593 Digital Publisher
CEIT
ISSN 2588-0705

Estrategias para Identificar y Mitigar Vulnerabilidades de Inyección SQL en Aplicaciones Móviles Android: Revisión Bibliográfica

A Strategies for Identifying and Mitigating SQL Injection Vulnerabilities in Android Mo-bile Applications: A Literature Review

Anthony German Arteaga-Barragán[1]
Pontificia Universidad Católica del Ecuador Sede Ambato - Ecuador
agarteaga@pucesa.edu.ec

José Marcelo Balseca-Manzano[2]
Pontificia Universidad Católica del Ecuador Sede Ambato - Ecuador
jbalseca@pucesa.edu.ec

doi.org/10.33386/593dp.2024.3.2300

V9-N3 (may-jun) 2024, pp 71-83 | Recibido: 27 de diciembre del 2023 - Aceptado: 10 de marzo del 2024 (2 ronda rev.)

1 ORCID: https://orcid.org/0000-0001-5033-1850
2 ORCID: https://orcid.org/my-orcid?orcid=0000-0003-1517-0013

Cómo citar este artículo en norma APA:

Arteaga-Barragán, A., Balseca-Manzano, J., (2024). Estrategias para Identificar y Mitigar Vulnerabilidades de Inyección SQL en Aplicaciones Móviles Android: Revisión Bibliográfica. 593 Digital Publisher CEIT, 9(3), 71-83, https://doi.org/10.33386/593dp.2024.3.2300

Descargar para Mendeley y Zotero

RESUMEN

Las aplicaciones móviles actualmente han tenido que implementar medidas correctivas contra amenazas que comprometen la seguridad de los datos personales de los usuarios. Las vulnerabi-lidades de inyección como la inyección SQL es una amenaza común que afecta a la seguridad de las aplicaciones móviles, y pueden dar pie a que los atacantes exploren, modifiquen o des-truyan los archivos personales importantes para los usuarios. De este modo, resalta la importan-cia del estudio brindado estrategias de mitigación e identificación para los desarrolladores co-mo seguridad para los usuarios comunes de aplicaciones. Por lo tanto, el objetivo de este estu-dio es llevar a cabo un análisis/recopilación/compilación de la literatura relacionada con las estrategias de identificación y mitigación de vulnerabilidades de inyección SQL en aplicaciones móviles Android. La metodología adoptada en este estudio es la Revisión Sistemática de la Literatura, también conocida por sus siglas RSL, que implica un proceso de recopilación y aná-lisis de información de fuentes primarias sobre este tema específico. Dado que, el alcance de esta investigación se centra en esta metodología, los resultados de este estudio destacan los riesgos asociados a esta vulnerabilidad, así como las estrategias de identificación y mitigación en base a los diferentes enfoques recopilados a través del proceso de búsqueda y selección de estudios.

Palabras claves: inyección SQL, aplicaciones móviles android, revisión sistemática de literatura, identificación y mitigación.

ABSTRACT

Mobile applications today have had to implement countermeasures against threats that com-promise the security of users' personal data. Injection vulnerabilities such as SQL injection is a common threat that affects the security of mobile applications, and can allow attackers to scan, modify or destroy personal files important to users. Thus, it highlights the importance of the study by providing mitigation and identification strategies for developers as security for com-mon app users. Therefore, the objective of this study is to conduct an analy-sis/collection/compilation of literature related to SQL injection vulnerability identification and mitigation strategies in Android mobile applications. The methodology adopted in this study is Systematic Literature Review, also known by its acronym SLR, which involves a process of collecting and analyzing information from primary sources on this specific topic. Since, the scope of this research focuses on this methodology, the results of this study highlight the risks associated with this vulnerability, as well as identification and mitigation strategies based on the different approaches gathered through the search and study selection process.

Keywords: SQL injection, android mobile applications, systematic literature review, identification and mitigation.

593 Digital Publisher CEIT ISSN 2588-0705

Introducción

La popularidad de las aplicaciones móviles con el paso del tiempo ha ido en aumento y han traído consigo desafíos de seguridad que ha ido escalando a medida que estas aplicaciones se vuelven más accesibles y se utilizan con mayor frecuencia, los riesgos asociados a las vulnerabilidades de seguridad representan una amenaza importante para la protección de los datos y disponibilidad de los datos que pertenecen a los usuarios.

Aunque existen métodos y mejores prácticas de seguridad disponibles, como la encriptación de datos, filtros de entrada y las pruebas de seguridad, muchas aplicaciones móviles Android siguen siendo vulnerables a ataques comunes. En particular, la inyección SQL es una amenaza de seguridad cada vez más grave. Según OWASP Foundation (2021), ocupando el tercer lugar entre los diez principales riesgos de seguridad. Esta técnica puede usarse para manipular las consultas SQL de la aplicación, lo que permite a los atacantes ver, modificar o eliminar datos en la base de datos. Además, en los últimos tres años, en el repositorio de datos (Google Android: Security vulnerabilities, CVEs sql injection, 2023), se han publicado una serie de vulnerabilidades relacionadas con la inyección SQL, que muestra lo susceptible que encuentra hoy en día las aplicaciones móviles Android, destacando así la relevancia de la seguridad en el sistema operativo Android.

Teniendo en cuenta lo anterior, se conoce como vulnerabilidad a la falta de seguridad en un sistema, que puede ser explotada por uno o varios atacantes para comprometer la seguridad de un sistema. Además, las vulnerabilidades no son propias de un sistema en particular sino también pueden estar presentes en una aplicación móvil es una herramienta informática diseñado para proporcionar servicios y funcionalidades para ejecutarse en teléfonos inteligentes o tabletas, que corre bajo un sistema operativo llamado Android que tiene sus orígenes en el sistema operativo Linux y diseñado por Google para dispositivos móviles (Srinivasa Rao Kotipalli, 2016).

Es un hecho bien conocido que ningún sistema es completamente seguro. Los atacantes siempre están buscando nuevas formas de explotar las vulnerabilidades, el riesgo de no explorar y comprender las estrategias disponibles para hacer frente a esta vulnerabilidad específica provoca un aumento en los ataques de inyección SQL. De hecho, según un informe de Check Point (2023) reveló que el 80% de las aplicaciones móviles presentan vulnerabilidades de inyección SQL. Además, según Android Developers (2023), estas vulnerabilidades son la principal causa de ataques a aplicaciones móviles Android. Este problema no se limita a las aplicaciones individuales. Según lo publicado por (Juan Manzano, 2021), en 2020, Accellion, una empresa de seguridad sufrió un ataque de inyección SQL que afectó a 100 de sus organizaciones en todo el mundo.

De este modo, el objetivo de este estudio es llevar a cabo un análisis/recopilación/compilación de la literatura relacionada con las estrategias de identificación y mitigación de vulnerabilidades de inyección SQL en aplicaciones móviles Android. En este sentido, este estudio busca explorar diferentes enfoques que ayudan a la identificación y mitigación de las vulnerabilidades de inyección SQL. Para ello, en donde la metodología RSL será la base para el desarrollo de esta investigación, que según (Carrizo et al., 2018), es un estudio secundario que tiene como objetivo identificar, evaluar y sintetizar la evidencia de investigaciones primarias.

Método

Esta investigación utiliza un enfoque cualitativo para comprender las estrategias existentes para la identificación y mitigación de vulnerabilidades de inyección SQL en aplicaciones móviles Android, que va de la mano con la metodología RSL. En particular, según (Narcisa Dolores Piza Burgos et al., 2019), es un enfoque cualitativo aquel que reconoce diversos contextos para comprender las diferentes perspectivas del fenómeno que se está investigando de fuentes, como artículos de revistas científicas, informes de investigación, tesis, y libros.

El método de la Revisión Sistemática de la Literatura (RSL), se divide en tres fases principales: planificación, realización y reporte de resultados. Se detalla a continuación cada una de las fases que componen esta metodología.

Ver tabla 1.

Tabla 1
Fases de la RSL

Fase	Métodos
Planificación	Fundamentación para la realización de la revisión. Interrogantes investigativas. Fuente de la búsqueda. Cadena de búsqueda
Realización	Búsqueda de documentos. Criterios de inclusión y exclusión. Criterios de la calidad. Extracción de datos. Síntesis de datos.
Reporte de resultados	Respuesta a las preguntas de investigación planteadas

Nota: La tabla muestra las fases de un RSL junto con los métodos que pertenecen a cada fase, recurso adaptado de: Velásquez & Juan D. (2015). 'Una Guía Corta para Escribir Revisiones Sistemáticas de Literatura Parte 3'.

Fundamentación de la revisión

El uso de aplicaciones móviles en la actualidad representa el día a día de la población en general, según el artículo publicado por (Andrew Buck, 2023), en el portal MobiLoud, las aplicaciones móviles son parte integral de vida diaria de las personas. En este sentido, destaca que los usuarios durante el 2022 pasaron un total de horas de 3,8 billones en aplicaciones móviles, con una tasa de retención del 22,6% un día después, 6,5% siete días después y del 2,6% 30 días después de la descarga de aplicaciones. Además, se destaca que el 88% del tiempo que las personas pasan en sus dispositivos móviles se invierte en el uso de aplicaciones.

En el ámbito de la ciberseguridad las aplicaciones móviles son un objetivo atractivo para los atacantes, dado que puede ser el medio para comprometer la información de un usuario que posee en su dispositivo móvil una gran cantidad de datos personales y sensibles, como información bancaria, contraseñas, contactos, etc. De este modo según (Arjona Pérez, 2014), aplicar medidas correctivas para evitar vulnerabilidades como, la inyección SQL, que es una de las amenazas más comunes y peligrosas para las aplicaciones móviles Android, puede brindar a los atacantes manipular y acceder a datos sensibles, comprometiendo así la seguridad y la privacidad de los usuarios.

A pesar de las medidas de seguridad existentes, las aplicaciones móviles Android siguen siendo el foco a este tipo de ataques. Por lo tanto, es imperativo realizar una revisión bibliográfica de las estrategias para identificar y mitigar las vulnerabilidades de inyección SQL.

Interrogantes investigativas

La presente investigación se centra en explorar las estrategias para identificar y mitigar las vulnerabilidades de inyección SQL en aplicaciones móviles Android. Para guiar este estudio, se han formulado las siguientes preguntas:

¿Cuáles son los riesgos de las vulnerabilidades de inyección SQL en aplicaciones móviles Android?

¿Cuáles son las estrategias de identificación de vulnerabilidades de inyección SQL en aplicaciones móviles Android, según la literatura actual?

¿Cuáles son las estrategias de mitigación de vulnerabilidades de inyección SQL en aplicaciones móviles Android, según la literatura actual?

Fuente de la búsqueda

Para la investigación, se seleccionó las bases de datos académicas, ACM Digital Library, ScienceDirect, para la recopilación de literatura, debido a la amplia gama de información de investigación que albergan en sus respectivos catálogos.

Cadena de búsqueda.

593 Digital Publisher CEIT ISSN 2588-0705

La estructura que compone la cadena de búsqueda para las bases de datos académicas se basa en los siguientes elementos: palabras clave, conectores y cadena de búsqueda.

Ver tabla 2.

Tabla 2

Cadenas de búsqueda resultante de la suma de palabras clave y conectores.

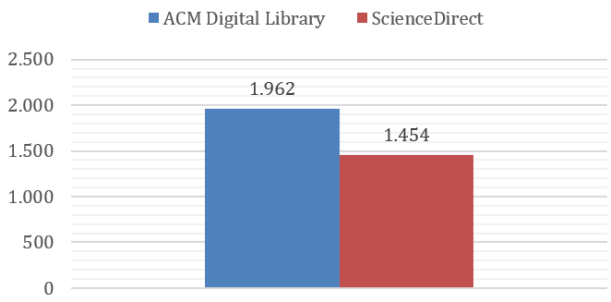
Palabras clave	Conectores	Cadenas de búsqueda
Inyección SQL Vulnerabilidades Aplicaciones Móviles Android	AND OR	**ACM Digital Library [** (SQL injection vulnerabilities) AND (android mobile applications) AND (mitigation AND identification) AND (risk AND vulnerabilities) AND (Author Keywords: "SQL injection" AND Android) **]**
Estrategias de Mitigación e Identificación Riesgos		**ScienceDirect [** ("SQL injection" AND mobile app OR mobile application) AND (mitigation OR detection OR identification) AND (risk OR vulnerability OR security) **]**

Búsqueda de documentos

En base a las cadenas de búsqueda aplicadas a las bases de datos ACM Digital Library y ScienceDirect, se obtuvo los siguientes artículos, ver figura 1.

Figura 1

Se muestran los resultados de aplicar las cadenas de búsqueda.



Nota: Se seleccionaron las cadenas de búsqueda que proporcionaban la mayor cantidad de artículos para la investigación, Fuente: Autoría propia.

Criterios de inclusión y exclusión

Para garantizar la validez y fiabilidad de los resultados de la revisión, se establecieron criterios para la selección de los estudios:

Criterios de exclusión:

Estudios que no estén enfocados en aplicaciones móviles Android.

Estudios que no aborden la temática de vulnerabilidades de inyección SQL.

Estudios que estén fuera del rango de 2020-2023

Criterios de inclusión:

Estudios que estén escritos en inglés.

Estudios que estén relacionados con las estrategias para identificar vulnerabilidades de inyección SQL

Estudios que estén relacionados estrategias para mitigar vulnerabilidades de inyección SQL

Investigaciones con acceso abierto.

Criterios de la calidad

Para evaluar la calidad de los estudios, se utilizará una escala de calificación de 1 a 6. La calificación de cada estudio se basará en el cumplimiento de los criterios de inclusión y exclusión mencionados anteriormente, el enfoque fue adaptado del estudio realizado por (Rodriguez-F et al., 2021), de este modo, si el estudio cumple un criterio, obtiene un '1', en caso contrario si el estudio no cumple con el criterio obtiene un '0', por lo tanto, los estudios que sumen '6', formaran parte del estudio y los estudios con criterios inferiores a '6' no formaran parte del estudio.

Extracción de datos

Para proceso de recopilación de la información, se aplicaron cadenas de búsqueda específicas a cada una de las bases de datos seleccionadas. Para la selección de los estudios debían cumplir con ciertos criterios de inclusión

y exclusión base, como: debían estar escritos en inglés, ser de acceso abierto y haber sido publicados entre 2020 y 2023. Los estudios que cumplían con estos criterios se descargaron y se almacenaron en carpetas específicas, nombrada de acuerdo con cada base de datos correspondiente. Esta organización facilitó el análisis posterior de los estudios.

Síntesis de datos

Durante esta etapa de síntesis de la información recopilada, se revisó los resúmenes y hallazgos reportados por cada uno de los estudios recopilados, para evaluar su cumplimiento en función de los criterios de inclusión como en los criterios de exclusión, con enfoque a dar respuesta a las preguntas de planteadas. Se presenta a continuación los resultados.

Ver tabla 3.

Tabla 3
Resultados del análisis efectuado a los diferentes estudios

Número total de artículos iniciales :	3.416	
Evaluación de los Artículos		
Criterio	Cumple	No cumple
Estudios dentro del rango de 2020-2023	1693	1723
Estudios que estén escritos en inglés	1693	0
Estudios con acceso abierto	755	938
Enfocado en aplicaciones móviles Android	245	510
Aborda la temática de vulnerabilidades de inyección SQL	22	223
Está relacionado con las estrategias para identificar como para mitigar vulnerabilidades de inyección SQL	11	11
Número total de artículos que cumplen con los criterios:	11	

Nota: El apartado de "Cumple" se basa en los criterios de calidad que representan a estudios con puntuación de 1, mientras tanto, el apartado de "No cumple", representa a los estudios con puntuación de 0, la parte que tiene de nombre "Número total de artículos que cumplen con los criterios" representa a los estudios válidos que darán respuesta a las preguntas planteadas.

Resultados

A continuación, se presentan los principales hallazgos clave derivados de la Revisión de literatura.

RQ 1: ¿Cuáles son los riesgos de las vulnerabilidades de inyección SQL en aplicaciones móviles Android?

Según el estudio publicado por (Han et al., 2023), el riesgo en una aplicación móvil pude estar presente en partes comunes que interactúan con regularidad los usuarios, como por ejemplo la entrada de usuario, si no se valida adecuadamente en el código posterior, puede facilitar a los atacantes manipular la lógica de ejecución de las consultas SQL para acceder a información confidencial y comprometer la integridad de la base de datos.

Tomando como base a lo anteriormente mencionado, se presenta los diferentes riesgos asociados a las vulnerabilidades de inyección SQL, en base a los diferentes estudios recopilados.

Ver tabla 4.

Tabla 4
Riesgos asociados a la inyección SQL.

Riesgo y Descripción	Referencia
Intrusión en bases de datos Ocurre cuando un atacante explota una vulnerabilidad para ganar acceso no autorizado.	E1, E6, E7, E8
Alteración de datos Ocurre cuando un atacante por medio de una vulnerabilidad modifica los datos de un sistema.	E1, E5, E6, E7, E8, E11
Ejecución remota de código Ocurre cuando un atacante por medio de una vulnerabilidad la ejecuta código en un sistema remoto sin permiso.	E1, E5, E6, E7, E8, E11
Instalación de malware Ocurre cuando un atacante por medio de una vulnerabilidad ingresa software en un sistema para causar daño o robo de información	E1, E4, E5, E8, E11
Fuga de información. Ocurre cuando, un atacante por medio de una intrusión a un sistema filtra información confidencial de manera intencionada.	E1, E5, E6, E8, E11
Encriptación de la información Ocurre cuando un atacante por medio de una vulnerabilidad ingresa encripta la información del sistema.	E1, E7, E11

593 Digital Publisher CEIT ISSN 2588-0705

RQ 2: ¿Cuáles son las estrategias de identificación de vulnerabilidades de inyección SQL en aplicaciones móviles Android, según la literatura actual?

Se presenta los enfoques más relevantes en cuanto a estrategias de identificación de vulnerabilidades de inyección SQL:

Para una identificación de vulnerabilidades de inyección SQL en Android el enfoque debe cubrir diversos apartados que conforman la arquitectura de una aplicación móvil, para así detectar y prevenir posibles amenazas que atentan con su disponibilidad, como es el caso de la publicación realizada por (Brun et al., 2023), donde se destacan estrategias como el uso de herramientas que advierten sobre posibles vulnerabilidades al emplear APIs, no confiar únicamente en la revisión de código por expertos, sino también considerar cómo se distribuyen las diferentes responsabilidades de escritura de código entre los desarrolladores, orientado al entendimiento de dónde podrían surgir las vulnerabilidades, para hacerle frente a los denominados "blindspot" o puntos ciegos, que consiste a esos vacíos de conocimiento donde no se consideran las diferentes posibilidades que una API puede verse afecta por los atacantes y la capacidad de desarrolladores para razonar adecuadamente sobre el código y detectar vulnerabilidades.

Por otro lado, el uso de herramientas para el análisis tanto dinámico como estático, brinda una visión amplia sobre el funcionamiento de las aplicaciones, facilitando la detección de posibles vulnerabilidades. En este sentido, el estudio realizado por (Deverashetti et al., 2022), resalta la ventaja que brindan Drozer y MobSF, como herramientas de análisis estático automatizado, para la identificación exhaustiva de vulnerabilidades en un gran número de aplicaciones de forma efectiva, herramientas que le permitieron inspeccionar inicialmente 219 aplicaciones de seguimiento menstrual de forma automatizada, descubriendo debilidades que luego fueron confirmadas y profundizadas a través de un análisis dinámico más meticuloso, la combinación de enfoques maximiza las posibilidades de encontrar vulnerabilidades al aprovechar las fortalezas de cada método.

El uso de herramientas no solo implica a la utilización de dispositivos físicos o software convencional, sino que también abarca tecnologías emergentes como el aprendizaje automático, que se convierte en una opción esencial en diversos campos, incluyendo la ciberseguridad. En este sentido el estudio realizado por (Apruzzese et al., 2023), resalta la importancia del aprendizaje automático en tareas de detección de amenazas, donde presenta una mayor eficacia que los métodos tradicionales, gracias a su capacidad de aprendizaje y adaptación continua, que aborda enfoques como la evaluación de vulnerabilidades mediante pruebas de penetración (penetration testing), como método de alto aporte a la ciberseguridad.

A la hora de la detección de vulnerabilidades se ve necesario tener un marco de trabajo que permita guiar el rumbo de manera integral, aplicando múltiples técnicas de manera coordinada y reduciendo las limitaciones de las aproximaciones individuales. Asimismo, el estudio realizado por (Gajrani et al., 2020), resalta la utilidad de Vulvet, la cual provee un tipo de marco de trabajo a través de una arquitectura de múltiples etapas. En particular, Vulvet propone: la combinación diferentes análisis estáticos de forma coordinada, realizando validaciones adicionales basadas en el flujo de control y la especificidad de Android. De esta forma, es capaz de detectar una amplia gama de vulnerabilidades de forma más precisa. Además, su implementación como extensión de una plataforma de análisis de bytecode también le otorga flexibilidad para el aprovechamiento de múltiples funcionalidades. De esta forma, Vulvet provee efectivamente un marco de trabajo integral que guía el proceso de detección de vulnerabilidades de una manera coordinada, evitando limitaciones de enfoques aislados y cerrando completamente el ciclo de análisis de seguridad.

Desde otro punto de vista orientado a herramientas de identificación, para determinar el originan de vulnerabilidades como, inyección

SQL, el estudio realizado por (Ami et al., 2021), resalta la importancia del framework µSE que permite generar de forma sistemática aplicaciones modificadas con comportamientos no deseados insertados, con el fin de evaluar de manera exhaustiva las capacidades y limitaciones de herramientas de análisis estático en la identificación de dichos comportamientos. A través del análisis pormenorizado de los casos falsos negativos, µSE facilita la detección de suposiciones y lagunas en las herramientas existentes, orientando al desarrollo de nuevas técnicas que mejoren la identificación de vulnerabilidades, en base a este antecedente, este método brinda la posibilidad de descubrir de dónde y cómo se producen las vulnerabilidades en las aplicaciones Android, por medio de la herramienta que genera sistemáticamente aplicaciones modificadas con comportamientos no deseados insertados.

A continuación, se presenta un resumen de las diferentes estrategias de identificación de vulnerabilidades de inyección SQL, junto con los diferentes enfoques recopilados no mencionados anteriormente.

Ver tabla 5.

Tabla 5

Resumen de las estrategias de identificación de vulnerabilidades de inyección SQL

Estrategias de identificación	Referencia
Herramientas que advierten sobre posibles vulnerabilidades al emplear APIs.	E5, E6, E7
No confiar únicamente en la revisión de código por expertos, sino también considerar cómo se distribuyen las diferentes responsabilidades de escritura de código entre los desarrolladores	E7
Herramientas de análisis estático automatizado como "Vulnerability Nets", para la búsqueda de posibles vulnerabilidades, seguido de un análisis dinámico, estimulando entradas de usuario y examinando su estado a la hora de la ejecución.	E1, E5, E6, E10
Herramientas basadas en ML para la detección de vulnerabilidades.	E2, E4, E5
Analizar datos no estructurados para predecir amenazas emergentes	E4
Utilización de un marco de trabajo como "Vulvet", que aplica análisis estáticos, como el análisis de flujo de datos, llamadas y parámetros, para detectar vulnerabilidades en las aplicaciones de Android.	E1, E2, E6
Utilización de guías especializadas en la seguridad, como las propuestas por OWASP y WASC, para la identificación de vulnerabilidades de inyección SQL.	E1, E3, E5, E6, E9

RQ 3: ¿ Cuáles son las estrategias de mitigación de vulnerabilidades de inyección SQL en aplicaciones móviles Android, según la literatura actual?

Se presenta los enfoques más relevantes en cuanto a estrategias de mitigación de vulnerabilidades de inyección SQL:

El papel de la identificación de vulnerabilidades es crucial para una mitigación eficaz. Esto es particularmente relevante cuando se trata de resolver problemas relacionados con vulnerabilidades como la inyección SQL. El estudio realizado por (Apruzzese et al., 2023), destaca el uso del aprendizaje automático (ML) en el monitoreo y correlación de alertas para mejorar la detección y respuesta, así como el análisis de datos no estructurados para predecir amenazas emergentes. Sin este paso previo de identificación, no sería posible determinar con precisión dónde aplicar medidas correctivas, como la evaluación de puntos débiles para instaurar controles que reduzcan los riesgos de seguridad. La información obtenida sobre los problemas de seguridad existentes, guía al proceso de mitigación, permitiendo abordar a estos problemas de manera puntual. De esta

593 Digital Publisher CEIT ISSN 2588-0705

forma, la identificación capacita a la mitigación para conseguir de manera efectiva eliminar estas vulnerabilidades de forma eficiente en la aplicación.

A la hora de realizado una previa identificación se presenta la problemática de solucionar, las brechas de seguridad que originan las vulnerabilidades, desde el enfoque del estudio realizado por (Gajrani et al., 2020), se propone el marco Vulvet como una solución integral para analizar y corregir automáticamente diversas categorías de vulnerabilidades en aplicaciones Android. Vulvet destaca la generación automática de parches mediante instrumentación de código, de tal manera que se evite cualquier interrupción, es decir lo que permite es mitigar las vulnerabilidades detectadas de forma eficiente sin afectar la funcionalidad de las aplicaciones.

Desde una perspectiva de análisis de seguridad orientada al código fuente de aplicaciones móviles, existen distintas estrategias y recomendaciones propuestas para mitigar vulnerabilidades comúnmente identificadas. Por otro lado, desde un enfoque orientado a la mitigación de vulnerabilidades en el código fuente de las aplicaciones móviles, destaca el estudio realizado por (Deverashetti et al., 2022), que propone mejorar el almacenamiento seguro de datos, restringir el acceso a componentes de aplicaciones solo mediante permisos para prevenir la interferencia de aplicaciones maliciosas con las consultas SQL. Si bien no están orientadas específicamente a la inyección SQL, contribuyen indirectamente a mitigar esta vulnerabilidad desde la perspectiva del desarrollo del código. Po otro lado, el estudio realizado por (Han et al., 2023), destaca como estrategia de mitigación fortalecer mecanismos de autenticación y autorización de usuarios, para restringir acceso a datos y funciones sensibles. Que en conjunto brinda soluciones, para así hacerle frente a estas vulnerabilidades.

No obstante, desde un enfoque orientado al análisis de código basado en el flujo de datos, el estudio realizado por (Krohmer et al., 2022), propone mejorar la efectividad de los analizadores estáticos o taint analyzers para identificar vulnerabilidades asociadas a validación insuficiente de entrada de datos. Esto permite derivar estrategias de mitigación como la corrección guiada de resultados, priorizando los hallazgos más críticos mediante su ordenación por riesgo e impacto. Asimismo, la integración del análisis en diferentes fases del ciclo de desarrollo, junto con la provisión de información técnica detallada para apoyar la reparación de vulnerabilidades, contribuirían a garantizar la detección y mitigación continua de deficiencias. La realización de actualizaciones periódicas del taint analyzer que incorporen nuevos patrones de detección permitiría identificar vulnerabilidades emergentes de forma más efectiva con el tiempo. El enfoque de especializar los analizadores propuesto en el estudio posibilita derivar estas estrategias de corrección guiada, integración en el ciclo de desarrollo y mejora continua, proveyendo soluciones completas para hacer frente a este tipo de problemas de seguridad.

A continuación, se presenta un resumen de las diferentes estrategias de mitigación de vulnerabilidades de inyección SQL.

Ver tabla 6.

Tabla 6

Resumen de las estrategias de mitigación de vulnerabilidades de inyección SQL

Estrategias de mitigación	Referencia
Herramientas que advierten sobre posibles vulnerabilidades y generen parches automáticos para corregirlas.	E2, E8
Mejora de métodos pedagógicos para reducir los impactos negativos de los puntos ciegos en las APIs	E8, E9
Corrección guiada de resultados, priorizando los hallazgos más críticos	E1, E8, E9
Integración del análisis en diferentes fases del ciclo de desarrollo	E1, E8
Provisión de información técnica detallada para apoyar la reparación de vulnerabilidades.	E1, E8, E9
Fortalecer mecanismos de autenticación y autorización de usuarios	E6
Mejorar el almacenamiento seguro de datos.	E5
Restringir el acceso a componentes de aplicaciones solo mediante permisos	E5
Utilización de guías especializadas en la seguridad, como las propuestas por OWASP y WASC, para la mitigación de vulnerabilidades de inyección SQL.	E1, E3, E5, E6, E9

Discusión

Se recopilo la información en función a las cadenas de búsqueda y con los criterios de exclusión e inclusión que juntos brindaron la información necesaria para responder de manera resumida dando repuesta a las tres preguntas de investigación formuladas. Se detallaron en base a los enfoques de cada estudio los principales riesgos, estrategias de detección y estrategias de mitigación relacionadas a vulnerabilidades de inyección SQL. Los hallazgos brindaron una visión integral sobre el tema que se orienta a la seguridad de aplicaciones móviles.

Durante el proceso de búsqueda de información en las bases de datos seleccionadas, arrojó diversos estudios relevantes sobre vulnerabilidades en aplicaciones móviles. Muchos de estos trabajos planteaban marcos de trabajo integrales o abordaban temáticas generales como la seguridad en Android, si bien enfoque era orientado a las vulnerabilidades, aportaban una fundamentación valiosa en su contexto.

Por ejemplo, varios estudios describieron plataformas de análisis automático que aplicaban diferentes técnicas de manera coordinada. Esto brindó las bases conceptuales para comprender cómo identificar deficiencias a través de enfoques múltiples. Asimismo, otros reportaron experiencias al evaluar categorías amplias de vulnerabilidades, lo cual enriqueció el panorama sobre métodos para detectar y mitigar la inyección SQL.

Limitaciones del estudio

Se limitó la revisión en función de los criterios tanto de inclusión como de exclusión, en donde se contempla el rango de años, el lenguaje, su tipo de acceso o su contexto no orientado a las estrategias de identificación y mitigación de vulnerabilidades inyección SQL en aplicaciones Android, es posible que algunas perspectivas o estudios valiosos que no cumplían con los criterios de inclusión hayan sido omitidos, ya sea por el idioma en el que estaban escritos o por el tipo de acceso que presentaban. Por lo tanto, los resultados de este estudio deben interpretarse teniendo en cuenta estas limitaciones.

Implicaciones de la investigación

Proporciona una visión actualizada de los principales riesgos asociados a la vulnerabilidad de inyección SQL en aplicaciones móviles Android, según la literatura revisada entre 2020-2023, permitiendo conocer los riesgos más relevantes a las que se enfrentan actualmente estas aplicaciones.

Identifica las distintas estrategias de identificación y mitigación de esta vulnerabilidad que han sido reportadas en estudios recientes, que brinda formas de fortalecer la seguridad en las aplicaciones.

Presenta de manera integral y sistemática el estado del conocimiento sobre este tema clave de seguridad en aplicaciones móviles, que remarca los principales hallazgos de múltiples fuentes en forma de revisión bibliográfica.

Futuras investigaciones

Tras revisar los resultados la información compilada, se sugiere que las futuras investigaciones se orienten hacia la creación de una guía que establezca las bases en temas de seguridad móvil. En donde se considere las distintas estrategias o técnicas emergentes no cubiertas en la revisión, para evaluar empíricamente su efectividad en aplicaciones móviles.

Conclusiones

Esta revisión sistemática de literatura permitió obtener una perspectiva integral actualizada acerca de los principales riesgos, estrategias de identificación como de mitigación frente a la vulnerabilidad de inyección SQL en aplicaciones móviles Android, conforme a la literatura más reciente. Los hallazgos proporcionan una guía orientada a los desarrolladores, a fin de incorporar las mejores prácticas que fortalecen la seguridad ante esta relevante amenaza. De esta manera, los resultados cumplen el objetivo de contribuir al estado del

593 Digital Publisher CEIT ISSN 2588-0705

conocimiento sobre este tema de interés para el aseguramiento de aplicaciones móviles.

En cuanto a los riesgos asociados a la vulnerabilidad, inyección SQL, se concluye que los más frecuentes reportados son el acceso no autorizado a la información almacenada y la edición de los registros en las bases de datos, denegación de servicio y ejecución de código malicioso, que pueden comprometer gravemente la privacidad, integridad y disponibilidad de las aplicaciones móviles. Respecto a las estrategias de identificación, las técnicas con mayor potencial incluyen pruebas de penetración automatizadas, análisis estático de código, detección asistida por herramientas y machine learning. Sin embargo, cada método presenta fortalezas y debilidades que deben de tomarse en cuenta. En cuanto a técnicas de mitigación, destaca las herramientas que advierten sobre posibles vulnerabilidades y generen parches automáticos para corregirlas, fortalecer mecanismos de autenticación y autorización de usuarios, restringir el acceso a componentes de aplicaciones solo mediante permisos, integración del análisis en diferentes fases del ciclo de desarrollo, utilización de guías especializadas en la seguridad, como las propuestas por OWASP Y WASC, para la mitigación de vulnerabilidades de inyección SQL, prácticas que según la literatura ayudan a prevenir inyección SQL.

Referencias

Ami, A. S., Kafle, K., Moran, K., Nadkarni, A., & Poshyvanyk, D. (2021). Systematic Mutation-Based Evaluation of the Soundness of Security-Focused Android Static Analysis Techniques. ACM Transactions on Privacy and Security, 24(3). https://doi.org/10.1145/3439802

Andrew Buck. (2023, December 8). 57 Mobile App Download, Usage and Revenue Statistics for 2024 | MobiLoud. https://www.mobiloud.com/es/blog/estad%C3%ADsticas-de-aplicaciones-m%C3%B3viles

Android Developers. (s/f). Inyección de SQL | App quality | Android Developers, de https://developer.android.com/privacy-and-security/risks/sql-injection?hl=es-419.

Apruzzese, G., Laskov, P., Montes De Oca, E., Mallouli, W., Brdalo Rapa, L., Grammatopoulos, A. V., & Di Franco, F. (2023). The Role of Machine Learning in Cybersecurity. Digital Threats: Research and Practice, 4(1). https://doi.org/10.1145/3545574

Arroyo Guardeño, D., Gayoso Martínez, V., & Hernández Encinas, L. (2020). Ciberseguridad. Editorial CSIC Consejo Superior de Investigaciones Científicas. https://elibro.puce.elogim.com/es/lc/puce/titulos/172144

Brun, Y., Lin, T., Somerville, J. E., Myers, E. M., & Ebner, N. (2023). Blindspots in Python and Java APIs Result in Vulnerable Code. ACM Transactions on Software Engineering and Methodology, 32(3). https://doi.org/10.1145/3571850

Carrizo, D., Moller, C., Carrizo, D., & Moller, C. (2018). Estructuras metodológicas de revisiones sistemáticas de literatura en Ingeniería de Software: un estudio de mapeo sistemático. *Ingeniare. Revista chilena de ingeniería*, *26*, 45–54. https://doi.org/10.4067/S0718-33052018000500045

Deverashetti, M., Ranjitha, K., & Pradeepthi, K. V. (2022). Security analysis of menstruation cycle tracking applications using static, dynamic and machine learning techniques. Journal of Information Security and Applications, 67. https://doi.org/10.1016/j.jisa.2022.103171

Gajrani, J., Tripathi, M., Laxmi, V., Somani, G., Zemmari, A., & Gaur, M. S. (2020). Vulvet: Vetting of Vulnerabilities in Android Apps to Thwart Exploitation. Digital Threats: Research and Practice, 1(2). https://doi.org/10.1145/3376121

Garg, S., & Baliyan, N. (2022). M2VMapper: Malware-to-Vulnerability mapping for Android using text processing. Expert Systems with Applications, 191. https://doi.org/10.1016/j.eswa.2021.116360

Google Android : Security vulnerabilities, CVEs sql injection. (2023). cvedetails.com. https://www.cvedetails.com/vulnerability-list/vendor_id-1224/product_id-19997/opsqli-1/Google-Android.html

Han, Y., Ji, X., Wang, Z., & Zhang, J. (2023). Systematic Analysis of Security and Vulnerabilities in Miniapps. Proceedings of the 2023 ACM Workshop on Secure and Trustworthy Superapps, 1–9. https://doi.org/10.1145/3605762.3624432

Juan Manzano. (2021, May 6). Ciberataque en proveedores: Caso Accellion - BDO. https://www.bdo.es/es-es/blogs/coordenadas-bdo/mayo-2021/ciberataque-en-proveedores-accellion

Justin Clarke-Salt. (2012). SQL Injection Attacks and Defense: Vol. 2nd ed. Syngress.

Kalouptsoglou, I., Siavvas, M., Ampatzoglou, A., Kehagias, D., & Chatzigeorgiou, A. (2023). Software vulnerability prediction: A systematic mapping study. In Information and Software Technology (Vol. 164). Elsevier B.V. https://doi.org/10.1016/j.infsof.2023.107303

Krohmer, D., Sharma, K., & Chen, S. (2022). Adapting Static Taint Analyzers to Software Marketplaces: A Leverage Point for Mass Vulnerability Detection? SCORED 2022 - Proceedings of the 2022 ACM Workshop on Software Supply Chain Offensive Research and Ecosystem Defenses, Co-Located with CCS 2022, 73–82. https://doi.org/10.1145/3560835.3564553

Narcisa Dolores Piza Burgos, Francisco Alejandro Amaiquema Márquez, & Gina Esmeralda Beltrán Baquerizo. (2019). *Métodos y técnicas en la investigación cualitativa. Algunas precisiones necesarias*. http://scielo.sld.cu/scielo.php?script=sci_arttext&pid=S1990-86442019000500455.

OWASP Foundation. (2021). A03 Injection - OWASP Top 10:2021. https://owasp.org/Top10/A03_2021-Injection/

Senanayake, J., Kalutarage, H., Al-Kadri, M. O., Petrovski, A., & Piras, L. (2023). Android Source Code Vulnerability Detection: A Systematic Literature Review. In ACM Computing Surveys (Vol. 55, Issue 9). Association for Computing Machinery. https://doi.org/10.1145/3556974

Srinivasa Rao Kotipalli, M. A. I. (2016). *Hacking Android: EBSCOhost*.

Urcuqui, C. C., & Navarro Cadavid, A. (2022). Ciberseguridad: los datos tienen la respuesta. Editorial Universidad Icesi.

Velásquez, & Juan D. (2015). Una Guía Corta para Escribir Revisiones Sistemáticas de Literatura Parte 3. https://doi.org/10.15446/dyna.v82n189.48931

Wang, P., Liu, S., Liu, A., & Jiang, W. (2023). Detecting Security Vulnerabilities with Vulnerability Nets. Journal of Systems and Software, 111902. https://doi.org/10.1016/j.jss.2023.111902

Yuri Diogenes, & Dr. Erdal Ozkaya. (2019). Cybersecurity – Attack and Defense Strategies: Counter Modern Threats and ...: EBSCOhost. https://pwebebsco.puce.elogim.com/ehost/detail/detail?vid=6&sid=a0357c86-9466-41c1-bca3-eb44829a02ac%40redis&bdata=JnNpdGU9ZWhvc3QtbGl2ZQ%3d%3d#AN=2344998&db=nlebk

593 Digital Publisher CEIT ISSN 2588-0705

Se presenta a continuación los diferentes estudios recopilados, ver anexo 1.

Anexo 1

Se presenta los estudios recopilados con su ID junto con datos del estudio

ID	Año	Autor	Titulo
E1	2022	Krohmer, Daniel, Sharma,Kunal, Chen, Shi	Adapting Static Taint Analyzers to Software Marketplaces:A Leverage Point for Mass Vulnerability Detection?
E2	2020	Gajrani, Jyoti, Tripathi, Meenakshi, Laxmi, Vijay, Somani, Gaurav, Zemmari, Akka, Gaur, Manoj Singh	Vulvet: Vetting of Vulnerabilities in Android Apps to Thwart Exploitation
E3	2023	Kalouptsoglou, Ilias, Siavvas, Miltiadis, Ampatzoglou, Apostolos, Kehagias, Dionysios, Chatzigeorgiou,Alexander	Software vulnerability prediction: A systematic mapping study
E4	2023	Apruzzese, Giovanni, Laskov, Pavel, Montes De Oca, Edgardo, Mallouli, Wissam, Brdalo Rapa, Luis ,Grammatopoulos, Athanasios Vasileios, Di Franco, Fabio	The Role of Machine Learning in Cybersecurity
E5	2022	Deverashetti, Mounika, Ranjitha, K, Pradeepthi, K. V	Security analysis of menstruation cycle tracking applications using static, dynamic and machine learning techniques
E6	2023	Han, Yuyang, Ji, Xu, Wang, Zhiqiang, Zhang, Jianyi	Systematic Analysis of Security and Vulnerabilities in Miniapps
E7	2022	Garg, Shivi, Baliyan, Niyati	M2VMapper:Malware-to-Vulnerability mapping for Android using text processing
E8	2023	Brun, Yuriy, Lin, Tian, Somerville, Jessie Elise, Myers, Elisha M, Ebner, Natalie	Blindspots in Python and Java APIs Result in Vulnerable Code
E9	2021	Ami, Amit Seal, Kafle, Kaushal, Moran, Kevin, Nadkarni, Adwait, Poshyvanyk, Denys	Systematic Mutation-Based Evaluation of the Soundness of Security-Focused Android Static Analysis Techniques
E10	2023	Wang, Pingyan, Liu, Shaoying, Liu, Ai, Jiang, Wen	Detecting Security Vulnerabilities with Vulnerability Nets
E11	2023	Senanayake, Janaka, Kalutarage, Harsha, Al-Kadri, Mhd Omar, Petrovski, Andrei, Piras, Luca	Android Source Code Vulnerability Detection: A Systematic Literature Review