Universidad del Istmo de Guatemala
Facultad de Ingeniería
Ing. en Sistemas y Ciencias de la Computación
Seguridad
Dylan Gabriel Rodas Samayoa – rodas171315@unis.edu.gt

SQL INJECTION

Parcial I

Casi todas las organizaciones tienen presencia en la web. Algunas organizaciones usan sus sitios web para ofrecer servicios o vender productos a clientes en línea. Organizaciones como las instituciones educativas tienen portales en línea para ayudarlos a administrar la información y mostrarla de varias maneras a diferentes usuarios. Los hackers comenzaron a apuntar a sitios web y sistemas basados en la web desde ya hace mucho tiempo, pero en ese entonces era solo por el placer del prestigio. Hoy en día, los sistemas basados en la web contienen datos muy valiosos y confidenciales.

Los hackers buscan estos datos para robarlos y venderlos a terceras partes o pedir un rescate por grandes sumas de dinero. No sería de extrañar, que los competidores recurran a los hackers para forzar los sitios web de sus competidores y dejarlos fuera de servicio. Hay varias formas en que los sitios web pueden verse comprometidos. La siguiente vulnerabilidad es una de las más comunes, de hecho, es la tercera vulnerabilidad más crítica en cuanto a riesgos de seguridad de las aplicaciones web se trata, según la organización de OWASP durante el 2,021.

SQL Injection

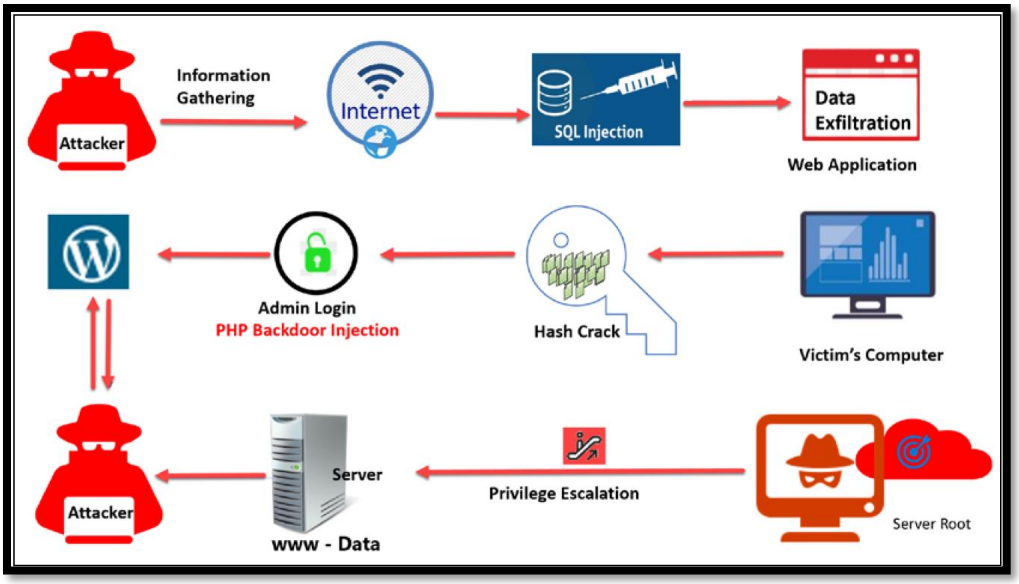
Se le denomina así, a todo tipo de ataque de inyección de código que tiene como objetivo la ejecución de los inputs proporcionados por los usuarios en el backend de los sitios web codificados en cualquier lenguaje de programación que se encuentre conectado a una base de datos bajo sintaxis SQL. Puede ser un ataque obsoleto en un sitio web bien sanitizado, pero algunas organizaciones son demasiado descuidadas o no se preocupan adecuadamente del mantenimiento y actualización

de su sitio web corporativo. Esto puede tener dos significados, primero: las organizaciones no examinan continuamente sus sistemas y, por lo tanto, el sitio web puede implantar algo que luego puede explotarse en un futuro cercano, y segundo: las organizaciones emplean desarrolladores que no siguen las pautas del código seguro y, como resultado, sus sitios web creados siguen siendo vulnerables.

Algunas organizaciones incluso ejecutan sitios web antiguos que siguen siendo vulnerables a este ataque. Los hackers suministran inputs que pueden manipular la ejecución de declaraciones SQL, lo que provoca que se produzca un compromiso en el backend y exponga incluso toda la base de datos. Las inyecciones de SQL se pueden usar para leer, modificar o eliminar bases de datos y su contenido. Para ejecutar un ataque de inyección SQL, un hacker debe crear un script SQL válido e ingresarlo en cualquier campo de entrada que genere una petición al backend.

Los ejemplos más comunes incluyen adicionar a la consulta el siguiente texto: *"OR "1"="1* y *"OR "a"="a*, que tergiversan a los códigos de SQL que se ejecutan en el backend. Esencialmente, lo que hacen los scripts anteriores es finalizar la consulta esperada y lanzar una declaración válida. Si fue en un campo de inicio de sesión, en el backend, los desarrolladores habrán codificado las sentencias SQL y del lenguaje de programación para verificar si los valores que el usuario ingresó en los campos de nombre de usuario y contraseña coinciden con los de la base de datos. En cambio, el script *'o '1'='1* le dice al SQL que finalice la comparación o que verifique si uno es igual a uno. Un hacker puede agregar un código aún más malicioso con comandos como *select* o *drop,* lo que puede hacer que la base de datos arroje su contenido o elimine tablas, respectivamente.

La siguiente ilustración demuestra el flujo de secuencia sobre cómo ocurre un ataque de inyección SQL. Resulta de gran interés observar la herramienta WordPress dentro de la ilustración, pero lo anterior, no es una simple conveniencia gráfica. En la actualidad, dicha herramienta abarca una buena parte de los sitios web convirtiéndola en la opción de CMS más utilizada del mercado o por lo menos, de las alternativas más populares para la creación de sitios web y generando una gran compañía que basa toda la programación de sus aplicaciones en el lenguaje PHP.



SQL Injection Scanners

Existen herramientas automatizadas que permiten agilizar el arduo proceso de evaluar si un sitio web es seguro contra inyecciones de SQL, desde herramientas en línea que comprueban vulnerabilidades en caliente hasta ejecutables para corroborar entornos locales más controlados. El sitio web de *pentest-tools.com* posee un escáner en línea que busca vulnerabilidades en aplicaciones web y encuentra SQL Injection, XSS, Server Side-Request Forgery, Directory Traversal y otros, además de problemas de configuración del servidor web. Esta herramienta de seguridad de aplicaciones web ejecuta comprobaciones de seguridad integrales del sitio web que detectan Log4Shell, OWASP Top 10 y más vulnerabilidades de alto riesgo.

Por otra parte, se encuentran herramientas como SQLI-SCANNER o SQLMAP, ambos son programas muy versátiles que pueden ayudar a escanear varios sitios web desde un archivo para ver si son vulnerables a la inyección de SQL. Las herramientas están diseñadas para listar *url's* mediante el uso de múltiples procesos de escaneo. Como resultado, los escaneos se ejecutan muy rápidamente, vienen con un potente motor de detección, muchas funciones de nicho para *pentesters* y una amplia gama de interruptores que van desde la toma de huellas dactilares de la base de datos, la obtención de datos de la base de datos, hasta el acceso al sistema de archivos subyacente y la ejecución de comandos en el sistema operativo a través de conexiones fuera de banda.

Métodos de mitigación del ataque SQL Injection

1. Lo primero que se debe hacer para evitar ataques de inyección es validar correctamente los inputs. En el lado del servidor, esto se puede hacer escribiendo las propias rutinas de validación, aunque la mejor opción es usar las propias rutinas de validación del lenguaje, ya que son más utilizadas y probadas. En el lado del cliente, la validación se puede lograr mediante la creación de funciones de validación de JavaScript, utilizando expresiones regulares.

2. Para la inyección de SQL, también es útil evitar la concatenación de valores de inputs a las consultas. En su lugar, se deben usar consultas parametrizadas, también llamadas declaraciones preparadas.

3. Siguiendo un enfoque de defensa en profundidad, también es útil restringir la cantidad de daño que se puede hacer en caso de que una inyección tenga éxito. Para ello, se utiliza un usuario del sistema con pocos privilegios para ejecutar la base de datos y los servidores web. Asegurar que el usuario al que las aplicaciones permiten conectarse al servidor de la base de datos no sea un administrador de la base de datos.

4. Deshabilitar o eliminar los procedimientos y comandos almacenados que permiten a un atacante ejecutar comandos del sistema o escalar privilegios.

5. Para inyección de SQL, utilizar siempre consultas parametrizadas o preparadas en lugar de concatenar sentencias y entradas de SQL. Las consultas parametrizadas insertan parámetros de función en lugares específicos de una oración SQL, lo que elimina la necesidad de que los programadores construyan la consulta ellos mismos mediante concatenación.

6. Además de realizar una validación correcta, también es necesario reducir el impacto del ataque en caso de que alguien logre inyectar algún código. Esto se hace configurando correctamente los privilegios de un usuario en el contexto de un sistema operativo para un servidor web y tanto para la base de datos como para el sistema operativo en el contexto de un servidor de base de datos.

7. Los Web Application Firewalls (**WAF**) pueden detectar ataques a la capa de aplicación, como inyección SQL, secuencias de comandos entre sitios, etc. Un WAF puede ayudar a detectar, denegar, interrumpir y degradar los ataques a la capa de aplicación. Algunos ejemplos de WAF incluyen:

- Amazon Web Services
- Barracuda
- Cloudflare
- F5
- Imperva
- Microsoft
- Oracle

8. **PHP-IDS** es un popular sistema de detección de intrusos (IDS) de PHP, también conocido como firewall de aplicaciones web (WAF). PHP-IDS funciona filtrando cualquier entrada proporcionada por el usuario contra una lista negra de código potencialmente malicioso. El IDS no elimina, desinfecta ni purifica ninguna entrada maliciosa, simplemente reconoce cuando un atacante intenta acceder al sitio web y reacciona exactamente de la manera que se desee.

Con base en un conjunto de reglas de filtro aprobadas y ampliamente probadas, a cualquier ataque se le otorga una calificación de impacto numérico que facilita decidir qué tipo de acción debe seguir al intento de hackeo. Esto puede variar desde un simple registro hasta el envío de un correo de emergencia al equipo de desarrollo, mostrar un mensaje de advertencia para el atacante o incluso finalizar la sesión del usuario. PHP-IDS permite ver quién está atacando el sitio web y cómo, todo sin el tedioso rastreo de archivos de registro o la búsqueda de foros de hackers para su dominio.

9. La herramienta más útil cuando se trata de validación de datos son las expresiones regulares. Cabe mencionar que facilitan mucho el trabajo de un *pentester* cuando se trata de procesar y filtrar grandes cantidades de información.

10. Por último, la implementación de la validación de parámetros puede ayudar a proteger contra ataques de inyección SQL, XSS e incluso DDoS.

Las implicaciones de no atender ataques de magnitud tipo SQL Injection

Ha habido varias lecciones dolorosas en el pasado reciente que se pueden usar para mostrar por qué las organizaciones deben tomarse en serio las respuestas a incidentes ante ataques cibernéticos. Incluso empresas de muy alto perfil pueden ser víctimas de protocolos de respuesta deficientes y el mismo destino puede ocurrirle a las PYME. Algunas de las razones clave por las que las organizaciones deben desarrollar planes de protección efectivos se detallan a continuación.

Protección de datos

Dichos datos confidenciales en las manos equivocadas podrían usarse como rescate, y los hackers solicitan grandes cantidades de dinero a cambio de no divulgar los datos al público o venderlos a otros actores malintencionados. Un sitio web bien sanitizado se puede utilizar para mitigar el robo de datos e intervenir en circunstancias en las que los hackers ya han extraído datos de la empresa. Las medidas podrían incluir detección de actividades maliciosas, administración estricta de identidades y accesos, copias de seguridad, registros de auditoría, alertas de seguridad de monitoreo e incluso cómo frustrar los intentos de transferir datos que se puedan haber robado, por ejemplo, encriptarlos.

Proteger la reputación y la confianza del cliente

Las personas prefieren realizar transacciones con empresas que les brinden una sensación de seguridad y responsabilidad por la privacidad. Como resultado de esto, muchos hackers que han incurrido en el pasado han degradado la valoración y la rentabilidad de las empresas afectadas. Por ejemplo, el caso de los hackeos a Yahoo demuestra el hecho de que las violaciones de datos tienen el efecto de reducir la valoración de una empresa.

Además, las brechas también dañan la confianza del cliente. Los clientes llevarán fácilmente su negocio a otras empresas si su preferida se ve afectada por una violación de datos. Es sabido que el efecto de las violaciones de datos es más pronunciado en las empresas que cotizan en bolsa. Una vez que se ha hecho pública

una brecha, las acciones de la organización afectada siempre caen drásticamente. Como por ejemplo Sony, cuando sus precios de acciones se desplomaron después de que fueron pirateados. Esto implica que las brechas afectan la confianza de inversionistas y accionistas. Por lo tanto, si una brecha no se maneja con atención, rapidez y cuidado, la empresa perderá su valoración y su base de clientes.

Proteger los ingresos

Claramente, las brechas cibernéticas son costosas y las organizaciones tienen mucho que perder cuando ocurren tales eventos. Los ingresos siempre están en juego, ya que inevitablemente se perderán directamente a través del ataque o indirectamente como parte de los gastos destinados a resolver el ataque. Los costos de cualquier ataque incluyen los montos robados por los hackers, los honorarios legales, los costos de remediación, los costos de investigación, las multas y la pérdida de negocios frente a los competidores. Un sitio web sanitizado puede ayudar a proteger los ingresos de una organización durante las brechas cibernéticas. El sitio puede tener algunos medios para prevenir la pérdida de dinero a través de herramientas de seguridad proactivas y políticas estrictas de transferencia de dinero.

REFERENCIAS

DIOGENES, Yuri y OZKAYA, Erdal. *Cybersecurity – Attack and Defense Strategies*. 2ª. Ed. Birmingham, UK: Packt Publishing Ltd, 2019. 635 p.

NAJERA-GUTIERREZ, Gilberto. *Kali Linux Web Penetration Testing Cookbook*. 2ª. Ed. Birmingham, UK: Packt Publishing Ltd, 2018. 394 p.

OZKAYA, Erdal. *Incident Response in the Age of Cloud*. 1ª. Ed. Birmingham, UK: Packt Publishing Ltd, 2021. 623 p.

RAINS, Tim. *Cybersecurity Threats, Malware Trends, and Strategies*. 1ª. Ed. Birmingham, UK: Packt Publishing Ltd, 2020. 429 p.