A Deep Dive into Nmap Scripts for Web Application Testing

Khaleel Khan

·

Follow

Published in

System Weakness

4 min read

·

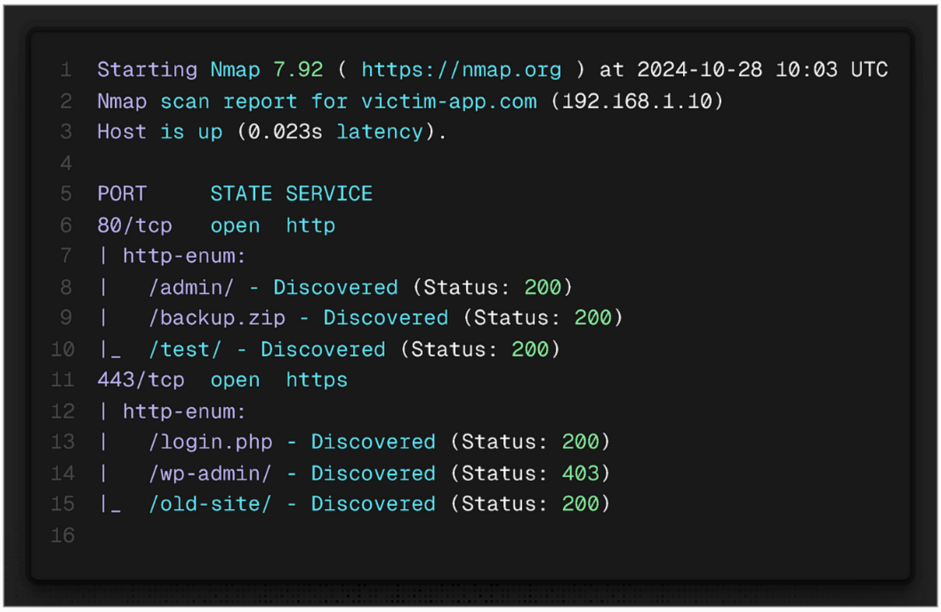
Nov 13, 2024

Listen

Share

More

A Step-by-Step Guide to Leveraging Nmap’s Most Advanced Scripts for Comprehensive Web Application Security Analysis



```
Starting Nmap 7.92 ( https://nmap.org ) at 2024-10-28 10:03 UTC
Nmap scan report for victim-app.com (192.168.1.10)
Host is up (0.023s latency).

PORT      STATE SERVICE
80/tcp    open  http
| http-enum:
|   /admin/ - Discovered (Status: 200)
|   /backup.zip - Discovered (Status: 200)
|_  /test/ - Discovered (Status: 200)
443/tcp   open  https
| http-enum:
|   /login.php - Discovered (Status: 200)
|   /wp-admin/ - Discovered (Status: 403)
|_  /old-site/ - Discovered (Status: 200)
```

Nmap Scripts for Web Application Testing

This scenario showcases how an experienced penetration tester could leverage Nmap's most powerful scripts to probe a web application thoroughly. Let's dive into how Nmap can uncover hidden weaknesses in a complex application, http://victim-app.com, revealing detailed methods and insights for each step.

The Target: http://victim-app.com

The target for this engagement is an e-commerce web application hosted on victim-app.com. The goal is to conduct a full security assessment by identifying open ports, enumerating services, and detecting potential vulnerabilities.

Step 1: Initial Recon — Scanning Ports and HTTP Titles

Starting with a basic port scan is crucial to identify active services. Adding the http-title script helps quickly gather insights about the services running on common web ports, such as 80, 443, and 8080.

```
nmap -p 80,443,8080 --script http-title victim-app.com
```

Output:

```
PORT STATE SERVICE
80/tcp open http
| http-title: Welcome to Victim App – Your One-Stop Shop!
443/tcp open https
| http-title: Victim App – Secure Shopping
8080/tcp open http-proxy
| http-title: Administration Login
```

This reveals a potentially interesting admin login page on port 8080, a detail to revisit later.

Step 2: Enumerating HTTP Methods

Next, the http-methods script is run to check for HTTP methods that the server allows. Unrestricted methods like PUT or DELETE could enable unauthorized actions on the server.

```
nmap -p 80,443,8080 --script http-methods victim-app.com
```

Output:

```
PORT STATE SERVICE
80/tcp open http
| http-methods:
| Supported Methods: GET POST OPTIONS HEAD
443/tcp open https
| http-methods:
| Supported Methods: GET POST OPTIONS HEAD PUT
DELETE
8080/tcp open http-proxy
| http-methods:
| Supported Methods: GET POST OPTIONS HEAD DELETE
```

The presence of DELETE on ports 443 and 8080 indicates a potential risk. This information is noted for further investigation.

Step 3: Testing for SQL Injection Vulnerabilities

To further probe the application, the http-sql-injection script is used to check for possible SQL injection vulnerabilities in endpoints where the server may not validate inputs properly.

```
nmap -p 80 --script http-sql-injection victim-app.com
```

Output:

```
PORT STATE SERVICE
80/tcp open http
| http-sql-injection:
| Possible SQL injection in /search.php?q=1:
| Payload: /search.php?q=1' OR '1'='1
```

A possible SQL injection vulnerability is flagged on the search endpoint, and this critical finding is logged for follow-up testing.

Step 4: Enumerating Sensitive Directories

The http-enum script is used next to search for common files and directories that could contain valuable information or unprotected content.

```
nmap -p 80,443 --script http-enum victim-app.com
```

Output:

```
PORT STATE SERVICE
80/tcp open http
| http-enum:
| /admin/ - Found
| /backup.zip - Found
| /test/ - Found
443/tcp open https
| http-enum:
| /login.php - Found
| /wp-admin/ - Found
| /old-site/ - Found
```

This scan reveals several intriguing paths, including an admin directory and backup.zip, which could contain sensitive data.

Step 5: Collecting Server Details with HTTP Headers

The http-headers script provides valuable server information, such as security-related HTTP headers, to understand the server’s configuration.\

```
nmap -p 80,443 --script http-headers victim-app.com
```

Output:

```
PORT STATE SERVICE
80/tcp open http
| http-headers:
| X-Frame-Options: SAMEORIGIN
| Content-Security-Policy: default-src 'self'
| Server: Apache/2.4.18 (Ubuntu)
443/tcp open https
| http-headers:
| Strict-Transport-Security: max-age=31536000;
includeSubDomains
| X-XSS-Protection: 1; mode=block
```

The results confirm the server is running Apache on Ubuntu and has security headers like Strict-Transport-Security, adding some layers of protection.

Step 6: Testing for XSS Vulnerabilities

Cross-Site Scripting (XSS) vulnerabilities can be highly damaging, so the http-xssed script is employed to check for XSS flaws within the application.

```
nmap -p 80 --script http-xssed victim-app.com
```

Output:

```
PORT STATE SERVICE
80/tcp open http
| http-xssed:
| /contact.php?message= - Vulnerable to reflected XSS
```

An XSS vulnerability is found on the contact page, a key risk that could allow attackers to execute arbitrary scripts.

Step 7: Checking for Local File Inclusion (LFI)

The http-lfi script is run next to test for Local File Inclusion vulnerabilities, which could expose sensitive server files.

```
nmap -p 80 --script http-lfi victim-app.com
```

Output:

```
PORT STATE SERVICE
80/tcp open http
| http-lfi:
| /includes.php?file=../../../../../../etc/passwd - Confirmed LFI vulnerability
```

The scan confirms an LFI vulnerability, with access to /etc/passwd, which could lead to further information exposure.

Step 8: Testing for Open Redirects

To finish, the http-open-redirect script is used to test for open redirect vulnerabilities that could redirect users to external, malicious sites.

```
nmap -p 80 --script http-open-redirect victim-app.com
```

Output:

```
PORT STATE SERVICE
80/tcp open http
| http-open-redirect:
```

```
| /redirect?url=http://evil-site.com - Vulnerable to open redirect
```

The application is found to be vulnerable to open redirects, a risk that could allow attackers to lure users to phishing sites or other malicious destinations.

Nmap’s comprehensive scripting capabilities make it an invaluable tool for penetration testers. These scripts, when used skillfully, enable testers to assess a target thoroughly, gathering detailed information that serves as the foundation for further exploitation. For web application security, Nmap remains an essential ally in identifying and understanding potential vulnerabilities.