“ANÁLISIS DE LOS ATAQUES A APLICACIONES WEB SQL INJECTION Y CROSS SITE SCRIPTING Y SUS MEDIDAS DE PRECAUCION Y DEFENSA "
2015-2016

Edgar Subía Ponce
Universidad Técnica del Norte, Carrera de Ingeniería en Sistemas Computacionales, Universidad Técnica del Norte, Avenida 17 de Julio 5-21, Ibarra,
Imbabura, Ecuador.
epsubia@utn.edu.ec

Resumen. Desde el descubrimiento del SEQUEL en el año de 1974, se dio origen al lenguaje que especifica las características de las base de datos que manejaban el modelo relación, es que con este descubrimiento nace el SQL Inyección un ataque que permite encontrar fallas de seguridad y poder modificar casi porte completa de la base de datos, así mismo Cross Site Scripting surge para poder inyectar código malicioso del lado del cliente como del servidor causando gran daño a empresas. El presente trabajo tiene como fin dar a conocer el funcionamiento de los ataques y como poder prevenir las aplicaciones web ante vulnerabilidades encontradas.

Palabras Claves

Seguridad Informática, SQLi Injection, Cross Site Scripting, Ciberdelincuencia, Ethical Hacking, Hacking

Abstract.
Since the discovery of SEQUEL in the year 1974, it gave rise to language that specifies the characteristics of the database that handled the relationship model, is that with this discovery comes the SQL injection attack to find security flaws and power modify almost full bearing of the database, also arises Cross Site Scripting to inject malicious code on the client side and the causing great damage to businesses server.
This paper aims to publicize the operation of the attacks and how to prevent web applications against vulnerabilities found.

Keywords
Security, SQLi Injection, Cross Site Scripting, Cybercrime, Ethical Hacking, Hacking

1. Introducción

A principios de los años 90s la Web tenía un formato muy sencilla y simple que no era nada más que documentos de texto planos que se enlazaban entre sí mediante hipervínculos, en la actualidad conforme avanza la tecnología, los simples documentos se han ido convirtiendo en grandes aplicaciones capaces de interactuar con el usuario permitiendo manejar y almacenar grandes cantidades de información de cualquier tipo, este avance ha traído en si problemas de seguridad en los contenidos que se publicaban, especialmente en aplicaciones que ofrecían servicios que requerían información muy sensible del cliente como lo eran tarjetas de crédito, datos personales, etc.

Por tal motivo se recurre a la seguridad informática que se encarga de proteger los datos de manera óptima con estándares, protocolos y reglas facilitando minimizar los posibles riesgos en la información. De igual manera con el paso del tiempo nace el Hacking Ético que ayuda a realizar pruebas de testeo en sistemas informáticos permitiendo descubrir las deficiencias y fallas que el desarrollador comete en la codificación de programas y configuración de servidores que permite al atacante comprometer la seguridad en una aplicación web, también ayuda a comprender de mejor manera las técnicas o métodos más utilizados por los hackers, tratando de poder resolver las vulnerabilidades que existen en las aplicaciones en un entorno web, a lo largo de este estudio se comprenderá y se podrá diferenciar a un pirata informático del verdadero rol de un Hacktivista[1].

1.1 Materiales y Métodos

El uso de metodologías de testeo (pentesting) son muy esenciales, debido a que ayudan a evaluar las debilidades de los sistemas informáticos, en este caso en aplicaciones web, Consiste en un modelo que reproduce intentos de acceso de un potencial intruso desde los diferentes puntos de entrada que existan, tanto internos como remotos, a cualquier entorno Informático. Permite detectar vulnerabilidades en los Sistemas Informáticos y corregirlas en forma rápida y eficaz. A los PenTest también se los denomina Hacking Ético o Test de Intrusión.

1.3 Metodologías a utilizar

Para la elaboración del análisis se ha optado por trabajar con dos metodologías la una que sirve para el desarrollo del demo, y la otra sirve para las pruebas de intrusión en los sistemas.

La metodología de desarrollo que se utiliza para la elaboración del demo es una metodología ágil llamada XP, dicho método se trabaja en parejas y con un grupo máximo de 10 personas en donde se aplican las 4 fases que son:

[1] **Hacktivista.-** Es toda actividad hacker motivada por fines políticos o sociales

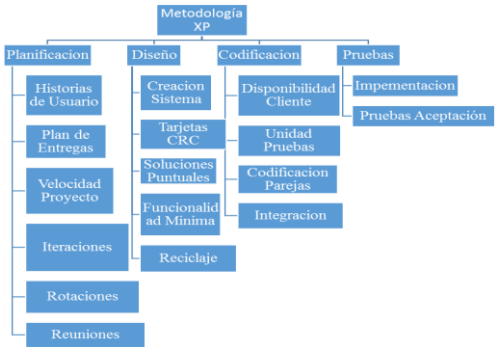

Ilustración 1Metodología XP

Fase 1: Planificación del Proyecto.

Historias de Usuario.- Se recolecta todas las requerimientos del cliente pero en lenguaje PNL[2] y sin detalles, se usa para poder estimar el esfuerzo y el tiempo que tomara desarrollar la aplicación.(Gonzalez, 2012)[1]. Son usadas en la fase de pruebas para observar y verificar el cumplimento del desarrollo.

Roles.- Los roles en este trabajo se los ha definido según (Beck, 1999)[2] que ya fueron tratados en temas anteriores.

Plan de Entrega.- Se exponen las historias definidas que se usaran para cada versión del programa, en esta fase toman un papel importante el cliente y el programador ya que ambos deciden el tiempo para la implantación del sistema

Iteraciones.- Se debe definir iteraciones de 3 semanas de duración del aplicativo aproximadamente, al inicio de cada iteración los clientes seleccionan las historias de usuario de acuerdo al plan de entrega, estas historias son divididas en tareas que por lo general dura de 1 a 3 días.

Velocidad del Proyecto.- Se emplea la velocidad para poder contralar que todas las tareas se ejecuten en el tiempo indicado por cada iteración.

Rotaciones.- La metodología estudiada aconseja a trabajar conjuntamente ya que ayuda a mejor la calidad y productividad del sistema desarrollado.

Reuniones Diarias.- Es aconsejable realizar reuniones diarias para intercambiar dudas e ideas para que la aplicación no sufra cambios al ya finalizar el trabajo.

Fase 2: Diseño.-

Diseño simple.- Elaborar un diseño simple y sencillo, ayuda a entender y reducir tiempo, tratando de evitar mucho esfuerzo en el desarrollo.

Riesgos.- Para evitar que existan riesgos, la metodología propone trabajar en pareja.

Re factorizar.- revisar una y otra vez el código para optimizar su rendimiento.

Fase 3: Codificación.- En esta fase el cliente juega un papel importante ya que la mayor parte del tiempo debe pasar y estar pendiente del proyecto, el cliente es quien agrega el tiempo máximo en el que se presenten avances.

Fase 4: Pruebas.- se realizan dos tipos de pruebas, la primera que corresponde al funcionamiento de cada versión validando y verificando el cumplimiento de las historias de usuario, y el segundo llamado test de aceptación en donde el cliente o usuario verifica que su funcionamiento.

En la metodología de pentesing llamado OTP (Owasp Top Project), Se realizan algunas fases las que son:

Recolección de Información.- Es una etapa que demanda mucha dedicación y tiempo ya que mayor sea la cantidad y la calidad de la información recauda, existirá la posibilidad de encontrar vulnerabilidades o backdoors, esta etapa es la integral dentro de cualquier metodología de testing de caja negra donde se asume que el atacante no cuenta con información técnica detallada que se utilizado para la construcción del sistema. Para lograr conseguir toda la información y su plataforma utilizada existen una variedad de recursos como pueden ser: inspección automatizada o manual de robots, uso de motores de búsqueda (google, Firefox), reconocimiento de parámetros GET/POST, descubrimiento detallado de la aplicación plataforma utilizada y lenguajes de programación.

Gestión de la configuración.- Permite realizar un análisis de la arquitectura y topología de la aplicación, el objetivo principal en esta etapa es el servidor en que se encuentra alojado la aplicación, algunas pruebas que se realizan en esta etapa son: verificación sobre los certificados tanto SSL/TLS, Verificación sobre el servicio de Base Datos (listener de conexiones) con el fin de buscar cualquier tipo de mala configuración o "exposición" de datos sensitivos, Verificación de extensiones de las paginas (*.php, *.asp, *.jsp, etc.) además de verificación sobre los tipos MIME[3] soportados por el servidor web

Pruebas de Autenticación.- Se realiza este tipo de pruebas en el caso de que la aplicación contenga formularios de accesos privilegiado para cada usuario se pueden realizar pruebas como por ejemplo: Verificación de credenciales viajando sobre canales cifrados y no cifrados, Si la aplicación permite la opción de "Recordar Password", analizar el comportamiento de dicha funcionalidad así como también analizar el comportamiento de otras funcionalidades relacionadas, Prueba de Cierre de Sesión y Gestión de Cache de Navegación.

Gestión de Sesión.- En esta etapa interviene un campo muy importante que es el protocolo HTTP ya que sirve para que interactúe el usuario con el servidor y se almacene toda la información que el usuario proporcione, es aquí donde el tester realiza pruebas para romper y obtener las sesiones de los usuario, las pruebas relacionadas a esta etapa son: Pruebas sobre los atributos de las cookies generadas

[2] **PNL.-** Programación Neurolingüística es una técnica usada para la recolección de información por medio de la comunicación en lenguaje normal para hacer entender lo que se requiere al cliente

[3] **MIME (*Multipurpose Internet Mail Extensions)*. -** Son especificaciones dirigidas al intercambio por medio de internet cualquier tipo de archivos sean estos texto, audio, video.

para mantener el estado de las sesiones en el lado del cliente, pruebas de XSS[4]

Pruebas de Autorización.- Es donde se consultan los roles, permisos y privilegios que tiene un usuario para poder acceder a diferentes recursos de la aplicación, las pruebas que se pueden realizar son: Prueba para evitar esquema de autorización, Pruebas de elevación de privilegios para determinar si es posible que un usuario pueda elevar su rol y acceder a recursos para los que inicialmente no debería acceder.

Pruebas de Lógica de Negocio.- Se emplea mucho tiempo y creatividad ya que es donde el tester o intruso verifica la funcionalidad del sistema realizándose preguntas como el que pasa si, es una de las pruebas más difíciles ya que no existen ayudas como herramientas automatizadas que ayuden a verificar las función del aplicativo, el propio Testeador debe hacer uso de sus conocimientos y habilidades para realizar este tipo de test.

Pruebas de Validación de datos.- Una debilidad muy común en las aplicaciones son las validaciones de las entradas que proporciona el cliente o a veces de la interfaz de la aplicación, es por esto que se deben tomar medidas de control sobre el tipo de dato que se ingresa en una aplicación, las pruebas más comunes que realizan son: inyecciones de código SQL e inyecciones de código JavaScript.

Pruebas de Denegación de Servicios.- Básicamente consiste en saturar con peticiones a la aplicación que se encuentra del lado del servidor. Las pruebas más usadas para este tipo de pruebas son el bloqueo de cuentas de usuarios, almacenamiento de demasiados datos en Sesión.

Pruebas de Servicios Web.- Las vulnerabilidades que existen en este tipo de pruebas van orientadas más a los archivos XML que son intercambiados con el cliente y el servidor, las pruebas que usan en este tipo de pruebas es: Prueba en la estructura del XML.

Pruebas Ajax.- Es una técnica que se usa mucho en páginas web para dinamizar las respuestas del lado del servidor, una técnica que se usa para este tipo de pruebas es la prueba de vulnerabilidades de Ajax, en concreto, las relacionadas con el objeto XMLHttpRequest.

Cabe mencionar que en esta sección de la metodología se utilizara las pruebas de validación a datos Inyección SQL y XSS.

[4] **XSS (Cross-Site Scripting).-** Es un tipo de vulnerabilidad donde se ejecuta código de JavaScript para poder obtener información, inicio de sesiones etc.

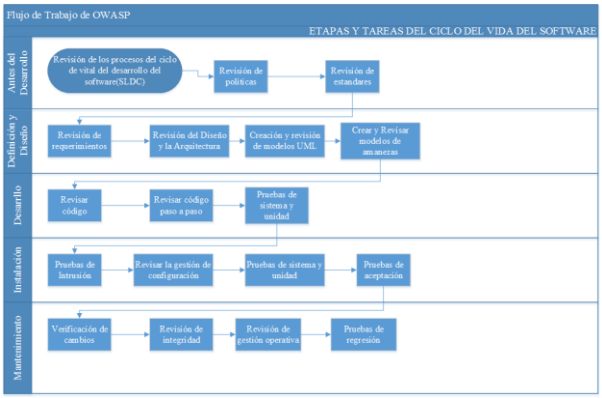

Ilustración 2 Test de Seguridad en el ciclo de desarrollo de una aplicación

2 Herramientas de Desarrollo

2.1 Kali Linux

Es una distribución Linux gratuita basada en Debian y desarrollado por una organización llamada Offensive Security expertos en auditoria y seguridad informática, es un proyecto que se creó a través de otro proyecto llamado backtrack que actualmente está fuera de servicio debido a que se unió con kali. Kali, actualmente consta con más de 300 programas destinados a la realización y verificación de ataques, básicamente a lo que corresponde con seguridad informática y no para delitos informáticos, siendo algunas de las más conocidas Nmap (un escáner de puertos), Wireshark (un sniffer), John the Ripper (Un crackeador de passwords) y la suite Aircrack-ng (Software para pruebas de seguridad en redes inalámbricas), además del framework Metasploit, que sirve para encontrar vulnerabilidades en sistemas informáticos especialmente enfocados a la web.

2.2 Leguaje PHP

Es un lenguaje de alto nivel y se ejecuta del lado del servidor y fue creado para la implementación de sitios web estáticos, interactivos y dinámicos se puede crear una variedad de páginas web deicidio a que posee una gran cantidad de librerías, entre su funcionalidad permite conectarse con base de datos relacionales.(Anabell, 2004)[3]

2.3 Servidor Apache

Es un programa que se ejecuta al lado del servidor, es de código abierto y se lo puede ejecutar en diversas plataformas como por ejemplo Linux, Microsoft y Mac. Cabe mencionar que este servidor es el más utilizado al nivel mundial debido a su gran estabilidad, versatilidad y confiabilidad.(Asensio Hildago, 2014)[4]

2.4 Framework Symfony

Es un framework PHP que utiliza un patrón de diseño MVC, Symfony es un completo framework diseñado para optimizar, gracias a sus características, el desarrollo de las aplicaciones web. Para empezar,

separa la lógica de negocio, la lógica de servidor y la presentación de la aplicación web.(Fabien Potencier, 2013).[5]

Este framework utiliza algunos componentes muy útiles para mejorar y optimizar el desarrollo y son los siguientes:

Tabla 1 Componentes de PHP y Symfony

COMPONENTES	DESCRIPCION
ORM	Es un objeto relacional mapeador que se utiliza en la programación para transformar en clases, atributos y datos, todas las tablas columnas y relaciones de una base de datos.
DOCTRINE	Es una librería imitada de hibernate, que ayuda a crear una capa de persistencia para trabajar con objetos en php.
DQL	Es un lenguaje de consulta basado en doctrine, es semejante a las sentencias de SQL y se emplea para obtener objetos en lugar de tablas.
YAML	Formato de serializacion de archivos que trabaja con datos nativos de lenguajes de programación

Fuente: Autoría Propia

2.5 Base de datos Postgresql

Es un motor de base de datos DBMS[5] que sirve para la creación y acceso a los datos, se compone de un lenguaje DDL[6], de un DML[7] y un SQL[8], y necesita de la ayuda de un intérprete como lo es PGADMIN que sirve para interactuar y con la base de datos.

POSTGRES fue pionera en muchos conceptos que sólo estuvo disponible en algunos sistemas de bases de datos comerciales mucho más tarde. (Momjian, 2014)[6]

3 Vulnerabilidades

Una vulnerabilidad es una debilidad de un sistema ya sea software, hardware, datos, estos últimos son los más importantes dentro de una organización porque estos no se los puede reparar y a los otros componentes si se los puede reparar, cuando existe debilidades un atacante puede quebrantar la integridad y cometer fraudes, alterar aplicaciones, una vulnerabilidad ocurre por un bug en la aplicación o algún fallo en la codificación o a veces por descuidos del usuario al recordar sus contraseñas en navegadores.

3.1 Sql Injection

Es un método muy utilizado a nivel mundial para detectar anomalías en aplicaciones web, este método consiste en la comunicación con la base de datos mediante sentencias SQL a través de un navegador web, cuyo objetivo es extraer información posible que se encuentra almacenado en una base de datos.(De La Quintanaillanes, 2013).[7]

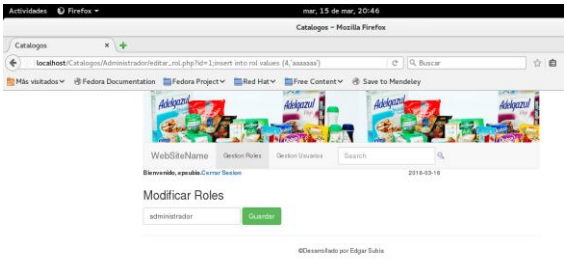

Ilustración 3 Inyección con dato numérico

3.2 Cross Site Scripting

Este ataque como el anterior estudiado es muy potente y peligroso ya que es explotado del lado del cliente y no del servidor, por lo que la seguridad de usuario queda muy vulnerable y expuesta a varios fraudes. Este método consiste en inyectar código HTML o JavaScript en una aplicación web, cuyo objetivo es consiste en que el navegador del usuario ejecute el código inyectado al momento de ver la página alterada. Comúnmente el XSS se utiliza para causa una acción indebida en el navegador de un usuario, pero depende del ataque de XSS que se realice, se puede explotar el fallo de un servidor o de la aplicación en sí. El XSS se puede utilizar para hacer phishing,[9] robo de credenciales, troyanizar navegadores, o simplemente para hacer un deface[10], todo depende de la página.

[5] **DBMS.-** Sistema de administración de base de datos.

[6] **DDL(Data Definition Language).-** describe las estructuras de información y programas que usan para construir y actualizar información que contiene la base de datos

[7] **DML(Data Manipulation Language).-**es utilizado para escribir programas que creen, extraigan y actualizasen información.

[8] **SQL(Structured Querry Language).-** sirve para extraer información de la base

[9] **Phishing.-** Es un ataque que suplanta la identidad de alguna persona con el fin de cometer un delito ya sea este robando contraseñas, tarjetas bancarias etc.

[10] **Deface.-** Es cuando se produce algún

Existen dos clases de XSS

- Persistente.- Este tipo de ataques se dan cuando existen huecos de seguridad, el servidor genera una página instantánea de resultados de acuerdo a información proporcionada, un ejemplo puede ser en los campos de búsqueda.

- Reflejado.- La información que proporciona el usuario es almacenada en la base de datos, en archivos de sistema o cualquier otro lugar que cause mucho daño, luego esa información es mostrada a otros usuarios para que visiten la página por eso se le conoce a este ataque como persistente. Con este tipo de ataques lo que se puede realizar es robar cookies, un ejemplo en donde se puede encontrar estas vulnerabilidades son los foros de discusión.

Ilustración 4 Inyección XSS en validaciones de texto

Conclusiones

- Aplicar una buena metodología de pentest que ayude a verificar los riesgos y vulnerabilidades, y corregir los errores más comunes que se comente al momento de programar.
- Las vulnerabilidades estudiadas causan mucho daño y afectan a toda la capa de la aplicación causando pérdidas de información vital para las empresas.
- Utilizar herramientas de testeo ya sean libres o de pago sirven de mucha ayuda para determinar vulnerabilidades.
- Tener una lista o bitácora bien argumentada de la situación actual de la empresa sirve para darse cuenta en qué estado se encuentra y en qué estado de vulnerabilidad se encuentra.

Agradecimientos

Gracias a la Universidad Técnica del Norte por haberme permitido ser parte de esta casona universitaria y poder formarme como persona con excelentes valores y a ser un gran profesional.
Un agradecimiento muy especial al Ing. Mauricio Rea, Director de Trabajo de Grado por haberme brindado su guía en la elaboración del presente trabajo de grado.

cambio en la interfaz de una aplicación por un atacante.

Referencias

[1] GONZALEZ, C. **Análisis,Diseño,Desarrollo e Implementación de una Aplicación Web para la Automatización de Clientes,Vehículos,Facturación,Inventario y Campañas para Autoservicios RBS**. 2012. Departamento de Ciencias de la Computación, Escuela Politécnica del Ejercito, Sangolquí.

[2] BECK, K. Embracing Change with Extreme Programming. **Computer,** v. 32, n. 10, p. 70-77, 1999. ISSN 0018-9162.

[3] ANABELL, C. Java o PHP. **Revista Digital Universitaria,** v. Volumen, 2004. ISSN 1067-6079. Disponível em: < http://www.revista.unam.mx/vol.7/num12/art104/dic_art104.pdf >.

[4] ASENSIO HILDAGO, L. Seguridad en aplicaciones web: una visión práctica. 2014.

[5] FABIEN POTENCIER, F. Z. Symfony 1.4, la guía definitiva. 2013. Disponível em: < http://librosweb.es/libro/symfony_1_4/ >.

[6] MOMJIAN, B. Postgresql. California, p. Documentation, 2014. Disponível em: < http://www.postgresql.org/docs/9.3/static/release-9-3-5.html >.

[7] DE LA QUINTANAILLANES, M. M. SQL INYECTION. **Revista de Información, Tecnología y Sociedad**, p. 38-40, 2013. ISSN 1997-4044. Disponível em: < http://www.revistasbolivianas.org.bo/scielo.php?script=sci_arttext&pid=S1997-40442013000100017&nrm=iso >