INYECCIÓN DE SQL, CASO DE ESTUDIO OWASP

Giovanny Chicaiza [1], Luis Ponce[2], Gabriela Velásquez Campos[3]

Unidad de Postgrados,

Gerencia de Sistemas, Universidad de las Fuerzas Armadas ESPE,

Sangolquí, Ecuador, giova236015@hotmail.com, lepjbien@hotmail.com, gabychy8@gmail.com

RESUMEN

El presente documento realiza un análisis de seguridad de las aplicaciones web, en donde se trata de encontrar la debilidades del sistema de seguridad y protección de las mismas por medio de pruebas de penetración enfocadas a explotar las vulnerabilidades de inyección SQL, para luego de esto implementar las recomendaciones de Open Web Application Security Project (OWASP), para realizar las mismo ataques a los sistemas web con las técnicas OWASP implementadas.

Una vez obtenidas las dos mediciones se comprueba la efectividad de OWASP en base a la comparación de los resultados, de esta forma se pretende verificar la efectividad de las técnicas OWASP para mitigar estas vulnerabilidades.

Keywords: *inyección SQL, seguridad, OWASP.*

1. INTRODUCCIÓN

La mayoría de las organizaciones en la actualidad administran y gestionan sus transacciones e información con sistemas informáticos y bases de datos. De esta manera los riesgos de ataques a la información clave es inminente por lo cual las organizaciones deben proteger sus datos implementando controles de seguridad.

Los ataques a las organizaciones para vulnerar los controles de seguridad, son más comunes de lo que se puede pensar. Las personas que tratan de vulnerar las seguridades para un fin malicioso buscan en los controles un hueco de seguridad para infectar al dispositivo victima con un software malicioso.

Para esto las personas primero deben buscar o tratar de establecer una comunicación con el dispositivo victima esta tarea se lo realiza con el escaneo de puertos, una vez encontrado el puerto a vulnerar empieza la infección del dispositivo víctima.

Las amenazas pueden llegar a través de internet o en línea, o desconectado de la red a través de medios magnéticos, tal como se puede observar en las gráficas obtenidas de Kaspersky (gráfica 1, 2 y 3).

Ranking mundial	País	% de usuarios con intentos de infección	Nº de incidentes
38	**Brasil**	32.0%	22122995
63	**Perú**	28.7%	4537175
64	**Panamá**	28.5%	929976
74	**México**	27.0%	17514481
80	**Honduras**	26.5%	458438
90	**El Salvador**	25.4%	439970
92	**Nicaragua**	24.9%	310517
95	**Ecuador**	24.6%	3157211
97	**Colombia**	24.4%	4991622
98	**Chile**	24.2%	1813276
99	**Guatemala**	24.2%	822242
112	**República Dominicana**	22.8%	435710
125	**Costa Rica**	21.6%	588932
132	**Argentina**	21.2%	1375126
148	**Uruguay**	19.6%	175873
165	**Paraguay**	17.9%	238189
232	**Cuba**	7.9%	30586

Gráfica 1 - Distribución geográfica de los incidentes de los ataques registrados en línea (on-line) Fuente Kaspersky Lab

Ranking mundial	País	% de usuarios con intentos de infección	Nº de incidentes
58	Brasil	46.8%	88852802
77	Perú	43.8%	11248606
92	México	41.6%	54488994
104	Colombia	39.9%	14147355
105	Ecuador	39.9%	7029962
110	Honduras	39.1%	1276765
123	Panamá	37.5%	2741821
126	República Dominicana	36.7%	1867047
127	Argentina	36.6%	4466697
134	Nicaragua	35.9%	736624
135	El Salvador	35.7%	1241320
141	Chile	34.8%	4593217
142	Guatemala	34.7%	2436633
154	Paraguay	32.5%	879106
165	Costa Rica	30.8%	1685651
181	Uruguay	28.4%	517721
225	Cuba	18.9%	66875

Gráfica 2 Distribución geográfica de los incidentes de los ataques registrados fuera de línea (off-line) Fuente Kaspersky Lab

Uno de los ataques comunes a sistemas WEB, es SQL INJECTION, que trata de insertar cadena de texto con sentencias SQL, a la página víctima, y con esto comenzar a obtener información de los sistemas con esta técnica, según el nivel de seguridad que este implementado será el impacto del ataque e información recolectada.

Como se dijo en el párrafo anterior, si la página no tiene las seguridades necesarias, el atacante puede llegar a conseguir credenciales relevantes, adicional borrado de información, como descubrir la estructura de base de datos como la información del servidor

2. MARCO TEÓRICO

2.1. SEGURIDAD DE LA INFORMACIÓN

La seguridad de la información es el conjunto de medidas preventivas y reactivas de las organizaciones y de los sistemas tecnológicos que permiten resguardar y proteger la información buscando mantener la confidencialidad, la disponibilidad e integridad de la misma.

Para el hombre como individuo, la seguridad de la información tiene un efecto significativo respecto a su privacidad, la que puede cobrar distintas dimensiones dependiendo de la cultura del mismo (1).

En la Seguridad Informática se debe distinguir dos propósitos de protección, la Seguridad de la Información y la Protección de Datos.

En la Seguridad de la Información el objetivo de la protección son los datos mismos y trata de evitar su perdida y modificación non-autorizado. La protección debe garantizar en primer lugar la confidencialidad, integridad y disponibilidad de los datos, sin embargo existen más requisitos como por ejemplo la autenticidad entre otros.

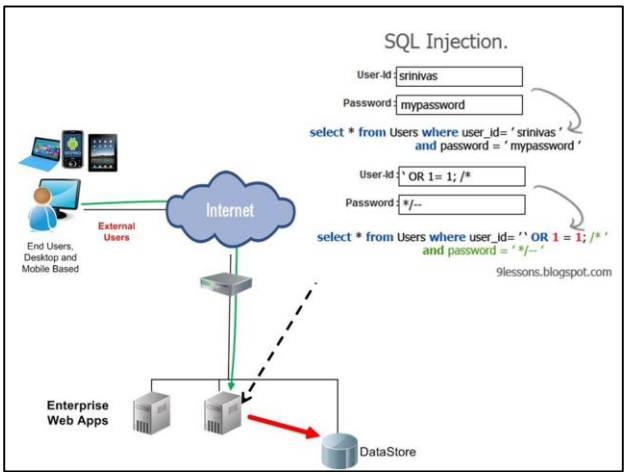
El motivo o el motor para implementar medidas de protección, que responden a la Seguridad de la Información, es el propio interés de la institución o persona que maneja los datos, porque la pérdida o modificación de los datos, le puede causar un daño (material o inmaterial). Entonces en referencia al ejercicio con el banco, la pérdida o la modificación errónea, sea causado intencionalmente o simplemente por negligencia humana, de algún récord de una cuenta bancaria, puede resultar en pérdidas económicas u otros consecuencias negativas para la institución.

En el caso de la Protección de Datos, el objetivo de la protección no son los datos en sí mismo, sino el contenido de la información sobre personas, para evitar el abuso de esta.

Esta vez, el motivo o el motor para la implementación de medidas de protección, por parte de la institución o persona que maneja los datos, es la obligación jurídica o la simple ética personal, de evitar consecuencias negativas para las personas de las cuales se trata la información. (2)

2.2. SQL INJECTION

Un ataque de inyección SQL consiste en la inserción o “inyección” de una consulta SQL a través de los datos de entrada del cliente de la aplicación. Una inyección SQL éxito exploit puede leer los datos sensibles de la base de datos, modificar datos de bases de datos (Insertar / Actualizar / Eliminar), ejecutar operaciones de administración en la base de datos (tales como apagar el DBMS), recuperar el contenido de un archivo determinado presente en el sistema de archivos DBMS y en algunos casos emitir comandos al sistema operativo. Los ataques de inyección SQL son un tipo de ataque de inyección, en el que comandos SQL se inyectan en la entrada de datos de plano a fin de efectuar la ejecución de comandos predefinidos SQL. (3)



Gráfica 3 Inyección SQL

Los ataques por inyección SQL permiten a los atacantes suplantar identidad, alterar datos existentes, causar problemas de repudio, permite la revelación de todos los datos en el sistema, destruir los datos o si no volverlos inasequibles, y convertirse en administradores del servidor de base de datos.

La inyección SQL es muy común con aplicaciones PHP y ASP. Debido a la naturaleza de las interfaces programáticas, las aplicaciones J2EE y ASP.NET tienen menor probabilidad de ser fácilmente atacadas por una inyección SQL.

La gravedad de una inyección SQL está limitada por la habilidad e imaginación del atacante, y en menor medida a las contramedidas, como por ejemplo las conexiones con bajo privilegio al servidor de bases de datos, entre otras. En general, se considera a la inyección SQL de alto impacto. (3)

La inyección SQL se ha convertido en un problema común con sitios web que cuentan con base de datos. La falla es fácilmente detectada y fácilmente explotada, y como tal, cualquier sitio o paquete de software con incluso una mínima base de usuario es propenso a ser objeto de un intento de ataque de este tipo. Esencialmente, el ataque es llevado a cabo mediante la colocación de una meta carácter en los datos de entrada para colocar comandos SQL en el plano de control, el cual antes no existía.

Este error depende del hecho de que SQL no hace real distinción entre los planos de datos y los de control.

Las principales consecuencias son:

Confidencialidad: Dado que las bases de datos SQL generalmente almacenan información sensible, la pérdida de la confiabilidad es un problema frecuente con las vulnerabilidades de inyección SQL.

Autenticación: Si se utilizan consultas SQL pobres para chequear nombres de usuarios u contraseñas, puede ser posible conectarse a un sistema como otro usuario sin conocimiento previo de la contraseña.

Autorización: Si la información de autorización es almacenada en una base de datos SQL, puede ser posible cambiar esta información mediante la explotación exitosa de una vulnerabilidad por inyección SQL.

Integridad: Así como puede ser posible leer información sensible, también es posible realizar cambios o incluso borrar esta información mediante un ataque por inyección SQL.

2.3. TIPO DE SQL INJECTION

Los ataques realmente varían de unos a otros, pero nos sentimos más se pueden clasificar en dos categorías:

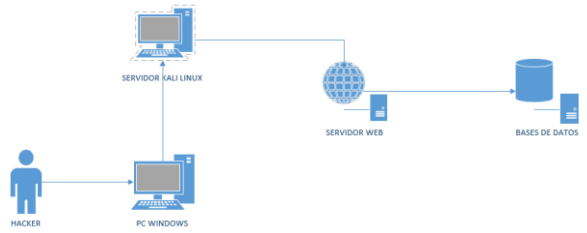
a) **Datos Exfiltración:** Exfiltración de datos a través de la inyección de SQL es lo que ha contribuido a algunas de las violaciones de datos más grandes hasta la fecha. Los atacantes encuentran una vulnerabilidad que les permite a la lista de todas las tablas y volcar todas las cuentas de usuario, correos electrónicos y contraseñas.

b) **Código Inyección:** No vemos esto muy a menudo, que a menudo se basan en algunas de vulnerabilidad inicial pre-pruebas que nos bloqueamos automáticamente a través de nuestro Sitio Web Firewall por lo que es mucho más difícil de grabar y tentativa.

3. CASO PRÁCTICO

3.1. DIAGRAMA DE RED

A continuación mostramos el diagrama de red que fue armado para el presente trabajo.



Hackers: persona experta en alguna rama de la tecnología, que se dedica a intervenir y/o realizar alteraciones técnicas con buenas o malas intenciones.

Kali Linux: es una distribución de Linux avanzada para pruebas de penetración y auditorías de seguridad.

Servidor Web: Un servidor web o servidor HTTP es un programa informático que procesa una aplicación del lado del servidor.

Bases de Datos: Es una colección de información organizada de forma que un programa de ordenador pueda seleccionar rápidamente los fragmentos de datos que necesite.

3.2. BÚSQUEDA DE URL CON VULNERABILIDADES

Mediante la utilización de un analizador de vulnerabilidades llamado w3af, procedimos a buscar una URL que nos sirva para poder hacer un ataque de inyección de SQL.

En la figura 1 podemos ver la ejecución del código para que se ejecute el programa w3af y poder encontrar la URL que servirá en toda la práctica.

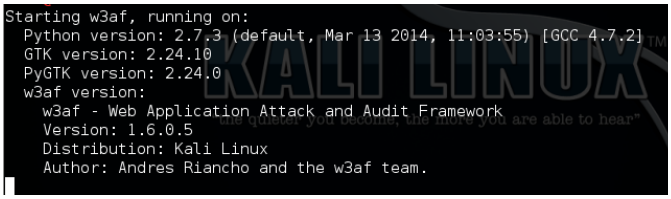


Figura 1 Ejecución del Analizador de Vulnerabilidades w3af

En la figura 2. Podemos ver la ejecución del analizador de vulnerabilidad aplicado a la URL http://fe.gms.com.ec/login.aspx

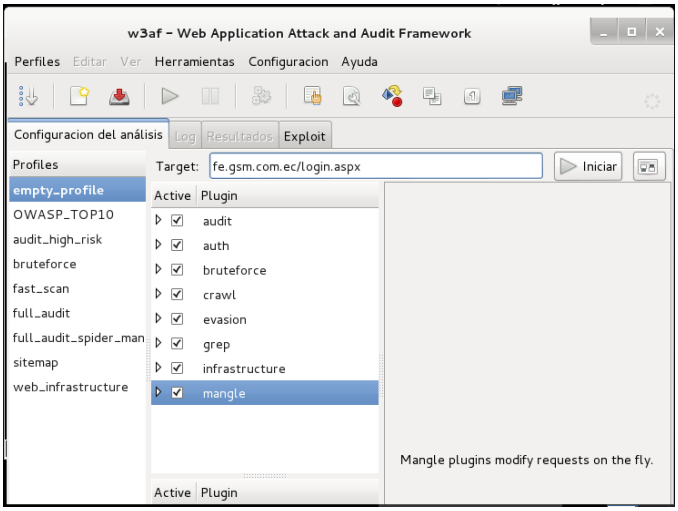


Figura 2 Ejecución del Analizador de Vulnerabilidades a la URL de GMS

En la figura 3. Podemos observar que a la url: no se puede atacar debido a que nos indica que no tiene vulnerabilidades.

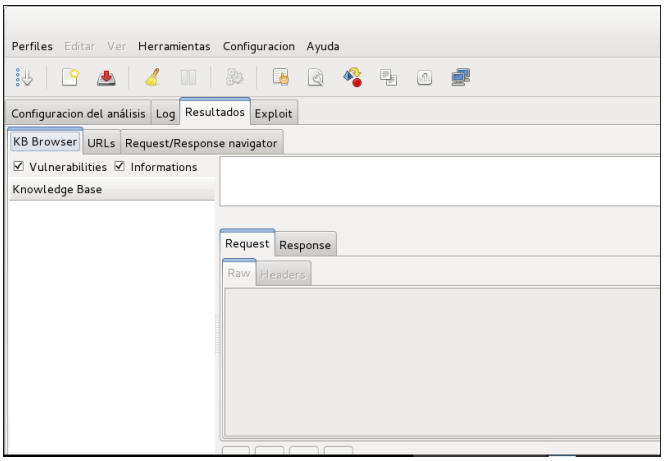


Figura 3. Resultado de Vulnerabilidades a la URL de GMS

En la figura 4. Podemos ver la ejecución del analizador de vulnerabilidad aplicado a la URL http://www.pichincha.com/

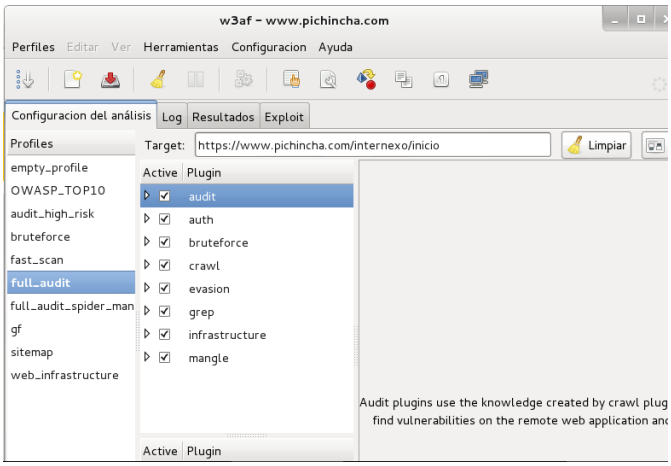


Figura 4. Ejecución del Analizador de Vulnerabilidades a la URL de Banco Pichincha

En la figura 5. Podemos observar que a la url: http://www.pichincha.com/ no se puede atacar debido a que nos indica que tiene 2 vulnerabilidades, pero las cuales no permiten un ataque SQL injection.

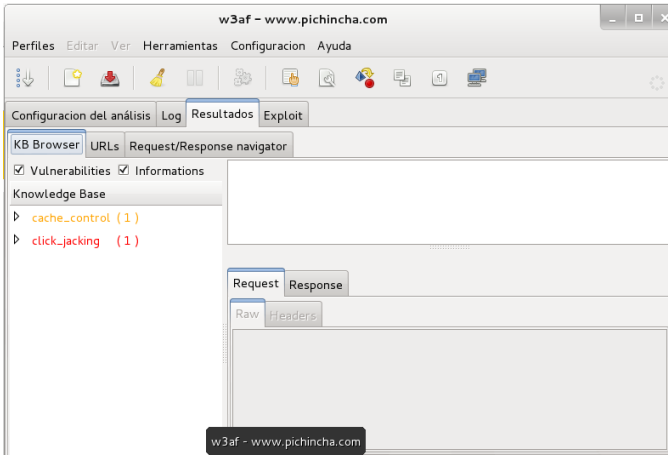


Figura 5. Resultado de Vulnerabilidades a la URL de Banco Pichincha

En la figura 6. Podemos ver la ejecución del analizador de vulnerabilidades a la URL http://www.hacerseoir.com.ar/ .

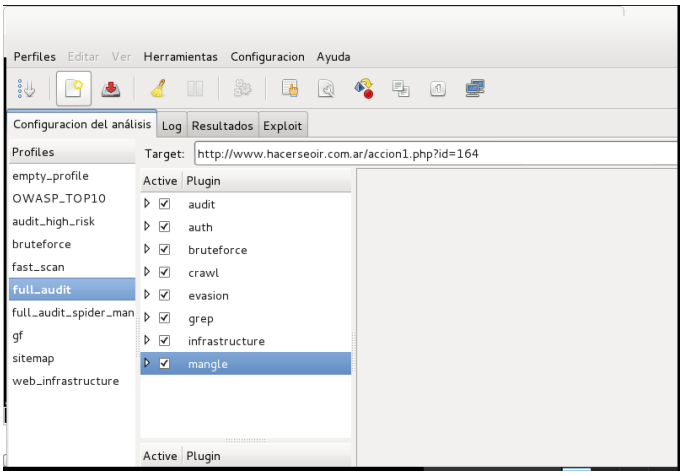


Figura 6. Ejecución del Analizador de Vulnerabilidades a la URL de Hacerse Oir

En la figura 7 podemos observar que nos indica el número de vulnerabilidades detectadas en la URL http://www.hacerseoir.com.ar/, la cual utilizaremos para hacerle un ataque de SQL injection.

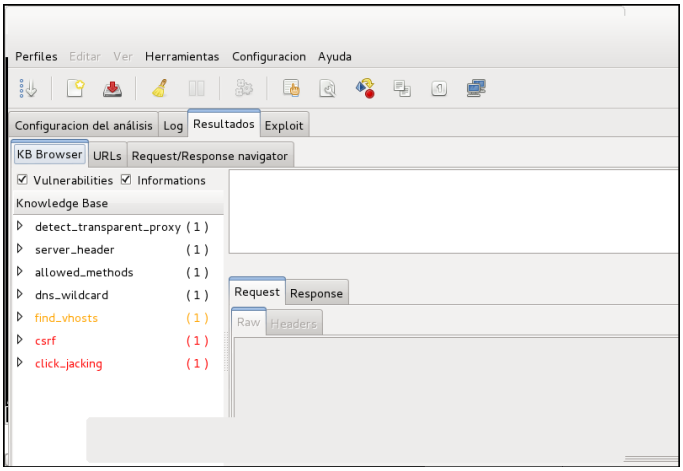


Figura 7. Resultado de Vulnerabilidades a la URL de Hacerse Oir

En la figura 8 podemos ver un log en donde se registra cada una de las vulnerabilidades encontradas en la url: http://www.hacerseoir.com.ar/.

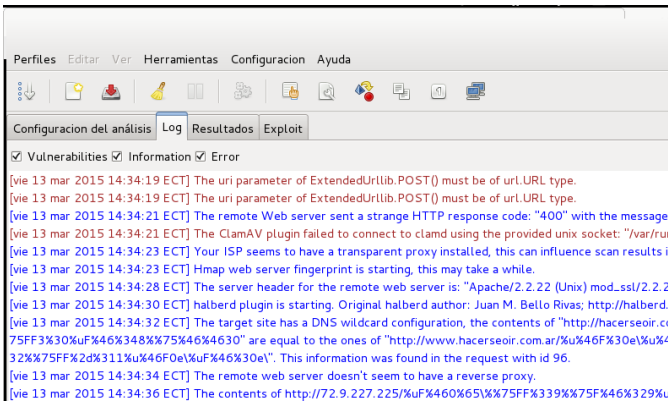


Figura 8. Log de registro de vulnerabilidades a URL de Hacerse Oir

En la figura 9 se puede observar con mayor detalle cual es la vulnerabilidad encontrada y la información encontrada en la misma.

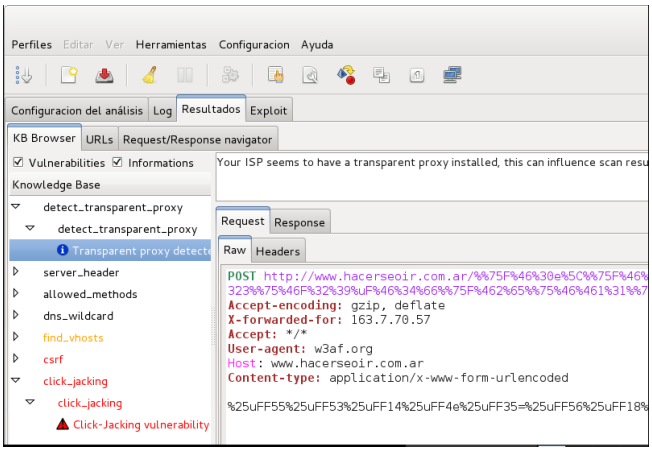


Figura 9. Descripción a detalle de vulnerabilidades a URL de Hacerse Oir

3.3. ATAQUE SQL INYECTION CON SQLMAP USANDO KALI LINUX

Así que una vez tenemos la URL: http://www.hacerseoir.com.ar/ la introducimos en SQL map de la siguiente manera, especificando "–dbs", lo cual indica que nos saque las bases de datos existentes, esto lo podemos ver en la figura 10.

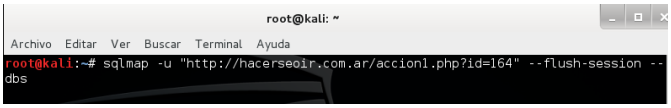


Figura 10. Comando para saber las bases de datos

Como vemos en la figura 11 la herramienta nos está indicando que el parámetro **id** es vulnerable y que según test heurísticos de SQL map, la base de datos parece ser MySQL, por lo que nos propone hacer las pruebas únicamente para esa plataforma.

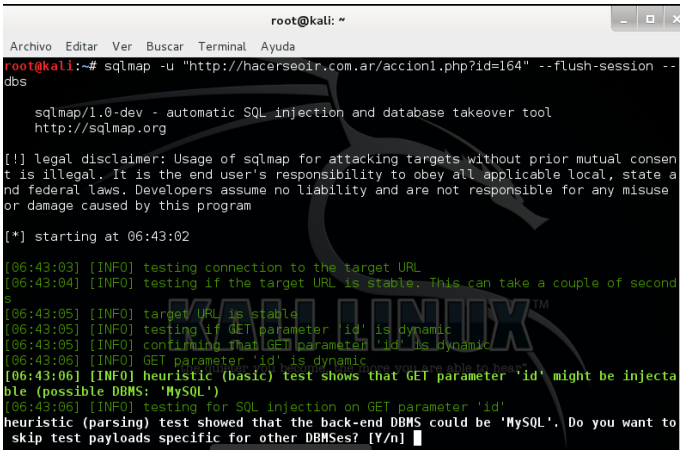


Figura 11. Parámetro id es vulnerable.

Tras aceptar (**poner Y**) y como vemos en la figura 12. Realiza gran cantidad de pruebas hasta obtener que realmente la web es vulnerable a SQL Injection mediante Union-Base, lo cual nos permitirá sacar los datos de una forma muy rápida.

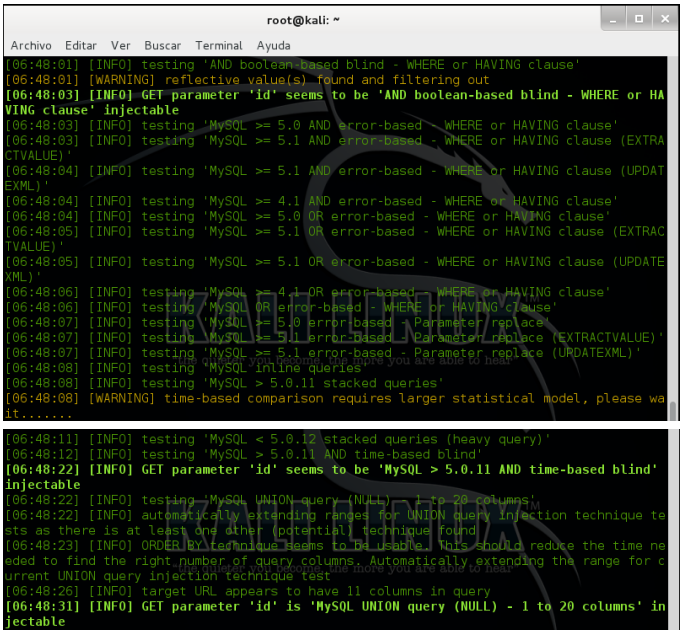


Figura 12. Pruebas para determinar que es posible SQL injection.

Tras dar explotar la SQL i vemos que nos saca las formas en las que es posible explotar la vulnerabilidad según las pruebas realizadas, y las bases de datos existentes, esto lo podemos ver en la figura 13.

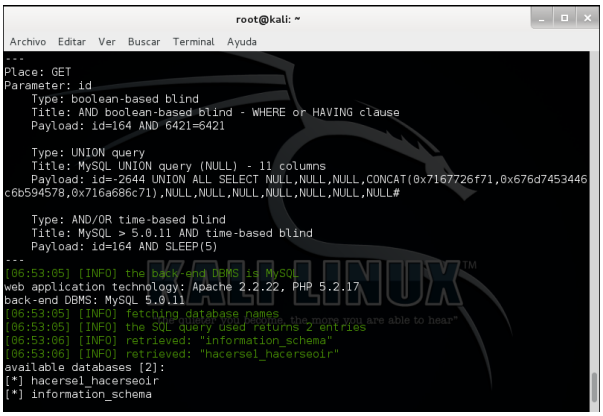


Figura 13. Formas de SQL injection y bases de datos existentes.

Ahora que ya tenemos las bases de datos vamos a ver cómo podríamos sacar las diferentes tablas, en este caso de la base de datos hacerce1_hacerseoir. Para ello ejecutamos la siguiente consulta con SQL map, como podemos ver en la figura 14.

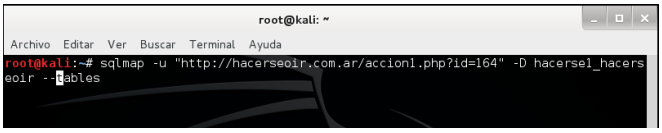


Figura 14. Comando para obtener las tablas de la base de datos hacerse1_hacerseoir.

Lo cual nos da como resultado las diferentes tablas (19 en total), que hay en dicha BD como vemos en la figura 15.

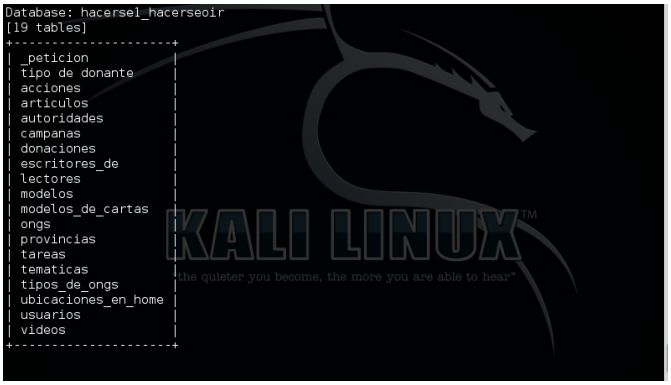


Figura 15. Tablas encontradas de la BDD hacerse1_hacerseoir.

Después lo que hicimos es sacar las columnas que tiene la tabla que nos interese, en este caso usuarios, para ello ejecutamos el siguiente comando mostrado en la figura 16.

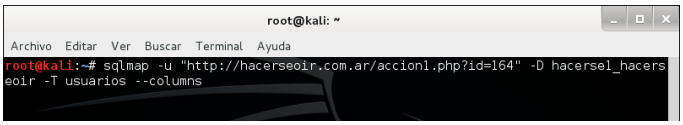


Figura 16. Comando para sacar las columnas

Y como esperábamos nos devuelve las columnas existentes en esa tabla **(2 columnas),** como podemos ver en la figura 17.

Figura 17. Columnas de la tabla usuarios.

Después de eso lo que se hizo fue realizar un volcado de la tabla completa, para de esta manera poder obtener los datos de dichos campos (id, nivel, nombre, password), el comando ejecutado se puede ver en la figura 18.

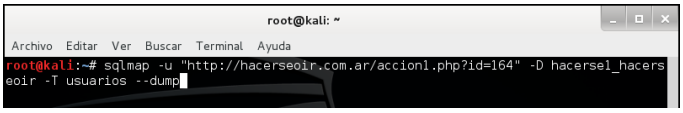


Figura 18. Comando para volcar la tabla.

Los datos obtenidos podemos ver en la figura 19, los cuales nos servirían para hacer otros tipos de ataques, debido a que se obtuvieron datos de acceso (login y usuario).

Figura 19. Datos obtenidos de la tabla.

Con esto hemos podido realizar con éxito un ataque SQL injection, como fue planteada en un principio.

4. RESULTADOS

4.1. MITIGACIÓN DE ATAQUES SQL INJECTION

En el presente trabajo se buscó páginas web, en las cuales se ejecutó el ataque para luego mitigarla, pero ninguna página de las empresas donde laboran los integrantes del presente trabajo presento este tipo de riesgos, lo que significa que algunas empresas en Ecuador ya tienen conciencia de seguridad y tienen puesto los mecanismos necesarios para la mitigación de este tipo de ataque.

A continuación se presentan algunas buenas prácticas, métodos o tecnologías para mitigar el ataque SQL INJECTION.

a) Procedimientos almacenados: se puede crear funciones estáticas de inserción, eliminación, actualización y selección en la base de datos que accedan a las tablas de la base de datos, estas funciones tienen parámetros con tipos definidos que al momento de realizar una sentencia SQL son tratadas en su totalidad como el tipo declarado evitando de esta forma inyección del código SQL , pues sería procesado como cadena de texto parámetro y no como parte de la sentencia

b) En lo posible en las páginas web colocar en los input, cuadros de lista, para que los usuarios solo puedan seleccionar datos predefinidos, s esto no es posible leer las variables de acuerdo al tipo definido en la base de datos y enviar esto como un solo campo es decir para campos tipo cadena se debe enviar entre comilla simple o doble según el lenguaje de base de datos, asegurando que si existiera dentro de una secuencia de caracteres estos símbolos se realice el respectivo escape para que sea tomado como parte de la cadena y no de la sentencia

c) Firewall WEB, este software se instala entre el servidor WEB y el usuario, este tipo de software tiene reglas de validación que proveen ataques de seguridad de diferente tipo entre ellas SQL INJECTION, lo importante de este tipo de implementación es que constantemente los proveedores de estas soluciones actualizan sus reglas para nuevos tipos de ataques.

d) Mantener actualizado el servidor y/o contenedor WEB, muchos de los servidores WEB Apache, Tomcat, Jboss, GlashFish, IIS actualmente tienen módulos que se encargan de prevenir diferentes tipos de ataques.

e) Cuando aparecen nuevos tipos de ataques los proveedores actualizan sus sistemas para ofrecer protección a la aplicación Web, por lo que es aconsejable mantener siempre actualizado los servidores de aplicación.

f) Evitar usar variables en la URL, en lo posible se debe usar variables dentro de la página URL, o en cookies haciendo que sea un poco más laborioso para el atacante encontrar la variable e inyectar código, además de esta técnica debe ser apoyada por la técnica 2.

g) Los mensajes de error deben indicar el error en detalle general es decir sin mostrar información relevante que puede ser usada por el atacante para vulnerar el sitio WEB.

h) Verificar los textos introducidos con un diccionario de cadenas usadas en ataques SQL INJECTION de esta forma si el atacante ingresa 1=1 y esta cadena está en el diccionario se enviar un mensaje de error al ejecutar la transacción

i) Sentencias preparadas, la mayoría de lenguaje de programación permite el uso de sentencias preparadas en donde se crea la sentencia con parámetros de ingreso y luego se asigna valor a estos parámetros, le interprete del lenguaje entiende que todo los caracteres que contiene el parámetro debe ser tratado como uno solo, y coloca automáticamente los caracteres de escape.

j) Realizar en la etapa de pruebas del software, pruebas de seguridad SQL INJECTION, de esta forma se asegura que la página o sitio WEB, no sea vulnerable por cualquier atacante

5. CONCLUSIONES Y TRABAJO FUTURO

El ataque de tipo SQL injection, permite obtener información de distinto tipo desde las páginas WEB, entre ellas datos de la base de datos, estructura del servidor de base de datos, IP de servidor de base de datos y aplicaciones.

En todo el planeta existen un sinnúmero de páginas con este tipo de problemas, lo que conlleva que este tipo de ataques estén entre los más frecuentes.

La mayoría de casos en los que se puede realizar ataques tipo SQL injection, es por errores de programación, descuidos del desarrollador, malas validaciones, uso de librerías estándar, entre otras.

OWASP, ofrece técnicas al desarrollador para tratar de mitigar los riesgos de este tipo de ataques

REFERENCIAS BIBLIOGRÁFICAS

1. **Wikipedia.** WIKIPEDIA. *WIKIPEDIA - SEGURIDAD DE LA INFORMACION.* [En línea] [Citado el: 15 de 03 de 2015.] http://es.wikipedia.org/wiki/Seguridad_de_la_informaci%C3%B3n.

2. **Wordpress.** WORDPRESS. *WORDPRESS - SEGURIDAD INFORMACION.* [En línea] [Citado el: 15 de 03 de 2015.] https://protejete.wordpress.com/gdr_principal/seguridad_informacion_proteccion/.

3. **OWASP.** OWASP BOOK. [En línea] [Citado el: 15 de 03 de 2015.] https://www.owasp.org/index.php/Inyecci%C3%B3n_SQL.

4. **Scholt, Theodoor, Balzarotti, Davide y Kirda, Engin.** *Have things changed now? An Empirical Study on Input Validation Vulnerabilities in Web Applications.* 2012.

5. *DECRETO por el que se expide la Ley Federal de Protección de Datos Personales en Posesión de los Particulares.* **Diario Oficial de la Federación.** 2010, Diario Oficial de la Federación, págs. 11-12.

6. **Tatum, Malcolm.** What Is a Cyberattack? *Wise Geek.* [En línea] Enero de 2013. http://www.wisegeek.com/what-is-a-cyberattack.htm.

7. **Harris, Shon.** *All in One CISSP Exam Guide.* s.l. : McGraw-Hill, 2008.

8. **Ponemon Institute.** *2012 Cost of Cyber Crime Study:.* s.l. : Ponemon Institute LLC, 2012.

9. **Hewlett-Packard.** *HP 2012 Cyber Risk Report.* 2012.

10. **The Open Source Vulnerability Database.** The Open Source Vulnerability Database. *The Open Source Vulnerability Database.* [En línea] 2012. http://www.osvdb.org.

11. **Williams, Jeff y Wichers, Dave.** *OWASP Top Ten - 2013.* s.l. : Creative Commons (CC) Attribution Share-Alike, 2013.

12. **Puppy, Rain Forest.** Phrack. *NT Web Technology Vulnerabilities.* [En línea] 24 de Diciembre de 1998. http://www.phrack.com/issues.html?issue=54&id=8#article.

13. **Clarke, Justin.** *SQL Injection Attacks and Defense, 2nd Edition.* s.l. : Syngress, 2012.

14. **van der Stock, Andrew.** *Una Guía para Construir Aplicaciones y Servicios Web Seguros.* s.l. : Free Software Foundation, 2005.

15. **Hernández Sampieri, Roberto, Fernández Collado, Carlos y Baptista Lucio, Pilar.** *Metodología de la investigación.* Naucalpan de Juárez, Edo. de México : McGRAW - HILL INTERAMERICANA DE MÉXICO, S.A. de C.V., 1991.

16. **OWASP.** *OWASP Application Security Verification Standard (ASVS).* s.l. : Creative Commons (CC) Attribution Share-Alike, 2009.

17. **Chandra, Pravir y Deleersnyder, Seba.** *OWASP Software Assurance Maturity Model.* s.l. : Creative Commons (CC) Attribution-Share Alike, 2012.