ISSN: 2594 - 2905

IMPLEMENTACIÓN DE TÉCNICAS DE ENCRIPTACIÓN EN LA SEGURIDAD DE LAS BASES DE DATOS

IMPLEMENTATION OF ENCRYPTION TECHNIQUES IN THE SECURITY OF DATABASES

Juan Manuel Bernal Ontiveros[1], Francisco Zorrilla Briones[2], Marisela Palacios Reyes[3], Noé Ramón Rosales Morales[4], Verónica Farías Veloz[5]

[1]Maestría en Ciencias en Ingeniería Industrial, [2]Doctorado en Ciencias en Ingeniería Industrial,
[3]Maestría en Ciencias en Ciencias de la Computación, [4]Maestría en Software Libre, [5]Maestría en Software Libre
[1,2,3,4,5]Instituto Tecnológico de Ciudad Juárez, Departamento de Sistemas y Computación, Avenida Tecnológico No. 1340 Fraccionamiento El Crucero Código Postal 32500, Teléfono (656) 6882500, Ciudad Juárez, Chihuahua México
[1]jbernal@itcj.edu.mx, [2]fzorrilla@itcj.edu.mx, [3]mpalacios@itcj.edu.mx [4]nrosales@itcj.edu.mx, [5]vfarias@itcj.edu.mx

Resumen – A través del tiempo debido a la evolución de los ataques informáticos a cualquier sistema computacional, se ha tenido la necesidad de estar a la vanguardia de la seguridad informática apoyado en la aplicación de algún mecanismo que coadyuve en proporcionar protección a la información en las bases de datos. La seguridad de los datos puede ser proporcionada por dos tipos de niveles: seguridad física y seguridad lógica. La seguridad en las bases de datos se compone por el nivel lógico establecido como el nivel dos. Se quiere mencionar que los dos métodos no pueden ofrecer una solución totalmente satisfactoria al problema de la protección de las bases de datos, las causas principales son que cuando los datos sin procesar, existen en una forma entendible dentro de una base de datos, es imposible que se pueda comprobar el origen auténtico de ciertos datos.
Es cuando en estas situaciones emerge el cifrado de los datos y se usa como la técnica de ocultación de la información en el ámbito de la seguridad de las bases de datos para ser encriptadas. Los requisitos impuestos que deben ser implementados a un sistema de cifrado, debido por las características de las bases de datos, y un algoritmo de cifrado que cumpla con la mayoría de estos requisitos propuestos. Si bien es poco probable que el cifrado por sí solo proporcione protección a la información de una base de datos y especialmente cuando ningún algoritmo de cifrado no garantice la seguridad de la información.

Palabras Clave: Criptografía, Seguridad, Bases de Datos, Encriptación, Cifrado.

Abstract -- Over time due to the evolution of computer attacks on any computer system, there has been a need to be at the forefront of computer security supported by the application of some mechanism that helps provide protection to information in databases. Data security can be provided by two types of levels: physical security and logical security. Database security is integrated by the aforementioned level two. It is wanted to mention that the two methods cannot offer a totally satisfactory solution to the problem of database protection, the main causes are that when raw data exists in an understandable form within a database, it is impossible to verify the true origin of certain data.
These are where encryption excels, and there is a tool in the database security compendium for encryption. The requirements imposed must be implemented on an encryption system, due to the characteristics of the databases, and an encryption algorithm that meets most of these proposed requirements. While encryption alone is unlikely to provide protection to the information in a database and especially when no encryption algorithm does not guarantee the security of the information.

Key words **–** Cryptography, Security, Databases, Encryption.

INTRODUCCIÓN

La información en las bases de datos es un activo muy importante para cualquier empresa o institución ya sea federal, educativa, de salud etc. Por lo que en la evolución de los sistemas informáticos, se han creado software que apoya el manejo y control de acceso legal y apegado a las políticas de seguridad. Por lo que también han creado tanto técnicas como software mal intencionados que buscan la infiltración a través de vulnerabilidades para cometer algún delito cibernético.
Debido a los ataques e infiltraciones, ha surgido la necesidad de la aplicación de técnicas y mecanismos que implícitamente están contenidos en la seguridad para apoyar la defensa y protección de la información incluida en las bases de datos. Por tanto, la seguridad de los datos puede ser proporcionado por tres tipos de métodos distintos superpuestos: seguridad hardware, Seguridad de software (operativo, bases de datos y cifrado de datos) y seguridad de red [1].
Al aplicar los métodos ya mencionados, ya va implícita la seguridad de la base de datos. Se aclara que ninguno de estos métodos ofrece una solución totalmente satisfactoria al problema de seguridad de la base de datos.

Las razones son que los datos contenidos sin procesar son visibles en forma legible dentro de una base de datos y que es difícil que se compruebe el origen auténtico de un elemento de datos.
Se considera que las áreas en la cuales de utiliza la encriptación de datos es la de desarrollo de aplicaciones, bases de datos, y la implementación de software, además se consideran un arsenal en la seguridad para cifrar las bases de datos. Se deben cumplir requisitos impuestos a un sistema de cifrado en base a las características requeridas por las bases de datos y un algoritmo de cifrado que cumple con la mayoría de estos requisitos. Las probabilidades de que un algoritmo de encriptación sea eficiente en el resguardo de las bases de datos no es del cien por ciento (no garantiza que ningún algoritmo de encriptación sea seguro), pero se tiene que su función principal es el de resolver los problemas en aspectos como el acceso no autorizado de usuarios ajenos al sistema, consultas confidenciales de la información, eliminación de la información, entre otros.

Técnicas de desarrollo criptográficas para las bases de datos.
Se han desarrollado una gran diversidad de herramientas para la creación de la automatización de los *Algoritmos Criptográficos* en las pruebas y análisis de las bases de datos posibilitando la detección de debilidades y/o vulnerabilidades criptoanalíticas que pudieran ser explotadas por un atacante, mediante técnicas de criptoanálisis. A continuación se mencionan algunas de ellas:

El lenguaje de programación *PHP*: se apoya en su programación para WEB, con el Gestor de Bases de Datos *MySQL*, y con *phpMyAdmin* herramienta administrativa que a través de la web maneja de forma simple las bases de datos *MySQL* y con una interfaz amistosa o *HeidiSQL* de software libre, ligera y desarrollada para que sea utilizada en el sistema operativo Windows para su manipulación en *MySQL* y *Microsoft SQL*.

Además se contemplan otros entornos de desarrollo como: *Apache* servidor WEB, *Xampp* distribución de *Apache* que incluye diferentes softwares libres, *Wamp* entorno de desarrollo web para Windows, *Mamp* herramienta de desarrollo web gratuita, que pueden ser configuradas en un momento y *EasyPHP* paquete integrador fácil de instalar y configurar cuya función es instalar *Apache*, *MySQL*, *PHP* y *PhpMyAdmin* en una máquina que disponga del S.O. Windows 9x, 2k, XP y versiones posteriores.

PHP es un lenguaje de programación WEB y como es del conocimiento se compone de una gran capacidad de conexión con la mayoría de motores de Gestores de Bases de datos, aunado a esto es de destacar la conexión con gestores como *MySQL* y *Postgre*. Además, existen muchos IDEs (Integrated Development Environment) con la integración de diversos lenguajes y como muestra se mencionan algunos: Eclipse, Dreamweaver, ActiveState Komodo, IntelliJ IDEA, MyEclipse, Oracle JDeveloper, NetBeans, Codenvy y Microsoft Visual Studio [3].

Descripción del Problema
La seguridad de la información siempre ha sido de gran importancia y más cuando se habla de información geográficamente distribuida, lo que quiere decir que no se encuentra en un mismo lugar, sino que está distante en diferentes sitios geográficos que manejan múltiples plataformas. En el manejo de bases de datos es de vital importancia el control que se debe hacer sobre los accesos dentro de las mismas bases de datos, lo que quiere decir que los datos almacenados en ellas deben estar protegidos contra cualquier ataque, pérdida accidental, acceso malintencionado y no autorizado.

La encriptación de las bases de datos, es considerada como el escudo protector contra injerencias de accesos de infiltraciones de ataques cibernéticos. Si alguien no autorizado intenta infiltrarse realizando una solicitud de acceso a la base de datos, el sistema debe de verificar la información y, si no es correcta, el acceso debe ser denegado, claro es que cuando se solicita el acceso, los datos deben estar en forma encriptada. Esto se realiza por el motivo de no visualizar la contraseña de la cuenta de usuario.

Filtración de datos Confidenciales de las Bases de Datos
Un ejemplo muy conocido de infiltración indebida fue la de *LinkedIn* donde el ataque se dio en el año 2012 cuando se descubrió que 6,5 millones de contraseñas asociadas a esta plataforma de red social profesional (ataque del tipo Hash *SHA-1*() sin el uso del argumento *sal*), existió una infiltración no autorizada de atacantes externos o espías a la compañía, las robaron y publicaron en un foro de atacantes piratas informáticos rusos. Después se supo en realidad la magnitud del daño, cuando en el año 2016 fue revelado el alcance real del ataque.

Los delincuentes informáticos vendieron los datos en la plataforma de red social *MySpace* de la cual ofrecían las contraseñas de las direcciones de correos electrónicos de un aproximado de 165 millones de usuarios de la red social profesional *LinkedIn*, por tan solo cinco bitcoins (alrededor de 2000 dólares de ese tiempo). Aunque *LinkedIn*, reconoció haberse enterado del ataque más tarde, debido a la violación de la información de su plataforma, por lo que la acción de correctiva fue la de restablecer las contraseñas de las cuentas afectadas [2].

iPSUMTEC
ISSN: 2594 - 2905

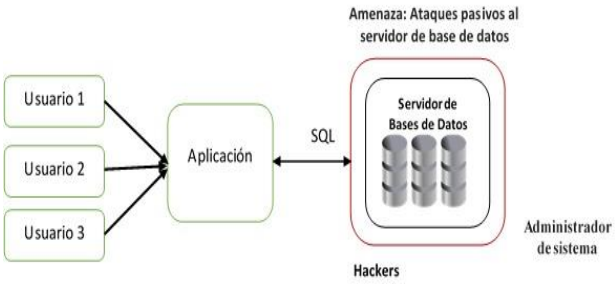

Figura 1. Procesamiento de consultas SQL en datos cifrados

Las bases de datos, dentro de su definición se consideran como contenedores de objetos, y dentro de estos están constituidos por datos, por los cuales siempre existe el interés de apropiarse ilegalmente de ellos. Se puede decir que la sensibilidad de los datos es una medida de la importancia de quienes son los propietarios o dueños de los datos, es por lo mismo de se debe indicar es imperante la necesidad de protección ante ataques.
Es importante detectar en las bases de datos y estar alerta sobre las vulnerabilidades o fallas que pueden romper con la confidencialidad, por lo que se muestra una lista de diferentes fallas que quebrantarían la seguridad de las bases de datos [5].

1. Gestión inadecuada de accesos

El acceso de los usuarios internos a una organización o institución, debe ser controlado, asignando los derechos adecuados, a las cuentas de los mismos, para que puedan realizar sus funciones cotidianas, y es así como las bases de datos, puedan estar protegidas y bloqueadas contra los ataques malintencionados. Los usuarios que accedan a las bases de datos deben estar contenidos como cuentas legalmente registradas, aunados a estos, también los administradores, deben tener un límite de acceso a redes y computadores. Las herramientas de seguridad como los firewalls impiden infiltraciones de direcciones IP no identificadas o no autorizadas accedan a las bases de datos. Un control de acceso demasiado restrictivo puede hacer que el trabajo de las actualizaciones de software retrase la solución de problemas. Aunque puede ser un fastidio, se debe implementar la restricción mereciendo el esfuerzo, porque acotará cualquier vía tomen las infiltraciones de los atacantes.

2. Datos no cifrados en reposo

La principal función de los algoritmos criptográficos, es la de cifrar y descifrar los datos, esto es con el fin de proteger los datos contra los ataques delictivos, es por tanto que la encriptación, es la técnica en la que se codifican o convierten los datos en un código no entendible. Se considera que estos algoritmos de protección pueden ser confiables aunque no al cien por ciento. Hay que tomar en cuenta que sus llaves deben ser protegidas con mucho cuidado y no ser visibles para cualquiera.

3. No Utilizar Algoritmos de Encriptación para la Protección de la Privacidad

El trabajo de los algoritmos criptográficos es el de apoyar en la ocultación de datos, y los cuales deben estar en un estado que sea indescifrable. Esto permitirá eliminar las preocupaciones sobre la privacidad de los datos. La criptografía es una tecnología bien conocida para proteger datos confidenciales. Cuando existe la combinación de los métodos de encriptación de clave pública y de encriptación de clave privada en el ocultamiento de los datos que son sensibles o confidenciales en la recuperación de texto plano encriptado, hace que sea más simple la distribución de las claves, otro aspecto es que es más seguro, ya que la clave privada nadie la conoce. Es de saber que los algoritmos de encriptación no son eficientes al cien por ciento en la resolución de los problemas, pero su efectividad es fuerte cuando no es necesario que todos los confidenciales estén disponibles en todo momento. Un ejemplo de ello, es el reemplazo de un nombre por un seudónimo aleatorio.

4. Falta de Control en las Bases de Datos

Los *Gestores de Bases de Datos*, cualesquiera que estos fuesen, deben contener herramientas de administración y sobre todo de la limitación de control de acceso. Por lo que al usar las aplicaciones apropiadas puedan ver las estructuras de las entidades (tablas) con sus respectivos datos. Debido a la seguridad informática es recomendable no utilizar la misma contraseña en todas las aplicaciones, por lo que la limitación de acceso a redes locales debe ser aplicada.

Revisión de Literatura

Los autores Ramez Elmasri y Shamkant B. Navathe en su libro *Fundamentos de los Sistemas de Bases de Datos*, describen los problemas de seguridad de las bases de datos incluidos los siguientes [6]:

- Los aspectos jurídicos y éticos que están relacionados con el derecho de acceso a la información: El mejor ejemplo expuesto en este punto es el que la información sensible considerada privada, no debe ser accedida de forma legal, sino existe una autorización permitida.
- La aplicación de la política en el gobierno, organizaciones privadas o corporativas, en cuanto al manejo de la información, no debe ser del uso público, tales como calificaciones crediticias, y registros médicos personales.
- Aspectos del Sistema, tales como los niveles en los que se debe fortalecer y ejecutar las funciones necesarias de seguridad, tal es el caso del manejo del nivel de hardware físico, el nivel del sistema operativo o el nivel de Sistema gestor de Bases de Datos (DBMS).

iPSUMTEC
ISSN: 2594 - 2905

- La exigencia de clasificar los niveles de seguridad categorizando los datos y usuarios en base a la función de estas clasificaciones, los ejemplos que expresan esta idea son: alto secreto, secreto, confidencial y no clasificado. Por esto es de aplicarse la política de seguridad en base a la autorización del acceso en varias categorías de datos.

Amenazas a las bases de datos: Los ataques a las bases de datos son riesgos causan la pérdida o degradación de la información con algunos o todos los objetivos de seguridad comúnmente aceptados: La integridad, disponibilidad y confidencialidad. Para proteger las bases de datos contra este tipo de amenazas, es común implementar cuatro tipos de medidas de control: control de acceso, control de inferencia, control de flujo y cifrado [6].

El gran desarrollo de las tecnologías de los sistemas de información, se han fijado nuevas metas en materia de seguridad de la información. Actualmente, mejorar la seguridad de las bases de datos se está convirtiendo en uno de los objetivos primordiales relacionado en el campo de la investigación y la industria de las bases de datos. Entre tanto, muchas organizaciones no pueden funcionar correctamente si su base de datos no funciona; normalmente se les conoce como sistema de *misión crítica*. Junto con la amplia aplicación de la base de datos surge la necesidad de su protección. El gasto, esfuerzo, tiempo y recursos ha sido enorme en el intento de que las bases de datos cumplan con todos los requisitos de seguridad y que incluyen [16]:

1. Evitar la divulgación no legal de la información.
2. Evitar de la alteración no legal de la información.
3. Evitar la denegación de servicio.
4. Evitar la penetración del sistema por parte de personas no autorizadas.
5. Prevenir el abuso de privilegios especiales.

Diseñar una base de datos que cumpla con los puntos anteriores en el aspecto de seguridad es muy difícil, ya que un sistema de base de datos procesa una gran cantidad de datos de manera compleja. El resultado es que la mayoría de los sistemas de bases de datos convencionales tienen fallas en el atacante puede usar para penetrar en la base de datos. Independientemente del grado de seguridad que se establezca, los datos confidenciales de las bases de datos siguen siendo vulnerables a los ataques.

Por lo tanto, un remedio es recurrir a la criptografía medio de almacenamiento de datos. Encriptar los datos de una base de datos, previene exponerlas a los espías o atacantes, inclusive si se evade el mecanismo de control de acceso. El encriptamiento de las bases de datos, garantiza la confidencialidad de los datos, protegiéndolos contra los ataques de intrusos externos como de usuarios internos maliciosos. Es compatible con casi todos los DBMS comerciales, incluidos Oracle, SQL Server y Mysql.

Método de la Criptografía de clave privada (llave simétrica)

Kerckhoffs es conocido por su principio de que los diseños criptográficos deben hacerse públicos. Sin embargo, en realidad enunció seis principios, los siguientes de los cuales es muy relevante para esta documentación [10]:

Un *cifrado* debe ser funcional y matemáticamente indescifrable.

El principio de Kerckhoff dice esencialmente que no es necesario utilizar un secreto perfecto en el esquema de encriptación, pero en su lugar es suficiente usar un esquema que no pueda ser descifrado en un "tiempo razonable" con cualquier "probabilidad razonable de éxito" (en el lenguaje de Kerckhoff, un esquema que es "prácticamente indescifrable"). En términos más concretos, basta con utilizar un esquema de cifrado que pueda (en teoría) ser roto, pero eso no se puede romper con una probabilidad mejor que 10^{-30} en 200 años usando la supercomputadora más rápida disponible [10].

El enfoque computacional incorpora dos relajaciones de la noción de seguridad perfecta:

1. La seguridad solo se preserva contra atacantes eficientes que se ejecutan en una forma factible, cantidad de tiempo, y
2. Los atacantes pueden potencialmente tener éxito con una probabilidad muy pequeña (que es lo suficientemente pequeño como para que no preocuparse con lo que realmente pueda suceder).

Para obtener una teoría significativa, se necesita definir con precisión lo que significa lo anterior. Hay dos enfoques comunes para hacerlo: el acercamiento concreto al enfoque asintótico. Se explica a continuación.

El enfoque concreto cuantifica la seguridad de un esquema criptográfico determinado, limitando explícitamente la probabilidad máxima de éxito de cualquier atacante en la ejecución de este ataque durante un período de tiempo específico como máximo. Esto es, dejar que t y $\mathcal{E}$ sean constantes positivas con $\mathcal{E} \leq 1$. Entonces, una definición concreta de seguridad, en términos generales, tomaría la siguiente forma:

Un esquema es seguro (t, $\mathcal{E}$) si todos los atacantes que intentan ejecutar una infiltración durante un tiempo máximo t tienen éxito en romper el esquema con una probabilidad de t en la mayoría $\mathcal{E}$.

iPSUMTEC
ISSN: 2594 - 2905

La definición de lo que significa "destrozar" el esquema. A continuación se describe una ejemplificación; se quiere tener la garantía de que un ataque llevado a cabo por un agresor, utilizando un algoritmo eficiente por un periodo máximo de 200 años usando la más veloz supercomputadora existente y puede tener éxito en quebrar el algoritmo de cifrado con una mayor probabilidad que 10^{-30}. O de otro modo es mejor la medición del tiempo en los términos de los ciclos de una computadora, y utilizar un algoritmo de tal manera que el agresor intente un ataque con un máximo a los 2^{80} ciclos y que su probabilidad sea mejor que 2^{-64}. Es explicativo para dar una muestra de los valores utilizados de t, $\mathcal{E}$ y que frecuentemente están incluidos dentro de los algoritmos modernos de la criptografía [10].

El enfoque asintótico es el enfoque, arraigado en la teoría de la complejidad, ver el tiempo de ejecución del atacante, así como su probabilidad de éxito como funciones de algún parámetro en vez de números concretos. Específicamente, un esquema de criptografía incorporará un parámetro de seguridad que es un número entero *n*.
Cuando las partes honestas inicializan el esquema (es decir, cuando generan claves), eligen algún valor *n* para el parámetro de seguridad; este valor se asume como conocido por cualquier atacante para tener acceso indebido al esquema. El tiempo de ejecución del agresor (y el tiempo de ejecución de las partes honestas) también como la probabilidad de éxito del agresor, se ven como funciones de *n* [10]. Entonces:

1. La homologación de la noción de "estrategias factibles" o "algoritmos eficientes" con algoritmos probabilísticos que se ejecutan en *tiempo polinomial en n*. (A veces se usa *PPT* para representar *tiempo polinomial probabilístico*). Significa que algunas constantes *a*, *c*, el algoritmo se ejecuta en tiempo $a \cdot n^c$ en parámetro de seguridad *n*. Es requerido que las partes honestas funcionen en tiempo *polinomial*, y por tanto sólo se tendrá preocupación de lograr la seguridad frente al *tiempo polinomial* de los agresores. Es importante recalcar que el atacante, aunque requiere ejecutar el *tiempo polinomial*, puede que sea mucho más poderoso (y ejecute mucho más tiempo) que las partes honestas. Además, las estrategias de los agresores requieren una cantidad de tiempo *super polinomial* por lo que no se consideran amenazas realistas (y por lo tanto son esencialmente ignoradas).
2. Igualamos la noción de "pequeña probabilidad de éxito" las probabilidades con éxito más pequeñas que cualquier *polinomio inverso* en *n*, lo que significa que para cada constante *c*, la probabilidad de éxito del agresor es menor que n^{-c} para valores suficientemente grandes de *n*. Una función que crece más lentamente que cualquier *polinomio inverso* se llama insignificante.

Por lo tanto, una definición de seguridad asintótica toma la siguiente forma general:

Un esquema es seguro si todos los éxitos del atacante contra PPT (*tiempo polinomial probabilístico*)) logra romper el esquema con sola una probabilidad despreciable.

Aunque muy limpio desde un punto de vista teórico (ya que en realidad se puede hablar de un esquema seguro o no), es importante entender que el *enfoque asintótico* solo "garantiza la seguridad" para valores suficientemente grandes de *n* [10].

Funciones para el registro de contraseñas.
El autor Ariel Maiorano en su libro *Criptografía: Técnicas de desarrollo para profesionales*, hace referencia a las funciones que se usan para llevar a cabo la encriptación de contraseñas disponibles en el motor de base de datos, se aclara, en primera instancia, que MySQL provee una función llamada *PASSWORD()*, pero ésta se utiliza para manejar los *passwords* o contraseñas de los usuarios del sistema (registrados en la columna *password* de la tabla *user* de la base de datos llamada *MySQL*, es decir, dentro del sistema de autentificación y validación del servidor MySQL) [8].

Describe en su documentación se sugiere no utilizarla en aplicaciones que sean propias. Se sugiere el uso de las funciones *MD5()* o *SHA1()*. Aunque no sea recomendada para usos distintos de la actualización o manejo de la información de autentificación de MySQL, con el ánimo de no omitir los detalles de una función relacionada con el tema en encriptación, se ejemplifica la función a continuación.
La función se define de la manera siguiente:

```
PASSWORD(cad_txt)
```

Lee la cadena de caracteres de un *password* o contraseña cifrada, que se muestra a partir de un texto plano de una contraseña en el parámetro *cad_txt*. La encriptación es de una sola vía (o sea no reversible).

Ejemplo:

```
SELECT PASSWORD('clave_secreta');
```

Maiorano aconseja que lo más adecuado para guardar *passwords* o contraseñas, de una forma rápida y fácil es el uso de la función *ENCRYPT( )*. En el gestor de bases de datos MySQL la integra para mantener acorde con los sistemas de contraseñas de los sistemas operativo Unix;

por otro lado en Windows, la función devolverá siempre NULL (Nulo). Su definición es como sigue:

```
ENCRYPT(cad_txt [,salt])
```

Encriptará la cadena de caracteres *cad_txt* usando la llamada al sistema (Unix) *crypt( )* y el resultado será la cadena encriptada. El parámetro que se muestra en la sintaxis *salt* es opcional deberá ser una cadena de, al menos con dos caracteres en la longitud. Si en la sintaxis no se especifica, el parámetro *salt* será aleatorio y también por lo tanto será utilizado.

Ejemplo de uso:

```
SELECT ENCRYPT('Contraseña');
```

A continuación se mostrarán las siguientes funciones que se utilizan en cuanto al grabar las contraseñas en las aplicaciones en MySQL. Éstas funciones son conocidas como *hashing* criptográfico o funciones de una vía que integran los algoritmos *MD5* y *SHA-1* llamadas funciones *MD5( ) y SHA1( )* o *SHA( )*, como se puede ver, las dos últimas funciones son sinónimos.
Si se da un vistazo al manual de MySQL hace referencia a una nota con respecto a los *exploits* (códigos o programas que aprovechan una vulnerabilidad o falla de seguridad), estos que son publicados respecto a los algoritmos *MD5* y *SHA-1*, y alerta el considerar la utilización de la alternativa *SHA-2*, disponible en la versión 6 del SGBD MySQL.

Hay que tomar en cuenta que aquí también se aplican las definiciones vistas relacionadas con el parámetro *salt* para contrarrestar la eficacia de los ataques por diccionario, aunque es responsabilidad del usuario quien usa el implementar alguna metodología que se apoye de este mecanismo.

```
MD5(cad_txt)
```

El uso de la definición de la función *MD5( )*. Calcula un resultado de 128 bits el cual es obtenido de la cadena de caracteres que es proporcionada como el argumento o parámetro en *cad_txt*. El valor obtenido es el de una cadena de 32 caracteres de longitud (ya que es convención, se presenta en el formato hexadecimal). Regresa un valor NULL (Nulo) solo y si su valor del parámetro sea NULL (Nulo). Si accedemos al manual de se describe y hace mención de que se trata de una implementación del algoritmo "*RSA Data Security Inc. MD5 Message-Digest Algorithm*".

Ejemplo:

```
SELECT MD5('Contenido de prueba');
```

La descripción de la función *SHA1( )* (o *SHA( )*, que se utiliza como sinónimo):

```
SHA1(cad_txt), SHA(cad_txt)
```

Da como resultado del algoritmo *SHA-1*, en base a RFC 3174 (*Secure Hash Algorithm*), de 160 bits de longitud. Obteniendo el valor como una cadena de 40 caracteres en forma hexadecimal. De la misma manera con la función *MD5( )*, si el parámetro es NULL (Nulo), la función devolverá un valor NULL (Nulo).

Ejemplos:

```
SELECT SHA1('Contenido de prueba');
SELECT SHA('Contenido de prueba');
```

Y la función *SHA2( )*, que se puede encontrar desde la versión 6.0.5 del gestor de bases de datos MySQL. Se le considera criptográficamente más segura que *MD5( )* y *SHA1( )* según Maiorano [8].

```
SHA2(cad_txt, hash_length)
```

El resultado de esta función es el *Hash de cifrado* (algoritmo matemático unidireccional, el cual su función es el de la asignación de datos de cualquier tamaño en una serie de bits de un tamaño fijo) utilizando cualquier algoritmo de la familia *SHA-2* (*SHA-224*, *SHA-256*, *SHA-384* y *SHA-512*). El primer argumento o parámetro es una cadena de caracteres de la cual se obtiene *hash* (cad_txt); el segundo parámetro es la longitud, en bits, del resultado obtenido, que se espera que sean los valores: 224, 256, 384 o 512. Si se obtiene algún parámetro como NULL (Nulo), o se indica una longitud de salida que no es válida, lo obtenido es un NULL (Nulo). Si no es así, entonces, muestra el valor del resultado, el cual es indicado en una cadena de caracteres en formato hexadecimal.

Ejemplo de uso:

```
SELECT SHA2('Contenido de prueba', 512);
```

Se quiere aclarar que esta función solo se ejecutará si el gestor de bases de datos MySQL fue configurado para el soporte del certificado digital que autentifica la identidad de un sitio web (Secure Sockets Layer o SSL)[8].

Método en el Encriptado Simétrico de Datos [8]**.**
El Gestor de Bases de datos MySQL provee, en la encriptación y la desencriptación simétrica, la aplicación de dos algoritmos: el *TripleDES* y *AES*. Estas están formadas a partir de un par de funciones (una para encriptar y otra para desencriptar) para cada uno de los dos algoritmos. Se describen a continuación.

```
DES_ENCRYPT(cad_txt [,{llave_num|llave_cad_txt}])
```

Su trabajo es el de encriptar una cadena de caracteres con su respectiva contraseña, llave que también en este caso,

iPSUMTEC
ISSN: 2594 - 2905

usa el algoritmo *triple-DES*. Si se llegase a encontrar error, regresa un valor NULL (Nulo). Hay que indicar que la función se ejecutará con éxito sólo y sí el gestor MySQL ha sido configurado para soportar el certificado para SSL (Secure Sockets Layer o Capa Sockets de seguridad). La contraseña o llave que se encripta es determinada en base al segundo argumento de la función *DES_ENCRYPT( )*: Haciendo notar que si no fue especificado ningún argumento, por default será usada la primera contraseña del archivo de contraseñas *DES*. Si el argumento es un número (0-9), entonces se utilizará esa contraseña, lo que significa que, corresponde a ese número, definido también en el archivo *DES*. Si es usada una cadena de caracteres, entonces se usará la misma cadena para encriptar el parámetro *cad_txt*.

El archivo de contraseñas o claves puede especificarse con la opción del servidor *--des-key-file*. La cadena de caracteres que se obtiene tiene como primer carácter un *CHAR(128/cve_num)*. Se agrega el 128 para no complicarla y reconocer una contraseña o clave encriptada. Si se usa una cadena de caracteres, cve_num es 127. La longitud de la cadena para el resultado es *new_len = orig_len + (8- (orig_len % 8))+1*. Cada línea en el archivo de contraseñas o claves *DES* tiene la siguiente instrucción:

key_num des_key_str

Cada argumento *cve_num* debe ser un número con el rango entre 0 a 9. Las instrucciones contenidas en el archivo puede que estén en cualquier orden; *des_key_str* es la instrucción de la cadena de caracteres usada en la encriptación de la información. Entre el número y la clave existe un espacio como mínimo. La primera clave se entiende que es usada por defecto, cuando no se especifica ningún argumento o parámetro para *DES_ENCRYPT( )*. Se le puede indicar al motor del gestor MySQL que lea un nuevo valor de la clave del archivo con el comando *FLUSH DES_KEY_FILE*. Forzosamente entonces necesita el privilegio *RELOAD*.

El beneficio de integrar el conjunto de contraseñas o claves por defecto, es el de permitir a las aplicaciones la manera de reafirmar la existencia de los valores de columna encriptados, sin dar al usuario la manera de desencriptarlos.

Ejemplo de comprobación de esta función:

```
mysql > SELECT campo1 FROM tabla1
      > WHERE campo2 = DES_ENCRYPT('4567');
```

DES_DECRYPT(encrip_cad_txt [,llave_cad_txt])

El trabajo que realiza es el de desencriptar una cadena de texto que ha sido encriptada con la función *DES_ENCRYPT()*. Si existe el surgimiento del caso de error, esta retornará NULL (Nulo).
Así también de la misma manera como se empleo con la función anterior, se debe notar que esta función actuará sólo si MySQL fue configurado con soporte para SSL(Secure Sockets Layer o Capa Sockets de seguridad). Si no se especifica el parámetro *llave_cad* de la función *DES_DECRYPT( )*, examinará el primer byte de la cadena encriptada para determinar el número de la clave *DES* que fue utilizada para encriptar la cadena original y, luego se puede leer la clave del archivo de contraseñas o claves *DES* para desencriptar la información. Para que esto funcione, el usuario debe tener el privilegio SUPER. El archivo puede especificarse con la opción del servidor *-- des-key-file*.

Si se especifica el argumento *llave_cad*, esta cadena de texto se usa como la contraseña (clave o llave, en este caso) para desencriptar la información. Si el argumento *crypt_cad* no parece una cadena encriptada, MySQL retorna *crypt_cad*.

AES_ENCRYPT(cad_txt, cve_ cad_txt)

La tarea de esta función es encriptar los datos utilizando el algoritmo oficial *AES* (*Advanced Encryption Standard/Encriptación Estándar Avanzada*), conocido anteriormente como el algoritmo ***"Rijndael"***. Se puso en práctica por defecto una encriptación con una clave o llave de 128 bits de longitud, pero es viable ampliarlo con una longitud de 256 bits cambiando el código fuente. En el manual de la documentación aclara que se ha elegido una longitud base de 128 bits por las razones de velocidad y considerado hoy en día como altamente seguro.

Los parámetros o argumentos proporcionados de entrada pueden ser de *n* longitud. Si algún argumento es NULL(Nulo), el resultado de esta función también es NULL(Nulo). El algoritmo *AES* es considerado como un cifrado por bloques, utiliza *padding* o relleno para las cadenas de longitud no correspondiente y así la longitud de la cadena resultante se calcula como:

16 (trunc(string_length / 16) + 1)*

Si en la función *AES_DECRYPT( )* se detectan datos que son inválidos o un *padding/relleno* incorrecto, entonces se retornará un NULL(Nulo). También es posible que *AES_DECRYPT( )* regrese un valor no NULL(Nulo) (considerado posiblemente basura) en tanto solo y si los datos de entrada o la clave/llave son inválidos.
Es factible utilizar la función *AES_ENCRYPT( )* para almacenar datos de forma encriptada modificando las sentencias SQL. Por ejemplo:

iPSUMTEC
ISSN: 2594 - 2905

```
INSERT INTO tabla1 VALUES
(1, AES  ENCRYPT('texto', 'contraseña'));
```

Es factible obtener, incluso, mayor seguridad, cuando no es transferida la contraseña a través de la conexión por cada consulta; esto se lleva acabo almacenando la clave en una variable del servidor cuando se realiza la conexión. Por ejemplo:

```
SELECT @contrasena:='contraseña';

INSERT INTO tabla1 VALUES
(1, AES_ENCRYPT('texto', @contrasena));
AES_DECRYPT(crypt_cad, llave_cad)
```

Esta función elabora la desencriptación de la información usando el algoritmo oficial *AES (Advanced Encryption Standard/Encriptación Estándar Avanzada)* o en caso de que se lleve a cabo una puesta en funcionamiento fuera de lo provisto por la base de datos, que ha sido encriptada con antelación mediante la función *AES_ENCRYPT( )*, descrita anteriormente.
En el manual oficial de MySQL, *AES_ENCRYPT( )* y *AES_DECRYPT( )* son consideradas funciones de cifrado simétrico, criptográficamente hablando, las más seguras disponibles en MySQL según lo menciona Maiorano [8].

En la computación moderna los algoritmos (aplicaciones) criptográficos, se basan en cálculos por lo que las computadoras están creadas para correr los algoritmos. Para poder entender lo anterior, describe la definición del concepto de algoritmo: es un programa que ejecuta una secuencia finita ordenada de comandos o instrucciones que al introducir datos o valores de entrada, y que en un instante dado finaliza devolviendo un resultado como solución.

La *Teoría de Algoritmos*: Ciencia en la que se estudia el desarrollo de aplicaciones o software para resolver problemas del mundo real, se considera como un pseudocódigo. En la *Teoría de Algoritmos* integra una variedad de herramientas aplicables, esto es con el fin de escoger el mejor algoritmo que se adapte a la mejor manera en resolver cualquier problema. La Criptografía se apoya en la *Teoría de Algoritmos*, evaluando aspectos de seguridad informática para evitar el daño a los datos, uno de ellos es el de asegurar el acceso de un usuario legítimo, que debe tener la llave, con la cual puede encriptar y desencriptar los datos de forma lícita, rápida y cómoda, garantizando que un atacante no dispondrá de ninguna aplicación que sea eficiente y capaz de comprometer el sistema según datos de Manuel J. Lucena López [7].

La Técnica de Ciberseguridad de Fuerza Bruta
Según el autor Tori [12] en su libro *Hacking Ético*, en la que describe que la *Fuerza bruta, es la* técnica de ciberseguridad, contenida dentro de los aspectos seguridad de la criptografía, y sobre todo del concepto del criptoanálisis (técnica que su función es la de infiltra o quebrar códigos de texto encriptados para hacerlos entendibles). Por medio de un programa intenta y prueba las probabilidades de ir clarificando cada carácter ya sea número o símbolo de una contraseña hasta descubrir de manera entendible el resultado esperado. La técnica de la fuerza bruta es la acción de generar código o claves (combinación secuencial de caracter por caracter) intentando determinar la autentificación o archivo para verificar la coincidencia con un login de acceso válido (usuario, clave o sólo clave). Y su función es la de apuntar para romper el encriptado de archivos que contienen los passwords (contraseñas) de los sistemas operativos o cuentas de usuario. Por lo que esta técnica debe aplicarse de una manera profesional y ética, puede utilizarse con sus respectivas herramientas solo en la verificación de las vulnerabilidades, de las cuales se debe proteger y dar solución a los ataques de infiltración.

Métodos Criptográficos
Según Arboledas en su libro *Criptografía sin Secretos con Python* equipara en forma paralela que a partir del desarrollo de la esteganografía dió inicio a la creación de los métodos criptográficos. En la protección de la información que es confidencial o secreta loe mensajes emitidos deben ser indescifrables, esto con el motivo para cualquiera que no entienda el algoritmo de codificación, el cual es basado en un protocolo compartido tanto por un emisor y un receptor de un mensaje. Es así como el receptor podrá revertir el proceso, haciendo que el mensaje sea comprensible (descifrable) [11].

La *Criptografía* es constituida de dos *técnicas de transformación* del texto plano: ***métodos de transposición*** y ***métodos de sustitución***. El primer método es el ***método de transposición*** donde las letras contenidas en el texto plano se desordenan o mezclan, ejecutando un determinado algoritmo para obtener un ***anagrama*** (*procedimiento consistente en crear una palabra a partir de la reordenación de las letras de otra palabra*). El ejemplo de este método es el criptograma *SEVALC* que es una factorización de 6! = 720 *transposiciones* o ***anagramas*** posibles de un texto plano *oculto*. En este caso aplica el algoritmo que consiste en escribir el texto al revés, de atrás hacia delante.

Solo existe un inconveniente que este método funciona usando mensajes cortos, se quiere aclarar que es un muy inseguro, debido a la existencia de un número limitado en cuanto a ***anagramas*** se refiere, específicamente la palabra *secreto*, donde *n* es el número de letras. Ahora

que, cuando la longitud del mensaje se incrementa, la cantidad del número de *transposiciones* crece aún más, por tanto es casi imposible descifrar el texto plano, cuando el código del proceso es desconocido.

El segundo método es el ***método de sustitución***, en este en oposición al primero, el contenido del texto plano preserva su orden en el mensaje tal cual, pero el texto es reemplazado con caracteres, letras o símbolos en el texto encriptado utilizando algún algoritmo específico (RSA, Clave única etc.). En el ejemplo, el criptograma *secreto* es una sustitución del texto plano *secreto*, en el que cada letra del texto ha sido sustituido por un respectivo carácter del alfabeto masónico (*el cifrado de francmasón conocido también con el nombre de "cifrado pocilga, cifrado masónico, cifrado de Napoleón o cifrado de tres en raya", es un algoritmo que funciona por el método de sustitución simple, el cual convierte las letras por símbolos usando la base de un diagrama*) [11].

La necesidad de cifrar datos antes de ser guardados en una base de datos, es el de restringir el acceso a través de la autorización y autenticación de datos, lo cual puede ayudar en cierto límite, pero ¿qué pasa si el intruso de alguna manera llega a la base de datos? Tiene todos los datos de la base de datos y puede hacer un mal uso de ellos como quiera, aquí entra en juego el cifrado de los datos antes ser almacenados en la base de datos. Si los datos están cifrados antes de guardarlos en la base de datos, incluso con acceso a la base de datos, el intruso no puede hacer un uso indebido de estos según Singh [4].

La autora Lucy Noemy Medina Velandia en su libro *Criptografía y Mecanismos de Seguridad* describe que los seres humanos han intentado dar protección a la información sensible o ya sea confidencial contra ataques mal intencionados, y sobre todo muy en especial, en la del conocimiento, siendo muy útil en la creación de mecanismos que son útiles en la defensa para salir victoriosos en los eventos bélicos. Es por eso que el conocimiento ha evolucionado en base a claves secretas, que son representadas en criptogramas. Debido a este antecedente surge la importancia de garantizar una eficiente protección a los datos en base a los protocolos de encriptamiento los que limitan el acceso a la información.
Es importante la protección a los datos al evitar ataques no autorizados, ya sea de robo, infiltración no autorizada para modificar la información, control de acceso de infiltraciones mal intencionadas, pero también es imperante el tener la información en disposición cuando esta sea necesitada. Y sobre todo que cuando un usuario autorizado no le sea negado el acceso cuando modifique o utilice la información legalmente [3].

Dan Boneh y Victor Shoup explican cómo se define la *seguridad perfecta* en base a la siguiente pregunta: ¿qué es un cifrado "seguro"? la respuesta es que un cifrado seguro es aquel para el cual un mensaje cifrado permanece "bien oculto", incluso después de verlo ya encriptado. Por tanto, convertir esta respuesta en una que sea matemáticamente significativa y prácticamente relevante es un verdadero desafío. De hecho, aunque los cifrados se han utilizado durante siglos, es sólo en las últimas décadas que se han desarrollado definiciones de seguridad matemáticamente aceptables [9].

La noción matemática de seguridad perfecta, es el estándar de oro para la seguridad (al menos, cuando la preocupación es cifrar un solo mensaje y no importa la integridad). Por lo que es posible alcanzar este nivel de seguridad; De hecho, se muestra que la libreta de un solo uso satisface la definición. Sin embargo, la libreta de un solo uso no es muy práctica, en el sentido de que las claves deben ser tan largas como los mensajes: Por ejemplo si Alicia quiere enviar un archivo de 1GB a Roberto, ya deben compartir una clave de 1GB Desafortunadamente, esto no se puede evitar: También cualquier cifrado perfectamente seguro debe tener un espacio clave al menos tan grande como su espacio de mensaje. Este hecho proporciona la motivación para desarrollar una definición de seguridad que es más débil, pero que es aceptable desde un punto de vista práctico, y que permite cifrar mensajes largos utilizando claves cortas [9].

Si Alicia encripta un mensaje m bajo una clave k, y un adversario (espía) que escucha a escondidas obtiene el texto cifrado c, Alicia solo tiene la esperanza de mantener m en secreto si la clave k es difícil de adivinar, lo que significa, como mínimo, que la clave k debe elegirse al azar de un gran espacio de claves. A decir que m está "bien oculto" debe significar al menos que es difícil determinar completamente m a partir de c, sin conocimiento de k; sin embargo, esto no es realmente suficiente. Aunque el adversario (espía) pueda que no sepa k, es de asumir que sí conoce el algoritmo de cifrado y la distribución de k. De hecho, se supone que cuando se encripta un mensaje, la clave k siempre se elige al azar, uniformemente entre todas las claves en el espacio de claves.

El adversario (espía) también puede tener algún conocimiento del mensaje cifrado debido a las circunstancias, puede saber que el conjunto de posibles mensajes es bastante pequeño, y puede saber algo acerca de la probabilidad de cada posible mensaje. Por ejemplo, suponer que sabe que el mensaje m es m_0= *"ATAQUE␣AL␣AMANECER"* o m_1= *"ATAQUE␣AL␣ANOCHECER"*, y con base en la inteligencia disponible del adversario (espía), es

igualmente probable que Alicia elija cualquiera de estos dos mensajes. Sin ver el texto cifrado *c*, el adversario (espía) solo puede tener un 50% posibilidad de adivinar qué mensaje envió Alicia. Pero se asume que el adversario sabe *c*. Incluso teniendo este conocimiento, ambos mensajes pueden ser posibles; es decir, pueden existir las claves k_0 y k_1 tal que $E(k_0, m_0)=c$ y $E(k_1, m_1)=c$, por lo que no puede estar seguro si $m = m_0$ o $m = m_1$ [9].

Medición de la Seguridad en Bits

La medición de un nivel de seguridad basado en bits, puede ser más pequeño que el tamaño de una clave por las siguientes motivos [18]:

- Cuando un ataque desencripta el texto cifrado en pocas operaciones de las que se pueden esperar, por lo que usando técnica que recupera la clave intentando probar no todas las claves de 2^n, sino solo una parte de ellas.
- Es claro que cuando el nivel de seguridad de encriptamiento discrepa del tamaño de su clave, como con la mayoría de los algoritmos de encriptación de clave pública. Y el ejemplo se da con el algoritmo RSA que tiene una clave secreta de 2048 bits brinda menos de 100 bits de protección de seguridad.

La seguridad de bits resulta útil cuando se comparan los niveles de seguridad de los cifrados, pero no brinda suficiente información sobre el costo real de un ataque. A veces es una abstracción demasiado simple porque simplemente asume que un cifrado seguro de *n* bits necesita 2^n operaciones para romperse, sean cuales sean estas operaciones. Por lo tanto, dos cifrados con el mismo nivel de seguridad de bits pueden tener niveles de seguridad muy diferentes en el mundo real cuando se tiene en cuenta el costo real de un ataque para un atacante real.

Magnitud Total del Ataque

El Bit de seguridad expresa el costo del ataque más rápido contra un cifrado, al estimar la magnitud de la cantidad de operaciones que necesita para tener éxito. Pero hay otros factores que afectan el costo de un ataque, y estos deben tenerse en cuenta al estimar el nivel de seguridad real. Los cuatro principales son: paralelismo, memoria, precálculo y número de objetivos [18].

Seguridad Comprobable

La seguridad comprueba que rompiendo un esquema criptográfico debe ser difícil tanto como el resolver otro problema, sabiendo que es complejo. La prueba garantiza que cuando se aplica la criptografía será tan segura, que entre más es complejo sea el descrifrar los datos será más efectivo. Este tipo de prueba es llamada *reducción*, y viene de la teoría de la complejidad. Se dice que desencriptar algún dato cifrado es acotar el problema *X* si cualquier técnica para resolver el problema *X* se aplica la técnica de desencriptación.

Las pruebas de seguridad se dividen en dos tipos, dependiendo del tipo de problema, si este es complejo que se: pruebas relativas a un problema matemático y el de pruebas relativas a un problema criptográfico [18].

El Modelo del Atacante

Los autores L. Bouganim y P. Pucheral describen que los atacantes se pueden clasificar en tres clases: intrusos, internos y administradores. Intruso es una persona que obtiene acceso a un sistema informático e intenta extraer información valiosa. Interno es una persona que pertenece al grupo de usuarios de confianza e intenta obtener información de la cual no está autorizado acceder. Administrador es una persona que tiene privilegios para administrar un sistema informático, pero abusa de sus derechos para extraer información valiosa [13]. En muchos casos, un Administrador de Bases de Datos tiene acceso a toda la base de datos y protegerla contra el mientras que simultáneamente le permite realizar sus tareas, por lo que se convierte en un gran desafío.

Todos los atacantes pueden utilizar diferentes estrategias de ataque: Los ataques de almacenamiento directo son ataques contra el almacenamiento que se pueden llevar a cabo accediendo a archivos de la base de datos por medios distintos al software de la base de datos, tal como físicamente eliminando los medios de almacenamiento o accediendo a los archivos de copia de seguridad de la base de datos. En ataques de almacenamiento indirecto, el atacante puede acceder a la información del esquema y metadatos, tal como los nombres de tablas y columnas, estadísticas de columnas y valores escritos en registros de recuperación, para estimar las distribuciones de datos. Un ejemplo que expresa lo anterior es el de los ataques de memoria, cuando el atacante tiene acceso ilegal directamente a la memoria del software de la base de datos. En muchos casos, la memoria contiene el caché el cual contiene grandes cantidades de la base de datos por razones de optimización [14].

Metodología

En la aplicación de la metodología pueden utilizarse cualquiera de los métodos más utilizados en la encriptación de datos, y a continuación se mecionan: *"llave pública"*, también conocida como *encriptación asimétrica*, y la *"llave privada"*, conocida también como *encriptación simétrica*. Ambas técnicas utilizan pares de llaves, pero la diferencia existe en cómo se manejan los emisores y receptores que comparten las llaves y manipulan el proceso de encriptado y desencriptado.

El método de la llave pública (encriptación asimétrica), aquel que envía conocido como emisor utiliza la llave pública, por lo que es conocida en la encriptación del texto de los datos. El receptor, tiene la llave privada que con la que conforma la otra parte del par público o privado. Con la combinación de la llave privada y la llave pública, el receptor puede desencriptar los datos. Si el caso es de una llave privada (encriptación simétrica), el emisor como el receptor manejan la misma llave privada. Como se describe, existe bastante organización involucrada tanto almacenar como transmitir llaves secretas.

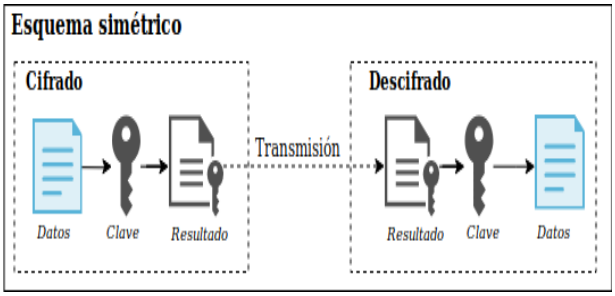

Figura 2. Esquema General de un Sistema de Cifrado Simétrico

Figura 3. Esquema General de un Sistema de Cifrado Asimétrico

En la actualidad en el concepto de Criptografía se manejan dos métodos para la encriptación de datos muy populares hoy en día, el primero llamado encriptación asimétrica conocido como el de "llave pública", y el segundo llamado encriptación simétrica, conocido como de "llave privada". Aunque en los dos métodos su base es el de la utilización de pares de llaves, pero la diferencia existente en cuanto al manejo de sus llaves es la manera en que el emisor y el receptor convergen con las llaves y por supuesto el manejo del proceso tanto de cifrado y descifrado de datos. A continuación se describen los diferentes algoritmos de encriptación. Ejemplo de algunos de ellos son:

***Triple-DES (3DES)*:** De tres llaves y es conocido como por fortalecer el método clásico *Digital Encryption Standard/Encriptación Estándar Digital* o llamado *DES*. Usa llave de 168 bits; por lo que también es susceptible a los ataques de llaves relacionadas. Este método se puede considerar un cifrado de 3 rondas con sub claves de rondas independientes de 56 bits del método DES y aumentando esta longitud a 168 bits para fortalecer que en cualquier ataque sea difícil de romper, teniendo en cuenta que cada ronda es considerada muy fuerte. Se podría usar el criptoanálisis de una clave rotacional relacionada; más sin embargo, tal enfoque requeriría muchos textos planos conocidos.

Advanced Encryption Standard/ (AES): Este método es de encriptamiento simétrico, el cual fue creado a partir del algoritmo de bloques conocido como "*Rijndael"*. Fue creado para reemplazar el método DES. *AES* usa una estructura diferente a DES. Por lo que no es un cifrado de ***Feistel***.

***RSA*:** El método RSA es similar, aunque muy diferente, al método *Diffie-Hellman* (DH abreviado). Se basa en una función unidireccional: suponiendo que p y g argumentos que se conocen públicamente, pueden calcular ($g^x \bmod p$ *"obtiene el residuo de una división"*) de x, pero no puede calcular x dado $g^x \bmod p$. RSA se basa en una función unidireccional de trampa.

Dada la información públicamente conocida n y e, es fácil calcular $m \bmod n$ a partir de m, pero no al revés. Sin embargo, si conoce la factorización de n, entonces es fácil hacer el cálculo inverso. La factorización de n es la información de trampa. Si se sabe, se puede invertir la función; si no se sabe, no se puede invertir la función.

Esta función de trampa permite a RSA que se utilice tanto para el cifrado como para firmas digitales. RSA fue inventado por *Ronald Rivest, Adi Shamir* y *Leonard Adleman*, y publicado por primera vez en 1978. Se usan valores p, q y n. Los valores p y q son números primos muy grandes y diferentes, cada uno 1000 bits o más. El valor n está definido por $n := pq$.

***Elliptic Curve Cryptography/ Criptografía de Curva Elíptica (ECC)*:** El método de *Criptografía de Curva Elíptica (ECC)* forma parte de las tres familias de los algoritmos de clave pública. *ECC* ofrece el mismo nivel de seguridad que el método RSA o los sistemas de logaritmo discreto, considerando operandos más cortos (aproximadamente 160-256 bits contra 1024-3072 bits). ECC se basa en el problema del logaritmo discreto generalizado, por lo que los protocolos DL(Discrete Logarithm/Logaritmo Discreto) tales como la llave de intercambio de *Diffie-Hellman* también pueden realizarse utilizando curvas elípticas.

En muchos casos, *ECC* ofrece ventajas de rendimiento (menos cálculos) y ventajas de ancho de banda (firmas y

claves más cortas) que los esquemas del método RSA y de Logaritmo Discreto (DL). Aunque las operaciones del método RSA que implican claves públicas cortas siguen siendo mucho más rápidas que las operaciones ECC. Los cálculos de las matemáticas de las *curvas elípticas* son considerablemente más complejas que las de los esquemas RSA y DL(Discrete Logarithm/Logaritmo Discreto).

En la metodología, los mecanismos de seguridad suelen introducir una sobrecarga computacional significativa. Esta sobrecarga puede constituir un problema fundamental para el Gestor de Bases de Datos (DBMS), ya que el rendimiento del Gestor (DBMS) tiene una influencia directa en el rendimiento de todo el sistema de información. Al intentar minimizar la sobrecarga de rendimiento que resulta del cifrado de la base de datos, se deben considerar los siguientes problemas [15]:

Cifrado selectivo: Sería deseable cifrar solo los datos confidenciales y mantener los datos no confidenciales sin cifrar. Además, solo los datos relevantes deben cifrarse/descifrarse al ejecutar una consulta. Por ejemplo, si solo un atributo participa en una consulta, sería innecesario cifrar/descifrar todo el registro.

Índices y otros mecanismos de optimización del Gestor de Bases de Datos (DBMS). Cifrar el contenido de la base de datos puede impedir que algunos mecanismos cruciales de optimización del *Gestor de Bases de Datos* (*DBMS*) funcionen correctamente. Por ejemplo, algunos proveedores del *Gestor* (*DBMS*) no permiten crear índices en columnas cifradas, mientras que otros lo permiten en función de los valores cifrados de la columna (en caso de que no estén salteados). El último enfoque da como resultado la pérdida de algunas de las características más obvias de los índices (búsquedas de rango), ya que un algoritmo de cifrado típico no conserva el orden.

Sobrecarga de cifrado. Es deseable que se minimice el tiempo dedicado a cifrar y descifrar datos. Por ejemplo, cifrar la misma cantidad de datos mediante una única operación de cifrado es más eficiente que cifrarlos por partes mediante varias operaciones de cifrado.

Modelo de Amenaza

Bastante tiempo la seguridad de la base de datos no se consideró un problema importante hasta que surgieron las amenazas a la confidencialidad e integridad de los datos [17].

En general, hay tres tipos de datos en la base de datos, que son datos privados, datos confidenciales y datos públicos. Los datos privados solo pueden ser controlados por el quien los creó y nunca se permite que otros, que son ajenos a la creación de la información sean expuestos o puedan ser modificado. Los datos confidenciales se refieren a los datos que contienen información confidencial que solo se puede compartir entre un grupo autorizado de usuarios, que son designados por el propietario de los datos, con lo cual otros usuarios no tienen el privilegio de acceder a ellos.

Los datos públicos son el caso común de datos, que podrían ser compartidos por todos los usuarios en el sistema de base de datos. Por lo tanto, se debe tener la concentración en los datos privados y los datos confidenciales, los cuales son el objetivo principal de los atacantes.

La clasificación de atacantes, los cuales se dividen en tres tipos, que son *intrusos*, *internos* y *administradores*. El intruso es aquel que es externo a una organización o institución y se infiltra en un servidor de base de datos para robar o manipular información de datos. *Internos* es un usuario que tiene una autorización en el sistema de base de datos, pero realiza una acción maliciosa. El *administrador* incluye el administrador de la base de datos (DBA) y el administrador del sistema (SA), y que tienen el poder absoluto sobre el sistema de base de datos. Si la base de datos está bajo el control de un administrador entonces su gestión considerada de un absoluto abuso del poder que tiene, por lo tanto no se puede garantizar una seguridad de base de datos confiable.

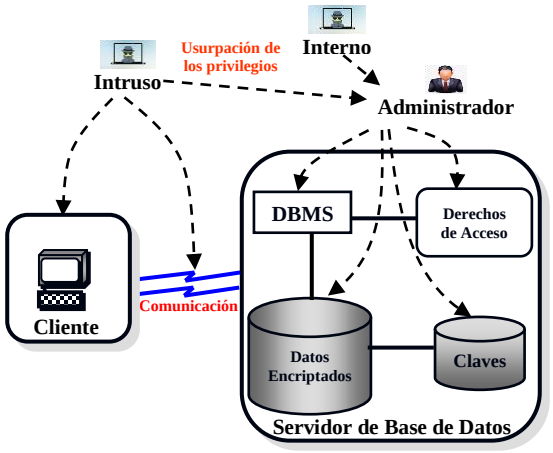

Figura 4 Modelo de Amenazas para el Cifrado de Bases de Datos

En el servidor de la base de datos, la base de datos cifrada puede sufrir los siguientes riesgos principales:

1. El administrador abusa de su poder. El poder de *superusuario* le da la capacidad de acceder y manipular datos confidenciales.
2. El intruso y el interno pueden tener la oportunidad de usurpar los privilegios de un administrador y eludir el control de acceso para realizar algunas tareas destructivas.

3. En el caso de que las claves se almacenen en el mismo servidor de base de datos con datos cifrados, es muy probable que ambas se revelen al mismo tiempo cada vez que la base de datos se vea comprometida.
4. Si el algoritmo de cifrado no es lo suficientemente fuerte o eficiente, la clave podría ser revelada de alguna manera.
5. Si se usa una clave para cifrar una gran cantidad de datos, es peligroso para todos los datos cifrados cuando los atacantes obtengan esta clave.

En el Cliente, los siguientes riesgos ponen en peligro la seguridad de la base de datos cifrada:

1. La protección de las llaves (Claves) es un gran problema. Es difícil guardar las llaves en un lugar seguro.
2. Algunos usuarios son propensos a olvidar sus contraseñas, que son importantes para el descifrado de datos.

En la Comunicación, el proceso de transmisión entre el cliente y el servidor de la base de datos, es probable que ocurra un peligro:

1. La información confidencial o las claves pueden ser robadas fácilmente por una persona que se infiltra en la forma en que se emiten los datos.
2. Todas las amenazas enumeradas que se mencionaron con anterioridad son problemas y hay que evitarlos utilizando el mecanismo de encriptación de la base de datos.

Limitación de los tipos y cantidades de datos (por ejemplo, nivel de columna) que se deben cifrar

Es importante en la definición de políticas la identificación de los datos que deben protegerse y los que pueden permanecer en texto claro. Por lo que muchas organizaciones o usuarios omiten este paso que es indispensable y solo cifran más datos de los necesarios. Por lo que puede parecer inofensivo y brindar protección adicional, el encriptamiento de más datos de los que pueden ser necesarios, será el resultado de un precio pagado en términos de rendimiento. El Cifrado a nivel de columna se debe implementar para respaldar la política de seguridad de las bases de datos una organización.

Realizar operaciones sin descifrar los datos

De forma similar a la búsqueda de datos cifrados, las soluciones avanzadas realizan operaciones como uniones de datos cifrados, sin necesidad de ser convertirlos a texto sin ser cifrados, lo que conserva el rendimiento del sistema.

Conclusiones

Para mantener la seguridad de una base de datos es importante y cada vez más difícil. El objetivo principal de los atacantes es obtener datos confidenciales de una base de datos por medio de una infiltración ilegal. Para mantener adecuadamente la seguridad, integridad y confidencialidad de los datos, es entonces cuando se convierte en uno de los desafíos cada vez más complicados.

Siendo uno de los requisitos, la seguridad de las bases de datos el cimiento para el uso del cifrado con que los datos se cifran, es una medida con la que son manejados y administrados a través de las redes, almacenados en sistemas de bases de datos, de tal manera que se puede impedir en un gran porcentaje de las infiltraciones de ataques de intruso externos como de usuarios maliciosos internos a instituciones u organizaciones (enemigo interno). La implementación de un cifrado en una base de datos, cien por ciento no se tendrá la protección contra todas las amenazas, aunque el cifrado de los datos almacenados puede aumentar la seguridad de una aplicación, siempre y cuando esté bien implementada y bien configurada, usando cualquier gestor de base de datos, llámese Oracle, SQL Server, Postgre, MySQL etc.

El concepto conocido como endurecimiento de los sistemas (*Hardening*). Como ejemplo, Oracle es un gestor de base de datos que proporciona capacidades de cifrado que son nativas del mismo y permiten a los desarrolladores de aplicaciones proporcionar medidas adicionales de seguridad mediante el cifrado selectivo de los datos almacenados. Al abordar el concepto de criptoanálisis es difícil, debido a que no hay un libro de texto estándar y no hay forma existente de conocer los riesgos del encriptamiento de las bases de datos si son los adecuados o no para los diferentes niveles.

Con la implementación de técnicas de la criptografía, se construyen en las bases de datos altamente eficientes, fuertes propiedades de seguridad con capacidades flexibles dentro de un buen control de acceso.

La encriptación de datos es una técnica elemental y necesaria en la ciberseguridad en general, sobre todo en la seguridad de bases de datos muy en particular. El proceso requiere tecnología altamente sofisticada, así como también el uso de los métodos matemáticos, por lo que las soluciones se están volviendo muy sencillas de usar, al menos al nivel de los usuarios.

En algunos casos, como en los sistemas operativos que utilizan iOS o Android, la encriptación ocurre automáticamente incluso aunque el usuario no lo sepa. Para las organizaciones, la encriptación de datos debería ser parte principal de las medidas de seguridad, y que sea aplicada de manera selectiva a los activos de información

sensibles contra los ataques de espías, atacantes o delincuentes cibernéticos.

En la búsqueda de palabras clave y consultas complejas, los protocolos se pueden considerar un método muy básico de acceso a las bases de datos en el cual el usuario ya conoce el índice del elemento (dato) a solicitar. Las aplicaciones prácticas de las bases de datos por lo regular exigen consultas complejas, como ejemplo, la búsqueda de palabras claves. También se pueden usar para implementar búsquedas privadas en datos que son encriptados, por lo que es cuestión personal permitir consultas aún más complejas.

BIBLIOGRAFÍA

[1] Apd (2019), "Tipos de seguridad informática: ¿Cuáles son y qué importancia tienen?", https://www.apd.es/tipos-de-seguridad-informatica/

[2] Cortés Mireya (2022), "Las 15 filtraciones de datos más grandes del siglo XXI", https://cio.com.mx/las-15-filtraciones-de-datos-mas-grandes-del-siglo-xxi/#:~:text=de%20datos%20personales.-,LinkedIn,foro%20de%20piratas%20inform%C3%A1ticos%20rusos, noviembre 2022.

[3] Medina Velandia Lucy Noemy (2017), "Criptografía y mecanismos de seguridad", Primera edición, Fundación Universitaria del Área Andina, noviembre de 2017, Bogotá D.C. ISBN: 978-958-5460-19-5

[4] Singh Prabhsimran (2015),"Database security using encryption", 2015 1st International conference on futuristic trend in computational analysis and knowledge management (ABLAZE 2015).https://www.researchgate.net/publication/282925678

[5] Wayner Peter (2021)," 12 fallas y errores de seguridad de las bases de datos", https://cioperu.pe/, CIO, Perú.

[6] Elmasri R., Navathe S.B. (2011), Fundamentals of Database Systems, 6th edition, Addison-Wesley, United States of America, 2011.

[7] Lucena Lopez Manuel J. (2022),"Criptografía y Seguridad en Computadores", 4ta Edición, CC Creative Commons, España.

[8] Maiorano, Ariel H. (2009). "Criptrografía. Técnicas de desarrollo para profesionales". 1era edición, AlfaOmega Buenos aires Argentina.

[9] Boneh Dan and Shoup Victor (2023),"A Graduate Course in Applied Cryptography".

[10]Katz Jonathan and Lindell Yehuda (2008), "Introduction to Modern Cryptography", Chapman & Hall CRC PRESS.

[11] Arboledas Brihuega David (2017), "Criptografía sin secretos con Python", RA-MA Editorial, Madrid España, ISBN: 978-84-9964-698-5.

[12] Tori Carlos (2008), Haking Ético, 1era Edición, Mastroianni Impresiones Buenos Aires Argentina, Mayo del 2008, ISBN 978-987-05-4364-0. pag 108

[13] L. Bouganim, P. Pucheral (2002), "Chip-secured data access: confidential data on untrusted servers", Proceedings of the 28th international conference on Very Large Data Bases-Volume 28 (2002).

[14] R. Agrawal, J. Kiernan, R. Srikant, Y. Xu (2004), Order preserving encryption for numeric data, Proceedings of the 2004 ACM SIGMOD international conference on Management of data (2004).

[15] Iyer B., Mehrotra S., Mykletun E., Tsudik G., Wu Y. (2004), "A Framework for Efficient Storage Security in RDBMS, Advances in Database Technology", EDBT 2004.

[16] Castano S., Fugini M., Martella G., Samarati P., "Database Security", Addison-Wesley, 1995.

[17] Chen Gang, Chen Ke, Dong Jinxiang, "A Database Encryption Scheme for Enhanced Security and Easy Sharing", Proceedings of the 10th International Conference on Computer Supported Cooperative Work in Design, 2006.

[18] Aumasson Jean-Philippe (2018), "SERIOUS CRYPTOGRAPHY: A Practical Introduction to Modern Encryption", No Starch Press, ISBN-13: 978-1-59327-826-7

ROLES DE CONTRIBUCIÓN

Rol	Autor (es)
Administración del proyecto Conceptualización	M.C. Juan Manuel Bernal Ontiveros
Redacción	Dr Francisco Zorrilla Briones
Supervisión	MSL Noé Ramón Rosales Morales
Redacción	MSL Verónica Farias Veloz
Metodología	M.C. Marisela Palacios Reyes