



Curso de Hacking ético Master. D

**Ejercicio 5 - ANÁLISIS DE
PUERTOS Y
VULNERABILIDADES**

Alumno: Julián Gordon

Índice

Introducción	3
Política de análisis de vulnerabilidades con Nessus.....	4
Análisis de vulnerabilidades encontradas con Nessus	10
Como funciona Nikto	12
Análisis de vulnerabilidades encontradas con Nikto	15
Como funciona Owasp-ZAP	18
Análisis de vulnerabilidades encontradas con Owasp-ZAP	22
Generar informe con Owasp-ZAP	24
Conclusiones	27


Introducción

En este ejercicio, sobre análisis de puertos y vulnerabilidades, trabajaremos contra la máquina de Metasploitable2 que tenemos en nuestro laboratorio. Para ello, crearemos una política de análisis de vulnerabilidades para aplicaciones web en Nessus y la ejecutaremos contra la aplicación web que está corriendo en el puerto 80 de nuestro laboratorio. Para llevar a cabo esta evaluación, emplearemos las herramientas de análisis de vulnerabilidades, Nessus, Nikto y OWASP-ZAP. Explicaremos las principales vulnerabilidades que encontremos y este análisis nos ayudará a comprender mejor las amenazas a las que se enfrentan las aplicaciones web. Esto nos permitirá no solo identificar los problemas, sino también proponer soluciones y recomendaciones para mitigar los riesgos y fortalecer la seguridad de la aplicación web.

Política de Análisis de vulnerabilidades para aplicaciones web con Nessus

Para explicar el análisis que vamos a realizar, debemos empezar por entender que tipo de protocolo es HTTP y que puerto utiliza. HTTP significa Protocolo de Transferencia de Hipertexto, es un protocolo de comunicación utilizado para la transferencia de datos en la World Wide Web (www). Permite la transferencia de información en forma de hipertexto, que incluye texto, imágenes, etc, entre un servidor web y un cliente web, como un navegador. HTTP utiliza el puerto 80 como puerto predeterminado para la comunicación entre el cliente y el servidor web.

Como hicimos en el trabajo anterior, ejecutaremos un análisis de vulnerabilidades con Nessus, sobre la máquina de Metasploitable2 de nuestro laboratorio. En este caso, la política que crearemos, será solamente para analizar la aplicación web que está corriendo en el puerto 80, por lo que el análisis será más corto y más específico. A continuación veremos imágenes de la configuración de la política y luego una imagen con algunas vulnerabilidades que encontró Nessus, sobre el puerto 80.

BASIC 

• General

Schedule

Notifications

DISCOVERY >

ASSESSMENT >

REPORT >

ADVANCED >


Name

Aplicaciones Web Metasploitable

Description

Un escaneo sobre la aplicación web que esta corriendo en el puerto 80 de Metasploitable2

Folder

My Scans 

Targets

10.0.2.4

Service Discovery

ASSESSMENT >

REPORT >

ADVANCED >

General Settings

☒ Test the local Nessus host

This setting specifies whether the local Nessus host should be scanned when it falls within the target range specified for the scan.

☐ Use fast network discovery

If a host responds to ping, Nessus attempts to avoid false positives, performing additional tests to verify the response did not come from a proxy or load balancer. Fast network discovery bypasses those additional tests.

Ping Methods

☒ ARP

☒ TCP

Destination ports

80

☒ ICMP

☐ Assume ICMP unreachable from the gateway means the host is down

- [Port Scanning](#)

Service Discovery

ASSESSMENT >

REPORT >

ADVANCED >

Port scan range:

80

Network Port Scanners



TCP



Override automatic firewall detection



Use soft detection



Use aggressive detection



Disable detection



SYN



Override automatic firewall detection



Use soft detection



Use aggressive detection



Disable detection

Settings

Credentials

Plugins 

BASIC >

DISCOVERY >

ASSESSMENT 

REPORT >

ADVANCED >

Scan Type

Scan for all web vulnerabilities (complex) ▼

General Settings:

Avoid potential false alarms

Enable CGI scanning

Perform thorough tests

Web Applications:

Start crawling from "/"

Crawl 1000 pages (max)

Traverse 6 directories (max)

Test for known vulnerabilities in commonly used web applications

Perform each generic web app test for 10 minutes (max)

Try all HTTP methods

Attempt HTTP Parameter Pollution

DISCOVERY



ASSESSMENT



REPORT



ADVANCED



Processing

- ☒ Override normal verbosity
 - ☐ I have limited disk space. Report as little information as possible
 - ☒ Report as much information as possible
- ☒ Show missing patches that have been superseded
- ☒ Hide results from plugins initiated as a dependency

Output

- ☒ Allow users to edit scan results
- ☒ Designate hosts by their DNS name
- ☒ Display hosts that respond to ping
- ☒ Display unreachable hosts
- ☒ Display Unicode characters

Aplicaciones Web Metasploitable

[Back to My Scans](#)

Conf

Hosts 1

Vulnerabilities 30

History 1

Filter

Search Vulnerabilities

30 Vulnerabilities

Sev	CVSS	VPR	Name	Family	Count
MIXED	4 Phpmyadmin (Multiple Issues)	CGI abuses	4
MIXED	3 PHP (Multiple Issues)	CGI abuses	3
MIXED	2 Twiki (Multiple Issues)	CGI abuses	2
MEDIUM	5.3		Browsable Web Directories	CGI abuses	1
MEDIUM	5.3		HTTP TRACE / TRACK Methods Allowed	Web Servers	1
MEDIUM	5.0 *		Backup Files Disclosure	CGI abuses	1
MEDIUM	4.3 *		CGI Generic Cookie Injection Scripting	CGI abuses	1

Scan Details

Policy: Web Application Tests

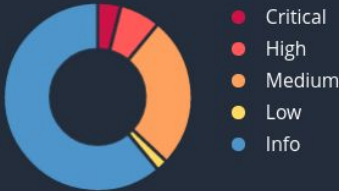
Status: Running

Severity Base: CVSS v3.0

Scanner: Local Scanner

Start: Today at 11:46 AM

Vulnerabilities



Análisis de vulnerabilidades encontradas con Nessus

phpMyAdmin SQL Injection (PMASA-2019-3)

La vulnerabilidad PMASA-2019-3 se refiere a una vulnerabilidad de inyección SQL (SQLi) que afecta a versiones anteriores a 4.8.6 de la aplicación phpMyAdmin. Esta aplicación es una interfaz web muy utilizada para gestionar bases de datos MySQL. La vulnerabilidad se encuentra en la función "designer" de phpMyAdmin. Podríamos aprovechar esta vulnerabilidad para inyectar o manipular consultas SQL en la base de datos en el servidor.

La solución que nos propone Nessus es actualizar phpMyAdmin a la versión 4.8.6 o una versión posterior, ya que los desarrolladores de phpMyAdmin han corregido esta vulnerabilidad en esas versiones.

Análisis de vulnerabilidades encontradas con Nessus

Apache PHP-CGI Remote Code Execution

Esta vulnerabilidad, conocida como "Apache PHP-CGI Remote Code Execution", afecta a servidores web que utilizan el intérprete PHP en modo CGI (Common Gateway Interface) y se refiere a una vulnerabilidad, que podría permitirnos ejecutar código en el servidor. Esto nos permite enviar argumentos de línea de comandos, como parte de una cadena de consulta (query string) en una solicitud web al programa PHP-CGI. También podríamos obtener acceso a código fuente de archivos PHP, lo que podría exponer información confidencial y lograr la caída del sistema interrumpiendo los servicios del servidor. La solución recomendada para esta vulnerabilidad es actualizar PHP a la versión 5.3.13 o 5.4.3, o a una versión posterior.

Como funciona - Nikto

Antes de entrar en detalle sobre las vulnerabilidades encontradas con Nikto, explicaremos un poco como funciona esta herramienta. Nikto es una herramienta de escaneo de vulnerabilidades diseñada específicamente para identificar posibles problemas de seguridad en servidores web. Funciona realizando una serie de solicitudes HTTP a un servidor web de destino y analizando las respuestas, en busca de indicadores de vulnerabilidades conocidas. Busca patrones que coincidan con estas vulnerabilidades, configuraciones inseguras y problemas comunes de seguridad. También compara las respuestas con una base de datos de firmas que se actualiza regularmente para identificar nuevas amenazas.

Lo primero que debemos hacer es indicarle una URL o IP en la que se encuentra la aplicación web que queremos analizar, en este caso será 10.0.2.4 que es la máquina de Metasploitable2 que tenemos en nuestro laboratorio. Luego Nikto genera un informe detallado que enumera las vulnerabilidades encontradas, junto con información adicional sobre el servidor web, como la versión, los encabezados HTTP y otros datos útiles. En el informe nos brinda enlaces en los que clasifica las vulnerabilidades detectadas en función de su gravedad y también proporciona recomendaciones y consejos sobre cómo abordar y solucionar los problemas identificados.

Ahora explicaremos el comando que utilizamos para realizar el análisis.

Hemos utilizado “ **nikto -Cgidirs all -mutate 12345 -Tuning x7 -host 10.0.2.4** ”.

-Cgidirs all: Esta opción indica que escanee todos los directorios comunes de CGI en busca de posibles scripts CGI, amplía la búsqueda de posibles vulnerabilidades en scripts CGI en el servidor web.

-mutate 12345: Realizará pruebas de mutación en los nombres de archivos y directorios en busca de posibles vulnerabilidades. La mutación implica generar combinaciones de nombres de archivos y directorios basados en números y letras, lo que ayuda a identificar posibles rutas de acceso ocultas o vulnerabilidades.

-Tuning x7: El parámetro "Tuning" se utiliza para ajustar el escaneo de Nikto para buscar vulnerabilidades específicas o realizar escaneos especializados. En este caso, "x7" es un ajuste predefinido que realiza un escaneo amplio y completo en busca de vulnerabilidades comunes, La x hace que se utilizan todos los parámetros que hay menos el número que le indiquemos, en este caso 7.

-host 10.0.2.4: Este parámetro especifica la dirección IP del host de destino que analizamos. .

A continuación se pueden ver imágenes con el informe que nos generó Nikto, luego lo analizaremos en detalle.

```
2 - Nikto v2.5.0
3
4 + Target IP:      10.0.2.4
5 + Target Hostname: 10.0.2.4
6 + Target Port:    80
7 + Using Mutation: Test all files with all root directories
8 + Using Mutation: Guess for password file names
9 + Using Mutation: Enumerate user names via Apache (/~user type requests)
10 + Using Mutation: Enumerate user names via cgiwrap (/cgi-bin/cgiwrap/~user type requests)
11 + Using Mutation: Attempt to brute force sub-domain names, assume that the host name is the parent domain
12 + Start Time:    2023-10-30 11:37:37 (GMT1)
13
14 + Server: Apache/2.2.8 (Ubuntu) DAV/2
15 + /: Retrieved x-powered-by header: PHP/5.2.4-2ubuntu5.10.
16 + /: The anti-clickjacking X-Frame-Options header is not present. See: https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/X-Frame-Options
17 + /: The X-Content-Type-Options header is not set. This could allow the user agent to render the content of the site in a different fashion to the MIME type. See:
https://www.netsparker.com/web-vulnerability-scanner/vulnerabilities/missing-content-type-header/
18 + /index: Uncommon header 'tcn' found, with contents: list.
19 + /index: Apache mod_negotiation is enabled with MultiViews, which allows attackers to easily brute force file names. The following alternatives for 'index' were
found: index.php. See: http://www.wisec.it/sectou.php?id=4698ebdc59d15, https://exchange.xforce.ibmcloud.com/vulnerabilities/8275
20 + Apache/2.2.8 appears to be outdated (current is at least Apache/2.4.54). Apache 2.2.34 is the EOL for the 2.x branch.
21 + /: Web Server returns a valid response with junk HTTP methods which may cause false positives.
22 + /: HTTP TRACE method is active which suggests the host is vulnerable to XST. See: https://owasp.org/www-community/attacks/Cross\_Site\_Tracing
23 + /phpinfo.php: Output from the phpinfo() function was found.
24 + /doc/: Directory indexing found.
25 + /doc/: The /doc/ directory is browsable. This may be /usr/doc. See: http://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-1999-0678
26 + /?PBPB8B5F2A0-3C92-11d3-A3A9-4C7B08C10000: PHP reveals potentially sensitive information via certain HTTP requests that contain specific QUERY strings. See:
OSVDB-12184
27 + /?PHPE9568F36-D428-11d2-A769-00AA001ACF42: PHP reveals potentially sensitive information via certain HTTP requests that contain specific QUERY strings. See:
OSVDB-12184
28 + /?PHPE9568F34-D428-11d2-A769-00AA001ACF42: PHP reveals potentially sensitive information via certain HTTP requests that contain specific QUERY strings. See:
OSVDB-12184
29 + /?PHPE9568F35-D428-11d2-A769-00AA001ACF42: PHP reveals potentially sensitive information via certain HTTP requests that contain specific QUERY strings. See:
OSVDB-12184
30 + /phpMyAdmin/changelog.php: phpMyAdmin is for managing MySQL databases, and should be protected or limited to authorized hosts.
31 + /phpMyAdmin/ChangeLog: Server may leak inodes via ETags, header found with file /phpMyAdmin/ChangeLog, inode: 92462, size: 40540, mtime: Tue Dec 9 18:24:00
2008. See: http://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2003-1418
32 + /phpMyAdmin/ChangeLog: phpMyAdmin is for managing MySQL databases, and should be protected or limited to authorized hosts.
33 + /test/: Directory indexing found.
34 + /test/: This might be interesting.
```

Análisis de vulnerabilidades encontradas con Nikto

The X-Content-Type-Options header is not set.

La vulnerabilidad que se informa en Nikto con respecto a la falta del encabezado "X-Content-Type-Options", se relaciona con la seguridad en el procesamiento de tipos MIME (Multipurpose Internet Mail Extensions) en un sitio web y tiene una severidad baja. Se señala que el sitio web, no ha configurado el encabezado "X-Content-Type-Options". Esto significa que el servidor web no proporciona instrucciones claras al navegador sobre cómo manejar el contenido. El encabezado "X-Content-Type-Options" es una medida de seguridad, que se puede configurar en un servidor web. Su propósito es asegurarse de que el navegador del cliente, interprete correctamente el tipo de contenido que se le está enviando. Cuando este encabezado se establece adecuadamente, evita que el navegador intente adivinar o interpretar incorrectamente el tipo de contenido según su contenido real. Podríamos intentar servir contenido malicioso que parezca ser de un tipo MIME diferente al real. El "MIME type sniffing" es una funcionalidad estándar en los navegadores, que permite encontrar la forma adecuada de representar datos cuando los encabezados HTTP enviados por el servidor, no son concluyentes o están ausentes. El problema surge cuando un sitio web permite, a los usuarios, cargar contenido que luego se publica en el servidor web. Por ejemplo hacer que un archivo JavaScript se interprete como una imagen, lo que podría ser utilizado para realizar ataques de tipo Cross-Site Scripting (XSS) u otros ataques. Nikto nos brinda un enlace en el cual nos explica este tipo de vulnerabilidad y la manera de corregirla. La solución podría ser agregar el encabezado "X-Content-Type-Options" con un valor de "nosniff" , para indicar al navegador que confíe en lo que el sitio ha enviado como el tipo de contenido adecuado y que no intente "hacer sniffing" del tipo de contenido real. X-Content-Type-Options: nosniff

<https://www.netsparker.com/web-vulnerability-scanner/vulnerabilities/missing-content-type-header/>

Análisis de vulnerabilidades encontradas con Nikto

Apache/2.2.8 appears to be outdated

Para entender de qué se trata esta vulnerabilidad, debemos entender que es Apache. El servidor web Apache es un software, que se utiliza para alojar sitios web y aplicaciones en servidores. Como en cualquier software, las versiones anteriores de Apache pueden contener vulnerabilidades de seguridad conocidas que se han corregido en versiones más recientes. La vulnerabilidad encontrada con Nikto, indica que la versión de Apache (2.2.8), que se está ejecutando en el servidor, es obsoleta y desactualizada. Además, señala que la versión 2.2.34 de Apache 2.2.x ya ha llegado al final de su vida útil (EOL). Eso significa que no se publican más actualizaciones, ni parches de seguridad, para esa versión. Esta es una vulnerabilidad crítica ya que las versiones obsoletas de software suelen ser propensas a múltiples vulnerabilidades conocidas que podemos explotar, para comprometer el servidor. La solución, en nuestro caso sería actualizar la versión de Apache. En el caso de que fuera en un servidor de una empresa, se recomendaría hacer una copia de seguridad y configuraciones antes de actualizar el software.

Análisis de vulnerabilidades encontradas con Nikto

HTTP TRACE method is active which suggests the host is vulnerable to XST

Para entender esta vulnerabilidad, debemos saber que el método HTTP TRACE se utiliza para diagnosticar y depurar problemas en las solicitudes y respuestas HTTP. Esta vulnerabilidad informa que el método HTTP TRACE está habilitado en el servidor. El método TRACE, puede ser aprovechado para llevar a cabo ataques de Cross-Site Tracing (XST), lo que podría revelar información sensible. Esta es otra vulnerabilidad crítica, ya que la habilitación del método TRACE puede exponer el sistema a riesgos de seguridad, lo que podría permitir ataques de tipo XST. Para solucionar esta vulnerabilidad, se recomienda desactivar el método HTTP TRACE en el servidor web. Este es el enlace con la información sobre la vulnerabilidad que nos brinda Nikto: https://owasp.org/www-community/attacks/Cross_Site_Tracing

Como funciona - Owasp-ZAP

Al igual que hicimos anteriormente con Nikto, explicaremos primero como funciona Owasp-ZAP y luego analizaremos las vulnerabilidades encontradas. OWASP ZAP (Zed Attack Proxy) es una herramienta de seguridad diseñada para ayudar en la identificación y mitigación de vulnerabilidades en aplicaciones web. Se utiliza comúnmente en la auditoría de seguridad y pruebas de penetración para evaluar la seguridad de aplicaciones web. OWASP ZAP actúa como un proxy, intermedio entre el navegador y la aplicación web que se está probando. Esto nos permite, interceptar y modificar las solicitudes y respuestas HTTP entre el navegador y el servidor, lo que es esencial para realizar pruebas de seguridad. Nos ofrece capacidades de escaneo automatizado que ayudan a identificar vulnerabilidades comunes, como inyección de SQL, XSS (Cross-Site Scripting), CSRF (Cross-Site Request Forgery), entre otras. Esta herramienta puede realizar rastreos de sitios web de manera automática, siguiendo enlaces y descubriendo páginas ocultas. Esto nos sirve para garantizar que todas las partes de la aplicación sean probadas. Owasp-Zap, al igual que otras herramientas que vimos anteriormente, genera informes detallados sobre las vulnerabilidades identificadas, su gravedad y sugerencias de mitigación.

Owasp-ZAP es una herramienta que utiliza interfaz gráfica, por lo que es bastante fácil de usar e intuitiva. A continuación se muestran algunas imágenes de cómo configurar un análisis.

En la pestaña Alerts, es donde van apareciendo las vulnerabilidades que encuentra. Podemos ver la gravedad, ID de CWE, descripción, solución y debajo enlace para obtener más información sobre esta vulnerabilidad.

The screenshot shows the ZAP 2.14.0 interface. The 'Alerts' tab is active, displaying a list of 23 alerts on the left and details for the selected 'Hash Disclosure - MD5 Crypt' alert on the right.

Alerts (23)

- > Hash Disclosure - MD5 Crypt
- > Path Traversal (18)
- > Absence of Anti-CSRF Tokens (3090)
- > Application Error Disclosure (181)
- > Content Security Policy (CSP) Header Not Set (2539)
- > Directory Browsing (56)
- > Missing Anti-clickjacking Header (2349)
- > Vulnerable JS Library
- > Cookie No HttpOnly Flag (18)
- > Cookie without SameSite Attribute (28)
- > Information Disclosure - Debug Error Messages (171)
- > Private IP Disclosure (24)
- > Server Leaks Information via "X-Powered-By" HTTP Response Header Field(s) (139)
- > Server Leaks Version Information via "Server" HTTP Response Header Field (2791)
- > Timestamp Disclosure - Unix (589)
- > X-Content-Type-Options Header Missing (2414)
- > Authentication Request Identified (8)
- > Information Disclosure - Sensitive Information in URL (9)
- > Information Disclosure - Suspicious Comments (90)
- > Modern Web Application (2260)
- > Session Management Response Identified (4)
- > User Controllable Charset (2)
- > User Controllable HTML Element Attribute (Potential XSS) (930)

Hash Disclosure - MD5 Crypt

URL: <http://10.0.2.4/mutillidae/index.php?page=source-viewer.php>
Risk: High
Confidence: High
Parameter:
Attack:
Evidence: \$1\$65505215\$Klj3LgdTa3z6bB0NYYfO/
CWE ID: 200
WASC ID: 13
Source: Passive (10097 - Hash Disclosure)
Input Vector:
Description:
A hash was disclosed by the web server. - MD5 Crypt

Other Info:

Solution:
Ensure that hashes that are used to protect credentials or other resources are not leaked by the web server or database. There is typically no requirement for password hashes to be accessible to the web browser.

Reference:
<http://projects.webappsec.org/w/page/13246936/Information%20Leakage>
<http://openwall.info/wiki/john/sample-hashes>

Alert Tags:

Key	Value
OWASP_2021_A04	https://owasp.org/Top10/A04_2021-Insecure_Design/
OWASP_2017_A03	https://owasp.org/www-project-top-ten/2017/A3_2017-Sensitive_Data_Expos...

Alerts: 2 6 8 7 Main Proxy: localhost:8080

Current Scans: 0 0 3 1 0 0 0 0 0 0

En la pestaña Active Scan, podemos ver el escaneo activo que estamos realizando en este momento, el método utilizado, la URL(que deriva de la que le pasamos antes de iniciar),etc

FileEditViewAnalyseReportToolsImportExportOnlineHelp

Standard Mode

Quick StartRequestResponseSitesHistorySearchAlertsRequesterSpiderOutputActive Scan

New ScanProgress: 0: http://10.0.2.42%Current Scans: 1Num Requests: 29280New Alerts: 18Export

Sent MessagesFiltered Messages

ID	Req. Timestamp	Resp. Timestamp	Method	URL	Code	Reason	RTT	Size Resp. Header	Size Resp. Body
35,628	11/3/23, 8:05:52 AM	11/3/23, 8:05:54 AM	POST	http://10.0.2.4/twiki/bin/preview/Sandbox/TestTopic6	200 OK		1.16 s	160 bytes	3,615 bytes
35,629	11/3/23, 8:05:53 AM	11/3/23, 8:05:54 AM	POST	http://10.0.2.4/twiki/bin/preview/Sandbox/TestTopic5	200 OK		665 ms	160 bytes	3,441 bytes
35,630	11/3/23, 8:05:53 AM	11/3/23, 8:05:54 AM	POST	http://10.0.2.4/twiki/bin/preview/Sandbox/TestTopic8	200 OK		1.02 s	160 bytes	3,417 bytes
35,631	11/3/23, 8:05:54 AM	11/3/23, 8:05:54 AM	POST	http://10.0.2.4/twiki/bin/preview/Sandbox/TestTopic6	200 OK		430 ms	160 bytes	3,441 bytes
35,632	11/3/23, 8:05:53 AM	11/3/23, 8:05:54 AM	POST	http://10.0.2.4/twiki/bin/preview/Sandbox/TestTopic7	200 OK		1.07 s	160 bytes	3,615 bytes
35,633	11/3/23, 8:05:53 AM	11/3/23, 8:05:54 AM	POST	http://10.0.2.4/twiki/bin/preview/Sandbox/TestTopic4	302 Moved		1.1 s	255 bytes	0 bytes
35,634	11/3/23, 8:05:54 AM	11/3/23, 8:05:54 AM	POST	http://10.0.2.4/twiki/bin/preview/Sandbox/TestTopic8	200 OK		416 ms	160 bytes	3,289 bytes
35,635	11/3/23, 8:05:54 AM	11/3/23, 8:05:54 AM	POST	http://10.0.2.4/twiki/bin/preview/Sandbox/TestTopic5	200 OK		768 ms	160 bytes	3,444 bytes
35,636	11/3/23, 8:05:53 AM	11/3/23, 8:05:55 AM	POST	http://10.0.2.4/twiki/bin/preview/Sandbox/WebHome	200 OK		1.17 s	160 bytes	3,383 bytes
35,637	11/3/23, 8:05:54 AM	11/3/23, 8:05:55 AM	POST	http://10.0.2.4/twiki/bin/preview/Sandbox/TestTopic7	200 OK		710 ms	160 bytes	3,441 bytes
35,638	11/3/23, 8:05:55 AM	11/3/23, 8:05:55 AM	POST	http://10.0.2.4/twiki/bin/preview/Sandbox/TestTopic7	200 OK		266 ms	160 bytes	3,444 bytes
35,639	11/3/23, 8:05:54 AM	11/3/23, 8:05:55 AM	POST	http://10.0.2.4/twiki/bin/preview/Sandbox/TestTopic6	200 OK		1.11 s	160 bytes	3,444 bytes
35,640	11/3/23, 8:05:54 AM	11/3/23, 8:05:55 AM	POST	http://10.0.2.4/twiki/bin/preview/Sandbox/TestTopic4	302 Moved		1.12 s	255 bytes	0 bytes
35,641	11/3/23, 8:05:55 AM	11/3/23, 8:05:55 AM	POST	http://10.0.2.4/twiki/bin/preview/Sandbox/WebHome	200 OK		799 ms	160 bytes	3,389 bytes
35,642	11/3/23, 8:05:54 AM	11/3/23, 8:05:55 AM	POST	http://10.0.2.4/twiki/bin/preview/Sandbox/TestTopic5	200 OK		965 ms	160 bytes	3,440 bytes
35,643	11/3/23, 8:05:55 AM	11/3/23, 8:05:56 AM	POST	http://10.0.2.4/twiki/bin/preview/Sandbox/TestTopic5	200 OK		328 ms	160 bytes	3,453 bytes
35,644	11/3/23, 8:05:55 AM	11/3/23, 8:05:56 AM	POST	http://10.0.2.4/twiki/bin/preview/Sandbox/TestTopic7	200 OK		812 ms	160 bytes	3,440 bytes
35,645	11/3/23, 8:05:54 AM	11/3/23, 8:05:56 AM	POST	http://10.0.2.4/twiki/bin/preview/Sandbox/TestTopic8	200 OK		1.7 s	160 bytes	3,295 bytes
35,646	11/3/23, 8:05:55 AM	11/3/23, 8:05:56 AM	POST	http://10.0.2.4/twiki/bin/preview/Sandbox/TestTopic4	302 Moved		780 ms	255 bytes	0 bytes
35,647	11/3/23, 8:05:56 AM	11/3/23, 8:05:56 AM	POST	http://10.0.2.4/twiki/bin/preview/Sandbox/TestTopic7	200 OK		487 ms	160 bytes	3,453 bytes
35,648	11/3/23, 8:05:55 AM	11/3/23, 8:05:56 AM	POST	http://10.0.2.4/twiki/bin/preview/Sandbox/TestTopic6	200 OK		1.17 s	160 bytes	3,440 bytes
35,649	11/3/23, 8:05:55 AM	11/3/23, 8:05:56 AM	POST	http://10.0.2.4/twiki/bin/preview/Sandbox/WebHome	200 OK		945 ms	160 bytes	3,264 bytes
35,650	11/3/23, 8:05:56 AM	11/3/23, 8:05:57 AM	POST	http://10.0.2.4/twiki/bin/preview/Sandbox/TestTopic8	200 OK		641 ms	160 bytes	3,287 bytes
35,651	11/3/23, 8:05:56 AM	11/3/23, 8:05:57 AM	POST	http://10.0.2.4/twiki/bin/preview/Sandbox/TestTopic5	200 OK		987 ms	160 bytes	3,456 bytes
35,652	11/3/23, 8:05:56 AM	11/3/23, 8:05:57 AM	POST	http://10.0.2.4/twiki/bin/preview/Sandbox/WebHome	200 OK		544 ms	160 bytes	3,270 bytes
35,653	11/3/23, 8:05:52 AM	11/3/23, 8:05:57 AM	POST	http://10.0.2.4/twiki/bin/preview/Main/WebChanges	200 OK		4.74 s	161 bytes	29,416 bytes
35,654	11/3/23, 8:05:52 AM	11/3/23, 8:05:57 AM	POST	http://10.0.2.4/twiki/bin/preview/Main/WebIndex	200 OK		5.37 s	161 bytes	28,853 bytes
35,655	11/3/23, 8:05:57 AM	11/3/23, 8:05:57 AM	POST	http://10.0.2.4/twiki/bin/preview/Sandbox/TestTopic8	200 OK		899 ms	160 bytes	3,313 bytes
35,656	11/3/23, 8:05:56 AM	11/3/23, 8:05:58 AM	POST	http://10.0.2.4/twiki/bin/preview/Sandbox/TestTopic6	200 OK		1.38 s	160 bytes	3,453 bytes
35,657	11/3/23, 8:05:57 AM	11/3/23, 8:05:58 AM	POST	http://10.0.2.4/twiki/bin/preview/Sandbox/TestTopic5	200 OK		986 ms	160 bytes	3,434 bytes
35,658	11/3/23, 8:05:56 AM	11/3/23, 8:05:58 AM	POST	http://10.0.2.4/twiki/bin/preview/Sandbox/TestTopic4	302 Moved		1.85 s	255 bytes	0 bytes

Alerts2687Main Proxy: localhost:8080Current Scans000100000000

En OWASP ZAP, el término "spider" se refiere a una de las funcionalidades clave de la herramienta: el "Spidering" o "Spider." El Spider de OWASP ZAP es una función que automatiza el proceso de rastreo y descubrimiento de las páginas y funcionalidades de una aplicación web. Esto es fundamental para realizar pruebas exhaustivas de seguridad en una aplicación web. El Spider inicia en la página web específica que le dimos al principio que fue 10.0.2.4 (nuestra máquina de Metasploitable2) y, a partir de ahí, sigue los enlaces y rutas posibles dentro del sitio web. Puede descubrir páginas ocultas, recursos, formularios, parámetros, y otras funcionalidades. A medida que navega por el sitio, registra todos los datos sobre las solicitudes y respuestas HTTP. Esto incluye parámetros, cookies, encabezados y otros detalles importantes. Se puede observar en la siguiente imagen.

File Edit View Analyse Report Tools Import Export Online Help

Standard Mode

Quick StartRequestResponseSitesHistorySearchAlertsRequesterSpiderOutputActive Scan

New ScanProgress: 0: http://10.0.2.4100%Current Scans: 0URLs Found: 3279Nodes Added: 1909Export

URLsAdded NodesMessages

Processed	Method	URI	Flags
	GET	http://10.0.2.4	Seed
	GET	http://10.0.2.4/robots.txt	Seed
	GET	http://10.0.2.4/sitemap.xml	Seed
	GET	http://10.0.2.4/twiki/	
	GET	http://10.0.2.4/phpMyAdmin/	
	GET	http://10.0.2.4/mutillidae/	
	GET	http://10.0.2.4/dvwa/	
	GET	http://10.0.2.4/dav/	
	GET	http://10.0.2.4/twiki/readme.txt	
	GET	http://10.0.2.4/twiki/license.txt	
	GET	http://10.0.2.4/twiki/TWikiDocumentation.html	
	GET	http://10.0.2.4/twiki/TWikiHistory.html	
	GET	http://10.0.2.4/twiki/bin/view/Main/WebHome	
	GET	http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd	Out of Scope
	GET	http://twiki.org/	Out of Scope
	GET	http://twiki.org/cgi-bin/view/TWiki/TWikiDocumentation	Out of Scope
	GET	http://twiki.org/cgi-bin/view/Main/TWikiInstallations	Out of Scope
	GET	http://twiki.org/cgi-bin/view/Main/PoweredByTWikiLogo	Out of Scope
	GET	http://twiki.org/cgi-bin/view/Support/	Out of Scope
	GET	http://twiki.org/cgi-bin/view/Codev/	Out of Scope
	GET	http://www.gnu.org/copyleft/gpl.html	Out of Scope
	GET	http://10.0.2.4/dav/?C=N;O=D	
	GET	http://10.0.2.4/dav/?C=M;O=A	
	GET	http://10.0.2.4/dav/?C=S;O=A	
	GET	http://10.0.2.4/dav/?C=D;O=A	
	GET	http://10.0.2.4/	
	GET	http://twiki.org/cgi-bin/view/TWiki/PeterThoeny	Out of Scope
	GET	http://10.0.2.4/dav/LEFW2PZb.htm/	
	GET	http://twiki.org/cgi-bin/view/TWiki/InterwikiPlugin	Out of Scope
	GET	http://twiki.org/cgi-bin/view/Codev.ReadmeFirst	Out of Scope
	GET	http://twiki.org/cgi-bin/view/TWiki/TWikiFuncModule	Out of Scope

Alerts 2 6 8 7 Main Proxy: localhost:8080Current Scans 0 0 0 3 1 0 0 0 0 0

Análisis de vulnerabilidades encontradas Owasp-ZAP

Hash Disclosure - MD5 Crypt

Para entender esta vulnerabilidad, primero debemos saber que, MD5 Crypt es un algoritmo de hash que se utiliza comúnmente para almacenar contraseñas de forma segura. En lugar de almacenar contraseñas en texto claro, los sistemas suelen almacenar un hash (una cadena de caracteres generada a partir de la contraseña) en su lugar. El MD5 Crypt es un método de hash que es más seguro que simplemente almacenar contraseñas en texto claro, pero aún así puede ser vulnerable si se expone.

La vulnerabilidad "Hash Disclosure - MD5 Crypt" se refiere a la revelación de un hash MD5 Crypt por parte del servidor web. Cuando un servidor web muestra o expone un hash MD5 Crypt en una respuesta, esto puede ser un riesgo de seguridad, porque los hashes de contraseñas deberían ser confidenciales y no estar al alcance de los usuarios o atacantes. Si obtenemos acceso a un hash MD5 Crypt, podríamos utilizar técnicas de fuerza bruta para intentar descifrar la contraseña original. Si la contraseña es débil, esto podría tener graves consecuencias de seguridad. Para mitigar esta vulnerabilidad, se debe asegurar que los hashes utilizados para proteger contraseñas u otros recursos no se filtren ni se muestran en las respuestas del servidor web. Esto se logra mediante una configuración adecuada del servidor y la aplicación web para evitar que los hashes se expongan a los usuarios no autorizados o limitando el acceso a las contraseñas y hashes solo a las partes del sistema que realmente los necesitan, como los componentes de autenticación y autorización. Las contraseñas y hashes deben almacenarse en la base de datos, utilizando técnicas seguras de almacenamiento de contraseñas, como el almacenamiento con sal (salted hashing). Esto garantiza que incluso si la base de datos se ve comprometida, los atacantes no pueden recuperar las contraseñas originales. Es una vulnerabilidad de gravedad alta, CVE 200 y Owasp-ZAP nos da estos 2 enlaces para obtener más información sobre esta vulnerabilidad:

<http://projects.webappsec.org/w/page/13246936/Information%20Leakage>

<http://openwall.info/wiki/john/sample-hashes>

Análisis de vulnerabilidades encontradas Owasp-ZAP

The Path Traversal attack

Esta vulnerabilidad se conoce como "Path Traversal" (Travesía de Rutas) y se trata de una técnica de ataque que nos permite acceder a archivos, directorios y comandos que potencialmente se encuentran fuera del directorio raíz de documentos web del servidor. Esto significa que podemos manipular una URL de tal manera que el sitio web ejecute o revele el contenido de archivos arbitrarios en cualquier lugar del servidor web. Cualquier dispositivo que exponga una interfaz basada en HTTP es potencialmente vulnerable a este tipo de ataque. La mayoría de los sitios web restringen el acceso de los usuarios a una parte específica del sistema de archivos, generalmente llamada "directorio raíz de documentos web" o "directorio raíz de CGI". Estos directorios contienen los archivos destinados para el acceso de los usuarios y los ejecutables necesarios para el funcionamiento de las aplicaciones web. Para acceder a archivos o ejecutar comandos en cualquier lugar del sistema de archivos, los ataques de Path Traversal utilizan secuencias de caracteres especiales. El ataque Path Traversal más básico utiliza la secuencia de caracteres "../" para alterar la ubicación de recursos solicitados en la URL. Aunque la mayoría de los servidores web populares evitan que esta técnica escape del directorio raíz de documentos web, existen variaciones que pueden ayudar a eludir los filtros de seguridad. Estas variaciones incluyen la codificación Unicode válida e inválida ("..%u2216" o "..%c0%af") del carácter de barra diagonal, caracteres de barra invertida ("..") en servidores basados en Windows, caracteres codificados en URL ("%2e%2e%2f"), y doble codificación en URL ("..%255c") del carácter de barra invertida.

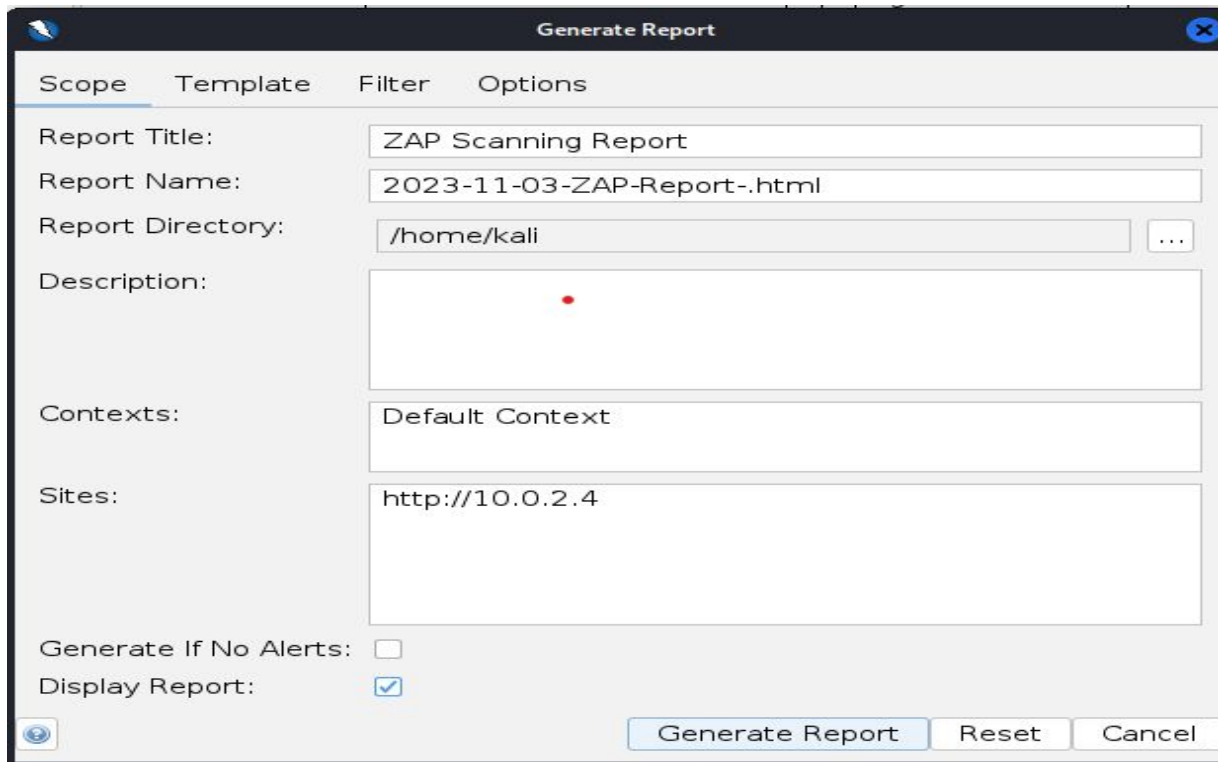
Esta es una vulnerabilidad de gravedad alta, CVE 22 y la solución que no sugiere Owasp-ZAP es adoptar una estrategia de validación de entrada, que se basa en asumir que toda la entrada es potencialmente maliciosa. Esto significa que en lugar de confiar en una lista de elementos que se deben bloquear o rechazar (deny list), se utiliza una lista de elementos aceptables conocidos (allow list) que cumplen estrictamente con las especificaciones del sistema. La estrategia consiste en aceptar solo la entrada que se ajusta a criterios predefinidos y rechazar cualquier entrada que no cumple con estos criterios. No se debe confiar únicamente en listas de denegación (deny lists) para bloquear entradas maliciosas. Las listas de denegación son menos efectivas porque los atacantes pueden encontrar formas de evadir los elementos en la lista. Owasp-ZAP nos brinda estos 2 enlaces para explicar en profundidad esta vulnerabilidad.

<http://projects.webappsec.org/Path-Traversal>

<https://cwe.mitre.org/data/definitions/22.html>

Generar informe con Owasp-ZAP

Además de lo que ya vimos que es posible esta herramienta, también nos da la posibilidad de exportar estos resultados en un informe en formato HTML. En las siguientes imágenes observamos la opción report y parte del informe generado.



Generate Report

Scope Template Filter Options

Report Title: ZAP Scanning Report

Report Name: 2023-11-03-ZAP-Report-.html

Report Directory: /home/kali ...

Description:

Contexts: Default Context

Sites: http://10.0.2.4

Generate If No Alerts: ☐

Display Report: ☒

Generate Report Reset Cancel

ZAP Scanning Report

Generated with  ZAP on Fri 3 Nov 2023, at 08:36:18

ZAP Version: 2.14.0

Contents

- [About this report](#)
 - [Report parameters](#)
- [Summaries](#)
 - [Alert counts by risk and confidence](#)
 - [Alert counts by site and risk](#)
 - [Alert counts by alert type](#)
- [Alerts](#)
 - [Risk=High, Confidence=High \(1\)](#)
 - [Risk=High, Confidence=Medium \(1\)](#)
 - [Risk=Medium, Confidence=High \(1\)](#)
 - [Risk=Medium, Confidence=Medium \(4\)](#)
 - [Risk=Medium, Confidence=Low \(1\)](#)
 - [Risk=Low, Confidence=High \(1\)](#)
 - [Risk=Low, Confidence=Medium \(6\)](#)
 - [Risk=Low, Confidence=Low \(1\)](#)
 - [Risk=Informational, Confidence=High \(1\)](#)
 - [Risk=Informational, Confidence=Medium \(3\)](#)
 - [Risk=Informational, Confidence=Low \(3\)](#)
- [Appendix](#)
 - [Alert types](#)

About this report

Report parameters

Contexts

No contexts were selected, so all contexts were included by default.

Sites

The following sites were included:

- `http://10.0.2.4`

(If no sites were selected, all sites were included by default.)

An included site must also be within one of the included contexts for its data to be included in the report.

Risk levels

Included: High, Medium, Low, Informational

Excluded: None

Confidence levels

Included: User Confirmed, High, Medium, Low

Summaries

Alert counts by risk and confidence

This table shows the number of alerts for each level of risk and confidence included in the report.

(The percentages in brackets represent the count as a percentage of the total number of alerts included in the report, rounded to one decimal place.)

		Confidence				
		User				
		Confirmed	High	Medium	Low	Total
Risk	High	0 (0.0%)	1 (4.3%)	1 (4.3%)	0 (0.0%)	2 (8.7%)
	Medium	0 (0.0%)	1 (4.3%)	4 (17.4%)	1 (4.3%)	6 (26.1%)
	Low	0 (0.0%)	1 (4.3%)	6 (26.1%)	1 (4.3%)	8 (34.8%)
	Informational	0 (0.0%)	1 (4.3%)	3 (13.0%)	3 (13.0%)	7 (30.4%)
	Total	0 (0.0%)	4 (17.4%)	14 (60.9%)	5 (21.7%)	23 (100%)

Conclusiones

En el transcurso de este trabajo, nos centramos en la evaluación de la seguridad de una aplicación web en el laboratorio de Metasploitable2. Nuestra tarea consistió en desarrollar una política de análisis de vulnerabilidades para aplicaciones web en Nessus y ejecutarla en la aplicación web que se aloja en el puerto 80 de Metasploitable2. Además, combinamos las herramientas de Nikto y OWASP-ZAP para realizar un análisis exhaustivo de la aplicación web en busca de posibles debilidades. Al final de este proceso, hemos identificado varias vulnerabilidades importantes. Algunas de las vulnerabilidades más destacadas, incluyen la falta de actualizaciones y parches en el software del servidor web, la exposición de información sensible, como directorios y archivos confidenciales, y problemas relacionados con la autenticación y la gestión de sesiones. También encontramos vulnerabilidades de inyección de código, que nos podrían permitir ejecutar comandos maliciosos en el servidor. La falta de actualizaciones y parches en el software del servidor web es como dejar una puerta abierta para los ciberdelincuentes. Sin las últimas correcciones de seguridad, el servidor es vulnerable a exploits conocidos.

La exposición de información sensible puede compararse con dejar documentos confidenciales en un lugar público. Cualquier persona puede acceder a ellos sin autorización, lo que podría llevar a la divulgación no autorizada de datos sensibles. Los problemas relacionados con la autenticación y la gestión de sesiones, son similares a una puerta sin cerradura. Un atacante podría encontrar una manera de eludir la autenticación y acceder a funciones o datos restringidos. Las vulnerabilidades de inyección de código son como un intruso que se cuela en una conversación y cambia el curso de la misma. Podríamos aprovechar estas vulnerabilidades para ejecutar comandos maliciosos en el servidor y comprometerlo. Este proyecto ha proporcionado una valiosa experiencia en la identificación de vulnerabilidades en aplicaciones web. Es crucial abordar y corregir estas vulnerabilidades para garantizar la integridad y la confidencialidad de los sistemas y los datos. Este trabajo, nos abre las puertas para la siguiente etapa que es la explotación de estas vulnerabilidades, y para ello iremos utilizando distintas técnicas de explotación que iremos aprendiendo con el avance del curso.