลอง

1) ศึกษาการโจมตีประเภท SQL Injection ที่เกิดขึ้น จาก CVE ย้อนหลัง 1-2 ปี และค้นหาประเภทของ SQL Injection ที่พบบ่อยที่สุด และประเภทของ Application ที่พบช่องโหว่ SQL Injection บ่อยที่สุด
2) ทำการทดลองโดยติดตั้ง Application ที่ได้จากข้อ 1
3) ทำการสร้าง Training set โดยนำ Crawler มาทำการทดสอบ Application ดังกล่าวที่ Enable General Query log
4) ดำเนินการสร้างแบบจำลอง Hidden Markov Model โดยนำมาเรียนรู้จาก Training set ดังกล่าว
5) ทำการทดลองโดย ผู้จัดทำจะโจมตี Application ดังกล่าวด้วย SQL Injection และนำแบบจำลองที่ได้จากข้อ 4 มาทำการตรวจจับการโจมตีที่เกิดขึ้น
6) บันทึกและสรุปผลการทดลอง

4.11.3 ผลการทดลอง

จากการศึกษาการโจมตี SQL Injection ที่เกิดขึ้นพบว่าการโจมตีที่พบบ่อยที่สุดเป็นการโจมตีประเภท Boolean based blind SQL injection และ Application ที่พบช่องโหว่ SQL Injection มากที่สุด คือ WordPress Plugin เราจึงนำ WordPress Plugin ตัวอย่างที่พบช่องโหว่ SQL injection ประเภท Boolean Based blind ชื่อ WordPress Photo Gallery 1.2.7 ที่พบใน CVE-2015-1055 มาทำการทดสอบการตรวจจับด้วยแบบจำลองของ Algorithm Hidden Markov Model และสร้าง Training set โดยใช้ Web data extraction ที่ชื่อ import.io

4.11.4 สรุปผลการทดลอง

1) Boolean based blind เป็นชนิดการโจมตีของ SQL injection ที่พบมากที่สุดจากการศึกษา CVE ย้อนหลัง 1-2 ปี
2) WordPress Plugin เป็นชนิดของ Application ที่เกิดช่องโหว่ SQL Injection มากที่สุดจากการศึกษา CVE ย้อนหลัง 1-2 ปี
3) Hidden Markov Model สามารถตรวจจับการโจมตีที่เกิดขึ้นจากการทดลองได้

4.12 การทดลองวัดประสิทธิภาพของ Hidden Markov Model ในการตรวจจับ SQL injection

4.12.1 วัตถุประสงค์

เพื่อวัดประสิทธิภาพในการตรวจจับการโจมตีประเภท SQL Injection โดยใช้ Hidden Markov Model

4.12.2 วิธีการดำเนินการ

ในการทดลองนี้ จะใช้ bWAPP [17] ซึ่งเป็น web application ที่มีช่องโหว่ SQL injection ในการวัดประสิทธิภาพในการตรวจจับ SQL injection ของ Hidden Markov Model โดยทำการรวบรวม general query log ของ bWAPP ซึ่งมีจำนวน query ที่แตกต่างกันทั้งหมด 387 query จากนั้นทดสอบทำการโจมตีด้วย SQL injection ทั้งหมด 4 รูปแบบ ได้แก่

4.12.2.1 boolean-based blind SQL injection

เป็นการโจมตีโดยการควบคุมให้ผลของ SQL query เป็นจริงหรือเท็จ จากนั้นสังเกตผลลัพธ์ของ HTTP response ที่แตกต่างกันระหว่างผลของ SQL query ที่เป็นจริงและเท็จ เพื่อให้ได้ข้อมูลที่ต้องการ โดยในการทดลองนี้จะใช้การโจมตีแบบ boolean based blind SQL injection ในการหาชื่อของฐานข้อมูลที่ใช้งานอยู่

query ที่เป็นการใช้งานปกติ: *SELECT * FROM movies WHERE title = 'Iron Man'*

query ที่เป็นการโจมตี:

*SELECT * FROM movies WHERE title = 'Iron Man' and substring(database(),1,1)='a'-- -'*

4.12.2.2 error-based SQL injection

เป็นการโจมตีโดยทำให้ query นั้นเกิด error เพื่อนำข้อมูลที่ต้องการมาแสดงผลใน error message โดยในการทดลองนี้จะใช้การโจมตีแบบ error-based SQL injction ในการหา version ของ database ที่ใช้งานอยู่

query ที่เป็นการใช้งานปกติ: *SELECT * FROM movies WHERE title LIKE '%a'*

query ที่เป็นการโจมตี:

*SELECT * FROM movies WHERE title LIKE '%a' and (select 1 from(select count(*),concat(version(),floor(rand(0)*2))x from information_schema.tables group by x)a);-- -%'*

4.12.2.3 time-based blind SQL injection

เป็นการโจมตีโดยการทำให้ผลลัพธ์ของ query ที่เป็นจริงหรือเท็จใช้เวลาประมวลผลที่แตกต่างกัน เพื่อให้ได้ข้อมูลที่ต้องการ โดยในการทดลองนี้จะใช้การโจมตีแบบ time-based blind SQL injection ในการหาชื่อของผู้ใช้งานคนแรกในฐานข้อมูล

query ที่เป็นการใช้งานปกติ: *SELECT * FROM movies where title='Iron Man'*

query ที่เป็นการโจมตี:

*SELECT * FROM movies where title='Iron Man' and (SELECT IF((substring((SELECT login from users LIMIT 1),1,1)='A'),SLEEP(5),'false'));-- -'*

4.12.2.4 union query-based SQL injection

เป็นการโจมตีโดยใช้คำสั่ง UNION ในการรวมผลจาก query ที่ดึงค่าของมูลที่ต้องการเข้ากับ query เดิมที่มีอยู่ โดยในการทดลองนี้ จะใช้การโจมตีแบบ union query-based SQL injection ในการดึงข้อมูลชื่อผู้ใช้และรหัสผ่านทั้งหมด

query ที่เป็นการใช้งานปกติ: *SELECT * FROM movies WHERE title LIKE '%a%'*

query ที่เป็นการโจมตี:

*SELECT * FROM movies WHERE title LIKE '%a%' UNION SELECT 1,login,password,4,5,6,7 from users;-- -%'*

4.12.3 ผลการทดลอง

ตาราง 4.27 ผลการตรวจจับการโจมตีประเภท SQL Injection ชนิดต่างๆ

รูปแบบ SQL injection	ผลการตรวจจับ
boolean-based blind SQL injection	ตรวจจับได้
error-based SQL injection	ตรวจจับได้
time-based blind SQL injection	ตรวจจับได้
UNION query-based SQL injection	ตรวจจับได้

4.12.4 สรุปผลการทดลอง

การตรวจจับ SQL Injection โดยใช้ Hidden Markov Model สามารถตรวจจับรูปแบบของ SQL Injection ได้ทั้ง 4 รูปแบบ คือ Boolean-based blind, error-based SQL Injection, Time-based blind SQl Injection และ UNION query-based SQL Injection

บทที่ 5
บทสรุปและข้อเสนอแนะ

5.1 สรุป

การวิเคราะห์และตรวจจับการบุกรุกประเภท SSH Brute Force ทางคณะผู้จัดทำได้ใช้อัลกอริทึม Support Vector Machine ซึ่งเป็นอัลกอริทึมที่มีความแม่นยำมากที่สุด ทางคณะผู้จัดทำได้สร้างแบบจำลองจากลักษณะของ Data set ทำให้แบบจำลองนั้นครอบคลุมลักษณะของการบุกรุกที่เกิดขึ้นจริง ผู้ใช้งานสามารถนำแบบจำลองนี้ไปใช้ได้ โดยไม่จำเป็นต้องสร้างแบบจำลองใหม่ ระบบของคณะผู้จัดทำจะทำการวิเคราะห์แฟ้มบันทึกข้อมูลย้อนหลัง 15 นาที (Time Windows = 15 นาที) ซึ่งมีค่าใกล้เคียงกับการทดลองของ Sperotto ที่กล่าวถึงพฤติกรรมของ SSH Brute Force Attack ที่มีช่วงของการโจมตี 16.67 นาที (ทางคณะผู้จัดทำได้ทำการวิเคราะห์ Data set ที่นำมาใช้ในการสร้างแบบจำลองพบว่ามีลักษณะเช่นเดียวกัน) นอกจากนี้ระบบของทางคณะผู้จัดทำสามารถตรวจจับการบุกรุกได้ทุกๆ 1 นาที (สามารถกำหนดค่าความถี่ในการตรวจจับเองได้)

การวิเคราะห์และตรวจจับการการบุกรุกประเภท SQL Injection ทางคณะผู้จัดทำได้ใช้ อัลกิริทึม Hidden Markov Model ซึ่งสามารถตรวจจับ SQL Injection ได้ทั้ง 4 แบบ คือ Boolean-based blind, Error based, Time-based blind และ UNION query-based ซึ่งการใช้งานระบบนั้นจำเป็นต้อง Enable General Query Log ซึ่งเป็นแฟ้มบันทึก SQL Statement ของการใช้งานแอปพลิเคชัน ทำให้ค่า Throughput ของแอปพลิเคชัน ลดลง 9.4% อีกทั้งการใช้งานระบบนั้นผู้ใช้งานจะต้องสร้างแบบจำลองใหม่ตาม Query Statement ของแต่ละแอปพลิเคชัน นอกจากนี้ระบบของทางคณะผู้จัดทำสามารถตรวจจับการบุกรุกได้ทุกๆ 1 นาที (สามารถกำหนดค่าความถี่ในการตรวจจับเองได้)

5.2 แนวทางในการพัฒนาต่อ

1) เพิ่มความสามารถให้กับระบบโดยระบบสามารถทำการวิเคราะห์และตรวจจับการบุกรุกการโจมตีอื่นๆ ได้ เช่น ตรวจจับการโจมตีประเภท Cross Site Scripting เป็นต้น
2) เพิ่มความสามารถให้กับระบบโดยระบบสามารถป้องกันการโจมตีที่เกิดขึ้นได้หลังจากตรวจจับเรียบร้อยแล้ว

เอกสารอ้างอิง

[1] Internet Storm Center, "InfoSec Reports," [Online]. Available: https://isc.sans.edu/reports.html.

[2] Solutionary, "Security Engineering Research Team (SERT) Quarterly Threat Report Q2," [Online]. Available: https://www.solutionary.com/_assets/pdf/research/sert-q2-2015-threat-report.pdf.

[3] OWASP, "OWASP Top 10 - 2013 The Ten Most Critical Web Application," 2013. [Online]. Available: http://owasptop10.googlecode.com/files/OWASP%20Top%2010%20-%202013.pdf.

[4] TechTerms, "SSH (Secure Shell) Definition," 2006. [Online]. Available: http://techterms.com/definition/ssh.

[5] B. Palace, "Data Mining: What is Data Mining?," 1996. [Online]. Available: http://www.anderson.ucla.edu/faculty/jason.frand/teacher/technologies/palace/datamining.htm.

[6] "Logistic Regression," Amos J Storkey, [Online]. Available: http://www.inf.ed.ac.uk/teaching/courses/mlpr/lectures/mlpr-logreg.pdf.

[7] D. SHIFFMAN, "THE NATURE OF CODE," 2012. [Online]. Available: http://natureofcode.com/book/chapter-10-neural-networks/.

[8] S. Inc, "Support Vector Machines," [Online]. Available: http://www.statsoft.com/Textbook/Support-Vector-Machines.

[9] M. Singh, "Topics in Computational Molecular Biology," 1999. [Online]. Available: https://www.cs.princeton.edu/~mona/Lecture/HMM1.pdf.

[10] A. Echeverri, "Linux logging basics," [Online]. Available: https://www.loggly.com/ultimate-guide/linux-logging-basics.

[11] MySQL, "The General Query Log," [Online]. Available: https://dev.mysql.com/doc/refman/5.7/en/query-log.html.

[12] Elastic.co, "Elasticsearch Reference [2.0]," [Online]. Available: https://www.elastic.co/guide/en/elasticsearch/reference/current/getting-started.html.

[13] B. B. a. D. K. B. Prasanta Gogoi, "Anomaly Detection Analysis of Intrusion Data using Supervised & Unsupervised Approach," *Journal of Convergence Information Technology,* vol. 5, no. 1.11, pp. 95-110, 2010.

[14] G. V. Christopher Kruegel, "Anomaly Detection of Web-based Attacks," 2003.

[15] A. K. J. P. E. Elisa Bertino, "Profiling Database Applications to Detect SQL Injection Attack," *IEEE,* pp. 449-458, 2007.

[16] R. S. P.-T. d. B. A. P. Anna Sperotto, "Hidden Markov Model Modeling of SSH Brute-Force Attacks," *International Workshop on Distributed Systems: Operations and Management: Integrated Management of Systems, Services, Processes and People in IT,* pp. 164-176, 2009.

[17] M. Mesellem, "bWAPP an extremely buggy web app !," [Online]. Available: http://www.itsecgames.com/.