



การพัฒนาเครื่องมือสแกนหาช่องโหว่ของเว็บเซิฟเวอร์แบบอัตโนมัติบน  
พื้นฐานของเดเบียน

โดย

นายนพพร ชมภูโคตร

โครงการพิเศษนี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตร  
วิทยาศาสตรบัณฑิต  
สาขาวิชาวิทยาการคอมพิวเตอร์  
คณะวิทยาศาสตร์และเทคโนโลยี มหาวิทยาลัยธรรมศาสตร์  
ปีการศึกษา 2567  
ลิขสิทธิ์ของมหาวิทยาลัยธรรมศาสตร์

การพัฒนาเครื่องมือสแกนหาช่องโหว่ของเว็บไซฟ์เวอร์แบบอัตโนมัติบัน  
พื้นฐานของเดเบียน

โดย

นายนพพร ชุมภูโคตร

โครงการพิเศษนี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตร  
วิทยาศาสตรบัณฑิต  
สาขาวิชาวิทยาการคอมพิวเตอร์  
คณะวิทยาศาสตร์และเทคโนโลยี มหาวิทยาลัยธรรมศาสตร์  
ปีการศึกษา 2567  
ลิขสิทธิ์ของมหาวิทยาลัยธรรมศาสตร์

THE DEVELOPMENT OF DEBIAN-BASED AUTOMATION WEB  
SERVER VULNERABILITY SCANNER NET-TOOL

BY

MR.NOPPORN CHOMPHUKOAT

A FINAL-YEAR PROJECT REPORT SUBMITTED IN PARTIAL FULFILLMENT  
OF THE REQUIREMENTS FOR THE DEGREE OF BACHELOR OF SCIENCE  
COMPUTER SCIENCE  
FACULTY OF SCIENCE AND TECHNOLOGY  
THAMMASAT UNIVERSITY  
ACADEMIC YEAR 2024  
COPYRIGHT OF THAMMASAT UNIVERSITY

(4)

มหาวิทยาลัยธรรมศาสตร์  
คณะวิทยาศาสตร์และเทคโนโลยี

รายงานโครงการพิเศษ

ของ

นายนพพร ชุมภูโคตร

เรื่อง

การพัฒนาเครื่องมือสแกนหาข่องโหว่ของเว็บเชิฟเวอร์แบบอัตโนมัตินับพื้นฐานของเดเบียน

ได้รับการตรวจสอบและอนุมัติ ให้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตร  
หลักสูตรวิทยาศาสตรบัณฑิต สาขาวิชาวิทยาการคอมพิวเตอร์  
เมื่อ วันที่ 30 พฤษภาคม พ.ศ. 2567

อาจารย์ที่ปรึกษา



(ผศ. ดร. วิลาวรรณ รักพกวางศ์)

กรรมการสอบโครงการพิเศษ



(ผศ. ดร. วนิดา พุทธิวิทยา)

กรรมการสอบโครงการพิเศษ



(ผศ. ดร. ฐานปนา บุญชู)

(5)

มหาวิทยาลัยธรรมศาสตร์  
คณะวิทยาศาสตร์และเทคโนโลยี

รายงานโครงการพิเศษ

ของ

นายนพพร ชมภูโคตร

เรื่อง

การพัฒนาเครื่องมือสแกนหาซ่องโหว่ของเว็บไซต์ฟเวอร์แบบอัตโนมัติในการพัฒนาพื้นฐานของเดเบียน

ได้รับการตรวจสอบและอนุมัติ ให้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตร

หลักสูตรวิทยาศาสตรบัณฑิต สาขาวิชาวิทยาการคอมพิวเตอร์

เมื่อ วันที่ 30 พฤษภาคม พ.ศ. 2567

อาจารย์ที่ปรึกษา



(ผศ. ดร. วิลาวรรณ รักพกวางศ์)

กรรมการสอบโครงการพิเศษ



(ผศ. ดร. วนิดา พุทธิวิจaya)

กรรมการสอบโครงการพิเศษ



(ผศ. ดร. ข้าปนา บุญรุ่ง)

หัวข้อโครงการพิเศษ	การพัฒนาเครื่องมือสแกนหาซ่องโหว่ของเรือเชิร์ฟเวอร์
ชื่อผู้เขียน	แบบอัตโนมัติบันทึกฐานของเดเบียน
ชื่อปริญญา	นายนพพร ชมภูโคตร
สาขาวิชา/คณะ/มหาวิทยาลัย	วิทยาศาสตรบัณฑิต สาขาวิชาวิทยาการคอมพิวเตอร์
อาจารย์ที่ปรึกษาโครงการพิเศษ	สาขาวิชาวิทยาการคอมพิวเตอร์
ปีการศึกษา	คณะวิทยาศาสตร์และเทคโนโลยี
	มหาวิทยาลัยธรรมศาสตร์
	พศ.ดร.วิภาวรรณ รักพากวงศ์
	2567

### บทคัดย่อ

ความปลอดภัยของข้อมูลและสารสนเทศเป็นเรื่องที่องค์กรควรให้ความสำคัญเป็นอันดับแรก เนื่องจากการใช้งานทรัพยากรของเทคโนโลยีสารสนเทศเพิ่มขึ้นอย่างก้าวกระโดด ทำให้เกิดการโจมตีทางไซเบอร์และซ่องโหว่ขึ้นภายในองค์กร การทดสอบเจาะระบบจึงเป็นกระบวนการที่จำเป็นเพื่อใช้ในการตรวจสอบ ประเมิน วิเคราะห์ และแก้ไขซ่องโหว่ที่เกิดขึ้น ป้อยครั้งการทดสอบเจาะระบบด้วยตนเองจำเป็นที่จะต้องใช้ผู้เชี่ยวชาญเฉพาะทางในการทดสอบซ่องโหว่เหล่านั้น หากซ่องโหว่มีมากขึ้น ทางองค์กรจำเป็นต้องใช้บุคลากร ทรัพยากร งบประมาณ และเวลาจำนวนมากในการดำเนินการแต่ละครั้ง โครงการนี้จึงนำมาตรฐานการทดสอบระบบที่ครอบคลุมมาประยุกต์ใช้กับระบบอัตโนมัติ เพื่อพัฒนาเครื่องมืออัตโนมัติที่สามารถเพิ่มประสิทธิภาพในการทดสอบเจาะระบบ ลดความซับซ้อนในการปฏิบัติงาน ลดการแหงลงของมนุษย์และข้อผิดพลาดจากมนุษย์ อีกทั้งสามารถทดสอบซ้ำได้โดยไม่ต้องจัดเตรียมทรัพยากรใหม่ตั้งแต่เริ่มต้น

**คำสำคัญ:** ความปลอดภัยของข้อมูลและสารสนเทศ, โจมตีทางไซเบอร์, ช่องโหว่, การทดสอบเจาะระบบ, การทดสอบเจาะระบบด้วยตนเอง, ระบบอัตโนมัติ, เครื่องมืออัตโนมัติ, การแหงลงของมนุษย์, ข้อผิดพลาดจากมนุษย์, ทดสอบซ้ำ

Thesis Title	The Development of Debian-based Automation Web Server Vulnerability Scanner Net-tool
Author	Mr. Noppoorn Chomphukoat
Degree	Bachelor of Science
Major Field/Faculty/University	Computer Science Faculty of Science and Technology Thammasat University
Project Advisor	Asst.Prof. Dr. Wilawan Rukpakavong
Academic Years	2024

## ABSTRACT

Information and data security is a top priority for organizations, especially as the usage of information technology resources grows significantly. This growth has resulted in more cyberattacks and vulnerability within the fields. As a result, penetration testing has become a crucial procedure for investigating, assessing, analyzing, and fixing these vulnerabilities. Manual penetration testing frequently requires the knowledge of vulnerability assessment specialists. As the number of vulnerabilities increases, organizations must dedicate considerable staff, resources, budgets, and time to each operation. This project applies extensive penetration testing standards to automated systems with the goal of developing an automated tool to improve penetration testing efficiency. It aims to streamline operations, reduce human interaction and human error, and allow for repetitive tests without requiring fresh resource preparation from the ground.

**Keywords:** Information and data security, Cyberattacks, Vulnerability, Penetration testing, Manual penetration testing, Automated systems, Automated tool, Human interaction, Human error, Repetitive tests.

## กิตกรรมประกาศ

โครงการนี้จัดทำขึ้นเพื่อเป็นแนวทางในการพัฒนาเครื่องมือบนพื้นฐานของเดเบียนโดยนำระบบการทำงานแบบอัตโนมัติเข้ามาประยุกต์ใช้ในการพัฒนา โครงการนี้จะไม่สามารถเกิดขึ้นได้ หากไม่ได้รับการสนับสนุนและอนุมัติจากทาง ภาควิชาวิทยาการคอมพิวเตอร์

ผู้จัดทำขอขอบพระคุณ ผู้ที่มีส่วนร่วมทุกท่าน ที่ให้คำแนะนำทำให้เกิดความคิดริเริ่มในการสร้างและพัฒนาเครื่องมือนี้ โดยเฉพาะอย่างยิ่ง ศ. ดร. วิล่าวรรณ รักพากวงศ์ ซึ่งเป็นอาจารย์ที่ปรึกษาโครงการ คอยให้คำปรึกษา ช่วยตรวจสอบอักษร และ กำหนดขอบเขตในการพัฒนาของ โครงการนี้ จนสามารถทำให้โครงการนี้เสร็จสมบูรณ์

นาย นพพร ชุมภูโคตร

## สารบัญ

	หน้า
บทคัดย่อ	6
ABSTRACT	7
กิตติกรรมประกาศ	8
สารบัญ	9
สารบัญตาราง	13
สารบัญภาพ	14
<b>บทที่ 1 บหนำ</b>	<b>1</b>
1.1 ที่มาและความสำคัญของโครงการ	1
1.2 วัตถุประสงค์	2
1.3 ขอบเขตของโครงการ	2
1.4 ประโยชน์ของโครงการ	3
1.5 ข้อจำกัดของโครงการ	3
<b>บทที่ 2 วรรณกรรมและงานวิจัยที่เกี่ยวข้อง</b>	<b>4</b>
2.1 แนวคิดทฤษฎีที่เกี่ยวข้อง	4
2.1.1 Penetration Test	4
2.1.2 Vulnerability Assessment	5
2.1.3 OWASP Top Ten	5
2.1.4 Kali Linux operating system	6
2.1.5 Software Automation	7
2.2 เทคโนโลยีที่ใช้	7
2.2.1 VirtualBox	7

2.2.2 Kali Linux 2024.3 installer amd64 Disc Image File	8
2.2.3 Network Mapper	9
2.2.4 Uniscan	9
2.2.5 Nikto	9
2.2.6 Wapiti	10
2.2.7 Gobuster	10
2.2.8 Acunetix Vulnerabilities Testing Website	11
2.2.9 เครื่องขยายแวร์ x86-64 H Architecture	11
2.3 งานวิจัยที่เกี่ยวข้อง	12
2.3.1 Kali Linux based Empirical Investigation on Vulnerability Evaluating using Pen-Testing tools	12
2.3.2 Automated versus Manual Approach of Web Application Penetration Testing	13
2.4 แอปพลิเคชันที่คล้ายคลึง	14
2.4.1 Rapidscan v1.2 The Multi-Tool Web Vulnerability Scanner	14
2.5 ความคล้ายคลึงและความแตกต่างระหว่าง VULNscan กับ Rapidscan	16
2.5.1 สิ่งที่คล้ายคลึงกัน	16
2.5.2 สิ่งที่แตกต่างกัน	16
บทที่ 3 วิธีการวิจัย	17
3.1 ภาพรวมของโครงงาน	17
3.2 การวิเคราะห์ขอบเขตและความต้องการของเครื่องมือ	18
3.2.1 Use case diagram	18
3.2.2 รายละเอียดแผนภาพ use case diagram	18
3.2.2.1 Actors	18

3.2.2.2 Version command use case	18
3.2.2.3 Update command use case	19
3.2.2.4 Help command use case	19
3.2.2.5 Target command use case	19
3.2.2.6 Use cases relationship	20
3.2.3 Activity diagram	20
3.2.3.1 Activity diagram คำสั่ง -v	20
3.2.3.2 Activity diagram คำสั่ง -h	21
3.2.3.3 Activity diagram คำสั่ง -n	21
3.2.3.4 Activity diagram คำสั่ง target	22
3.3 การดำเนินงาน	22
3.3.1 การรวบรวมข้อมูล	22
3.3.1.1 ข้อมูลสถิติช่องไฟด้านความปลอดภัย	22
3.3.1.2 เครื่องมือที่ใช้ในการพัฒนา	23
3.3.1.3 การวิเคราะห์และจัดลำดับความร้ายแรงของช่องไฟ	23
3.3.2 ตัวอย่างข้อมูล	23
3.3.2.1 ตัวอย่างข้อมูลเอกสาร OWASP Top Ten	23
3.3.2.2 ตัวอย่างข้อมูลผลลัพธ์ที่ได้จากเครื่องมือ	24
3.3.2.3 ตัวอย่างข้อมูลคะแนน Common Vulnerability Scoring System	31
3.4 ประเด็นที่น่าสนใจและท้าทาย	32
3.4.1 การจัดการข้อมูลผลลัพธ์ที่ได้จากเครื่องมือ	32
3.4.2 ระยะเวลาในการทำงานของเครื่องมือที่ใช้ในการพัฒนา	32
3.5 ผลลัพธ์ที่คาดหวัง	32
3.6 ระบบต้นแบบและผลลัพธ์เบื้องต้น	33

บทที่ 4 ผลการดำเนินงาน	40
4.1 ผลการพัฒนาโครงงาน	40
4.2 ภาพตัวอย่างเครื่องมือ	40
บทที่ 5 สรุป	49
5.1 สรุปผลการพัฒนาโครงงาน	49
5.1.1 การระบุช่องโหว่ได้อย่างครอบคลุม	49
5.1.2 การทำงานอัตโนมัติที่ช่วยลดภาระการทำงานของนักทดสอบ	49
5.1.3 การแสดงลำดับความสำคัญของปัญหาช่องโหว่	49
5.1.4 การรายงานผลที่เข้าใจง่าย	49
5.2 แนวคิดในการต่อยอดและปรับปรุง	50
5.2.1 แนวทางการเพิ่มความน่าเชื่อถือของช่องโหว่ที่พบ	50
5.2.2 การขยายขอบเขตและเพิ่มความครอบคลุมของช่องโหว่	50
5.2.3 การเปิดรับ Feedback และความร่วมมือจากผู้ใช้งาน	50
5.2.4 พัฒนาปรับปรุงเครื่องมือจากรูปแบบ CLI ให้เป็นเครื่องมือแบบ GUI	51
5.3 สิ่งที่ได้เรียนรู้หลังจากพัฒนาโครงงาน	51
5.3.1 Environment ที่เหมาะสมในการใช้งานเครื่องมือ VULNscan	51
5.3.2 การเลือกเป้าหมายสำหรับการทดสอบ	52
5.3.3 การนำเครื่องมือไปใช้งานต้องปฏิบัติอย่างไร	52
5.3.4 รูปแบบการทำงานของเครื่องมือ	52
รายการอ้างอิง	54

(13)

สารบัญตาราง

หน้า

ตารางที่ 2.1 ตารางผลลัพธ์ของการทดสอบเจาะระบบแบบอัตโนมัติและการทดสอบ

13

## สารบัญภาพ

	หน้า
ภาพที่ 2.1 ตัวอย่างเริ่มใช้ต์ที่ใช้ในการทดสอบเครื่องมือของ Acunetix	12
ภาพที่ 2.2 คำสั่งเรียกใช้งาน Rapidscan	15
ภาพที่ 2.3 หน้าหลักเมื่อเรียกใช้งาน Rapidscan	15
ภาพที่ 2.4 ผลลัพธ์การสแกนของ Rapidscan	15
ภาพที่ 3.1 แผนภาพกรณีใช้งานของเครื่องมือ	17
ภาพที่ 3.2 แผนภาพพฤติกรรมการใช้งานคำสั่ง -v	19
ภาพที่ 3.3 แผนภาพพฤติกรรมการใช้งานคำสั่ง -h	20
ภาพที่ 3.4 แผนภาพพฤติกรรมการใช้งานคำสั่ง -n	20
ภาพที่ 3.5 แผนภาพพฤติกรรมการใช้งานคำสั่ง target	21
ภาพที่ 3.6 ตัวอย่างช่องโหว่ที่เกิดขึ้นบนเว็บไซฟ์เวอร์ในเอกสาร OWASP Top Ten	22
ภาพที่ 3.7 ตัวอย่างรายละเอียด OWASP Top Ten	23
ภาพที่ 3.8 ผลลัพธ์จากการใช้เครื่องมือ nmap เพื่อหา MySQL, ORACLE, MS-SQL	23
ภาพที่ 3.9 ผลลัพธ์จากการใช้เครื่องมือ Uniscan เพื่อทำ SQL Injection	24
ภาพที่ 3.10 ผลลัพธ์จากการใช้เครื่องมือ Uniscan เพื่อทำ Cross-site Scripting	24
ภาพที่ 3.11 ผลลัพธ์จากการใช้เครื่องมือ Uniscan เพื่อทำ Brute Force File Name	25
ภาพที่ 3.12 ผลลัพธ์จากการใช้เครื่องมือ Uniscan เพื่อทำ Brute Force Directory	25
ภาพที่ 3.13 ผลลัพธ์จากการใช้เครื่องมือ nikto เพื่อทำ Server-Side Request Forgery (SSRF)	26
ภาพที่ 3.14 ผลลัพธ์จากการใช้เครื่องมือ Wapiti ในการสรุปผลการตรวจสอบ	27
ภาพที่ 3.15 ผลลัพธ์จากการใช้เครื่องมือ Wapiti ในการตรวจสอบช่องโหว่เพิ่มเติม	27
ภาพที่ 3.16 ผลลัพธ์จากการใช้เครื่องมือ Wapiti ในการตรวจสอบ SQL Injection	28
ภาพที่ 3.17 ผลลัพธ์จากการใช้เครื่องมือ Wapiti ในการตรวจสอบ Cross Site Scripting	28
ภาพที่ 3.18 ผลลัพธ์จากการใช้เครื่องมือ Wapiti ในการตรวจสอบ Internal Server Error	29
ภาพที่ 3.19 ผลลัพธ์จากการใช้เครื่องมือ gobuster เพื่อทำ directory traversal	29
ภาพที่ 3.20 ผลลัพธ์จากการใช้เครื่องมือ gobuster เพื่อทำ File Brute Force	30
ภาพที่ 3.21 ตัวอย่างเครื่องคำนวณคะแนน CVSS	30

ภาพที่	3.22 ตัวอย่างเครื่องคำนวณคะแนน CVSS เมื่อกรอกค่าตามมาตรฐาน	31
ภาพที่	3.23 การติดตั้งเครื่องมือบนระบบปฏิบัติการ Kali Linux	32
ภาพที่	3.24 โครงสร้างของไดเรกทอรีเครื่องมือเบื้องต้น	33
ภาพที่	3.25 ใช้คำสั่ง python3 scanner.py -v เพื่อเรียกตรวจสอบเวอร์ชัน	33
ภาพที่	3.26 ใช้คำสั่ง python3 scanner.py -u เพื่อเรียกอัพเดตเวอร์ชัน	34
ภาพที่	3.27 ภาพคู่มือที่แสดงเมื่อเรียกใช้คำสั่ง -h	34
ภาพที่	3.28 ใช้คำสั่ง python3 scanner.py www.example.com	35
ภาพที่	3.29 การตรวจสอบสภาพแวดล้อมว่าพร้อมใช้งานหรือไม่	35
ภาพที่	3.30 สแกนหาช่องโหว่และไม้พบรหัสช่องโหว่	36
ภาพที่	3.31 สแกนแล้วพบช่องโหว่บนเว็บเซิร์ฟเวอร์	36
ภาพที่	3.32 รายงานผลลัพธ์เมื่อเสร็จสิ้นกระบวนการ	37
ภาพที่	3.33 ตัวอย่างของ log file	37
ภาพที่	3.34 หน้าเว็บเพจ Vulnerabilities Summary ของไฟล์ report.html	38
ภาพที่	3.35 หน้าเว็บเพจ log visualization ของไฟล์ report.html	38
ภาพที่	4.1 การแสดงผลของคำสั่ง -V หรือ Version	39
ภาพที่	4.2 การแสดงผลของคำสั่ง -U หรือ Update	40
ภาพที่	4.3 การแสดงผลของคำสั่ง -H หรือ Help	40
ภาพที่	4.4 การแสดงผลเมื่อใส่เป้าหมายเว็บเซิร์ฟเวอร์ที่ต้องการตรวจสอบ	41
ภาพที่	4.5 การแสดงผลการแจ้งเตือนเมื่อตรวจพบช่องโหว่	41
ภาพที่	4.6 การแสดงผลข้อมูลโดยภาพรวมและการบันทึกข้อมูล	42
ภาพที่	4.7 ตัวอย่างรายละเอียดโดยสรุปจากไฟล์ HTML	42
ภาพที่	4.8 ตัวอย่างรายละเอียดแผนภูมิจากไฟล์ HTML	42
ภาพที่	4.9 ตัวอย่างรายละเอียดเชิงเทคนิคจากไฟล์ HTML	43
ภาพที่	4.10 ตัวอย่างรายละเอียดเชิงเทคนิคจากไฟล์ scan_report.txt	43
ภาพที่	4.11 ผลลัพธ์ทั้งหมดจากการทำงานของเครื่องมือจากไฟล์ raw_report.txt	44

## บทที่ 1

### บทนำ

#### 1.1 ที่มาและความสำคัญของโครงการ

การทดสอบเจาะระบบถูกใช้เพื่อตรวจสอบหาข้อบกพร่องหรือช่องโหว่ที่เกิดขึ้นภายในระบบขององค์กรและยังสามารถช่วยให้นักพัฒนานำผลการทดสอบไปใช้เพื่อปรับปรุงระบบรักษาความปลอดภัยของข้อมูลให้ตรงกับมาตรฐานขององค์กร จึงเป็นเรื่องที่สำคัญสำหรับทุก ๆ องค์กรที่ต้องปกป้องรายละเอียดและข้อมูลภายในการโจรตีของผู้คุกคามจากภายนอกและพยายามติดตามปัญหาที่เกี่ยวข้องกับความปลอดภัยอยู่เสมอ อีกทั้งการทดสอบระบบทำให้การจัดระดับความสำคัญของเหตุการณ์ที่เกิดขึ้นนั้นง่ายขึ้น ทางองค์กรสามารถวางแผนสำหรับการบริหารผลกระทบที่เกิดขึ้นได้ถูกจุดและมีประสิทธิภาพ (Abu-Dabaseh & Alshammary, 2018)

การทดสอบระบบสามารถทำได้โดยการนำช่องโหว่มาใช้ในการโจมตีระบบขององค์กรนั้น ๆ และทำการทดสอบรูปแบบเพื่อแจ้งไปยังองค์กร ความแตกต่างของการทดสอบเจาะระบบกับการโจมตีจากภายนอกคือ การทดสอบเจาะระบบนั้นทางองค์กรจะมีรอบของการทดสอบ มีเป้าหมายที่ชัดเจนให้กับนักทดสอบเพื่อประเมินความเสี่ยงขององค์กรและได้รับความยินยอมจากองค์กรเหล่านั้นในการทดสอบเจาะระบบ

ประเภทของการทดสอบเจาะระบบสามารถแบ่งออกได้เป็นสามประเภทที่แตกต่างกันได้แก่ gray hat, black hat และ white hat การทดสอบประเภท white hat จะเป็นการทดสอบที่ปฏิบัติตามกฎขององค์กรที่ให้ไว้และทางพนักงานภายในองค์กรสามารถช่วยทำได้ ในขณะที่ black hat เป็นการทดสอบเจาะระบบที่ไม่สนใจกฎขององค์กร โดยทางพนักงานขององค์กรนั้นจำเป็นที่จะต้องรับมือกับการเจาะระบบจากนักทดสอบเอง แต่การทดสอบเจาะระบบแบบ gray hat จะเป็นการสมมติฐานว่า white hat และ black hat กล่าวคือทางนักทดสอบและพนักงานขององค์กรช่วยกันวางแผนในการทำการทดสอบเจาะระบบนั้น ๆ (Shebli & Beheshti, 2018)

ถึงแม้การทดสอบเจาะระบบจะเป็นกระบวนการที่สำคัญเป็นอย่างมากต่อองค์กร แต่การนำมาประยุกต์ใช้ในองค์กรนั้นมีปัญหาที่ตามมา เนื่องจากการดำเนินการทดสอบด้วยตนเองต้องการนักทดสอบที่มีความเชี่ยวชาญเกี่ยวกับช่องโหว่แต่ละด้านที่เกิดขึ้น เมื่อช่องโหว่มากขึ้น ทางองค์กรจึงต้องใช้ทรัพยากร งบประมาณ และเวลาจำนวนมากในการทำการทดสอบเจาะระบบ ดังนั้นการนำ

ระบบอัตโนมัติมาใช้ในการทดสอบเจาะระบบจึงเป็นทางออกสำหรับองค์กรที่ต้องการประหยัดทั้งทรัพยากร งบประมาณและเวลาในการดำเนินการ (Kumar et al., 2023)

โครงการนี้จะนำพื้นฐานในการทำการทดสอบเจาะระบบมาประยุกต์ใช้ร่วมกับระบบอัตโนมัติ เพื่อพัฒนาและออกแบบเครื่องมือทางด้านความปลอดภัยที่สามารถตรวจหาช่องโหว่และข้อผิดพลาดที่อาจเกิดขึ้นจากมนุษย์ โดยนำมาตรฐานในการทดสอบเจาะระบบมาใช้ ส่งผลให้สามารถเพิ่มประสิทธิภาพในการดำเนินการของทีมรักษาความปลอดภัย ลดความซับซ้อนในการปฏิบัติงาน และสามารถทดสอบช้าได้โดยไม่ต้องจัดเตรียมทรัพยากรใหม่ตั้งแต่เริ่มต้น

## 1.2 วัตถุประสงค์ของโครงการ

1. เพื่อพัฒนาเครื่องมือสแกนอัตโนมัติสำหรับตรวจสอบช่องโหว่ของเว็บไซต์ฟิวเวอร์ ที่สามารถระบุช่องโหว่อย่างน้อย 5 ประเภท ตามมาตรฐาน OWASP Top 10 ได้โดยอัตโนมัติ
2. เพื่อออกแบบและพัฒนาเครื่องมือสแกนอัตโนมัติที่รวมฟังก์ชันของหลายเครื่องมือทดสอบเจาะระบบให้สามารถทำงานร่วมกันในระบบเดียวได้อย่างมีประสิทธิภาพ
3. เพื่อพัฒนาเครื่องมือสแกนอัตโนมัติสำหรับตรวจสอบช่องโหว่ของเว็บไซต์ฟิวเวอร์ ที่มีระบบแจ้งเตือนให้แก่ผู้ใช้งานเมื่อตรวจพบช่องโหว่ต่าง ๆ ระหว่างการสแกน

## 1.3 ขอบเขตของโครงการ

ขอบเขตของโครงการนี้ครอบคลุมการพัฒนาเครื่องมือตรวจสอบช่องโหว่ของเว็บไซต์ฟิวเวอร์ โดยเน้นการตรวจหาช่องโหว่หลัก 5 ประเภท ได้แก่ Broken Access Control, SQL Injection, Cross-Site Scripting (XSS), Remote Code Execution (RCE), และ Server-Side Request Forgery (SSRF) โดยการตรวจสอบจะสามารถระบุและป้องกันช่องโหว่สำคัญ เช่น การเข้าถึงข้อมูลที่ไม่ถูกต้องผ่านช่องโหว่ของการควบคุมสิทธิ์การเข้าถึง (Path Traversal, OpenSSL CCS Injection), การโจมตี SQL Injection, Cross-site scripting (XSS), Remote code execution (RCE) และการใช้ Server-side request forgery (SSRF) นอกจากนี้ เครื่องมือนี้จะสามารถทำการตรวจสอบช่องโหว่เกี่ยวกับ SSL และ Load Balancer ของเซิร์ฟเวอร์

เครื่องมือต่าง ๆ ที่นำมาใช้ในโครงการประกอบด้วย Nmap, Uniscan, Nikto, Wapiti, Gobuster และ Lbd ซึ่งแต่ละเครื่องมือมีความสามารถในการตรวจจับและวิเคราะห์ช่องโหว่ที่แตกต่างกันไป เช่น Nmap สามารถตรวจสอบช่องโหว่ OpenSSL CCS Injection และฐานข้อมูล SQL, Uniscan ใช้สำหรับ Brute Force ไฟล์และโฟลเดอร์บนโดเมน รวมถึงตรวจสอบ XSS และ SQLi, Nikto สามารถตรวจสอบปัญหาต่าง ๆ บนเซิร์ฟเวอร์ รวมถึงทำการทดสอบ SSL, ส่วน Wapiti เน้นการตรวจสอบช่องโหว่ SQL Injection และ XSS โดยเฉพาะ ในขณะที่ใช้ Gobuster ช่วยในการ

Brute Force ไฟล์และโฟลเดอร์ และ Lbd ตรวจสอบการใช้ Load Balancer เครื่องมือที่จะพัฒนาขึ้นมาได้สามารถใช้ร่วมกับเซิร์ฟเวอร์ที่นิยมใช้ในปัจจุบัน ได้แก่ Nginx, Apache, Apache Tomcat, และ IIS อีกทั้งยังสามารถใช้กับเว็บเซิร์ฟเวอร์ที่อยู่นอกวง LAN ได้อีกด้วย

#### 1.4 ประโยชน์ของโครงการ

1. องค์กรสามารถจัดการความปลอดภัยของเว็บเซิร์ฟเวอร์ได้ง่ายและมีประสิทธิภาพมากขึ้น จากรายงานอัตโนมัติที่นำเสนอด้วยข้อมูลอย่างชัดเจน โดยระบุประเภท ความรุนแรง และลำดับความสำคัญของช่องโหว่ ช่วยในการวางแผนจัดการความปลอดภัยได้ตรงจุดมากยิ่งขึ้น
2. องค์กรสามารถตอบสนองภัยคุกคามได้รวดเร็วและแม่นยำมากขึ้น ผ่านการแจ้งเตือนช่องโหว่ทันทีที่ตรวจพบ ช่วยลดความเสี่ยงที่อาจขยายวงกว้าง และยังช่วยรักษาภาพลักษณ์และความน่าเชื่อถือในเรื่องความปลอดภัยของข้อมูลต่อผู้เกี่ยวข้องอีกด้วย
3. ช่วยนักทดสอบเจาะระบบประหดเวลาในการทำงานพื้นฐานที่ซ้ำๆ และสามารถนำเวลาไปวิเคราะห์ปัญหาเชิงลึก วางแผนการป้องกันและแก้ไขปัญหาอย่างมีประสิทธิภาพมากขึ้น ส่งผลให้องค์กรได้รับการปกป้องอย่างยั่งยืน
4. องค์กรสามารถประหดเวลา ทรัพยากร และงบประมาณในการทดสอบเจาะระบบ จากการใช้งานกระบวนการทดสอบแบบอัตโนมัติที่ลดความซับซ้อน ช่วยบริหารจัดการงบประมาณและทรัพยากรด้านความปลอดภัยสารสนเทศได้คุ้มค่าและมีประสิทธิภาพยิ่งขึ้น

#### 1.5 ข้อจำกัดของโครงการ

1. เครื่องคอมพิวเตอร์ที่สามารถใช้งานเครื่องมือได้คือ Debian-based Linux Distros machine เท่านั้น เนื่องจากมีเครื่องมือเครื่องมือโอเพ่นซอร์สที่หลากหลายทำให้เหมาะสมแก่การพัฒนาเครื่องมือ
2. ระบบปฏิบัติการที่สามารถใช้งานเครื่องมือได้คือ Kali Linux OS
3. เครื่องมือจะถูกพัฒนาขึ้นมาจากการภาษาโปรแกรม Python เนื่องจากความหลากหลายของ Library ในภาษา ทำให้เป็นภาษาที่ยืดหยุ่นต่อการพัฒนาเครื่องมือ
4. ผู้ใช้งานจำเป็นที่จะต้องมีพื้นฐานเรื่องความปลอดภัยทางไซเบอร์ ภัยคุกคาม ความเสี่ยง และช่องโหว่ต่าง ๆ ที่เกิดขึ้นภายในระบบ

## บทที่ 2

### วรรณกรรมและงานวิจัยที่เกี่ยวข้อง

#### 2.1 แนวคิดทฤษฎีที่เกี่ยวข้อง

##### 2.1.1 Penetration Test

Penetration test (Pen Test) เป็นหลักการที่ถูกสร้างขึ้นเพื่อทดสอบโครงสร้างภายในองค์กรที่เชื่อมโยงกับคอมพิวเตอร์และเครือข่าย ไม่ว่าจะเป็นฮาร์ดแวร์ ซอฟต์แวร์ หรือแม้กระทั่งบุคลากรภายในองค์กร รวมทั้งการวิเคราะห์ทั้งระบบภายในองค์กรเพื่อตรวจสอบหาช่องโหว่ เช่นการตั้งค่าที่ผิดพลาดภายในระบบ การทำงานที่ผิดพลาดของฮาร์ดแวร์และซอฟต์แวร์ อีกทั้งยังมีการตรวจสอบหาจุดบกพร่องภายในองค์กรได้ (Shebli & Beheshti, 2018)

การทดสอบเจาะระบบ (Penetration Test หรือ Pen Test) ช่วยให้องค์กรสามารถระบุและประเมินความเสี่ยงที่อาจเกิดขึ้นเมื่อผู้โจมตีเข้าถึงระบบและเครือข่ายขององค์กร สามารถวางแผนแก้ไขซ่องโหว่ก่อนเกิดการโจมตีขึ้นจริง ซึ่งช่วยลดความสูญเสียทางการเงินและป้องข้อมูลสำคัญที่อาจส่งผลกระทบต่อความเชื่อมั่นของลูกค้า (Moreno et al., 2025) นอกจากนี้ยังแก้ไขปรับปรุงนโยบายและกฎระเบียบภายในองค์กรให้สอดคล้องกับข้อกำหนดของหน่วยงานกำกับดูแลและช่วยสร้างภาพลักษณ์ที่ดีให้กับองค์กร

การทดสอบเจาะระบบเป็นกระบวนการเชิงรุกที่ช่วยสร้างความตระหนักรู้เรื่องความปลอดภัยทางไซเบอร์ในทุกระดับขององค์กรผ่านการฝึกอบรมและการประเมินความรู้ด้านความปลอดภัยของพนักงาน รวมถึงประสิทธิภาพของนโยบาย (Policy) และผลิตภัณฑ์ด้านความปลอดภัย การทดสอบเจาะระบบยังมีส่วนช่วยในการวางแผนการลงทุนด้านความปลอดภัยทางไซเบอร์และการปรับปรุงการตั้งค่าการทดสอบเพื่อป้องกันความเสี่ยงที่ตรวจพบอย่างมีประสิทธิภาพ แม้ว่าการทดสอบนี้จะต้องใช้เวลา ความเชี่ยวชาญ และความพยายาม แต่ถือเป็นเครื่องมือประกันคุณภาพที่มีประโยชน์สำหรับทั้งธุรกิจและการดำเนินงาน

### 2.1.2 Vulnerability Assessment

การประเมินช่องโหว่ (Vulnerability Assessment : VA) คือการประเมินหาข้อมูลที่สำคัญต่าง ๆ บนเป้าหมายและการสแกนเป้าหมายเพื่อค้นหาช่องโหว่ที่เป็นข้อบกพร่องบนระบบ การประเมินหาสาเหตุที่ทำให้เกิดช่องโหว่เหล่านั้น เพื่อป้องกันการถูกโจมตีจากบุคคลภายนอก (Scarfone et al., 2008)

การประเมินช่องโหว่เป็นวิธีการเชิงรุกที่มีระบบในการค้นหาช่องโหว่ต่าง ๆ ในระบบทั้งที่รู้จักและไม่รู้จัก โดยสอดคล้องกับข้อกำหนดของมาตรฐานอุตสาหกรรมที่กำหนดให้มีการประเมินช่องโหว่เพื่อให้เป็นไปตามข้อกำหนดด้านความปลอดภัย การประเมินนี้ทำได้โดยใช้เครื่องมือในการสแกน ซึ่งเป็นแนวทางแบบสมมุติว่าการทดสอบอัตโนมัติและการวิเคราะห์จากผู้เชี่ยวชาญในองค์กร (Scarfone et al., 2008)

ข้อดีของการประเมินช่องโหว่คือ ช่วยให้สามารถตรวจสอบความปลอดภัยได้จำนวนมากผ่านระบบอัตโนมัติ และสามารถใช้เป็นการทดสอบการแก้ไขขั้นต้นที่มีประโยชน์เนื่องจากเครื่องมือที่สามารถหาได้จ่ายอย่างไรก็ตามการประเมินช่องโหว่ก็มีข้อเสีย เช่น การสร้างข้อมูลที่ไม่จำเป็นมากเกินไปและอาจมีผลลัพธ์ที่ผิดพลาดเนื่องจากเป็นการประเมินขั้นต้น ไม่สามารถระบุการโจมตีที่ซับซ้อนได้ อีกทั้งคำแนะนำที่ได้รับนั้นยังขึ้นอยู่กับเครื่องมือที่ใช้งานซึ่งอาจจะเป็นคำแนะนำที่ไม่เหมาะสม

### 2.1.3 OWASP Top Ten

OWASP Top Ten คือเอกสารที่จัดทำโดยองค์กรไม่แสวงหาผลกำไร Open Web Application Security Project (OWASP) ซึ่งมุ่งเน้นในการปรับปรุงความปลอดภัยของเว็บไซต์และเว็บเซิร์ฟเวอร์ (OWASP, 2021) โดยในทุกปี OWASP จะจัดประชุมเพื่อรวบรวมข้อมูลเกี่ยวกับช่องโหว่ ความเสี่ยงและภัยคุกคามที่เกี่ยวข้องกับเว็บแอปพลิเคชัน ซึ่งจากการประชุมนี้จะถูกจัดทำเป็นเอกสาร OWASP Top Ten เพื่อเตือนภัยและแนะนำแนวทางป้องกันช่องโหว่ที่พบในระบบต่าง ๆ

OWASP Top Ten เป็นเอกสารที่จัดทำขึ้นเพื่อจัดหมวดหมู่ 10 อันดับความเสี่ยงที่สำคัญที่สุดที่เกี่ยวข้องกับเว็บแอปพลิเคชันและเว็บเซิร์ฟเวอร์ ข้อมูลในเอกสารนี้มาจากการวิเคราะห์และประเมินภัยคุกคามที่พบในระบบจริง โดยมีจุดประสงค์เพื่อเป็นคู่มือ (Handbook) ในการป้องกันและแก้ไขช่องโหว่ที่อาจเกิดขึ้น หลักการสำคัญของเอกสารนี้คือการสร้างความเข้าใจให้กับนักพัฒนา

และเจ้าของระบบเกี่ยวกับช่องโหว่ที่อาจส่งผลกระทบต่อระบบ มุ่งเน้นการป้องกันและปรับปรุงความปลอดภัยเพื่อปกป้องข้อมูลส่วนตัวและระบบขององค์กร

OWASP Top Ten ประกอบด้วย 10 ประเภทช่องโหว่ที่เกิดขึ้นบ่อย เช่น Broken Access Control (BAC), SQL Injection (SQLi), Cross-Site Scripting (XSS), Remote Code Execution (RCE) และ Server-Side Request Forgery (SSRF) ข้อมูลนี้ถูกรวบรวมจากการประชุมประจำปีและการวิเคราะห์ภัยคุกคาม (Threat) ที่เกิดขึ้นจริง جانนั้นจัดเรียงเป็นลำดับสิบอันดับที่มีความเสี่ยงสูงที่สุด นักพัฒนาและผู้ดูแลระบบสามารถนำเอกสารนี้ไปประยุกต์ใช้ในกระบวนการพัฒนาและทดสอบความปลอดภัยของระบบเพื่อเพิ่มประสิทธิภาพในการป้องกันและลดความเสี่ยงจากการถูกโจมตี (OWASP, 2021)

#### 2.1.4 Kali Linux operating system

Kali Linux เป็นระบบปฏิบัติการโอเพ่นซอร์สที่พัฒนาขึ้นบนพื้นฐานของ Debian โดยเน้นการใช้งานในด้านการทดสอบความปลอดภัยทางไซเบอร์ ซึ่งมีความเสถียรและมีประสิทธิภาพในการจัดการแพ็กเกจจากคลังซอฟต์แวร์ของ Debian ทำให้การติดตั้งเครื่องมือต่าง ๆ ทำได้อย่างง่ายดาย (Hertzog et al., 2017) ไม่ว่าจะเป็น Nmap สำหรับตรวจสอบช่องโหว่ของพอร์ตที่เปิดไว้, Uniscan สำหรับตรวจสอบ Cross-Site Scripting และ SQLi, รวมถึง Nikto, Wapiti, และ Gobuster ที่ครอบคลุมการตรวจสอบช่องโหว่หลายประเภท เช่น SQL Injection และ Cross-Site Scripting ระบบปฏิบัติการนี้ยังรองรับการทำงานในสภาพแวดล้อมหลากหลาย เช่น เครื่องจำลองเสมือน (Virtual Machine : VM) และ Cloud-based ช่วยลดความซับซ้อนในการทดสอบและเพิ่มความเร็วในการดำเนินงาน เพียงใช้คำสั่ง apt-get install ในการติดตั้งเครื่องมือก็สามารถเริ่มการทดสอบได้อย่างรวดเร็วและมีประสิทธิภาพ

Kali Linux จึงเป็นระบบปฏิบัติการโอเพ่นซอร์สที่เหมาะสมทั้งสำหรับนักทดสอบมืออาชีพและผู้เริ่มต้น โดยระบบนี้ถูกพัฒนาขึ้นบนพื้นฐานของ Debian ซึ่งมีความเสถียรและประสิทธิภาพสูงในการจัดการแพ็กเกจ ทำให้การติดตั้งเครื่องมือต่าง ๆ สำหรับการตรวจสอบช่องโหว่หลายประเภททำได้อย่างง่ายดาย รองรับการทำงานในสภาพแวดล้อมที่หลากหลายทั้ง Virtual Machine และ Cloud-based Kali Linux ช่วยให้การทดสอบและการวิเคราะห์ช่องโหว่สามารถทำได้อย่างรวดเร็วและมีประสิทธิภาพ ด้วยคำสั่งเพียงไม่กี่ขั้นตอน ลดความซับซ้อนของการทดสอบความ

ปลอดภัยไซเบอร์ อีกทั้งยังครอบคลุมทั้งการประเมินความเสี่ยงและการทดสอบความปลอดภัยบนระบบเซิร์ฟเวอร์อีกด้วย

### 2.1.5 Software Automation

ซอฟต์แวร์แบบอัตโนมัติ (Software Automation) คือการใช้ซอฟต์แวร์ เครื่องมือ และเทคโนโลยี เพื่อปรับปรุง ควบคุมและดำเนินงานหรือกระบวนการต่าง ๆ โดยไม่มีการแทรกแซงของมนุษย์โดยตรง โดยส่วนใหญ่เกี่ยวข้องกับการทำงานแบบบวนซ้ำด้วยตนเองหรือดำเนินงานตามขั้นตอนการทำงานที่ถูกออกแบบโดยอัตโนมัติ ที่สร้างโดยศูนย์อัลกอริทึมหรือระบบที่สามารถดำเนินโดยอัตโนมัติได้ จุดมุ่งหมายของซอฟต์แวร์แบบอัตโนมัติที่สำคัญ คือเพิ่มประสิทธิภาพความแม่นยำและเพิ่มอัตราการผลิตโดยการลดความผิดพลาดของมนุษย์ลง อีกทั้งยังประหยัดเวลา และสามารถดำเนินงานได้ในวงกว้าง ครอบคลุมการใช้งานที่หลากหลาย (Ariah & Brightwood, 2024)

ซอฟต์แวร์แบบอัตโนมัติจะมีประโยชน์ที่โดดเด่นในด้านการเพิ่มประสิทธิภาพและการผลิตภาพ (Efficiency and Productivity) ยกตัวอย่างเรื่องความไหหลีนในการทำงาน การนำระบบอัตโนมัติมาประยุกต์ใช้ช่วยลดการทำงานด้วยตนเอง (Manual) ลดการทำซ้ำ (Repetitive) ประหยัดเวลาอีกทั้งยังสามารถลดข้อผิดพลาดที่เกิดจากมนุษย์ (Human error) ได้อีกด้วย และอีกหนึ่งตัวอย่างประโยชน์ที่เห็นได้ชัดคือความสามารถในการปรับขยายขนาด (Scalability) และความยืดหยุ่น (Flexibility) หากทางองค์กรต้องการปรับเปลี่ยนหรือปรับปรุงให้กระบวนการเป็นไปตามมาตรฐานสากลต่าง ๆ สามารถปรับได้ตามความเหมาะสมขององค์กร และสามารถนำบุคลากรไปดำเนินงานที่มีความสำคัญกว่าได้ ส่งผลให้องค์กรเติบโตและมีคุณภาพ

## 2.2 เทคโนโลยีที่ใช้

### 2.2.1 VirtualBox

VirtualBox เป็นซอฟต์แวร์โอเพ่นซอร์ส การจำลองเสมือนที่มีประสิทธิภาพ ซึ่งพัฒนาโดยบริษัท Oracle ช่วยให้ผู้ใช้สามารถใช้งานและติดตั้งระบบปฏิบัติการหลายตัวพร้อมกันบนเครื่องคอมพิวเตอร์จริงเครื่องเดียว (Oracle Corporation, 2022) โดยเปิดตัวครั้งแรกในปี 2007 VirtualBox ได้รับความนิยมอย่างแพร่หลายจากนักพัฒนา ผู้เชี่ยวชาญด้านไอที และนักการศึกษาเนื่องจากใช้งานง่ายและมีฟีเจอร์ที่แข็งแกร่ง ในสภาพแวดล้อมการคอมพิวเตอร์ที่มีการจำลองเสมือน

เพิ่มมากขึ้นในปัจจุบัน การทำความเข้าใจเกี่ยวกับ VirtualBox เป็นสิ่งจำเป็นสำหรับการใช้ทรัพยากรอย่างมีประสิทธิภาพและการทดสอบระบบ

VirtualBox ถูกออกแบบให้เป็นแอปพลิเคชันการจำลองเสมือนข้ามแพลตฟอร์ม ซึ่งช่วยให้ผู้ใช้สามารถสร้างและจัดการเครื่องเสมือนที่ทำงานเป็นคอมพิวเตอร์อิสระภายในระบบปฏิบัติการของเครื่องหลัก (Host Machine) หลักการสำคัญของ VirtualBox ประกอบด้วยการแอปเรียกใช้อาร์ดแวร์ ช่วยให้เครื่องเสมือน สามารถแบ่งปันทรัพยากรทางกายภาพ (Physical Resource) ได้ การแยกสภาพแวดล้อม (Environment) เพื่อให้มั่นใจว่าการกระทำในสภาพแวดล้อมเสมือน (Virtual Environment) หนึ่งไม่ส่งผลกระทบต่ออีกสภาพแวดล้อมหนึ่ง และฟังก์ชัน snapshots เพื่อการกู้คืนไปยังสถานะก่อนหน้าได้ องค์ประกอบของ VirtualBox ประกอบด้วยเครื่องเสมือน ภาคิดสก์เสมือน (VDIs) และอินเทอร์เฟซเครือข่ายเสมือนซึ่งทั้งหมดมีส่วนช่วยในฟังก์ชันการทำงานของมัน (Oracle Corporation, 2022)

กระบวนการที่นำไปในการใช้ VirtualBox เริ่มจากการดาวน์โหลดและติดตั้งซอฟต์แวร์ การสร้างและกำหนดค่าเครื่องเสมือน การติดตั้ง Guest OS และการจัดการเครื่องเสมือนผ่านฟีเจอร์ต่างๆ เช่น สแนปช็อตและการโคลน ด้วยแอปพลิเคชันที่ครอบคลุมตั้งแต่การพัฒนาซอฟต์แวร์ การศึกษา การบริหารจัดการระบบไอที ไปจนถึงการทดสอบด้านความปลอดภัย VirtualBox จึงเป็นเครื่องมือที่สำคัญในการสร้างสภาพแวดล้อมที่แยกออกจากกันสำหรับความต้องการในการทดสอบและพัฒนาระบบต่าง ๆ ซึ่งช่วยเพิ่มความยืดหยุ่นและประสิทธิภาพในการจัดการเครือข่ายและระบบ

### 2.2.2 Kali Linux 2024.3 installer amd64 Disc Image File

ไฟล์อิมเมจิติสก์ของระบบปฏิบัติการ Kali Linux เวอร์ชัน 2024.3 ที่ออกแบบมาเพื่อใช้ติดตั้งเป็นเครื่องเสมือนโดยเฉพาะ (Installer) เป็นระบบปฏิบัติการแบบ 64 บิต ซึ่งออกแบบมาสำหรับชิปปี้ที่รองรับสถาปัตยกรรม x86\_64 โดยติดตั้งผ่านแอปพลิเคชัน VirtualBox ที่ทำให้สามารถสร้างเครื่องเสมือนบนเครื่องหลักได้ เมื่อเครื่องเสมือนได้รับผลกระทบจากการพัฒนาเครื่องมือสแกน หากของไหว ผลกระทบเหล่านั้นจะไม่ส่งผลต่อเครื่อง Host และสามารถลบทิ้งแล้วติดตั้งใหม่ได้

### 2.2.3 Network Mapper

Network Mapper หรือ Nmap คือเครื่องมืออุปกรณ์ที่เป็นโอเพ่นซอร์ส ถูกออกแบบมาสำหรับการท่องเครือข่ายและประเมินความปลอดภัย นิยมใช้งานในด้านการจัดการเครือข่ายและตรวจสอบเวลาการทำงานของเครื่องคอมพิวเตอร์ต่าง ๆ (Lyon, 2009) โดย Nmap ใช้ IP Packet ที่ยังไม่ผ่านการประมวลผลมาใช้ในการตรวจสอบว่ามีอุปกรณ์ (Device) ใดบ้างที่เชื่อมต่อ กับเครือข่ายขององค์กร ตรวจสอบแอปพลิเคชันและบริการที่อยู่บนเครือข่าย ตรวจสอบระบบปฏิบัติการของผู้ใช้ อีกทั้งยังสามารถตรวจสอบ Firewall ที่ใช้ได้อีกด้วย

ความโดดเด่นของ Nmap คือความยืดหยุ่นและความทรงพลัง ตัวอย่างความยืดหยุ่น ของเครื่องมือนี้ ได้แก่ การรองรับระบบปฏิบัติการต่าง ๆ มากมาย ไม่ว่าจะเป็น Linux, Microsoft Window, Kali Linux OS, Ubuntu และอื่น ๆ อีกมากมาย ความทรงพลังของ Nmap คือ สามารถวางแผนและประยุกต์ใช้งานได้กับหลายอุปกรณ์ ไม่ว่าจะเป็นการตรวจสอบ Net filter, Router, Firewall หรือแม้กระทั่งการตรวจสอบ Port ที่ทางเว็บเซิร์ฟเวอร์เปิดที่ไว้ได้ (Lyon, 2009)

### 2.2.4 Uniscan

Uniscan คือ เครื่องมือโอเพ่นซอร์สที่ออกแบบมาสำหรับการทดสอบเจาะระบบที่ออกแบบมาสำหรับตรวจสอบช่องโหว่ของเว็บแอปพลิเคชันโดยเฉพาะ โดยความโดดเด่นของ Uniscan คือการทำ Web Crawling, Remote File Include (RFI) และ Remote File Execution (RFE) (Offensive Security, n.d., 2024)

การทำ Web Crawling ช่วยในการตรวจสอบหาช่องโหว่เป็นอย่างดี ไม่ว่าจะเป็นการทำ Directory Brute-Forcing Traversal เพื่อหาไดเรกทอรี่ที่ลึกลับ จำกัดสิทธิการเข้าถึง หรือการทำ File Brute-Forcing Traversal เพื่อหาไฟล์เอกสารสำคัญต่าง ๆ ที่ทางผู้ดูแลไม่ได้ตรวจสอบและเปิดสิทธิการเข้าถึงให้บุคคลภายนอกสามารถเข้าใช้งานได้

### 2.2.5 Nikto

Nikto เป็นเครื่องมือโอเพ่นซอร์สที่ถูกออกแบบมาใช้ด้านการสแกนเว็บเซิร์ฟเวอร์อย่างครอบคลุม ความโดดเด่นของเครื่องมือนี้คือการตรวจสอบปัญหาและช่องโหว่ที่เกี่ยวข้องกับ Secure Socket Layer Protocol (SSL) และการตรวจสอบช่องโหว่เกี่ยวกับ Server-Side Request

Forgery (SSRF) การตรวจสอบไฟล์หรือโปรแกรมที่อาจจะเป็นอันตรายต่อเว็บไซต์ ตรวจสอบเวอร์ชันของเว็บไซต์ (Offensive Security, n.d., 2024)

แม้ว่าการตรวจสอบของ Nikto จะเป็นการตรวจสอบด้านความปลอดภัย จะมีบางรายการที่เป็นการตรวจสอบในลักษณะของข้อมูลเท่านั้น (Info Only) เป็นการตรวจสอบสิ่งผิดปกติที่เกิดขึ้นบนเว็บไซต์ตามเวลาที่กำหนด แต่ข้อควรระวังของ Nikto คือสามารถถูกตรวจจับได้ด้วยระบบ Intrusion Detection หรือ IDS ผู้ใช้งานควรตระหนักรู้และยอมรับผลที่ตามมาหากนำไปใช้โดยไม่ได้รับอนุญาตจากเจ้าของเว็บไซต์

### 2.2.6 Wapiti

Wapiti เป็นเครื่องมือโอเพ่นซอร์สที่ใช้ในการตรวจสอบเว็บไซต์หรือเว็บแอปพลิเคชันการทำงานของเครื่องมือนี้จะเป็นการทดสอบแบบ Black-Box เป็นการทดสอบโดยใช้วิธีการการรวบรวมข้อมูลจากหน้าเว็บมาใช้งานแบบไม่สนใจการทำงานของโค้ดหลังบ้าน เมื่อได้รับ URL ของเว็บไซต์แล้วเครื่องมือจะทำการป้อน script ไปตามแบบฟอร์มหรือช่องใส่ข้อมูลต่าง ๆ เพื่อตรวจหาช่องโหว่ (Offensive Security, n.d., 2024)

จุดเด่นของ Wapiti คือเป็นเครื่องมือทดสอบระบบที่รอบด้านโดยความโดยความโดยเด่นของเครื่องมือนี้จะเป็นการทดสอบช่องโหว่ในด้าน Cross-site scripting, SQL injection, Remote File Execution และการตรวจสอบไฟล์ข้อมูลสำคัญขององค์กรที่เปิดให้ผู้ใช้ภายนอกเข้าไปใช้งานได้ การทดสอบแบบ Black-Box ตรวจจับได้ด้วยระบบ Intrusion Detection หรือ IDS ผู้ใช้งานควรตระหนักรู้และยอมรับผลที่ตามมาหากนำไปใช้โดยไม่ได้รับอนุญาตจากเจ้าของเว็บไซต์

### 2.2.7 Gobuster

Gobuster เป็นเครื่องมือโอเพ่นซอร์สที่ถูกออกแบบมาสำหรับการทดสอบระบบด้านเจาะระบบแบบ Brute Force โดยเฉพาะไม่ว่าจะเป็น URLs Brute Force ที่เป็นการสุ่มชื่อไฟล์และไดเรกทอรี่ทุกอย่างที่เป็นไปได้ การตรวจหา DNS subdomain หรือตรวจหาชื่อของเครื่องเสมือนบนระบบคลาวด์ (Cloud) (Offensive Security, n.d., 2024)

ความโดยเด่นของ Gobuster ที่ถูกนำมาใช้คือความรวดเร็วของการทำงานแม้จะเป็นการทำงานแบบ Brute Force ก็ตาม เนื่องจากเป็นเครื่องมือที่ถูกพัฒนามาจากภาษา GO (GO Language) ที่ส่งเสริมการทำงานแบบ Multi Thread คือการทำงานหลายกระบวนการในเวลา

เดียวกัน ส่งผลให้เวลาที่ใช้ในการดำเนินการลดน้อยลง แต่ข้อควรระวังที่อันตรายของเครื่องมือนี้คือ กระบวนการทำงานของเครื่องมือเป็นการทำงานแบบ Attacker surface กล่าวคือเป็นการทำงาน เช่นกันว่าไปโผล่ตัวเว็บแอปพลิเคชันหรือเว็บเซิร์ฟเวอร์โดยตรง เป็นเครื่องมือที่อันตรายหากไม่ได้รับ การยินยอมในการดำเนินการทดสอบ ต้องยอมรับโทษและผลของการกระทำการที่ตนเองทำ

#### 2.2.8 Acunetix Vulnerabilities Testing Website

Acunetix Vulnerabilities Testing Website ถูกพัฒนาขึ้นมาเพื่อใช้ในการทดสอบ ซึ่งเป็นซอฟต์แวร์ที่พัฒนาโดย Invicti องค์กรที่เป็นผู้ให้ คำปรึกษาทางด้านความปลอดภัยทางไซเบอร์ โดยแอปพลิเคชันทำงานเกี่ยวกับการตรวจสอบให้ ของเว็บไซต์และเว็บแอปพลิเคชัน (Invicti Security. n.d.)

เว็บไซต์ทดลองของ Acunetix Software เป็นเว็บไซต์ที่อนุญาตให้บุคคลทั่วไป สามารถทำการทดสอบเจาะระบบ ทดลองเครื่องมือทางความปลอดภัย หรือแม้กระทั่งนำ malicious script ไปปล่อยบนเว็บไซต์ได้ ยกตัวอย่าง เช่น <http://testphp.vulnweb.com> ที่ใช้ในการทำ SQL injection attack, Path traversal attack, และใช้ประโยชน์ของห่วงโซ่ SSL ที่เกี่ยวข้อง ข้อควร ระวังของการใช้งานคือไม่ควรคลิกทุกอย่างที่มีบนหน้าเว็บเนื่องจากมีการใช้งาน malicious code ฝังไว้เกือบทุกอย่างที่ประกอบของหน้าเว็บไซต์



Vulnerable test websites for [Acunetix Web Vulnerability Scanner](#).

Name	URL	Technologies	Resources
SecurityTweets	<a href="http://testhtml5.vulnweb.com">http://testhtml5.vulnweb.com</a>	nginx, Python, Flask, CouchDB	<a href="#">Review</a> Acunetix HTML5 scanner or <a href="#">learn more</a> on the topic.
Acuart	<a href="http://testphp.vulnweb.com">http://testphp.vulnweb.com</a>	Apache, PHP, MySQL	<a href="#">Review</a> Acunetix PHP scanner or <a href="#">learn more</a> on the topic.
Acuforum	<a href="http://testasp.vulnweb.com">http://testasp.vulnweb.com</a>	IIS, ASP, Microsoft SQL Server	<a href="#">Review</a> Acunetix SQL scanner or <a href="#">learn more</a> on the topic.
Acublog	<a href="http://testaspnet.vulnweb.com">http://testaspnet.vulnweb.com</a>	IIS, ASP.NET, Microsoft SQL Server	<a href="#">Review</a> Acunetix network scanner or <a href="#">learn more</a> on the topic.
REST API	<a href="http://rest.vulnweb.com/">http://rest.vulnweb.com/</a>	Apache, PHP, MySQL	<a href="#">Review</a> Acunetix scanner or <a href="#">learn more</a> on the topic.

ภาพที่ 2.1 ตัวอย่างเว็บไซต์ที่ใช้ในการทดสอบเครื่องมือของ Acunetix

#### 2.2.9 เครื่องハードแวร์ x86-64 H Architecture

เครื่องมือที่จะพัฒนาขึ้นจะใช้ Acer Swift 3 ที่มี Processor 11th Gen Intel(R) Core(TM) i7-11370H @ 3.30GHz 3.30 GHz เป็นระบบปฏิบัติการ Microsoft Window ที่มี

สถาปัตยกรรมประเภท 64 บิต ซึ่งสามารถใช้งาน VirtualBox และ Kali Linux disc image file เพื่อพัฒนาเครื่องมือตรวจจับช่องโหว่ของเว็บไซต์ฟิล์มได้

### 2.3 งานวิจัยที่เกี่ยวข้อง

#### 2.3.1 Kali Linux based Empirical Investigation on Vulnerability Evaluation using Pen-Testing tools

งานวิจัยนี้มุ่งเน้นไปที่การประเมินความเปราะบางของระบบโดยการใช้เครื่องมือการทดสอบเจาะระบบ (Penetration Testing) ซึ่งเป็นกระบวนการที่ช่วยในการตรวจสอบความปลอดภัยของเครือข่ายและระบบคอมพิวเตอร์ โดยงานวิจัยนี้ได้อธิบายถึงวิธีการและเครื่องมือที่ใช้ในการทดสอบเพื่อค้นหาจุดอ่อนที่อาจถูกโจมตีได้ การทดสอบนี้มีความสำคัญอย่างยิ่งในยุคที่การโจมตีทางไซเบอร์มีแนวโน้มเพิ่มขึ้น และองค์กรต่างๆ ต้องการแนวทางในการป้องกันและรักษาความปลอดภัยของข้อมูลและระบบของตนให้ปลอดภัยจากภัยคุกคามที่เกิดขึ้น (Kumar et al., 2023)

ในการทำวิจัยนี้ ผู้วิจัยเลือกใช้ Kali Linux ซึ่งเป็นระบบปฏิบัติการที่ออกแบบมาเฉพาะสำหรับการทดสอบเจาะระบบ โดยมีเครื่องมือหลายตัวที่ติดตั้งไว้ล่วงหน้า เช่น Wapiti, Skipfish, Nessus และ Nmap เครื่องมือเหล่านี้ช่วยให้สามารถทำการสแกนและประเมินความปลอดภัยของระบบได้อย่างมีประสิทธิภาพ โดย Kali Linux มีคุณสมบัติที่ยืดหยุ่นและสามารถปรับใช้ได้ตามความต้องการของผู้ทดสอบ นอกจากนี้ การใช้ Kali Linux ยังช่วยให้ผู้วิจัยสามารถจำลองสถานการณ์การโจมตีในโลกจริงเพื่อประเมินประสิทธิภาพของมาตรการรักษาความปลอดภัยที่มีอยู่

ผลการวิจัยแสดงให้เห็นว่าการทดสอบเจาะระบบโดยใช้เครื่องมือบนระบบปฏิบัติการ Kali Linux สามารถช่วยในการตรวจจับและประเมินความเสี่ยงจากช่องโหว่ในระบบได้อย่างมีประสิทธิภาพ โดยเฉพาะเมื่อใช้เครื่องมือที่เหมาะสม ผู้วิจัยพบว่าการดำเนินการทดสอบเหล่านี้ไม่เพียงแต่ช่วยให้ระบุช่องโหว่ได้เท่านั้น แต่ยังช่วยให้สามารถปรับปรุงมาตรการรักษาความปลอดภัยให้มีประสิทธิภาพและประหยัดงบประมาณมากยิ่งขึ้น นอกจากนี้ ผลลัพธ์ยังชี้ให้เห็นถึงความจำเป็นในการดำเนินการตรวจสอบความปลอดภัยอย่างสม่ำเสมอเพื่อป้องกันไม่ให้เกิดเหตุการณ์ร้ายแรงจากการโจมตีทางไซเบอร์ในอนาคต

### 2.3.2 Automated versus Manual Approach of Web Application Penetration Testing

งานวิจัยนี้จัดทำขึ้นเพื่อค้นหาและอธิบายสถานการณ์ที่สามารถสาธิตความแตกต่างระหว่างการทดสอบเจาะระบบแบบอัตโนมัติและการทดสอบเจาะระบบด้วยตนเอง โดยจะมีสถานการณ์ที่บ่งบอกว่าการทดสอบเจาะระบบด้วยตนเองนั้นดีกว่าการทดสอบเจาะระบบแบบอัตโนมัติ อีกทั้งยังมีสถานการณ์ที่การทดสอบเจาะระบบแบบอัตโนมัติดีกว่าการทดสอบเจาะระบบด้วยตนเอง ซึ่งให้ไว้ในงานวิจัยจะนำมาจากเอกสาร OWASP Top 10 security เป็นเอกสารที่จัดลำดับช่องโหว่ทางด้านความปลอดภัยของเว็บแอปพลิเคชันสิบอันดับ ซึ่งบทสรุปของงานวิจัยนี้จะแยกออกเป็นข้อดีและข้อเสียของการทดสอบเจาะระบบแบบอัตโนมัติและการทดสอบเจาะระบบด้วยตนเอง (Singh, Meherhomji & Chandavarkar, 2020)

ตารางที่ 2.1 ตารางผลลัพธ์ของการทดสอบเจาะระบบแบบอัตโนมัติและการทดสอบเจาะระบบด้วยตนเอง

Scenario	Vulnerability	Manual	Automatic
Clickjacking	sameorigin	✓	✗
File upload	XSS	✓	✗
Sensitive files	public domain	✗	✓
Business logic	logic failure	✓	✗

การทดสอบเจาะระบบแบบอัตโนมัติและการทดสอบเจาะระบบด้วยตนเองทั้งสองกระบวนการมีข้อได้เปรียบและเสียเปรียบแตกต่างกันไปตามสถานการณ์ ข้อได้เปรียบของการทดสอบเจาะระบบแบบอัตโนมัติคือการตรวจสอบที่รวดเร็วและประหยัดทรัพยากรในการสแกนหาช่องโหว่จากการตรวจหาในหลาย ๆ จุดของเว็บแอปพลิเคชัน มีความต่อเนื่องและสามารถทำซ้ำได้ทันที ข้อเสียที่เห็นได้ชัดคือไม่สามารถสแกนช่องโหว่ที่นักโจมตีกำหนดไว้ได้ อีกทั้งยังมีการแจ้งเตือนที่ผิดพลาดจากการทำงาน ซึ่งแตกต่างจากการทดสอบเจาะระบบด้วยตนเองที่มีข้อได้เปรียบคือการนำประสบการณ์และกระบวนการคิดวิเคราะห์ของมนุษย์มาใช้ ทำให้สามารถระบุช่องโหว่ที่เกิดขึ้นได้ครอบคลุมมากกว่าการทดสอบเจาะระบบแบบอัตโนมัติ สามารถนำเทคนิคและกระบวนการที่หลากหลายมาใช้ในการสแกนหาช่องโหว่เพื่อยืนยันว่าช่องโหว่เหล่านั้นเกิดขึ้นจริง เพื่อลดการแจ้งเตือนที่ผิดพลาด แต่ข้อเสียเปรียบของการทดสอบเจาะระบบด้วยตนเองที่เห็นได้ชัดคือ ใช้เวลาในการ

ทดสอบที่ yuananmag กว่าการทดสอบเจาะระบบแบบอัตโนมัติ เพื่อจะทดสอบทุกฟังก์ชันที่มีในเว็บแอปพลิเคชัน

โดยภาพรวมการทดสอบเจาะระบบด้วยตนเองยังถือว่ามีประสิทธิภาพสูงกว่าเนื่องจากการใช้ประสบการณ์และการคิดวิเคราะห์ช่วยให้ผู้ทดสอบสามารถตรวจจับช่องโหว่ที่เครื่องมืออัตโนมัติไม่สามารถตรวจจับได้ สามารถนำเครื่องมืออัตโนมัติช่วยเร่งกระบวนการทดสอบได้แต่ไม่สามารถรับประกันได้ว่าเว็บแอปพลิเคชันจะปลอดภัยอย่างสมบูรณ์ ดังนั้น การใช้ข้อดีของทั้งสองวิธีร่วมกันจึงเป็นทางเลือกที่เหมาะสม การทดสอบเจาะระบบแบบอัตโนมัติสามารถเสริมบทบาทในการตรวจจับช่องโหว่ได้ ผู้ทดสอบสามารถใช้เครื่องมืออัตโนมัติเพื่อช่วยตรวจหาช่องโหว่ โดยการเขียน скриปต์แบบเฉพาะเจาะจงสำหรับค้นหาช่องโหว่เหล่านั้น เครื่องมืออัตโนมัติมีประโยชน์สำหรับผู้เริ่มต้นที่กำลังเรียนรู้การทดสอบแอปพลิเคชันและหาช่องโหว่ เมื่อมีประสบการณ์มากขึ้น การพึ่งพาเครื่องมือเหล่านี้อาจลดลง แต่การนำเครื่องมืออัตโนมัติมาใช้งานร่วมกันย่อมส่งผลดีกว่าการทำงานด้วยตนเองเพียงอย่างเดียว

## 2.4 แอปพลิเคชันที่คล้ายคลึง

### 2.4.1 Rapidscan v1.2 The Multi-Tool Web Vulnerability Scanner

Rapidscan เป็นเครื่องมือโอเพ่นซอร์สสำหรับการสแกนหาช่องโหว่บนเว็บไซต์ที่พัฒนาขึ้นมาด้วยภาษาไพทอน (Python Language) ซึ่งถูกออกแบบมาเพื่อช่วยให้การสแกนความปลอดภัยที่ซับซ้อนสามารถทำได้โดยอัตโนมัติ โดยการรวบรวมเครื่องมือหลาย ๆ เครื่องมือรวมกันอยู่ในแอปพลิเคชันของ Rapidscan ช่วยลดเวลาในการหาข้อมูล ลดเวลาในการติดตั้งเครื่องมือ และมีการวิเคราะห์ผลลัพธ์ที่ได้ ทำให้สามารถนำไปใช้งานได้ทันที

RapidScan มี妃เจอร์เด่นในการติดตั้งแบบขั้นตอนเดียว พร้อมทั้งการสแกนช่องโหว่โดยใช้เครื่องมือด้านความปลอดภัยหลายตัวในครั้งเดียว เช่น nmap, dnsrecon, wafw00f, sslyze, และ nikto โดยที่สามารถตรวจหาช่องโหว่ได้อย่างรวดเร็วและประหยัดเวลา นอกจากนี้ยังช่วยตรวจสอบช่องโหว่เดียวกันด้วยหลายเครื่องมือเพื่อลดความเสี่ยงของการเกิด false positives โปรแกรมมีน้ำหนักเบาและไม่กินทรัพยากรามาก พร้อมมีเครื่องหมายบอกว่าแบบทดสอบได้ใช้เวลานาน ผู้ใช้สามารถกดข้ามได้ถ้าต้องการ นอกจากนี้ประเมินช่องโหว่ตามมาตรฐาน OWASP Top 10 และ CWE 25 และมีการจัดอันดับความร้ายแรงของช่องโหว่เป็น critical, high, medium, low

และ informational พร้อมคำอธิบายของช่องโหว่และแนวทางการแก้ไข ทั้งนี้ยังมีการสรุปภาพรวมของการสแกนในรูปแบบ executive summary ที่บอกถึงช่องโหว่ที่ค้นพบทั้งหมด

```
(kali㉿kali)-[~/Desktop]
$ cd rapidscan

(kali㉿kali)-[~/Desktop/rapidscan]
$ python3 rapidscan.py sh4.in
```

ภาพที่ 2.2 คำสั่งเรียกใช้งาน Rapidscan

```
/_ )_ • /(_ _  
/ ( (//)/( /_) ( (//)  
/  
(The Multi-Tool Web Vulnerability Scanner)  
  
Check out our new software, NetBot for simulating DDoS attacks  
- https://github.com/skavngr/netbot  
  
[ Checking Available Security Scanning Tools Phase... Initiated. ]  
Some of these tools ['xsser', 'dnswalk', 'golismero', 'uniscan', 'dnsmap'] are unavailable or will be skipped. RapidScan will still perform the rest of the test s. Install these tools to fully utilize the functionality of RapidScan.  
[ Checking Available Security Scanning Tools Phase... Completed ]  
  
[ Preliminary Scan Phase Initiated... Loaded 80 vulnerability checks. ]  
[● < 30s] Deploying 1/80 | Joomla Checker - Checks for Joomla Installation.
```

ภาพที่ 2.3 หน้าหลักเมื่อเรียกใช้งาน Rapidscan

```
Vulnerability Threat Level
info Does not have an IPv6 Address. It is good to have one.

Vulnerability Definition
Not a vulnerability, just an informational alert. The host does not have IPv6 support. IPv6 provides more security as IPSec (responsible for CIA - Confidentiality, Integrity and Availability) is incorporated into this model. So it is good to have IPv6 Support.

Vulnerability Remediation
It is recommended to implement IPv6. More information on how to implement IP v6 can be found from this resource. https://www.cisco.com/c/en/us/solutions/collateral/enterprise/cisco-on-cisco/IPv6-Implementation\_CS.html

[● < 2m] Deploying 5/80 | Uniscan - Brutes for Filenames on the Domain.
...Scanning Tool Unavailable. Skipping Test...
[● < 15s] Deploying 6/80 | Nmap - Checks for Remote Desktop Service over UDP
...Completed in 1s
[● < 25s] Deploying 7/80 | SSLyze - Checks for OCSP Stapling.
...Completed in 1s
```

ภาพที่ 2.4 ผลลัพธ์การสแกนของ Rapidscan

## 2.5 ความคล้ายคลึงและความแตกต่างระหว่าง VULNscan กับ RapidScan

### 2.5.1 สิ่งที่คล้ายคลึงกัน

- 1) พัฒนาโดยใช้ภาษา Python และใช้งานได้ดีบน Kali Linux ทั้งสองเครื่องมือเขียนด้วยภาษา Python และออกแบบมาให้ทำงานได้ดีในระบบปฏิบัติการ Kali Linux ซึ่งเป็นระบบที่นิยมใช้ในการทดสอบเจาะระบบ
- 2) สามารถสแกนหาช่องโหว่ของเว็บเซิร์ฟเวอร์ได้ โดย RapidScan และ VULNscan ต่างก็สามารถสแกนหาช่องโหว่จากเว็บเซิร์ฟเวอร์โดยอ้างอิงจากมาตรฐาน OWASP Top Ten เหมือนกัน
- 3) มีการจัดระดับความเสี่ยงของช่องโหว่ที่ตรวจพบ โดยทั้งคู่จัดประเภทความเสี่ยงเป็นระดับ Critical, High, Medium, Low เพื่อช่วยให้ผู้ใช้งานสามารถประเมินความสำคัญของช่องโหว่แต่ละรายการได้ชัดเจน
- 4) มีการสรุปผลการสแกนอย่างชัดเจน เมื่อการสแกนเสร็จ เครื่องมือทั้งสองจะแสดงสรุปผล เช่น จำนวนช่องโหว่ที่พบในแต่ละระดับความรุนแรง ระยะเวลาที่ใช้ในการสแกน และจำนวนงานที่ใช้ในการสแกน
- 5) มีไฟล์ Raw Report สำหรับผลการสแกนโดยละเอียด เครื่องมือทั้งสองสามารถสร้างไฟล์รายงานดิบ (Raw Report) ซึ่งแสดงผลลัพธ์จากแต่ละเครื่องมือที่ใช้ในการสแกนในรูปแบบไฟล์ข้อความ (.txt) สำหรับการตรวจสอบเพิ่มเติมภายหลัง
- 6) มีระบบการแจ้งเตือนผลสแกน ระหว่างหรือหลังการสแกน หากพบช่องโหว่ ระบบจะแจ้งเตือนในคอนโซลทันที พร้อมแสดงรายละเอียด เช่น ช่องโหว่ที่ตรวจพบและระดับความรุนแรง

### 2.5.2 สิ่งที่แตกต่างกัน

- 1) เครื่องมือ VULNscan มีการบันทึกไฟล์รายละเอียดและวิธีการกำกับดูแลช่องโหว่ในรูปแบบไฟล์ข้อความ (.txt) ซึ่งแตกต่างจากเครื่องมือ RapidScan ที่ไม่มีการบันทึกรายละเอียดและวิธีการกำกับดูแลช่องโหว่แต่แสดงบนหน้าจอแทน
- 2) เครื่องมือ VULNscan แสดงช่องโหว่ที่พบแบบละเอียดบนหน้าจอทันทีที่สแกนเสร็จ ซึ่งแตกต่างจาก RapidScan ที่ไม่มีการแสดงผลนิ่งหน้าจอ
- 3) เครื่องมือ VULNscan มีบันทึกการรายงานผลเชิงกราฟและตารางในรูปแบบของ dashboard บนไฟล์ html ทำให้ผู้ทดสอบทุกระดับสามารถเข้าใจรายละเอียดภาพรวมของการสแกนได้ดียิ่งขึ้น

## บทที่ 3

### วิธีการวิจัย

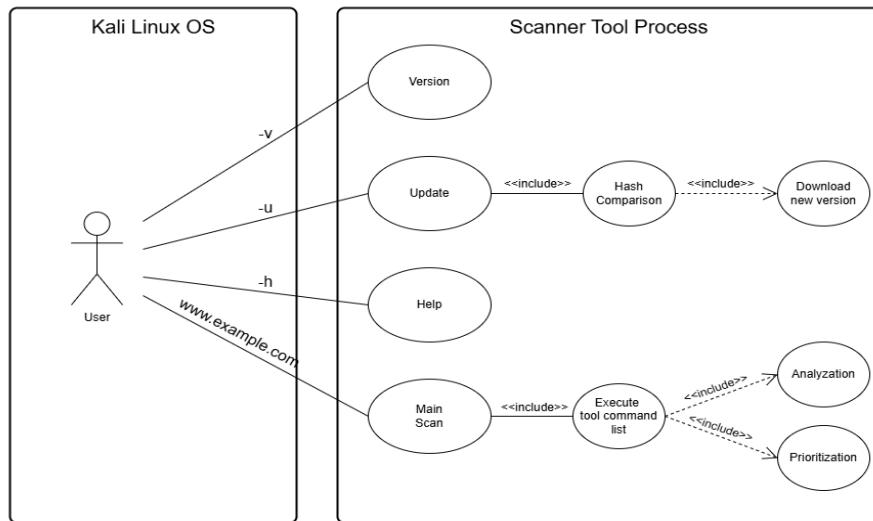
#### 3.1 ภาพรวมของโครงการ

เครื่องมือสแกนหาช่องโหว่ของเว็บเซิร์ฟเวอร์อัตโนมัติเป็นเครื่องมือแบบมัลติทูล เมื่อดำเนินการสแกนจะไปเรียกใช้เครื่องมือต่าง ๆ ตามรายการคำสั่งที่เขียนไว้ เครื่องมือนี้สามารถใช้งานผ่านระบบปฏิบัติการ Kali Linux โดยพัฒนาขึ้นจากภาษาโปรแกรมไพธอน (Python programming language) และเป็นเครื่องมือที่ใช้งานผ่านชุดคำสั่งแบบตัวอักษร หรือที่เรียกว่า command-line interface (CLI) สามารถติดตั้งได้โดยใช้คำสั่ง git clone เพื่อทำการคัดลอกเครื่องมือมาจาก git repository ซึ่งช่องโหว่ที่ครอบคลุมได้แก่ Broken Access Control, SQL Injection (SQLi), Cross-Site Scripting (XSS), Remote Code Execution (RCE) และ Server-Side Request Forgery (SSRF)

ไฟเจอร์ของเครื่องมือนี้จะประกอบไปด้วยสามไฟเจอร์หลักคือ สแกนหาช่องโหว่ ผู้ใช้งานสามารถสแกนข้าม LAN ของเครือข่ายได้ ไม่จำเป็นที่จะต้องอยู่ในเครือข่ายเดียวกันกับเว็บเซิร์ฟเวอร์ไฟเจอร์การประเมินผล เมื่อสแกนหาช่องโหว่เสร็จจะได้รับไฟล์ช่วยวิเคราะห์ซึ่งเป็นข้อมูลดิบ ไฟเจอร์ถัดไปคือ การประมวลผลและวิเคราะห์ข้อมูล ทางเครื่องมือจะนำข้อมูลดิบนั้นมาวิเคราะห์ ประมวลผลและจัดลำดับความสำคัญของปัญหา โดยขึ้นอยู่กับความร้ายแรงและความเสียหายที่จะเกิดขึ้นหากปล่อยให้มีช่องโหว่ดังกล่าวเกิดขึ้นบนเว็บเซิร์ฟเวอร์ และไฟเจอร์สุดท้ายคือการนำเสนอผลลัพธ์ ผลลัพธ์ที่ประมวลจะมีทั้งรูปแบบ log file และรูปแบบ report.html ที่ใช้สำหรับการทำ log visualization ซึ่งเป็นรูปแบบการนำเสนอที่บุคคลทั่วไปสามารถเข้าใจได้ในเวลาอันสั้น

### 3.2 การวิเคราะห์ขอบเขตและความต้องการของเครื่องมือ

#### 3.2.1 Use case diagram



ภาพที่ 3.1 แผนภาพกรณีใช้งานของเครื่องมือ

#### 3.2.2 รายละเอียดแผนภาพ use case diagram

คำสั่ง CLI ใน การใช้งานของเครื่องมือจะถูกแบ่งออกเป็นสามอาร์กิวเม้นท์ได้แก่ python3 ซึ่งเป็นคำสั่งที่เปลี่ยนให้เป็น interpreter mode เพื่อประมวลผลไฟล์ Python โปรแกรม อาร์กิวเม้นท์ลำดับถัดไปคือชื่อของไฟล์ Python โปรแกรมที่จะประมวลผล และอาร์กิวเม้นท์ที่สามารถแบ่งออกได้สี่คำสั่งตามการใช้งาน

จากแผนภาพกรณีใช้งานของเครื่องมือสามารถวิเคราะห์กระบวนการทำงานผ่านคำสั่งต่าง ๆ บนระบบปฏิบัติการ Kali Linux โดยมีรายละเอียดดังนี้

##### 3.2.2.1 Actors

1) User ผู้ใช้งานเครื่องมือสแกน โดยจะเป็นผู้ที่ส่งคำสั่งต่าง ๆ ให้กับเครื่องมือเพื่อให้เครื่องมือทำงานตามที่ต้องการ

##### 3.2.2.2 Version command use case

1) คำสั่ง python3 scanner.py -v

2) การใช้งาน ผู้ใช้สามารถใช้คำสั่ง -v เพื่อแสดงเวอร์ชันของเครื่องมือที่กำลังใช้งานอยู่ โดยฟีเจอร์นี้จะเป็นการตรวจสอบว่าเครื่องมือที่ใช้งานนั้นเป็นเวอร์ชันล่าสุดหรือไม่

3) ความสำคัญ ช่วยให้ผู้ใช้ทราบว่าเครื่องมือสแกนที่กำลังใช้อยู่เป็นเวอร์ชันใด

### 3.2.2.3 Update command use case

- 1) คำสั่ง python3 scanner.py -u
- 2) การใช้งาน ผู้ใช้สามารถใช้คำสั่ง -u เพื่ออัปเดตเครื่องมือ โดยจะมีการตรวจสอบไฟล์หรือเวอร์ชันปัจจุบันผ่านการ Hash Comparison (การเปรียบเทียบแฮช) เพื่อดูว่า ต้องการยัปเดตหรือไม่
- 3) กระบวนการทำงาน ทำการ Hash Comparison เพื่อเปรียบเทียบแฮชของไฟล์เวอร์ชันปัจจุบันกับเวอร์ชันล่าสุดหากพบว่าเวอร์ชันปัจจุบันไม่ตรงกับเวอร์ชันล่าสุด จะดำเนินการ Download new version เพื่อติดตั้งเวอร์ชันใหม่
- 4) ความสำคัญ ช่วยให้ผู้ใช้สามารถใช้งานเครื่องมือในเวอร์ชันล่าสุดที่มีประสิทธิภาพและความปลอดภัยที่ดียิ่งขึ้น

### 3.2.2.4 Help command use case

- 1) คำสั่ง python3 scanner.py -h
- 2) การใช้งาน คำสั่ง -h ช่วยให้ผู้ใช้เข้าถึงเอกสารหรือข้อมูลการใช้งาน เครื่องมือได้ เช่น คู่มือคำสั่ง หรือรายละเอียดเพิ่มเติมในการใช้งานฟีเจอร์ต่าง ๆ ของเครื่องมือ
- 3) ความสำคัญช่วยให้ผู้ใช้เข้าใจวิธีการใช้งานเครื่องมือและคำสั่งต่าง ๆ ง่ายขึ้น

### 3.2.2.5 Target command use case

- 1) คำสั่ง python3 scanner.py www.example.com
- 2) การใช้งาน ผู้ใช้สามารถสั่งให้เครื่องมือสแกนซ่องโหว่ของระบบเป้าหมายได้ โดยระบบจะทำการ Execute tool command list เพื่อดำเนินการสแกนตามชุดคำสั่งที่กำหนดไว้ใน เครื่องมือ
- 3) กระบวนการทำงาน แบ่งออกได้เป็นสามขั้นตอน ขั้นตอนแรกคือ Execute tool command list เรียกใช้คำสั่งที่จำเป็นสำหรับการสแกน ขั้นตอนถัดไปคือ Analyzation หลังจากการสแกนเสร็จสิ้น ระบบจะทำการวิเคราะห์ข้อมูลที่ได้รับจากการสแกนเพื่อหาช่องโหว่ต่าง ๆ

และขั้นตอนสุดท้ายคือ Prioritization จัดลำดับความสำคัญของช่องโหว่ตามระดับความเสี่ยหายที่จะเกิดขึ้นทำให้ทราบว่าควรแก้ไขอะไรก่อน

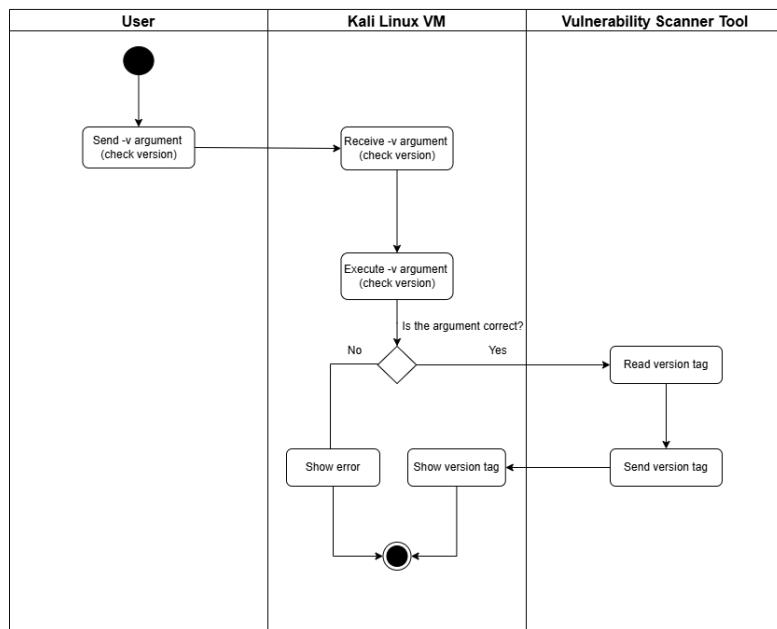
4) ความสำคัญ ไฟล์เจอร์นีเป็นหัวใจหลักของเครื่องมือ เพราะจะช่วยให้ผู้ใช้สามารถสแกนและระบุช่องโหว่ในระบบเป้าหมายได้อย่างมีประสิทธิภาพ

### 3.2.2.6 Use cases relationship

1) มีการใช้ความสัมพันธ์แบบ <<include>> ในหลายจุด หมายถึงว่ากระบวนการทำงานหลักนั้นจะต้องมีการเรียกใช้กระบวนการรายอื่น เช่น Update จะเป็นต้องเรียก Hash Comparison ก่อนเพื่อเปรียบเทียบแฮช และถ้าเวอร์ชันไม่ตรงกันจะเรียก Download new version ต่อไป และ Main Scan มีการเรียก Execute tool command list เพื่อสแกน จากนั้นจะเรียก Analyzation และ Prioritization เพื่อประเมินผลและจัดลำดับความสำคัญของช่องโหว่

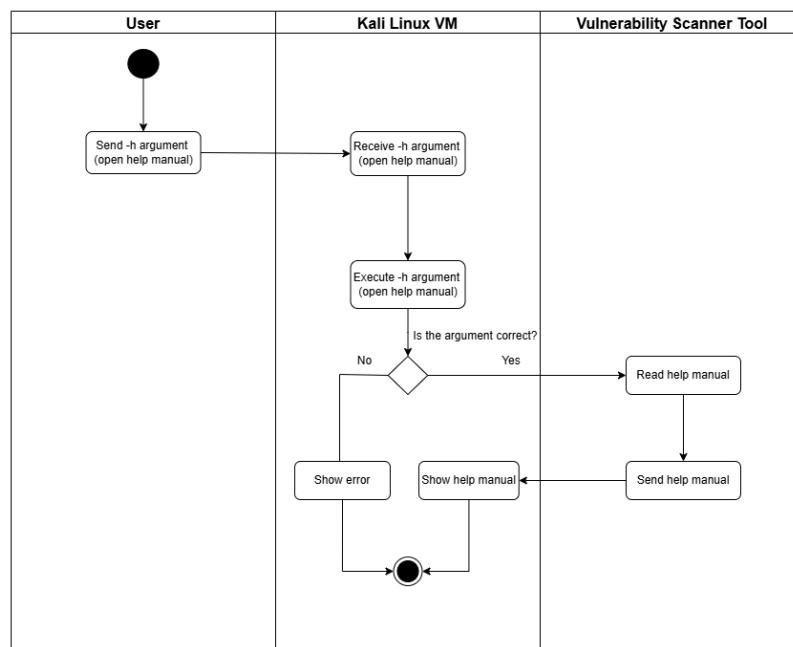
### 3.2.3 Activity diagram

#### 3.2.3.1 Activity diagram คำสั่ง -v



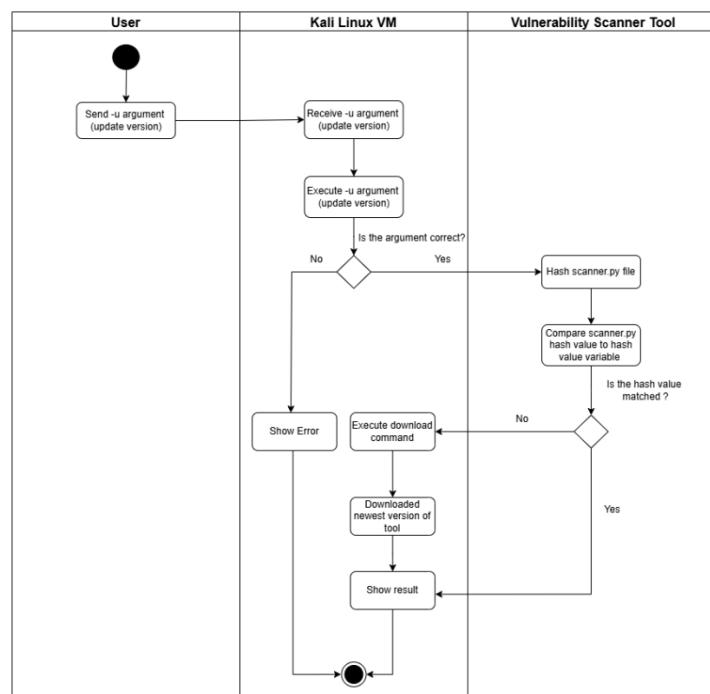
ภาพที่ 3.2 แผนภาพพื้นฐานการใช้งานคำสั่ง -v

### 3.2.3.2 Activity diagram คำสั่ง -h



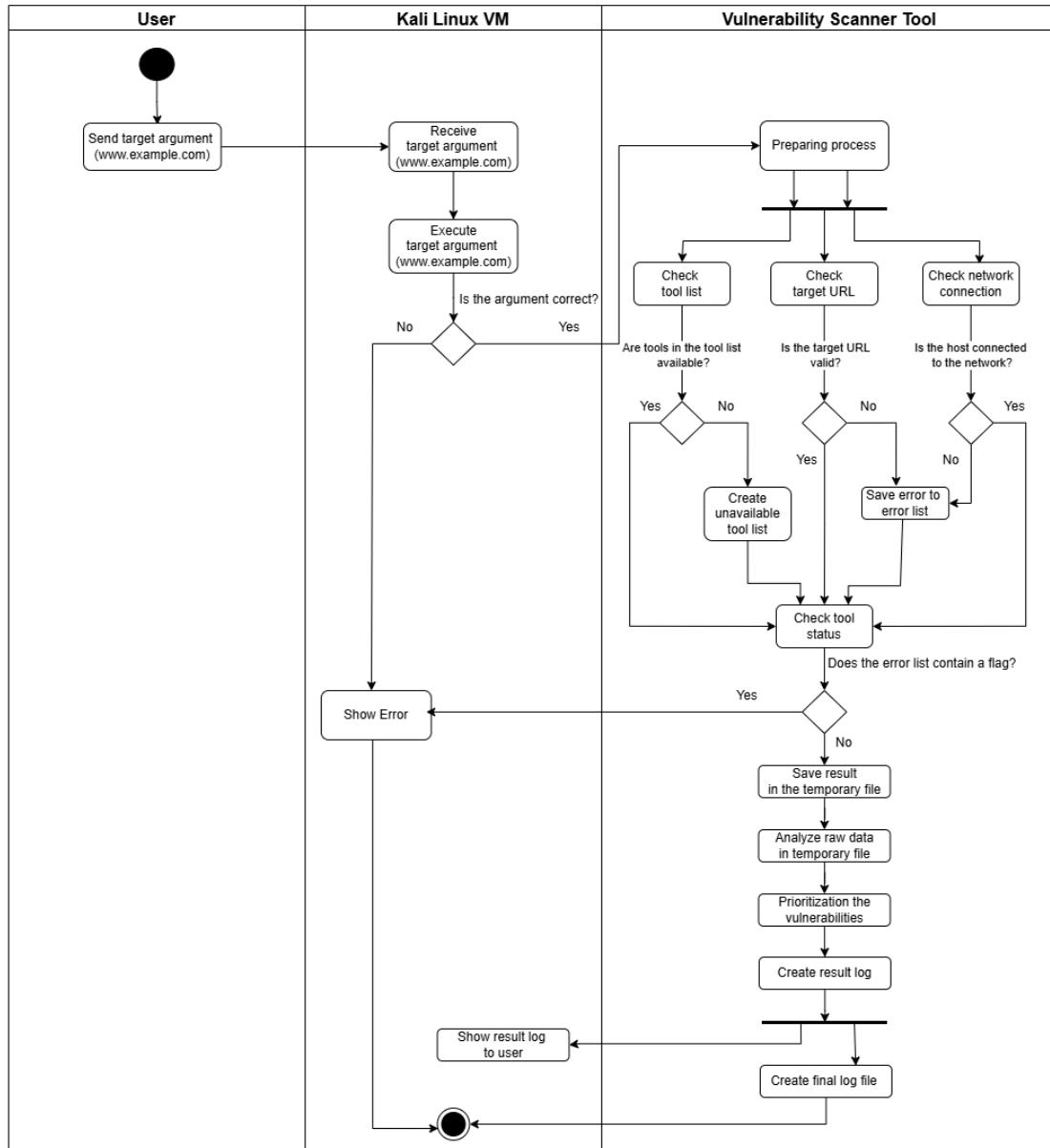
ภาพที่ 3.3 แผนภาพพื้นต์กรรมการใช้งานคำสั่ง -h

### 3.2.3.3 Activity diagram คำสั่ง -n



ภาพที่ 3.4 แผนภาพพื้นต์กรรมการใช้งานคำสั่ง -n

### 3.2.3.4 Activity diagram คำสั่ง target



ภาพที่ 3.5 แผนภาพพัฒนาระบบการใช้งานคำสั่ง target

## 3.3 การดำเนินงาน

### 3.3.1 การรับร่วมข้อมูล

#### 3.3.1.1 ข้อมูลสถิติของโหวตด้านความปลอดภัย

ข้อมูลสถิติของโหวตที่เกิดขึ้นนั้นจะใช้เอกสารจากองค์กรไม่แสวงหาผลกำไร Open Web Application Security Project (OWASP) โดยทางองค์กรจะจัดประชุมเพื่อ

รวบรวมข้อมูลเกี่ยวกับช่องโหว่ ความเสี่ยง และภัยคุกคามที่เกี่ยวข้องกับเว็บแอปพลิเคชัน ซึ่งเนื้อหาจากการประชุมจะถูกจัดทำเป็นเอกสาร OWASP Top Ten และนำมาเปิดเผยแพร่แบบสาธารณะเพื่อเตือนภัยและแนะนำวิธีการป้องกัน

### 3.3.1.2 เครื่องมือที่ใช้ในการพัฒนา

เครื่องมือที่ใช้ในการพัฒนานั้นจะมุ่งเน้นไปทางเครื่องมือโอเพ่นซอร์สที่สามารถใช้ได้ฟรี และมีอยู่บนระบบปฏิบัติการ Kali Linux เพราะเครื่องมือที่อยู่ในระบบปฏิบัติการนั้นมีการทำแพทช์ (Patching) อยู่เสมอไม่เหมือนกับเครื่องมือที่เผยแพร่อยู่บนโลกอินเทอร์เน็ต โดยเครื่องมือที่ใช้จะมี Nmap, Uniscan, Nikto, Wapiti, Gobuster, และ Lbd ซึ่งแต่ละเครื่องมือมีความสามารถในการตรวจจับ วิเคราะห์ช่องโหว่ และรูปแบบของผลลัพธ์ที่แตกต่างกันไป แต่บางเครื่องมือจะถูกใช้งานเพื่อตรวจสอบช่องโหว่เดียวกันเพื่อลดการรายงานผลที่ผิดพลาด

### 3.3.1.3 การวิเคราะห์และจัดลำดับความร้ายแรงของช่องโหว่

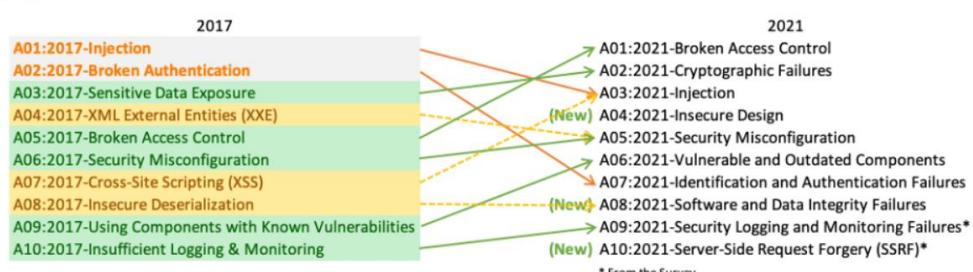
การวิเคราะห์ข้อมูลที่ได้รับจากเครื่องมือนั้นจะใช้มาตรฐานของสถาบันมาตรฐานและเทคโนโลยีแห่งชาติ (National Institute of Standard and Technology : NIST) และใช้ฐานข้อมูลช่องโหว่แห่งชาติ (National Vulnerability Database : NVD) ที่มีเครื่องคำนวณค่าการวัดและประเมินความรุนแรงของช่องโหว่ (Common Vulnerability Scoring System : CVSS) ให้เป็นพื้นฐานในการวิเคราะห์และจัดลำดับความร้ายแรงของช่องโหว่

## 3.3.2 ตัวอย่างข้อมูล

### 3.3.2.1 ตัวอย่างข้อมูลเอกสาร OWASP Top Ten

#### Top 10 Web Application Security Risks

There are three new categories, four categories with naming and scoping changes, and some consolidation in the Top 10 for 2021.



ภาพที่ 3.6 ตัวอย่างช่องโหว่ที่เกิดขึ้นบ่อยบนเว็บไซต์ฟอร์มเอกสาร OWASP Top Ten

- **A01:2021-Broken Access Control** moves up from the fifth position; 94% of applications were tested for some form of broken access control. The 34 Common Weakness Enumerations (CWEs) mapped to Broken Access Control had more occurrences in applications than any other category.
- **A02:2021-Cryptographic Failures** shifts up one position to #2, previously known as Sensitive Data Exposure, which was broad symptom rather than a root cause. The renewed focus here is on failures related to cryptography which often leads to sensitive data exposure or system compromise.
- **A03:2021-Injection** slides down to the third position. 94% of the applications were tested for some form of injection, and the 33 CWEs mapped into this category have the second most occurrences in applications. Cross-site Scripting is now part of this category in this edition.
- **A04:2021-Insecure Design** is a new category for 2021, with a focus on risks related to design flaws. If we genuinely want to "move left" as an industry, it calls for more use of threat modeling, secure design patterns and principles, and reference architectures.
- **A05:2021-Security Misconfiguration** moves up from #6 in the previous edition; 90% of applications were tested for some form of misconfiguration. With more shifts into highly configurable software, it's not surprising to see this category move up. The former category for XML External Entities (XXE) is now part of this category.
- **A06:2021-Vulnerable and Outdated Components** was previously titled Using Components with Known Vulnerabilities and is #2 in the Top 10 community survey, but also had enough data to make the Top 10 via data analysis. This category moves up from #9 in 2017 and is a known issue that we struggle to test and assess risk. It is the only category not to have any Common Vulnerability and Exposures (CVEs) mapped to the included CWEs, so a default exploit and impact weights of 5.0 are factored into their scores.

ภาพที่ 3.7 ตัวอย่างรายละเอียด OWASP Top Ten

### 3.3.2.2 ตัวอย่างข้อมูลผลลัพธ์ที่ได้จากการเครื่องมือ

#### 1) Nmap ใช้ในการทำ OpenSSL CCS Injection (Man-in-the-middle)

และ Checks for MySQL, ORACLE, and MS-SQL

```
$ nmap -p1433 --open -Pn -v testaspnet.vulnweb.com
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-09-25 09:10 EDT
Initiating Parallel DNS resolution of 1 host. at 09:10
Completed Parallel DNS resolution of 1 host. at 09:10, 0.05s elapsed
Initiating Connect Scan at 09:10
Scanning testaspnet.vulnweb.com (44.238.29.244) [1 port]
Completed Connect Scan at 09:10, 2.00s elapsed (1 total ports)
Read data files from: /usr/bin/../share/nmap
Nmap done: 1 IP address (1 host up) scanned in 2.08 seconds

(park㉿kali)-[~] HTTP (Hypertext Transfer Protocol)
└─$ nmap -p3306 --open -Pn -v testaspnet.vulnweb.com
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-09-25 09:14 EDT
Initiating Parallel DNS resolution of 1 host. at 09:14
Completed Parallel DNS resolution of 1 host. at 09:14, 0.00s elapsed
Initiating Connect Scan at 09:14
Scanning testaspnet.vulnweb.com (44.238.29.244) [1 port]
Completed Connect Scan at 09:14, 2.00s elapsed (1 total ports)
Read data files from: /usr/bin/../share/nmap
Nmap done: 1 IP address (1 host up) scanned in 2.04 seconds

(park㉿kali)-[~] might want to run another scan targeting that port directly.
└─$ nmap -p1521 --open -Pn -v testaspnet.vulnweb.com
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-09-25 09:14 EDT
Initiating Parallel DNS resolution of 1 host. at 09:14
Completed Parallel DNS resolution of 1 host. at 09:14, 0.01s elapsed
Initiating Connect Scan at 09:14
Scanning testaspnet.vulnweb.com (44.238.29.244) [1 port]
Completed Connect Scan at 09:14, 2.00s elapsed (1 total ports)
Read data files from: /usr/bin/../share/nmap
Nmap done: 1 IP address (1 host up) scanned in 2.04 seconds

(park㉿kali)-[~]
```

ภาพที่ 3.8 ผลลัพธ์จากการใช้เครื่องมือ nmap เพื่อหา MySQL, ORACLE, MS-SQL

2) Uniscan ใช้ในการทำ Brute Forces for Filenames on the Domain, Brute Forces Directories on the Domain และ Checks for XSS, SQLi, RCE.

**SQL Injection:**

```
http://testphp.vulnweb.com/listproducts.php?cat=1'
http://testphp.vulnweb.com/listproducts.php?cat=1"
http://testphp.vulnweb.com/listproducts.php?cat=2'
http://testphp.vulnweb.com/listproducts.php?cat=2"
http://testphp.vulnweb.com/listproducts.php?cat=3'
http://testphp.vulnweb.com/listproducts.php?cat=3"
http://testphp.vulnweb.com/listproducts.php?cat=4'
http://testphp.vulnweb.com/listproducts.php?cat=4"
http://testphp.vulnweb.com/listproducts.php?artist=2'
http://testphp.vulnweb.com/listproducts.php?artist=1'
http://testphp.vulnweb.com/listproducts.php?artist=2"
http://testphp.vulnweb.com/listproducts.php?artist=3'
http://testphp.vulnweb.com/listproducts.php?artist=3"
http://testphp.vulnweb.com/listproducts.php?artist=1"
http://testphp.vulnweb.com/secured/newuser.php
Post data: &uname=123&upass=123&upass2=123&urname=123&ucc=123&uemail=123&uphone=123&
signup=123&uaddress=123
```

ภาพที่ 3.9 ผลลัพธ์จากการใช้เครื่องมือ Uniscan เพื่อทำ SQL Injection

**Cross-Site Scripting (XSS):**

```
http://testphp.vulnweb.com/hpp/?pp="><script>alert('XSS')</script>
http://testphp.vulnweb.com/hpp/?pp="><IMG SRC="javascript:alert('XSS');">
http://testphp.vulnweb.com/hpp/?pp="><LINK REL="stylesheet" HREF="javascript:alert('XSS');">
http://testphp.vulnweb.com/hpp/?pp="><META HTTP-EQUIV="refresh" CONTENT="0; URL=http://;
URL=javascript:alert('XSS');">
http://testphp.vulnweb.com/hpp/?pp="><body onload="javascript:alert('XSS')"></body>
http://testphp.vulnweb.com/hpp/?pp="><DIV STYLE="background-image: url(javascript:alert('XSS'))">
http://testphp.vulnweb.com/hpp/?pp="><table background="javascript:alert('XSS')"></table>
http://testphp.vulnweb.com/search.php?test=query
Post data: &searchFor=<script>alert('XSS')</script>&goButton=123
http://testphp.vulnweb.com/search.php?test=query
Post data: &searchFor=<IMG SRC="javascript:alert('XSS');">&goButton=123
http://testphp.vulnweb.com/search.php?test=query
Post data: &searchFor=<LINK REL="stylesheet" HREF="javascript:alert('XSS');">&goButton=123
http://testphp.vulnweb.com/search.php?test=query
Post data: &searchFor=<META HTTP-EQUIV="refresh" CONTENT="0; URL=http://;URL=javascript:alert('XSS');">&
goButton=123
http://testphp.vulnweb.com/search.php?test=query
Post data: &searchFor=<DIV STYLE="background-image: url(javascript:alert('XSS'))">&goButton=123
http://testphp.vulnweb.com/search.php?test=query
Post data: &searchFor=<body onload="javascript:alert('XSS')"></body>&goButton=123
http://testphp.vulnweb.com/search.php?test=query
Post data: &searchFor=<table background="javascript:alert('XSS')"></table>&goButton=123
```

ภาพที่ 3.10 ผลลัพธ์จากการใช้เครื่องมือ Uniscan เพื่อทำ Cross-site Scripting

<b>SCAN TIME</b>
<b>Scan Started:</b> 20/9/2024 6:36:34
<b>TARGET</b>
<b>Domain</b> http://testphp.vulnweb.com/ <b>Server Banner:</b> nginx/1.19.0 <b>Target IP:</b> 44.228.249.3
<b>CRAWLING</b>
<b>File check:</b> CODE: 200 URL: http://testphp.vulnweb.com/CVS/Entries CODE: 200 URL: http://testphp.vulnweb.com/favicon.ico CODE: 200 URL: http://testphp.vulnweb.com/index.php CODE: 200 URL: http://testphp.vulnweb.com/login.php CODE: 200 URL: http://testphp.vulnweb.com/search.php CODE: 200 URL: http://testphp.vulnweb.com/userinfo.php?uid=1;
<b>SCAN TIME</b>
<b>Scan Finished:</b> 20/9/2024 6:37:32

ภาพที่ 3.11 ผลลัพธ์จากการใช้เครื่องมือ Uniscan เพื่อทำ Brute Force File Name

<b>- SCAN TIME</b>
<b>Scan Started:</b> 20/9/2024 6:39:10
<b>- TARGET</b>
<b>Domain</b> http://testphp.vulnweb.com/ <b>Server Banner:</b> nginx/1.19.0 <b>Target IP:</b> 44.228.249.3
<b>- CRAWLING</b>
<b>Directory check:</b> CODE: 200 URL: http://testphp.vulnweb.com/AJAX/ CODE: 200 URL: http://testphp.vulnweb.com/Flash/ CODE: 200 URL: http://testphp.vulnweb.com/admin/ CODE: 200 URL: http://testphp.vulnweb.com/hpp/ CODE: 200 URL: http://testphp.vulnweb.com/images/ CODE: 200 URL: http://testphp.vulnweb.com/pictures/ CODE: 200 URL: http://testphp.vulnweb.com/secured/
<b>- SCAN TIME</b>
<b>Scan Finished:</b> 20/9/2024 6:43:48

ภาพที่ 3.12 ผลลัพธ์จากการใช้เครื่องมือ Uniscan เพื่อทำ Brute Force Directory

3) Nikto ใช้สำหรับ Checks for Injectable Paths, Checks if Server is Outdated.Cheks for Server Issues และ Perform SSL checks.

```

└$ nikto -Plugins 'outdated' -h http://testaspnet.vulnweb.com
- Nikto v2.5.0
-----
+ Target IP:      44.238.29.244
+ Target Hostname: testaspnet.vulnweb.com
+ Target Port:    80
+ Start Time:    2024-09-25 04:48:58 (GMT-4)
-----
+ Server: Microsoft-IIS/8.5
+ : Server banner changed from 'Microsoft-IIS/8.5' to 'ATS/7.0.0'.
+ 239 requests: 0 error(s) and 1 item(s) reported on remote host
+ End Time:      2024-09-25 04:50:38 (GMT-4) (100 seconds)
-----
+ 1 host(s) tested

└(park㉿kali)-[~]
└$ nikto -Plugins 'paths' -h http://testaspnet.vulnweb.com
- Nikto v2.5.0
-----
+ Target IP:      44.238.29.244
+ Target Hostname: testaspnet.vulnweb.com
+ Target Port:    80
+ Start Time:    2024-09-25 05:00:05 (GMT-4)
-----
+ Server: Microsoft-IIS/8.5
+ : Server banner changed from 'Microsoft-IIS/8.5' to 'ATS/7.0.0'.
+ 240 requests: 0 error(s) and 1 item(s) reported on remote host
+ End Time:      2024-09-25 05:01:47 (GMT-4) (102 seconds)
-----
+ 1 host(s) tested

└(park㉿kali)-[~]
└$
```

ภาพที่ 3.13 ผลลัพธ์จากการใช้เครื่องมือ nikto เพื่อทำ Server-Side Request Forgery (SSRF)

4) Wapiti ใช้สำหรับ SQL Injections (Error based, boolean based, time based), Cross Site Scripting (XSS) แบบ reflected และ permanent, File disclosure detection (local and remote include, require, fopen) และ Checking remote code execution (RCE).

---

## Wapiti vulnerability report

Target: <http://testaspnet.vulnweb.com/>

Date of the scan: Sun, 22 Sep 2024 12:27:01 +0000. Scope of the scan: folder

---

### Summary

Category	Number of vulnerabilities found
Backup file	0
<a href="#">Blind SQL Injection</a>	34
Weak credentials	0
CRLF Injection	0
<a href="#">Content Security Policy Configuration</a>	1

---

ภาพที่ 3.14 ผลลัพธ์จากการใช้เครื่องมือ Wapiti ในการสรุปผลการตรวจสอบ

---

### [HTTP Secure Headers](#)

HttpOnly Flag cookie

Open Redirect

---

### [Secure Flag cookie](#)

SQL Injection

Server Side Request Forgery

---

### [Cross Site Scripting](#)

XML External Entity

---

### [Internal Server Error](#)



4

0

0

1

0

0

9

0

18

ภาพที่ 3.15 ผลลัพธ์จากการใช้เครื่องมือ Wapiti ในการตรวจสอบช่องโหว่เพิ่มเติม

## Blind SQL Injection

### Description

Blind SQL injection is a technique that exploits a vulnerability occurring in the database of an application. This kind of vulnerability is harder to detect than basic SQL injections because no error message will be displayed on the webpage.

### Vulnerability found in /Comments.aspx

Description      HTTP Request      cURL command line

Blind SQL vulnerability via injection in the parameter id

ภาพที่ 3.16 ผลลัพธ์จากการใช้เครื่องมือ Wapiti ในการตรวจสอบ SQL Injection

## Cross Site Scripting

### Description

Cross-site scripting (XSS) is a type of computer security vulnerability typically found in web applications which allow code injection by malicious web users into the web pages viewed by other users. Examples of such code include HTML code and client-side scripts.

### Vulnerability found in /Comments.aspx

Description      HTTP Request      cURL command line

XSS vulnerability found via injection in the parameter tbComment

### Vulnerability found in /Comments.aspx

Description      HTTP Request      cURL command line

XSS vulnerability found via injection in the parameter tbComment

ภาพที่ 3.17 ผลลัพธ์จากการใช้เครื่องมือ Wapiti ในการตรวจสอบ Cross Site Scripting

## Internal Server Error

### Description

An error occurred on the server's side, preventing it to process the request. It may be the sign of a vulnerability.

### Anomaly found in /ReadNews.aspx

Description	HTTP Request	cURL command line
The server responded with a 500 HTTP error code while attempting to inject a payload in the parameter NewsA		

### Anomaly found in /ReadNews.aspx

Description	HTTP Request	cURL command line
The server responded with a 500 HTTP error code while attempting to inject a payload in the parameter NewsA		

ภาพที่ 3.18 ผลลัพธ์จากการใช้เครื่องมือ Wapiti ในการตรวจสอบ Internal Server Error

- 5) Gobuster ใช้สำหรับการทำ Brute Forces for certain files on the Domain และ Brute Forces for certain directories on the Domain

```
(park㉿kali)-[~]
$ gobuster dir -u http://testphp.vulnweb.com -w /usr/share/wordlists/dirb/common.txt -t 50
=====
Gobuster v3.6
by OJ Reeves (@TheColonial) & Christian Mehlmauer (@firefart)
=====
[+] Url:          http://testphp.vulnweb.com
[+] Method:       GET
[+] Threads:      50
[+] Wordlist:     /usr/share/wordlists/dirb/common.txt
[+] Negative Status codes: 404
[+] User Agent:   gobuster/3.6
[+] Timeout:      10s
=====
Starting gobuster in directory enumeration mode
=====
/admin           (Status: 301) [Size: 169] [--> http://testphp.vulnweb.com/admin/]
/cgi-bin/         (Status: 403) [Size: 276]
/cgi-bin          (Status: 403) [Size: 276]
/crossdomain.xml (Status: 200) [Size: 224]
/CVS             (Status: 301) [Size: 169] [--> http://testphp.vulnweb.com/CVS/]
/CVS/Entries      (Status: 200) [Size: 1]
/CVS/Repository   (Status: 200) [Size: 8]
/CVS/Root         (Status: 200) [Size: 1]
/favicon.ico      (Status: 200) [Size: 894]
/images           (Status: 301) [Size: 169] [--> http://testphp.vulnweb.com/images/]
/index.php        (Status: 200) [Size: 4958]
/pictures         (Status: 301) [Size: 169] [--> http://testphp.vulnweb.com/pictures/]
/secured          (Status: 301) [Size: 169] [--> http://testphp.vulnweb.com/secured/]
/vendor           (Status: 301) [Size: 169] [--> http://testphp.vulnweb.com/vendor/]
Progress: 4614 / 4615 (99.98%)
```

ภาพที่ 3.19 ผลลัพธ์จากการใช้เครื่องมือ gobuster เพื่อทำ directory traversal

```
[+] Threads:          40
[+] Wordlist:         /usr/share/wordlists/dirb/common.txt
[+] Negative Status codes: 404
[+] User Agent:       gobuster/3.6
[+] Extensions:      png,js,csv,php,txt,jpg,jpeg
[+] Timeout:          10s
=====
Starting gobuster in directory enumeration mode
=====
/404.php           (Status: 200) [Size: 5271]
/admin              (Status: 301) [Size: 169] [--> http://testphp.vulnweb.com/admin/]
/artists.php        (Status: 200) [Size: 5328]
/cart.php           (Status: 200) [Size: 4903]
/categories.php    (Status: 200) [Size: 6115]
/cgi-bin            (Status: 403) [Size: 276]
/cgi-bin/           (Status: 403) [Size: 276]
/comment.php        (Status: 302) [Size: 1246] [--> ./index.php]
/crossdomain.xml   (Status: 200) [Size: 224]
/CVS                (Status: 301) [Size: 169] [--> http://testphp.vulnweb.com/Cvs/]
/CVS/Entries        (Status: 200) [Size: 1]
/CVS/Repository     (Status: 200) [Size: 8]
/CVS/Root            (Status: 200) [Size: 1]
/disclaimer.php    (Status: 200) [Size: 5524]
/favicon.ico        (Status: 200) [Size: 894]
Progress: 12751 / 36920 (34.54%) [ERROR] Get "http://testphp.vulnweb.com/energy.jpeg": context deadline exceeded (Client.Timeout exceeded while awaiting headers)
/guestbook.php      (Status: 200) [Size: 5391]
/images              (Status: 301) [Size: 169] [--> http://testphp.vulnweb.com/images/]
/index.php           (Status: 200) [Size: 4958]
/index.php           (Status: 200) [Size: 4958]
/login.php           (Status: 200) [Size: 5523]
/logout.php          (Status: 200) [Size: 4830]
/pictures             (Status: 301) [Size: 169] [--> http://testphp.vulnweb.com/pictures/]
/product.php         (Status: 200) [Size: 5056]
```

ภาพที่ 3.20 ผลลัพธ์จากการใช้เครื่องมือ gobuster เพื่อทำ File Brute Force

### 3.3.2.3 ตัวอย่างข้อมูลคะแนน Common Vulnerability Scoring System

The screenshot shows the NIST CVSS v4.0 calculator interface at the URL <https://nvd.nist.gov/vuln-metrics/cvss/v4-calculator?name=CV-2024-10421>. The interface is divided into two main sections: Base Metrics and Vulnerable System Impact Metrics.

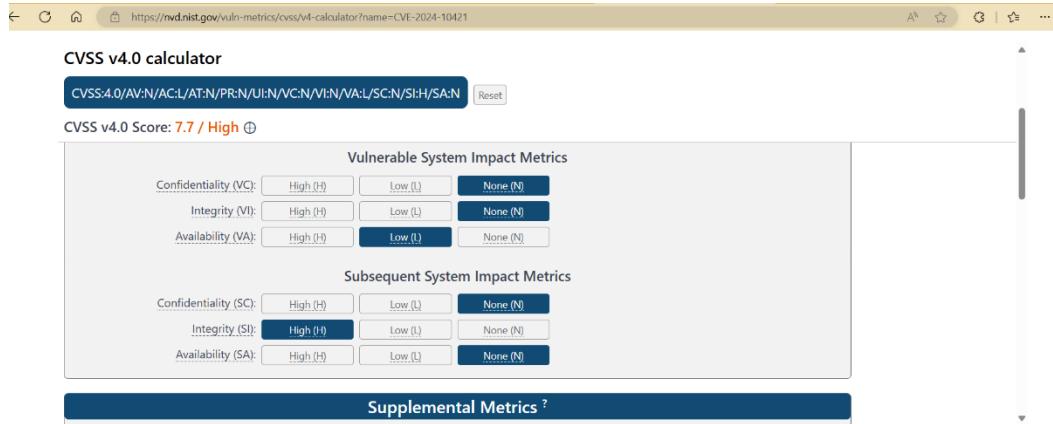
**Base Metrics:**

- Attack Vector (AV):** Network (N)
- Exploitability Metrics:**
  - Attack Complexity (AC): Low (L)
  - Attack Requirements (AT): None (N)
  - Privileges Required (PR): None (N)
  - User Interaction (UI): None (N)

**Vulnerable System Impact Metrics:**

- Confidentiality (VC):** High (H)
- Impact (IV):** Low (L)

ภาพที่ 3.21 ตัวอย่างเครื่องคำนวณคะแนน CVSS



ภาพที่ 3.22 ตัวอย่างเครื่องคำนวณคะแนน CVSS เมื่อกรอกค่าตามมาตรฐาน

### 3.4 ประเด็นที่น่าสนใจและท้าทาย

#### 3.4.1 การจัดการข้อมูลผลลัพธ์ที่ได้จากเครื่องมือ

ข้อมูลผลลัพธ์ที่ได้จากเครื่องมือนั้นมีรูปแบบของไฟล์ที่ไม่เหมือนกัน ซึ่งการจัดทำ Log เพื่อให้ผู้ใช้งานสามารถอ่านได้นั้นจำเป็นที่จะต้องนำข้อมูลในรูปแบบที่ต่างกันมารวมกันให้เป็นรูปแบบเดียว ทำให้การรวมข้อมูลเป็นไฟล์เดียวกันนั้นค่อนข้างที่จะท้าทายต่อการพัฒนาเครื่องมือ สแกนหาช่องโหว่

#### 3.4.2 ระยะเวลาในการทำงานของเครื่องมือที่ใช้ในการพัฒนา

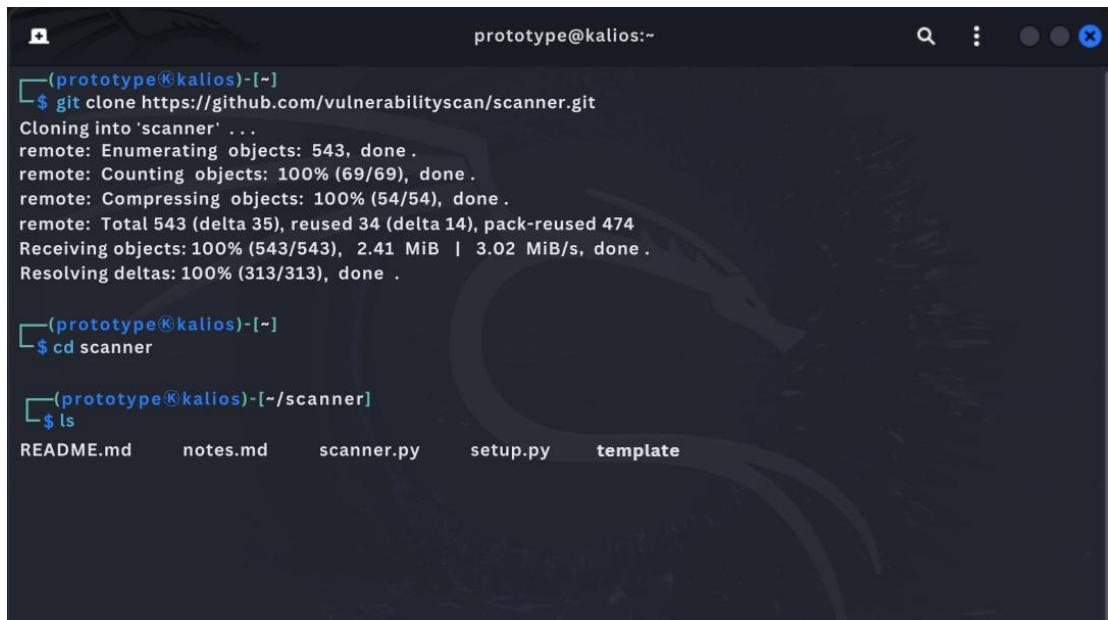
แต่ละเครื่องมือที่นำมาใช้นั้นมีระยะเวลาการทำงานที่แตกต่างกันตามความซับซ้อนของคำสั่งที่ใช้ บางเครื่องมือมีการประมวลผลแบบ multithreading มาให้ ส่งผลให้ใช้เวลาในการทำงานที่น้อย แต่บางเครื่องมือไม่ได้มีการอัปเดตระบบประมวลผลแบบ multithreading ทำเวลาในการใช้งานโดยรวมของเครื่องมือที่จะพัฒนาเพิ่มขึ้นตาม

### 3.5 ผลลัพธ์ที่คาดหวัง

- การระบุช่องโหว่ได้อย่างครอบคลุม เครื่องมือสามารถตรวจจับช่องโหว่ต่าง ๆ ได้อย่างมีประสิทธิภาพครอบคลุมประเภทช่องโหว่ที่สำคัญ เช่น Broken Access Control, SQL Injection, Cross-Site Scripting (XSS), Remote Code Execution (RCE) และ Server-Side Request Forgery (SSRF) เพื่อให้มั่นใจว่าระบบได้รับการป้องกันจากช่องโหว่ที่สำคัญ

2. การทำงานอัตโนมัติที่ช่วยลดภาระการทำงานของนักทดสอบ เครื่องมือช่วยให้สามารถสแกนและตรวจหาช่องโหว่ได้โดยอัตโนมัติ ลดความจำเป็นในการสแกนด้วยตนเองและเพิ่มความแม่นยำในการประเมินผล
3. การแสดงลำดับความสำคัญของปัญหาช่องโหว่ เครื่องมือสามารถจัดลำดับความสำคัญของช่องโหว่ตามความเสี่ยงและความร้ายแรง เพื่อช่วยให้องค์กรสามารถตัดสินใจในการแก้ไขปัญหาได้อย่างรวดเร็วและมีประสิทธิภาพ
4. การรายงานผลที่สามารถเข้าใจได้ง่าย เครื่องมือสามารถแสดงผลลัพธ์ในรูปแบบของ IoT ที่ชัดเจน เช่น จัดลำดับความสำคัญของช่องโหว่ตามระดับความเสี่ยงและสถานะของระบบโดยไม่จำเป็นต้องมีความรู้เชิงลึก
5. เพิ่มประสิทธิภาพการทำงาน การสแกนด้วยเครื่องมือนี้จะช่วยให้กระบวนการสแกนช่องโหว่เร็วขึ้น ลดเวลาและทรัพยากรที่ต้องใช้ในการทดสอบด้วยตนเอง

### 3.6 ระบบต้นแบบและผลลัพธ์เบื้องต้น



```
(prototype㉿kalios)-[~]
$ git clone https://github.com/vulnerabilityscan/scanner.git
Cloning into 'scanner' ...
remote: Enumerating objects: 543, done.
remote: Counting objects: 100% (69/69), done.
remote: Compressing objects: 100% (54/54), done.
remote: Total 543 (delta 35), reused 34 (delta 14), pack-reused 474
Receiving objects: 100% (543/543), 2.41 MiB | 3.02 MiB/s, done.
Resolving deltas: 100% (313/313), done.

(prototype㉿kalios)-[~]
$ cd scanner

(prototype㉿kalios)-[~/scanner]
$ ls
README.md      notes.md      scanner.py      setup.py      template
```

ภาพที่ 3.23 การติดตั้งเครื่องมือบนระบบปฏิบัติการ Kali Linux

```
prototype@kalios:~
```

```
(prototype㉿kalios) [~/scanner]
```

```
$ ls
```

```
README.md  notes.md  scanner.py  setup.py  template
```

ภาพที่ 3.24 โครงสร้างของไดเรกทอรีเครื่องมือเบื้องต้น

```
prototype@kalios:~
```

```
/ \
```

```
| |
```

```
| >_ COMMAND - v
```

```
| >_ CHECKING ...
```

```
\ /
```

```
Protecting your web server by scanning Broken Access Control (BAC),  
Path traversal, SQL injection, Cross-site Scripting, Remote code execution (RCE), and Server-side request  
forgery (SSRF).
```

```
> [ Checking version tag of the scanner.py file... Reading ... ]
```

```
[ tag v1.0 ] --> scanner.py
```

```
> [ Checking version tag of the scanner.py file... Complete ... ]
```

ภาพที่ 3.25 ใช้คำสั่ง python3 scanner.py -v เพื่อเรียกตรวจสอบเบื้องต้น

```

prototype@kalios:~         

/ _____ \
| |   | |
| >_ COMMAND -U
| |   | |
| >_ CHECKING ...
| |   | |
\_____|_
      ||

Protecting your web server by scanning Broken Access Control (BAC),
Path traversal, SQL injection, Cross-site Scripting, Remote code execution (RCE), and Server-side request
forgery (SSRF).

>> [ Compare hash of the scanner.py file ... Reading ... ]
[ latest version ] --> scanner.py
>> [ Checking version tag of the scanner.py file ... Complete ... ]

```

ภาพที่ 3.26 ใช้คำสั่ง python3 scanner.py -u เพื่อเรียกอัพเดตเวอร์ชัน

```

prototype@kalios:~         

VULNSCAN
Protecting your web server by scanning Broken Access Control (BAC), Path
traversal, SQL injection, Cross-site Scripting, Remote code execution (RCE), and
Server-side request forgery (SSRF).

SYNOPSIS
python3 [scanner.py] [COMMAND]

COMMAND
-v, -V           checking the version tag of scanner.py file
-u, -U           updating the scanner .py version to the latest version
-h, -H           opening help manual for command details
target          choosing the website to use for scanner.py
                [www.example.com], [https://www.example.com] or
                [http://www.example.com]

```

ภาพที่ 3.27 ภาพครุ่นวือที่แสดงเมื่อเรียกใช้คำสั่ง -h

```
prototype@kalios:~ 
[>] WWW.EXAMPLE.COM
[>] SCANNING ...
Protecting your web server by scanning Broken Access Control (BAC),
Path traversal, SQL injection, Cross-site Scripting, Remote code execution (RCE), and Server-side request
forgery (SSRF).
> [ Checking available security tools in tool list phase ... Checking. ]
All tools are available . The scanner will perform the task normally .
> [ Checking available security tools in tool list phase ... Complete. ]
```

ภาพที่ 3.28 ใช้คำสั่ง python3 scanner.py www.example.com

```
prototype@kalios:~ 
[>] WWW.EXAMPLE.COM
[>] SCANNING ...
Protecting your web server by scanning Broken Access Control (BAC),
Path traversal, SQL injection, Cross-site Scripting, Remote code execution (RCE), and Server-side request
forgery (SSRF).
> [ Checking available security tools in tool list phase ... Checking ... ]
Some of these tools [ 'wafw00f', 'gobuster', 'nmap' ] are unavailable or will be skipped.
The scanner will still perform the rest of the tasks. Install these tools to utilize the functionality of the
SCANNER fully.
> [ Checking available security tools in tool list phase ... Complete. ]
```

ภาพที่ 3.29 การตรวจสอบสภาพแวดล้อมว่าพร้อมใช้งานหรือไม่

```

prototype@kalios:~ > [ Main scan Phase Checking... Loading 16 tasks... ]
>_ ( Broken Access Control ) | 1/16 | Uniscan brute force files on the domain.
scanning [*****]
Task complete. No vulnerability was found in this task.

>_ ( Broken Access Control ) | 2/16 | Uniscan brute force directory on the domain.
scanning [*****]

```

ภาพที่ 3.30 สแกนหาช่องโหว่และไม่พบช่องโหว่

```

prototype@kalios:~ > [ Main scan Phase Checking... Loading 16 tasks... ]
>_ ( Broken Access Control ) | 1/16 | Uniscan brute force files on the domain.
scanning [*****]
Task complete. No vulnerability was found in this task.

>_ ( Broken Access Control ) | 2/16 | Uniscan brute force directory on the domain.
scanning [*****]

Trigger high Private directories are publicly open to outsider.

>_ ( Broken Access Control ) | 3/16 | Gobuster brute force files on the domain.
scanning [*****]

```

ภาพที่ 3.31 สแกนแล้วพบช่องโหว่บนเว็บเซิร์ฟเวอร์

```

prototype@kalios:~         

-> [ Creating log file from all task... Generating ... ]
Creating a Log file for www.example.com named log.www.example.com.txt
The log file is available in the same scanner directory.

Vulnerability Task Check      : 16
Vulnerability Tasks Skipped   : 14
Vulnerability Threat Detected : 1
Total Time                     : 5m 21sec

You can view the report by using the report.html file for the log visualization
presentation.

-> [ Creating log file from all task... Complete ... ]

```

ภาพที่ 3.32 รายงานผลลัพธ์เมื่อเสร็จสิ้นกระบวนการ

```

prototype@kalios:~         

-> ( Broken Access Control ) | 2/16 | Uniscan brute force directory on the domain.

cmd -> uniscan -w -u

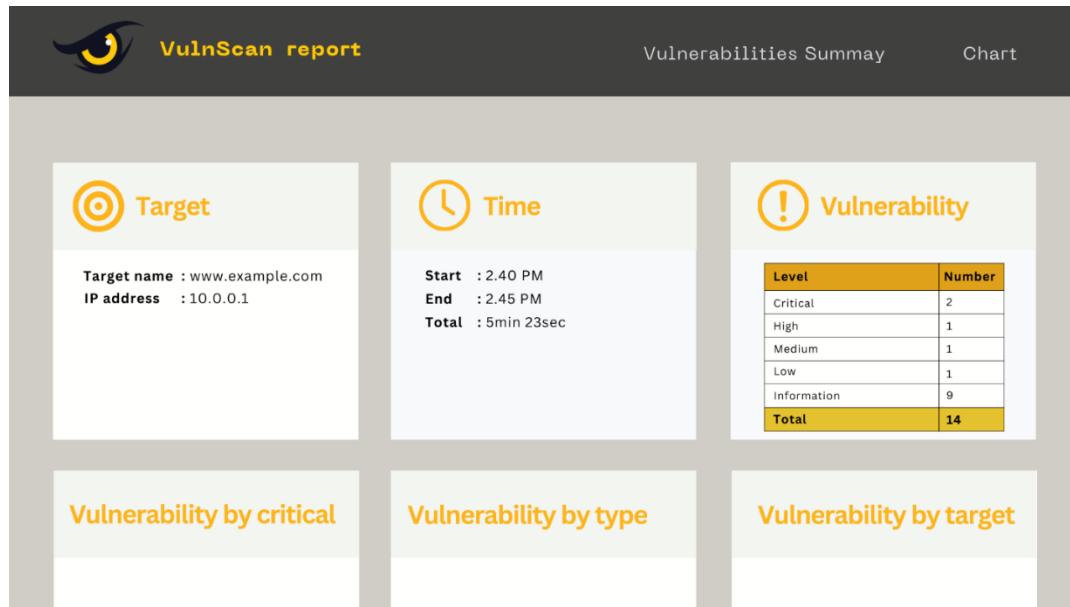
Vulnerability Threat Level
high Private directories are publicly open to everyone.

Vulnerability Information
Path traversal attack (directory traversal) aims to access files and directories stored outside the web root folder. By manipulating variables that reference files with "dot-dot-slash (..)" sequences and their variations or by using absolute file paths, it may be possible to access arbitrary files and directories stored on the file system including application source code or configuration and critical system files. It should be noted that access to files is limited by system operational access control (such as in the case of locked or in-use files on the Microsoft Windows operating system).

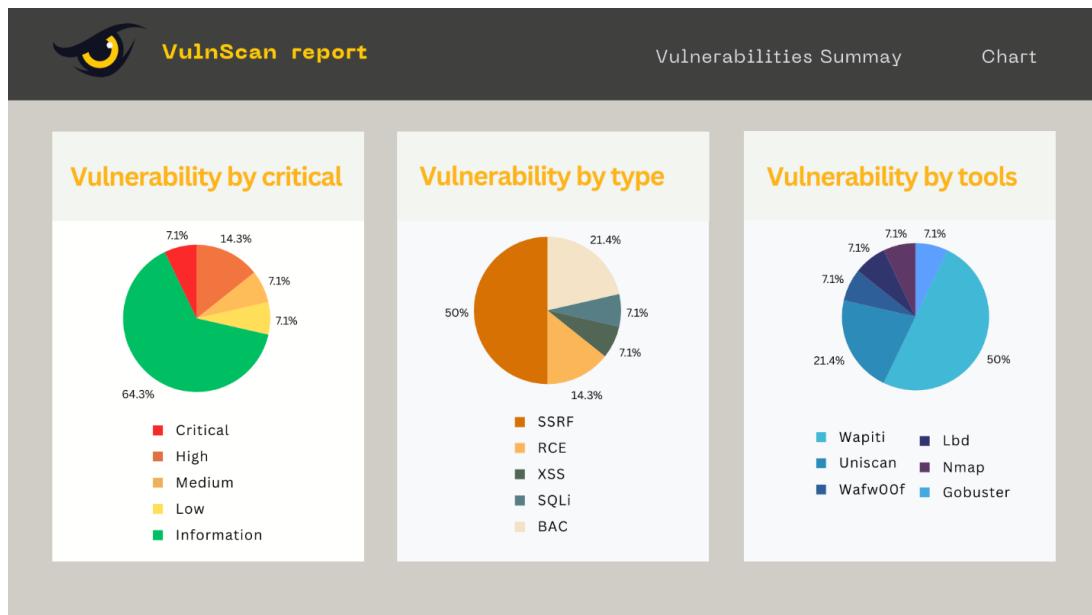
Vulnerability Remediation
The most effective way to prevent path traversal vulnerabilities is to avoid passing user-supplied input to filesystem APIs altogether. Many application functions that do this can be rewritten to deliver the same behavior more safely. If you can't avoid passing user-supplied input to filesystem APIs, we recommend using two layers of defense to prevent attacks

```

ภาพที่ 3.33 ตัวอย่างของ log file



ภาพที่ 3.34 หน้าเว็บเพจ Vulnerabilities Summary ของไฟล์ report.html



ภาพที่ 3.35 หน้าเว็บเพจ log visualization ของไฟล์ report.html

## บทที่ 4

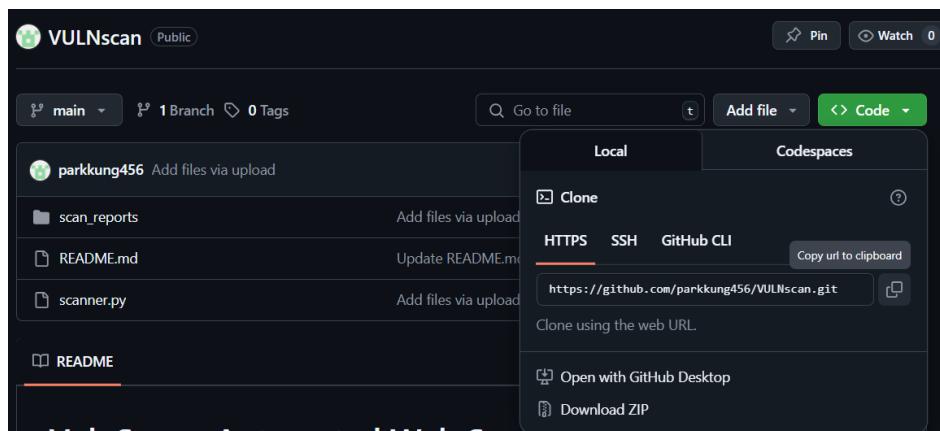
### ผลการดำเนินงาน

#### 4.1 ผลการพัฒนาโครงการ

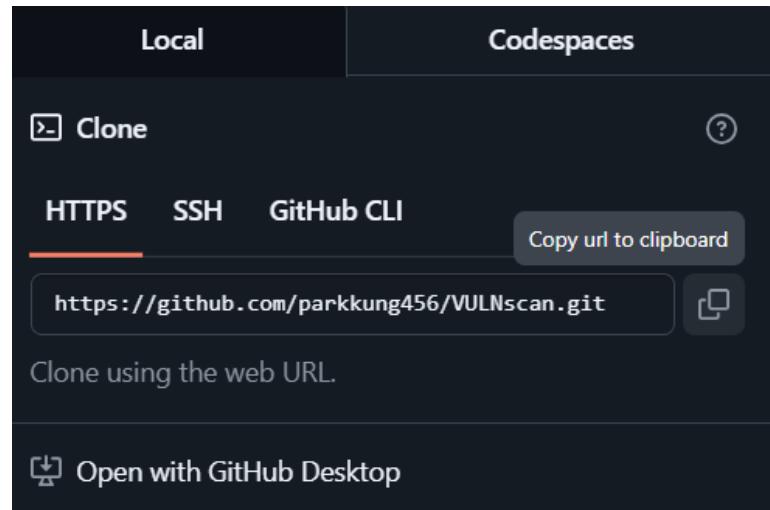
ผลการพัฒนาโครงการนี้ ได้ถูกเผยแพร่เป็นเครื่องมือแบบโอเพ่นซอร์สผ่านแพลตฟอร์ม GitHub ในชื่อ VulnScan ซึ่งเป็นเครื่องมือที่ช่วยในการตรวจสอบช่องโหว่ของ Web Server แบบอัตโนมัติ โดยมีจุดเด่นที่การรวมเครื่องมือตรวจสอบด้านความมั่นคงปลอดภัยที่มีชื่อเสียงหลากหลายชนิดไว้ในโปรแกรมเดียว เช่น Nmap, Nikto, Uniscan, Wapiti, Gobuster และ Lbd เป็นต้น ทำให้ผู้ใช้งานสามารถดำเนินการสแกนช่องโหว่ได้อย่างครอบคลุมภายในขั้นตอนเดียว ผลลัพธ์จากการสแกนจะถูกจัดหมวดหมู่ชัดเจนตามระดับความรุนแรงของช่องโหว่ ได้แก่ Low, Medium, High, และ Critical พร้อมแสดงคำแนะนำในการแก้ไขเพื่อให้ผู้ใช้งานนำไปปรับปรุงความปลอดภัยของระบบได้อย่างรวดเร็ว

ตัวเครื่องมือยังโดยเด่นด้วยระบบรายงานผลที่ครบถ้วน โดยสามารถสร้างรายงานได้หลายรูปแบบ ไม่ว่าจะเป็นไฟล์สรุปแบบข้อความ (TXT) และรายงาน HTML แบบ Interactive ซึ่งประกอบด้วยกราฟและแผนภูมิแสดงข้อมูลที่ชัดเจน เข้าใจง่าย ทำให้สามารถวิเคราะห์และประเมินระดับความเสี่ยงได้อย่างมีประสิทธิภาพ นอกจากนี้ VulnScan ยังรองรับการอัปเดตเครื่องมือให้เป็นเวอร์ชันล่าสุดจาก GitHub แบบอัตโนมัติ จึงเป็นเครื่องมือที่ใช้งานได้สะดวก ง่ายต่อการดูแลรักษา และพร้อมรองรับการใช้งานในอนาคต

#### 4.2 ภาพตัวอย่างเครื่องมือ VulnScan



ภาพที่ 4.1 git repository ของเครื่องมือ VulnScan



ภาพที่ 4.2 git repository url สำหรับติดตั้งเครื่องมือ

```
(root㉿kali)-[~/home/kali/demo]
# git clone https://github.com/parkkung456/VULNscan.git
Cloning into 'VULNscan' ...
remote: Enumerating objects: 79, done.
remote: Counting objects: 100% (79/79), done.
remote: Compressing objects: 100% (75/75), done.
remote: Total 79 (delta 28), reused 0 (delta 0), pack-reused 0 (from 0)
Receiving objects: 100% (79/79), 137.45 KiB | 1.60 MiB/s, done.
Resolving deltas: 100% (28/28), done.

(root㉿kali)-[~/home/kali/demo]
# ls
VULNscan
```

ภาพที่ 4.3 ใช้คำสั่ง git clone เพื่อติดตั้งเครื่องมือ

การติดตั้งเครื่องมือ VULNscan สามารถติดตั้งได้ผ่านคำสั่ง git clone โดยสามารถคัดลอก git repository url เพื่อใช้ในคำสั่งได้ตามภาพที่ 4.2 คือ “https://github.com/parkkung456/VULNscan.git” และนำ url ดังกล่าวไปต่อท้ายคำสั่ง git clone จะได้คำสั่งดังนี้ “git clone https://github.com/parkkung456/VULNscan.git” เป็นอันเสร็จการติดตั้ง

```
(root@kali)-[~/home/kali/cs403/VULNscan]
# python3 scanner.py -V
Places
  Computer
  kali
  Desktop
  Recent
  Trash
  Documents
  Music
  Pictures
  Videos
>_ CHECKING . . .
>_
Version: 1.0
```

ภาพที่ 4.4 การแสดงผลของคำสั่ง -V หรือ Version

การใช้งานเครื่องมือสามารถพิมพ์คำสั่งได้ดังนี้ “python3 scanner.py” และตามด้วยพาร์meter ข้อต่อไปนี้ 4.4 เป็นคำสั่ง -V ใช้เพื่อตรวจสอบเวอร์ชันของเครื่องมือ ซึ่งคำสั่งที่ใช้คือ “python3 scanner.py -V”

```
(root@kali)-[~/home/kali/cs403/VULNscan]
# python3 scanner.py -U
Places
  Computer
  kali
  Desktop
  Recent
  Trash
  Documents
  Music
  Pictures
  Videos
>_ UPDATING . . .
>_
Cloning into 'VULNscan' ...
remote: Enumerating objects: 73, done.
remote: Counting objects: 100% (73/73), done.
remote: Compressing objects: 100% (69/69), done.
remote: Total 73 (delta 24), reused 0 (delta 0), pack-reused 0 (from 0)
Receiving objects: 100% (73/73), 134.79 KiB | 1.62 MiB/s, done.
Resolving deltas: 100% (24/24), done.
```

ภาพที่ 4.5 การแสดงผลของคำสั่ง -U หรือ Update

อ้างอิงจากภาพที่ 4.5 เป็นคำสั่ง -V ใช้เพื่ออัพเดตเครื่องมือให้เป็นเวอร์ชันปัจจุบัน ซึ่งคำสั่งที่ใช้คือ “python3 scanner.py -U” ซึ่งตัวโปรแกรมจะใช้คำสั่ง “git clone https://github.com/parkkung456/VULNscan.git” เพื่อดำเนินการติดตั้งเครื่องมือเวอร์ชันใหม่

```
(root㉿kali)-[~/home/kali/cs403/VULNscan] # python3 scanner.py -H
Usage: python3 scanner.py [options]
Options:
  -V, --version      Print the version of this program.
  -U, --update       Replace every file and directory with:
                    git clone https://github.com/parkkung456/VULNscan.git
  -H, --help          Show this help message and exit.
  target            Must be a DNS name string. for examples "goooole.com", "testphp.vulnweb.com"
If no option is provided, the program runs the normal vulnerability scan.
```

ภาพที่ 4.6 การแสดงผลของคำสั่ง -H หรือ Help

อ้างอิงจากภาพที่ 4.6 เป็นคำสั่ง -H ใช้เพื่อแสดงคู่มือการใช้งานและรายละเอียดฟังก์ชันต่าง ๆ ที่สามารถใช้งานได้ ซึ่งคำสั่งที่ใช้คือ “python3 scanner.py -H”

```
(root㉿kali)-[~/home/kali/cs403/VULNscan] # python3 scanner.py testphp.vulnweb.com
Starting security scan on http://testphp.vulnweb.com (IP: 44.228.249.3) ...
→ [ Checking available security tools in tool list phase . . . Checking. ]
All tools are available. The scanner will perform the task normally .
→ [ Checking available security tools in tool list phase . . . Complete. ]
→ [ Main scan Phase Checking . . . Loading Task . . . ]
→ [NMAP] - Checking for open ports...
[nmap] 80/tcp open http nginx 1.19.0 detected as LOW
→ [NMAP_SQLSERVER] - Checking for SQL Server...
Task complete.No vulnerability found for nmap_sqlserver.
→ [NMAP_MYSQL] - Checking for MySQL Server...
Task complete.No vulnerability found for nmap_mysql.
→ [NMAP_ORACLE] - Checking for Oracle Server...
Task complete.No vulnerability found for nmap_oracle.
→ [NIKTO] - Checking for Apache Expect XSS Header...
[nikto] + 240 requests: 0 error(s) and 0 item(s) reported on remote host detected as LOW
→ [UNISCAN_RCE] - Performing RCE & RFI scan...
Task complete.No vulnerability found for uniscan_rce.
```

ภาพที่ 4.7 การแสดงผลเมื่อใส่เป้าหมายเว็บเชิร์ฟเวอร์ที่ต้องการตรวจสอบ

อ้างอิงจากภาพที่ 4.7 เป็นคำสั่ง -Target ใช้เพื่อสแกนหาช่องโหว่ของเว็บเซิร์ฟเวอร์ โดยป้อน Domain Name เข้าไปเพื่อเริ่มสแกนซึ่งคำสั่งที่ใช้คือ “python3 scanner.py -testphp.vulnweb.com”

```

→ [GOBUSTER_DIRECTORY_TRAVERSAL] - Checking for Directory Traversal ...
[gobuster_directory_traversal] /cgi-bin          (Status: 403) [Size: 276] detected as LOW
[gobuster_directory_traversal] /images           (Status: 301) [Size: 169] [→ http://testphp.vulnweb.com/images/] detected as LOW
[gobuster_directory_traversal] /admin            (Status: 301) [Size: 169] [→ http://testphp.vulnweb.com/admin/] detected as LOW
[gobuster_directory_traversal] /pictures          (Status: 301) [Size: 169] [→ http://testphp.vulnweb.com/pictures/] detected as LOW
[gobuster_directory_traversal] /vendor            (Status: 301) [Size: 169] [→ http://testphp.vulnweb.com/vendor/] detected as LOW
[gobuster_directory_traversal] /Templates          (Status: 301) [Size: 169] [→ http://testphp.vulnweb.com/Templates/] detected as LOW
[gobuster_directory_traversal] /Flash              (Status: 301) [Size: 169] [→ http://testphp.vulnweb.com/Flash/] detected as LOW
[gobuster_directory_traversal] /CVS                (Status: 301) [Size: 169] [→ http://testphp.vulnweb.com/CVS/] detected as LOW
[gobuster_directory_traversal] /AJAX               (Status: 301) [Size: 169] [→ http://testphp.vulnweb.com/AJAX/] detected as LOW
[gobuster_directory_traversal] /secured             (Status: 301) [Size: 169] [→ http://testphp.vulnweb.com/secured/] detected as LOW
[gobuster_directory_traversal] /14726              (Status: 500) [Size: 177] detected as LOW
[gobuster_directory_traversal] /snscan             (Status: 500) [Size: 177] detected as LOW
[gobuster_directory_traversal] /ebay1               (Status: 500) [Size: 177] detected as LOW

→ [UNISCAN_DIRECTORY_TRAVERSAL] - Checking for Accessible Directory ...
[uniscan_directory_traversal] | [+| CODE: 200 URL: http://testphp.vulnweb.com/AJAX/ detected as HIGH
[uniscan_directory_traversal] | [+| CODE: 200 URL: http://testphp.vulnweb.com/Flash/ detected as HIGH
[uniscan_directory_traversal] | [+| CODE: 200 URL: http://testphp.vulnweb.com/admin/ detected as HIGH
[uniscan_directory_traversal] | [+| CODE: 200 URL: http://testphp.vulnweb.com/hpp/ detected as HIGH
[uniscan_directory_traversal] | [+| CODE: 200 URL: http://testphp.vulnweb.com/images/ detected as HIGH
[uniscan_directory_traversal] | [+| CODE: 200 URL: http://testphp.vulnweb.com/pictures/ detected as HIGH
[uniscan_directory_traversal] | [+| CODE: 200 URL: http://testphp.vulnweb.com/secured/ detected as HIGH
[uniscan_directory_traversal] | [+| CODE: 200 URL: http://testphp.vulnweb.com/secured/ detected as HIGH

→ [UNISCAN_FILE_TRAVERSAL] - Checking for Accessible File ...
[uniscan_file_traversal] | [+| CODE: 200 URL: http://testphp.vulnweb.com/CVS/Entries detected as HIGH
[uniscan_file_traversal] | [+| CODE: 200 URL: http://testphp.vulnweb.com/favicon.ico detected as HIGH
[uniscan_file_traversal] | [+| CODE: 200 URL: http://testphp.vulnweb.com/index.php detected as HIGH
[uniscan_file_traversal] | [+| CODE: 200 URL: http://testphp.vulnweb.com/login.php detected as HIGH
[uniscan_file_traversal] | [+| CODE: 200 URL: http://testphp.vulnweb.com/search.php detected as HIGH
[uniscan_file_traversal] | [+| CODE: 200 URL: http://testphp.vulnweb.com/userinfo.php?uid=1; detected as HIGH

→ [NIKTO_OUTDATED] - Checking for Outdated components...
[nikto_outdated] + 239 requests: 0 error(s) and 0 item(s) reported on remote host detected as LOW

→ [NIKTO_ACCESSIBLE_PATHS] - Checking for Accessible Paths ...
[nikto_accessible_paths] + 240 requests: 0 error(s) and 0 item(s) reported on remote host detected as LOW
→ [ Main scan Phase Checking . . . Complete ]

```

ภาพที่ 4.8 การแสดงผลการแจ้งเตือนเมื่อตรวจพบช่องโหว่

จากการสแกนเว็บเซิร์ฟเวอร์ด้วยเครื่องมือ VULNSCAN มีการใช้เครื่องมือย่อยอย่าง Gobuster, Uniscan และ Nikto ในการตรวจสอบความปลอดภัย โดย Gobuster ใช้ brute-force เพื่อค้นหา directory และไฟล์ที่ซ่อนอยู่ ซึ่งตรวจพบหลาย path ที่น่าสงสัย เช่น /admin/, /Flash/, และ /secured/ โดยมีสถานะ HTTP เป็น 301 และ 403 ซึ่งระบุความเสี่ยงระดับต่ำ (LOW) แต่เมื่อวิเคราะห์ด้วย Uniscan กลับพบว่า directory เหล่านี้สามารถเข้าถึงได้ (HTTP 200) ทำให้ถูกจัดเป็นความเสี่ยงระดับสูง (HIGH) เนื่องจากอาจเปิดเผยข้อมูลสำคัญโดยไม่ตั้งใจ นอกจากนี้ Uniscan ยังตรวจพบไฟล์ที่สามารถเข้าถึงได้โดยตรง เช่น login.php, index.php, และ userinfo.php?uid=1 ซึ่งเพิ่มความเสี่ยงต่อการโจมตี เช่น SQL Injection หรือการเข้าถึงโดยไม่ได้รับอนุญาต ขณะที่ Nikto ตรวจสอบการตั้งค่าของเว็บเซิร์ฟเวอร์และไม่พบการใช้ชอฟต์แวร์ที่ล้าสมัย รวมถึงไม่มีการค้นพบ path ที่เป็นอันตรายเพิ่มเติม ทำให้ระบุความเสี่ยงอยู่ในระดับต่ำ (LOW) เช่นกัน

```

→ [ Creating log file from all task . . . Generating . . . ]

Final Summary:
_____
Total Vulnerability Task check      : 16
Total Tool skipped                  : 0
Total Vulnerability Thread detected: CRITICAL(0), HIGH (13), MEDIUM(84), LOW (80)
Total time from scan                : 1212.28 seconds

_____
Scan completed.
Report saved as scan_reports/21-04-2025-0249-scan_report_testphp.vulnweb.com.txt
Full raw output saved in scan_reports/21-04-2025-0229-raw_report_testphp.vulnweb.com.txt
HTML report generated: scan_reports/21-04-2025-0249-scan_report_testphp.vulnweb.com.html

→ [ Creating log file from all task . . . Complete . . . ]

```

ภาพที่ 4.9 การแสดงผลข้อมูลโดยภาพรวมและการบันทึกข้อมูล

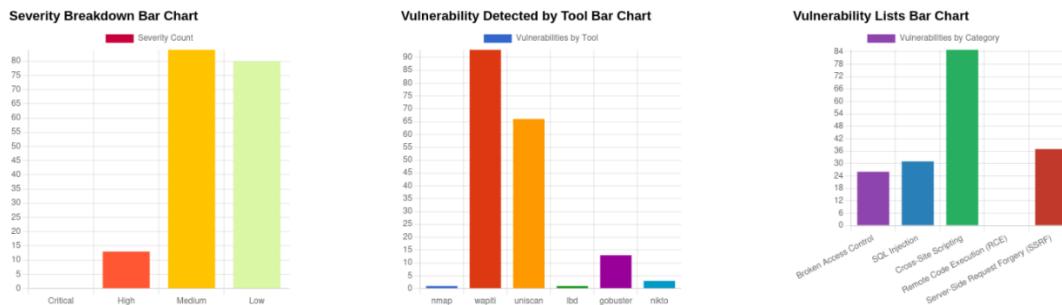
จากภาพที่ 4.9 แสดงเป็นช่วงสรุปผลสุดท้ายของกระบวนการสแกนซึ่งให้ผลลัพธ์ของเครื่องมือ VULNSCAN ซึ่งแสดงผลการสแกนแบบรวมทั้งหมด โดยมีการดำเนินการตรวจสอบทั้งหมดจำนวน 16 รายการ ไม่มีเครื่องมือใดที่ถูกข้าม (Tool skipped: 0) และพบช่องโหว่รวมทั้งหมด 177 รายการ แบ่งออกเป็นระดับความรุนแรง สูง (HIGH) จำนวน 13 รายการ, ระดับปานกลาง (MEDIUM) จำนวน 84 รายการ และ ระดับต่ำ (LOW) จำนวน 80 รายการ โดยไม่พบช่องโหว่ที่มีความรุนแรงระดับ วิกฤต (CRITICAL) เลย การสแกนใช้เวลาทั้งสิ้นประมาณ 1,212.28 วินาที หรือราว 20 นาที ผลการสแกนได้ถูกจัดเก็บในรูปแบบรายงานข้อความธรรมดา (TXT) และ HTML เพื่อให้สามารถดูรายละเอียดเชิงลึกและวิเคราะห์ต่อไปได้ในภายหลัง รายงานเหล่านี้ถูกจัดเก็บไว้ในโฟลเดอร์ scan\_reports โดยมีชื่อไฟล์ที่สอดคล้องกับวันที่และโดเมนเป้าหมายของการสแกน (testphp.vulnweb.com)

VulnScan Report		Vulnerabilities Summary Charts
Vulnerabilities Summary		
Target URL:	http://testphp.vulnweb.com	
Resolved IP:	44.228.249.3	
Date and Time of Scan:	21-04-2025-0249	
Total Scan Duration:	1212.28 seconds	
Total Tasks Executed:	16	
Total Tools Skipped:	0	
Severity Breakdown:	Critical: 0 High: 13 Medium: 84 Low: 80	
Overall Risk Rating:	High	

ภาพที่ 4.10 ตัวอย่างรายละเอียดโดยสรุปจากไฟล์ HTML

จากภาพที่ 4.10 คือตัวอย่างไฟล์ dashboard.html โดยส่วนแรกที่จะแสดงคือตารางแสดงรายละเอียดของการสแกนหาช่องโหว่ เครื่องมือ VULNscan จะรวบรวมข้อมูลเกี่ยวกับเว็บเซิร์ฟเวอร์

ที่เป็นเป้าหมายในการทดสอบไว้เป็นตาราง ซึ่งข้อมูลที่แสดงจะประกอบไปด้วย Target URL, IP Address, date-month-year and time of the scan, Total Scan Duration, Total Task Scan, Total Tools Skipped, Severity Breakdown และ Overall Risk Rating



ภาพที่ 4.11 ตัวอย่างรายละเอียดแผนภูมิจากไฟล์ HTML

จากภาพที่ 4.11 คือส่วนที่ 2 ของไฟล์ dashboard.html ตัวอย่างรายละเอียดแผนภูมิจากไฟล์ HTML เป็นแผนภูมิแท่งที่นำข้อมูลจากไฟล์รายงานผล raw\_report.txt มาประกอบผลการแสดง แบ่งออกเป็น 3 แผนภูมิได้แก่ Severity Breakdown Bar Chart คือแผนภูมิแท่งที่จะนำระดับความเสี่ยงของช่องโหว่ที่พบมาใช้ในการแสดงข้อมูล ลำดับถัดไปคือแผนภูมิ Vulnerability Detected by Tool Bar Chart เป็นแผนภูมนีจะนำเสนอว่าแต่ละเครื่องมืออยู่ใน VULNscan มีปริมาณช่องโหว่ที่ตรวจพบเท่าใด และแผนภูมิสุดท้ายคือ Vulnerability Lists Bar Chart เป็นการนำข้อมูลช่องโหว่ที่พบมาจัดหมวดหมู่ ได้แก่ Broken Access Control, SQL Injection, Cross Site Scripting, Remote Code Execution และ Server Side Request Forgery เป็นต้น

#### Detailed Tool Results

Command Executed	Threat Level	Vulnerability Information	Recommended Remediation
nmap -sV http://testphp.vulnweb.com	LOW	Nmap (Low): Open port detected. Low risk if no sensitive services are running.	Review the open ports and disable services that are unnecessary.
nikto -Plugins 'apache_expect_xss' -host http://testphp.vulnweb.com	LOW	Nikto (Low): Minor misconfigurations noted in HTTP headers.	Verify web server configurations and minimize banner information.
uniscan -d -u http://testphp.vulnweb.com	MEDIUM	Uniscan XSS (Medium): XSS vulnerability that may be exploitable under certain conditions.	Implement stricter input validation and output encoding.
lbd http://testphp.vulnweb.com	MEDIUM	LBD (Medium): Minor load balancing misconfiguration detected.	Review load balancer settings and apply necessary security updates.
wapiti -m sql -u http://testphp.vulnweb.com --verbose 2	MEDIUM	Wapiti SQL (Medium): Potential SQL injection vulnerability requiring further analysis.	Apply input sanitization and update vulnerable SQL queries.
wapiti -m ssrf -u http://testphp.vulnweb.com --verbose 2	MEDIUM	Wapiti SSRF (Medium): Potential SSRF vulnerability identified.	Implement input validation and restrict server-side requests.
wapiti -m xss -u http://testphp.vulnweb.com --verbose 2	MEDIUM	Wapiti XSS (Medium): Potential XSS vulnerability that should be investigated.	Patch the application to fix XSS vulnerabilities.
gobuster dir -w /usr/share/wordlists/dirbuster/directory-list-2.3-small.txt -t 100 -u http://testphp.vulnweb.com	LOW	Gobuster (Low): Directory listing appears benign.	Review directory permissions and ensure that only necessary directories are exposed.
uniscan -q -u http://testphp.vulnweb.com	HIGH	No information available.	No remediation available.
uniscan -w -u http://testphp.vulnweb.com	HIGH	No information available.	No remediation available.
nikto -Plugins 'outdated' -host http://testphp.vulnweb.com	LOW	Nikto Outdated (Low): Outdated software components detected with minimal risk.	Verify the software components and update if necessary.
nikto -Plugins 'paths' -host http://testphp.vulnweb.com	LOW	Nikto Accessible Paths (Low): Minor exposure of non-sensitive accessible paths detected.	Review file and directory permissions to ensure non-essential paths are not publicly accessible.

ภาพที่ 4.12 ตัวอย่างรายละเอียดเชิงเทคนิคจากไฟล์ HTML

จากภาพที่ 4.12 คือส่วนสุดท้ายของไฟล์ dashboard.html เป็นการนำข้อมูลจากไฟล์ scan\_report.txt มาแสดงในรูปแบบตาราง ซึ่งประกอบไปด้วย Command Executed คือคำสั่งที่ VULNscan เรียกใช้งานเพื่อเรียกเครื่องมืออยู่ ๆ มาสแกนหาช่องโหว่ คอลัมน์ถัดไปคือ Threat Level ที่จะแสดงระดับความรุนแรงของช่องโหว่ที่พบ ตามด้วยคอลัมน์ Vulnerability Information ที่จะอธิบายข้อมูลของช่องโหว่เหล่านั้นคืออะไร ลำดับสุดท้ายคือ Recommended Remediation ที่จะอธิบายวิธีการดูแลและแก้ไขช่องโหว่ที่พบ

```

Scan Report for http://testphp.vulnweb.com
Task : Running nmap - Checking for open ports ...
Command : nmap -sV http://testphp.vulnweb.com
Vulnerability threat level : low
Vulnerability Information: Nmap (Low): Open port detected. Low risk if no sensitive services are running.
Vulnerability remediation: Review the open ports and disable services that are unnecessary.

Task : Running nikto - Checking for Apache Expect XSS Header ...
Command : nikto -Plugins 'apache_expect_xss' -host http://testphp.vulnweb.com
Vulnerability threat level : low
Vulnerability Information: Nikto (Low): Minor misconfigurations noted in HTTP headers.
Vulnerability remediation: Verify web server configurations and minimize banner information.

Task : Running uniscan_xss - Performing BSQLi, SQLi, & XSS scan...
Command : uniscan -d -u http://testphp.vulnweb.com
Vulnerability threat level : medium
Vulnerability Information: Uniscan XSS (Medium): XSS vulnerability that may be exploitable under certain conditions.
Vulnerability remediation: Implement stricter input validation and output encoding.

Task : Running lbd - Checking for load balancing ...
Command : lbd http://testphp.vulnweb.com
Vulnerability threat level : medium
Vulnerability Information: LBD (Medium): Minor load balancing misconfiguration detected.
Vulnerability remediation: Review load balancer settings and apply necessary security updates.

Task : Running wapiti_sqli - Checking for SQL Injection ...
Command : wapiti -m sql -u http://testphp.vulnweb.com --verbose 2
Vulnerability threat level : medium
Vulnerability Information: Wapiti SQLI (Medium): Potential SQL injection vulnerability requiring further analysis.
Vulnerability remediation: Apply input sanitization and update vulnerable SQL queries.

Task : Running wapiti_ssrf - Checking for Server-side Request Forgery ...
Command : wapiti -m ssrf -u http://testphp.vulnweb.com --verbose 2
Vulnerability threat level : medium
Vulnerability Information: Wapiti SSRF (Medium): Potential SSRF vulnerability identified.
Vulnerability remediation: Implement input validation and restrict server-side requests.

```

ภาพที่ 4.13 ตัวอย่างรายละเอียดเชิงเทคนิคจากไฟล์ scan\_report.txt

ภาพที่ 4.13 เป็นรายละเอียดของไฟล์ scan\_report.txt ที่จะสร้างขึ้นเมื่อเครื่องมือ VULNscan ตรวจพบช่องโหว่ โดยรายละเอียดภายในไฟล์จะประกอบไปด้วย Task, Command, Vulnerability threat level, Vulnerability Information, และ Vulnerability remediation ที่จะถูกสร้างขึ้นเมื่อช่องโหว่เหล่านั้นถูกพบ ยกตัวอย่างจากภาพที่ 4.13 บรรทัดที่ 21 ถึง 25

บรรทัดที่ 21 “Task : Running lbd – Checking for load balancing...” เป็นการกล่าวถึงว่าช่องโหว่นี้ใช้เครื่องมือ lbd ในการตรวจสอบว่าเป้าหมายในการทดสอบนั้นมีการใช้งาน Load Balancer หรือไม่

บรรทัดที่ 22 “Command : lbd http://testphp.vulnweb.com” คือคำสั่งที่จะถูกเรียกใช้ในการสแกนหาช่องโหว่ ในที่นี้คือคำสั่ง “lbd” และตามด้วย “http://testphp.vulnweb.com” ที่เป็นเพ้าหมายในการสแกน การแสดงคำสั่งที่ใช้งานจะช่วยให้นักทดสอบเข้าใจช่องโหว่มากยิ่งขึ้น

บรรทัดที่ 23 “Vulnerability threat level : medium” คือการวิเคราะห์ระดับความเสี่ยงและความรุนแรงของช่องโหว่ โดยเครื่องมือวิเคราะห์ได้ว่าอยู่ในระดับปานกลาง (medium) การกำหนดค่า Load Balancer ที่ไม่เหมาะสมอาจทำให้เกิดพฤติกรรมที่ผิดปกติ เช่น การส่งทรัพฟิกไปยังเซิร์ฟเวอร์ที่ไม่พร้อมใช้งาน หรือการกระจายโหลดที่ไม่สม่ำเสมอ

บรรทัดที่ 24 “Vulnerability Information” เป็นการอธิบายว่าช่องโหว่ที่เกิดขึ้นคืออะไร ซึ่งจากภาพที่ 4.13 ระบุว่า การตั้งค่าของ Load Balancer มีข้อผิดพลาดดังนี้

บรรทัดที่ 25 “Vulnerability remediation” เป็นการอธิบายวิธีการดูแลและแก้ไขช่องโหว่เหล่านั้น จากภาพที่ 4.13 ระบุว่า โปรดตรวจสอบการตั้งค่าและอัพเดตความปลอดภัยที่จำเป็นต่อ Load Balancer

```

1 Raw Scan Report for http://testphp.vulnweb.com (IP: 44.228.249.3)
2 =====
3
4 = nmap Scan Output =
5 Starting Nmap 7.94SVN ( https://nmap.org ) at 2025-04-21 02:29 EDT
6 Nmap scan report for ec2-44-228-249-3.us-west-2.compute.amazonaws.com (44.228.249.3)
7 Host is up (0.0078s latency).
8 Not shown: 998 filtered tcp ports (no-response)
9 PORT      STATE SERVICE      VERSION
10 25/tcp    open  tcpwrapped
11 80/tcp    open   http        nginx 1.19.0
12
13 Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
14 Nmap done: 1 IP address (1 host up) scanned in 78.05 seconds
15
16 80/tcp    open   http        nginx 1.19.0 - Severity: l
17
18 = nmap_sqlserver Scan Output =
19 Starting Nmap 7.94SVN ( https://nmap.org ) at 2025-04-21 02:30 EDT
20 Nmap done: 1 IP address (1 host up) scanned in 2.12 seconds
21
22
23 = nmap_mysql Scan Output =
24 Starting Nmap 7.94SVN ( https://nmap.org ) at 2025-04-21 02:30 EDT
25 Nmap done: 1 IP address (1 host up) scanned in 2.11 seconds
26
27
28 = nmap_oracle Scan Output =
29 Starting Nmap 7.94SVN ( https://nmap.org ) at 2025-04-21 02:30 EDT
30 Nmap done: 1 IP address (1 host up) scanned in 2.10 seconds
31
32

```

ภาพที่ 4.14 ผลลัพธ์ทั้งหมดจากการบันกราบการทำงานของเครื่องมือจากไฟล์ raw\_report.txt

ภาพที่ 4.14 เป็นรายละเอียดของไฟล์ raw\_report.txt ซึ่งเป็นไฟล์ที่จะเก็บการทำงานทุกบรรทัดของเครื่องมืออยู่อย่างต่อๆ เพื่อใช้สำหรับตรวจสอบขั้นตอนการประมวลผลของ Task นั้น ๆ

## บทที่ 5

### สรุป

#### 5.1 สรุปผลการพัฒนาโครงการ

โครงการนี้ได้พัฒนาเครื่องมือ VULNscan ซึ่งเป็นเครื่องมือสแกนช่องโหว่ของเว็บไซต์ฟอร์มแบบอัตโนมัติ บนพื้นฐานของระบบปฏิบัติการ Debian โดยผลลัพธ์จากการพัฒนาเครื่องมือนี้สามารถตอบโจทย์ที่คาดหวังไว้ได้ อย่างมีประสิทธิภาพดังนี้

##### 5.1.1 การระบุช่องโหว่ได้อย่างครอบคลุม

เครื่องมือ VULNscan มีความสามารถเครื่องมือสแกนหลายเครื่องมือ ได้แก่ nmap, nikto, uniscan, wapiti, lbd และ gobuster เพื่อระบุช่องโหว่สำคัญได้อย่างครบถ้วน ไม่ว่าจะเป็น Broken Access Control, SQL Injection, Cross-Site Scripting (XSS), Remote Code Execution (RCE) และ Server-Side Request Forgery (SSRF) ทำให้ระบบได้รับการป้องกันจากช่องโหว่ที่สำคัญได้อย่างครอบคลุม

##### 5.1.2 การทำงานอัตโนมัติที่ช่วยลดภาระการทำงานของนักทดสอบ

เครื่องมือสามารถทำงานแบบอัตโนมัติเต็มรูปแบบ ช่วยลดภาระงานของนักทดสอบโดยสามารถดูแลกระบวนการสแกนด้วยตนเอง เพิ่มความแม่นยำในการประเมินผล และสามารถอัปเดตเครื่องมืออัตโนมัติได้ด้วย คำสั่ง -U (Update) เพื่อให้ได้รับเครื่องมือรุ่นใหม่อยู่เสมอ

##### 5.1.3 การแสดงลำดับความสำคัญของปัญหาช่องโหว่

เครื่องมือ VULNscan สามารถจัดลำดับช่องโหว่ตามระดับความร้ายแรง ตั้งแต่ระดับต่ำ (Low) ไปจนถึง ระดับวิกฤต (Critical) โดยแสดงผลอย่างชัดเจนผ่านรายงานที่ช่วยให้องค์กรสามารถจัดการและแก้ไขช่องโหว่ได้อย่างรวดเร็วและตรงจุด อีกทั้งยังมีการแจ้งเตือนช่องโหว่ที่พบในขณะที่กำลังสแกนหาช่องโหว่อีกด้วย

##### 5.1.4 การรายงานผลที่สามารถเข้าใจได้ง่าย

เครื่องมือได้ออกแบบให้รายงานผลลัพธ์ในรูปแบบ ISO ที่เข้าใจง่าย โดยผู้ใช้งาน

ทั่วไปสามารถอ่านและเข้าใจสถานะระบบได้โดยไม่จำเป็นต้องมีความรู้เชิงลึกด้านความปลอดภัย อีกทั้งยังมีการสร้าง รายงานในรูปแบบ HTML และ TXT ที่มีการแสดงผลเป็นกราฟ ข้อมูลรายละเอียดเชิงลึกและตารางรายงานเชิง เทคนิค เพื่อให้ง่ายต่อการวิเคราะห์ผลลัพธ์และการนำเสนอ

## 5.2 แนวคิดในการต่อยอดและปรับปรุง

### 5.2.1 แนวทางการเพิ่มความน่าเชื่อถือของช่องโหว่ที่พบ

เพื่อเพิ่มความน่าเชื่อถือและยืนยันช่องโหว่ที่เครื่องมือพบว่ามีอยู่จริง ในอนาคตควรเพิ่มขั้นตอนการตรวจสอบช้า (Verification) หรือ Task เช่นที่ทำการตรวจสอบยืนยันช่องโหว่ด้วยเทคนิคอื่นหรือเครื่องมือเสริมต่าง ๆ เช่น การนำเครื่องมือเช่น Burp Suite หรือ OWASP ZAP เข้ามาช่วยทำการทดสอบช่องโหว่แบบ Active Verification โดยตรง เพื่อให้มั่นใจได้ว่าช่องโหว่ที่รายงานออกมานั้นสามารถเกิดขึ้นได้จริง ไม่ใช่เพียงแค่ False Positive ที่อาจเกิดจากข้อผิดพลาดของการสแกนเบื้องต้นเท่านั้น นอกจากนี้ อาจเพิ่มกระบวนการยืนยันแบบ Proof-of-Concept (PoC) ในรูปแบบที่แสดงหลักฐานเป็นรูปภาพหรือวิดีโอสั้นๆ ประกอบรายงาน เพื่อให้ผู้ใช้งานสามารถนำไปปรับปรุงและแก้ไขได้ตรงประเด็นและรวดเร็วขึ้น

### 5.2.2 การขยายขอบเขตและเพิ่มความครอบคลุมของช่องโหว่

แม้ว่าเครื่องมือ VULNscan จะมีประสิทธิภาพในการตรวจสอบช่องโหว่ตาม OWASP Top 10 ในปัจจุบันแล้ว แต่เพื่อให้ครอบคลุมร้ายแรงมากขึ้น ใหม่ ๆ ที่เกิดขึ้นอย่างต่อเนื่อง จึงมีความจำเป็นที่จะขยายขอบเขตการตรวจสอบเพิ่มเติมจาก OWASP Top 10 ไปสู่มาตรฐานด้านความปลอดภัยอื่น ๆ ที่เป็นที่ยอมรับในระดับสากล เช่น CWE/SANS Top 25 Most Dangerous Software Errors, OWASP API Security Top 10 หรือแม้กระทั่งช่องโหว่เฉพาะทางที่เกี่ยวข้องกับเทคโนโลยีใหม่ๆ เช่น ช่องโหว่ที่เกี่ยวข้องกับ Cloud Security หรือ Microservices เพื่อให้ VULNscan มีความสามารถในการตรวจจับช่องโหว่ที่ครอบคลุมและทันสมัยยิ่งขึ้น อีกทั้งสามารถตอบโจทย์ด้านความปลอดภัยสำหรับองค์กรที่ใช้งานระบบเทคโนโลยีที่หลากหลายได้อย่างทั่วถึง

### 5.2.3 การเปิดรับ Feedback และความร่วมมือจากผู้ใช้งาน

เพื่อส่งเสริมการพัฒนาอย่างต่อเนื่องและทำให้ VULNscan เป็นเครื่องมือที่

ตอบสนองความต้องการของผู้ใช้อย่างแท้จริง ควรเปิดรับ Feedback หรือข้อเสนอแนะจากผู้ใช้งานโดยตรงผ่านช่องทางของ GitHub หรือ Community Platform อื่น ๆ ที่เกี่ยวข้อง ผู้ใช้งานสามารถเสนอแนะการปรับปรุง ฟีเจอร์ที่ต้องการเพิ่มเติม รายงานบັກ และการสนับสนุนความสามารถใหม่ๆ ได้อย่างสะดวกและรวดเร็ว การเปิดรับ Feedback อย่างเปิดกว้างจะช่วยสร้างชุมชนผู้ใช้งานที่แข็งแกร่ง และช่วยให้โครงการได้รับข้อมูลเชิงลึกที่มีคุณค่าในการพัฒนาเครื่องมือให้ดียิ่งขึ้น รวมถึงยังเป็นการสร้างความเชื่อมั่นและความสัมพันธ์อันดีระหว่างทีมพัฒนาและผู้ใช้งานอย่างยั่งยืน

#### 5.2.4 พัฒนาปรับปรุงเครื่องมือจากรูปแบบ CLI ให้เป็นเครื่องมือแบบ GUI

ในปัจจุบันเครื่องมือ VULNscan ทำงานผ่าน Command Line Interface (CLI) ซึ่งจะเหมาะสมสำหรับผู้ใช้งานระดับมืออาชีพ แต่ยังไม่เป็นมิตรกับผู้ใช้งานทั่วไปหรือผู้ที่ไม่มีพื้นฐานด้านการพัฒนาและความปลอดภัยไซเบอร์มากนัก ดังนั้น การพัฒนาต่อยอดให้ VULNscan มี Graphical User Interface (GUI) จะเป็นแนวทางที่สามารถเพิ่มประสิทธิภาพและประสบการณ์ของผู้ใช้งานได้อย่างมาก GUI ควรมีระบบแบบฟอร์มที่ผู้ใช้งานสามารถเลือกเครื่องมือย่อยที่ต้องการใช้ เช่น nmap, nikto, หรือ wapiti ได้ตามความต้องการของผู้ใช้งาน และสามารถเลือกประเภทช่องโหว่ย่อยที่ต้องการทดสอบได้ เช่น SQL Injection, XSS, หรือ SSRF โดยไม่จำเป็นต้องใช้ร้อให้เครื่องมือทำงานครบทุกเครื่องมือ

### 5.3 สิ่งที่ได้เรียนรู้หลังจากการพัฒนาโครงการ

#### 5.3.1 Environment ที่เหมาะสมในการใช้งานเครื่องมือ VULNscan

เครื่องมือ VULNscan ได้รับการออกแบบและพัฒนาบนระบบปฏิบัติการ Debian ซึ่งเป็นระบบปฏิบัติการแบบโอเพ่นซอร์สที่มีเสถียรภาพและได้รับการยอมรับอย่างแพร่หลายในแวดวงการทดสอบเจาะระบบและความปลอดภัยไซเบอร์ ดังนั้น Environment ที่เหมาะสมที่สุดในการใช้งานเครื่องมือนี้ คือ ระบบที่ใช้พื้นฐาน Debian เช่น Kali Linux, Ubuntu, Parrot OS และ Linux Mint โดยเฉพาะ Kali Linux ซึ่งถูกออกแบบมาสำหรับงานทดสอบความปลอดภัยโดยเฉพาะ เครื่องมือสามารถทำงานได้อย่างเต็มประสิทธิภาพในสภาพแวดล้อมที่ติดตั้งเครื่องมือที่ใช้ในการทดสอบเจาะระบบขั้นพื้นฐานมาอย่างครบถ้วน และสิทธิ์ของผู้ใช้งาน (user privilege) ควรมีความเหมาะสม เช่น การรันคำสั่งด้วยสิทธิ์ root หรือ sudo เพื่อให้เครื่องมือสามารถทำงานและเข้าถึงข้อมูลที่จำเป็นได้แบบอัตโนมัติ

### 5.3.2 การเลือกเป้าหมายสำหรับการทดสอบ

เป้าหมายที่จะใช้สำหรับการทดสอบเครื่องมือ VULNscan ต้องเป็นเว็บไซต์ เว็บเซิร์ฟเวอร์ หรือเว็บแอปพลิเคชันที่ทางองค์กรต่าง ๆ อนุญาตเท่านั้น หรือจะเป็นเว็บไซต์ เว็บเซิร์ฟเวอร์ หรือเว็บแอปพลิเคชันที่ทางองค์กรเหล่านั้นออกแบบและพัฒนา เพื่อให้นักทดสอบสามารถฝึกหักษะ หรือทดลองเครื่องมือโดยไม่ผิดกฎหมาย การเลือกเป้าหมายที่มีเอกสารหรือคู่มือเกี่ยวกับซองโหวตจะช่วยให้ผู้ทดสอบสามารถประเมินประสิทธิภาพของเครื่องมือได้อย่างมีประสิทธิภาพ

### 5.3.3 การนำเครื่องมือไปใช้งานต้องปฏิบัติอย่างไร

การนำเครื่องมือ VULNscan ที่เป็นเครื่องมือโอเพ่นซอร์สไปใช้งานในสถานการณ์จริง เช่น การทดสอบระบบขององค์กรหรือบริษัท จำเป็นต้องดำเนินการภายใต้กรอบของกฎหมาย และจรรยาบรรณด้านความปลอดภัยไซเบอร์อย่างเคร่งครัด โดยต้องมีการจัดทำข้อตกลงหรือสัญญา (Agreement) ระหว่างผู้ทดสอบและเจ้าของระบบอย่างเป็นทางการ ระบุขอบเขตการทดสอบ (Scope), ระยะเวลา, และความรับผิดชอบในกรณีที่เกิดผลกระทบต่อระบบหรือเกิดความเสียหายที่ไม่ได้ตั้งใจ การกำหนดขอบเขตเหล่านี้จะช่วยป้องกันไม่ให้เกิดการเข้าใจผิดหรือการกระทำที่เข้าข่ายผิดกฎหมาย เช่น Unauthorized Access การนำเครื่องมือโอเพ่นซอร์สไปใช้งานในทางที่ไม่เหมาะสม อาจส่งผลเสียทั้งทางกฎหมายและชื่อเสียงขององค์กรหรือผู้ใช้ ดังนั้นการใช้งานต้องมีความรับผิดชอบสูง และควรคำนึงถึงจริยธรรมเป็นสำคัญ

### 5.3.4 รูปแบบการทำงานของเครื่องมือ

เครื่องมือ VULNscan ประกอบด้วยเครื่องมือย่อยหลายตัวที่ทำงานร่วมกันโดยมีรูปแบบการทำงานแบบลองทุกวิธีที่เป็นไปได้ หรือ Brute-Force เป็นหลัก ยกตัวอย่างเช่น gobuster ใช้การ brute-force เพื่อค้นหา Directory และไฟล์ที่ซ่อนอยู่ในเซิร์ฟเวอร์ โดยนำไฟล์ wordlist.txt ที่รวบรวมรายชื่อ Directory หรือไฟล์ที่ใช้บอยมาจากการ Kali Linux มาใช้ในการ brute-force, nikto ใช้วิธีการสุมร่องขอ (request) เพื่อหาช่องโหว่ใน HTTP headers และเว็บเซิร์ฟเวอร์, ในขณะที่ wapiti และ uniscan ตรวจสอบพารามิเตอร์ต่างๆ ด้วยเทคนิค brute-force เพื่อหาช่องโหว่ในเว็บแอปพลิเคชัน

รูปแบบการทำงานแบบ Brute-force ทำให้สามารถระบุช่องโหว่ได้อย่างครอบคลุม แต่ต้องแลกมากับการใช้เวลาและทรัพยากรอย่างมาก ผู้ใช้งานจึงควรเข้าใจว่า VULNscan มีลักษณะการโจมตีเชิงรุก (Active Scan) ซึ่งอาจส่งผลกระทบต่อระบบเป้าหมายโดยตรง เช่น โหลดสูงหรืออาจจะทำให้เว็บเซิร์ฟเวอร์ล่มได้หากไม่กำหนดค่าให้เหมาะสม ดังนั้นควรวางแผนการใช้งานอย่างระัดระวัง เช่น ทำการทดสอบในช่วงเวลาที่ระบบไม่ใช้งานจริง (off-peak hours) หรือในสภาพแวดล้อมก่อนที่จะเปิดใช้งานบริการเหล่านั้น (Staging Environment)

## รายการอ้างอิง

- Abu-Databaseh, F., & Alshammari, E. (2018). *Automated Penetration Testing : An Overview* Computer Science & Information Technology,
- Ariah, S., & Brightwood, S. (2024). Exploring the Evolution and Impact of Software Automation: From Manual Tasks to Autonomous Systems.
- Hertzog, R., O'Gorman, J., & Aharoni, M. (2017). *Kali Linux revealed: Mastering the penetration testing distribution*. Offsec Press. (*Official guide to Kali Linux, including installation and usage.*)
- Invicti Security. (n.d.). Acunetix Vulnerable Test Sites. <http://www.vulnweb.com>
- Kumar, B., Bejo, S. P., Kedia, R., Banerjee, P., Jha, P., & Dehury, M. K. (2023, 14-16 July 2023). Kali Linux based Empirical Investigation on Vulnerability Evaluation using Pen-Testing tools. 2023 World Conference on Communication & Computing (WCONF),
- Lyon, G. F. (2009). *Nmap network scanning: The official Nmap project guide to network discovery and security scanning*. Insecure.Com LLC.
- Moreno, A. C., Hernandez-Suarez, A., Sanchez-Perez, G., Toscano-Medina, L. K., Perez-Meana, H., Portillo-Portillo, J., Olivares-Mercado, J., & García-Villalba, L. J. (2025). Analysis of autonomous penetration testing through reinforcement learning and recommender systems. *Sensors*, 25(1), 211.  
<https://doi.org/10.3390/s25010211>
- Oracle Corporation. (2022). *Oracle VM VirtualBox* (Version 7.0) [Computer software].  
<https://www.virtualbox.org>
- OWASP. (2021). *OWASP Top 10 – 2021: The ten most critical web application security risks*. OWASP Foundation. <https://owasp.org/Top10/>
- Offensive Security. (n.d.). (2024, March 11). Uniscan. *Kali Linux Tools*.  
<https://www.kali.org/tools/uniscan/>
- Offensive Security. (n.d.). (2024, March 11). Nikto. *Kali Linux Tools*.  
<https://www.kali.org/tools/nikto/>

Offensive Security. (n.d.). (2024, March 11). Wapiti. Kali Linux Tools.

<https://www.kali.org/tools/wapiti/>

Offensive Security. (n.d.). (2024, March 11). Gobuster. *Kali Linux Tools*.

<https://www.kali.org/tools/gobuster/>

Scarfone, K., Souppaya, M., Cody, A., & Orebaugh, A. (2008). *Technical guide to information security testing and assessment* (NIST Special Publication 800-115). National Institute of Standards and Technology.

Shebli, H. M. Z. A., & Beheshti, B. D. (2018, 4-4 May 2018). A study on penetration testing process and tools. 2018 IEEE Long Island Systems, Applications and Technology Conference (LISAT),