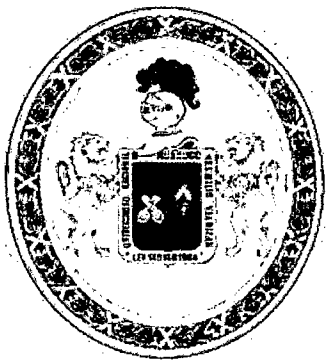
UNIVERSIDAD NACIONAL HERMILIO VALDIZÁN
HUANUCO



FACULTAD: INGENIERÍA INDUSTRIAL Y DE SISTEMAS

ESCUELA ACADÉMICO PROFESIONAL DE: INGENIERÍA DE SISTEMAS

TESIS

ALGORITMO DE PROTECCCION DE PUERTA TRASERA CONTRA ATAQUES DE AUTENTIFICACION EN SISTEMAS DE BASES DE DATOS EXPUESTOS A SERVICIOS WEB

PARA OPTAR EL TITULO PROFESIONAL DE INGENIERO DE SISTEMAS

TESISTA: WILMER OCHOA ALVARADO

ASESOR: Ing. ALCIDES BERNARDO TELLO

HUÁNUCO – PERÚ

2015

DEDICATORIA:

A Dios nuestro creador que nos dio la vida.

A mis padres, que me apoyan de día en día para poder cumplir las metas y sueños.

A los Docentes que con su empeño y dedicación me brindaron sus conocimientos para ser profesionales en el fututo.

AGRADECIMIENTO

En primer lugar a Dios por darnos la vida, protegernos, por ser nuestra fortaleza y guía en nuestro camino a fin de cumplir nuestras metas y sueños.

A mis padres y hermanos por brindarme su apoyo incondicional en cada instante de nuestra vida.

A nuestra Alma Mater, que es una institución formadora de profesionales altamente competitivos en la sociedad.

A mis docentes por brindarme su enseñanza con paciencia y dedicación a fin de formar profesionales de éxito al servicio de la comunidad.

A mi asesor de proyecto de tesis, ING. Alcides Bernardo Tello, no solo por las sugerencias, orientadas y aclaraciones brindadas, sino por el trato cortés que le caracteriza y porque ha sido de apoyo para llevar acabo la ejecución y culminación de la investigación.

AUTOR

RESUMEN

El presente estudio tiene por objetivo implementar el algoritmo de protección de puerta trasera contra ataques de autentificación en sistemas de bases de datos expuestos a servicios web. Se utilizó dos tipos de metodología, la primera relacionada a la investigación exploratorio y la segunda al desarrollo del algoritmo En este estudio se encontróque el sistema front-end de Joomla posee ciertas fallas en cuanto al acceso de los dorks(escalar privilegios) e inyecciones SQL, haciendo que el sistema sea vulnerable y accesible al backend(vista del administrador) de Joomla. Para contrarrestar se realizó el algoritmo que protege la puerta trasera del sistema front-end de Joomla, permitiendo bloquear los dorks e ingresos de robots de autentificación automático. En el sistema se impiementó dos textbox para usuario, contraseña y recaptcha. Y como resultado se logró proteger el front-end de joomla con acceso seguro al backend. Siendo una solución contra los Hackers.

Palabras Claves: Sistema front-end, backend,recaptcha, autenticación, puerta trasera

ABSTRACT

This study aims to implement the algorithm backdoor protection against attacks authentication database systems exposed to Web services. Two types of methodology, the first related to exploratory research and the other development of the algorithm In this study we found that the front-end system Joomla has some faults in access of dorks (escalate privileges) and injections was used SQL, making the system vulnerable and accessible to the backend (admin view) Joomla. To counter the algorithm that protects the back door of front-end system Joomla, allowing block the dorks and income robots automatic authentication is performed. In the system, two textbox for username, password and recaptcha was implemented. And as a result it was possible to protect the front-end of joomla with secure access to backend. As a solution against Hackers.

Keywords: System front-end, backend, recaptcha, authentication, backdoor

CONTENIDO

CAPITULO I 9

PROBLEMA DE INVESTIGACIÓN 9

1.1. Antecedentes y fundamentación del problema 9

1.2. Formulación del problema 10

1.3. Problema general 10

1.4. Problema específico 10

1.5. Objetivo de la Investigación 10

1.5.1. Objetivo general 10

1.5.2. Objetivos específicos 10

1.6. Justificación e importancia 11

1.7. Limitaciones 11

CAPITULO II 12

MARCO TÓRICO 12

2.1. ANTECEDENTES DEL PROBLEMA 12

2.1.1. Antecedentes a nivel mundial 12

2.1.2. Antecedentes a nivel nacional 17

2.2. Conceptos fundamentales 30

2.3. Marco situacional 30

2.4. Definición de términos 32

2.4.1. Servicio Web: 32

2.4.2. Puerta Trasera: 32

2.4.3. Base de Datos: 33

2.4.4. Algoritmo: 33

2.4.5. Inyección SQL 33

CAPITULO III 34
MARCO METODÓLOGICO 34
3.1. Tipo y nivel de investigación 34
3.2. Métodos de investigación 34
3.2.1. Investigación 34
3.2.1.3. La estructura de carpetas y archivos 47
3.2.1.4. El frontend y el Backend 51
3.2.2. DESARROLLO 60
CAPITULO IV 60
RESULTADOS 60

INTRODUCCIÓN

En el mundo globalizado, hoy en día, cada vez se utiliza la tecnología de la información y comunicación tal es el caso de los portales webs, donde se brinda la mayor información que uno busca o desea saber. Pero esto también podría ser perjudicial, debido a que muchos sistemas de hoy en día en Internet poseen información llamada "basura" en el argot informático y esto conlleva a dos puntos de vista como: ***el sistema podria estar vulnerado por los presuntos Hackers o simplemente ingresó la información sin tener en cuenta la veracidad de la misma.***

En el primer capítulo se plantea el problema; fundamentando toda la situación problemática con respecto al mundo de tecnologías de información y comunicación (TIC), y por ende se plantea el problema, los objetivos y las delimitaciones.

En el segundo capítulo se desarrolla los conceptos y teorías que se usa en el estudio.

El tercer capítulo se analizará profundamente el CMS Joomla para posteriormente desarrollar el algoritmo e implementarlo, para dar la solución al defecto que posee Joomla.

En el cuarto capítulo se indica los resultados obtenidos después del desarrollo

Hoy en día existe muchos cms (*Content Management System:* Sistema de gestión de contenidos) Y uno de ellos es Joomla que cumple con el sistema MVC (Modelo vista controlador), en esta investigación me centraré en el Framework del CMS de Joomla y las fallas que presentan y luego realizar las posibles soluciones elaborando un algoritmo. Para que de esta manera pueda solucionar la autentificación de puerta trasera del CMS Joomla.

AUTOR

CAPITULO I

PROBLEMA DE INVESTIGACIÓN

1.1. Antecedentes y fundamentación del problema

Hoy por hoy, en el mundo de TI (Tecnología de información) y el conocimiento, sigue siendo, un papel importante en la vida real en todo el ámbito de la supervivencia humana. Siendo así también que la TI se hace importante para las organizaciones en un mundo cambiante.

Así como aumenta la necesidad del uso de la TI, también aumenta el riesgo de vulnerabilidad de los sistemas, para ello muchas organizaciones contratan en temas de seguridad informática o realizando publicidad de correcto uso de su sistema. Pero las personas mal intencionadas tratan de ingresar a sus sistemas consiguiendo beneficios personales, es así que la seguridad es imprescindible para las organizaciones.

Muchos sistemas expuestos a web, están vulnerables a los ataques de los hackers, esto se debe a que muchas organizaciones contratan a empresas o terceras personas para su desarrollo de sus sistema, olvidando en simple y elemental sobre acceso de sus sistema (seguridad del sistema). Es así que en Febrero del año 2013 y 15 de Enero del 2015, la página del gobierno regional de Huánuco, fue hackeado, mediante inyección sql. Esto se debe a que no consideraron seguridad sobre las inyecciones.

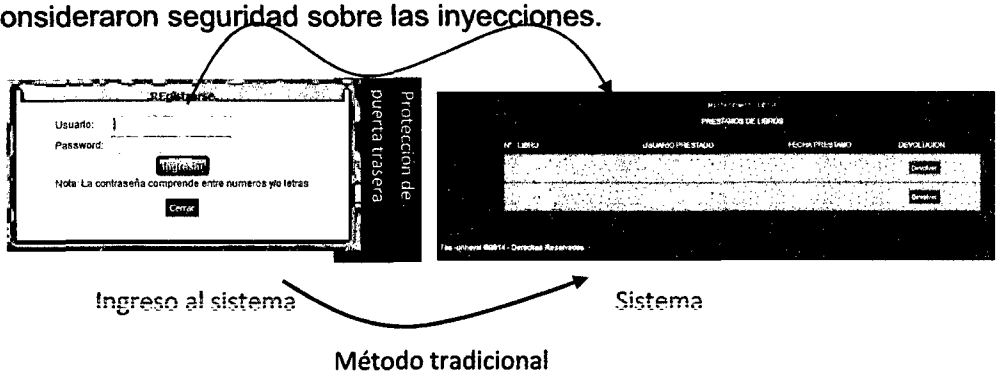


Imagen 01: Funcionamiento de la seguridad sistema

Imagen. Fuente: *Elaboración propia.*

1.2. Formulación del problema

1.3. Problema general

¿Cuál es el algoritmo que permite bloquear la puerta trasera contra ataques de autentificación de puerta trasera en sistemas de bases de datos expuestos a servicios web?

1.4. Problema específico

- ¿Cuál es el modelo del algoritmo que permitirá dar seguridad sobre ataques de autentificación en sistemas de bases de datos expuestos a servicios web?

- ¿Qué tipo de algoritmo sería adecuado para dar seguridad sobre ataques de autentificación en sistemas de bases de datos expuestos a servicios web?

1.5. Objetivo de la Investigación

1.5.1. Objetivo general

- Implementar el algoritmo de protección de puerta trasera contra ataques de autentificación en sistemas de bases de datos expuestos a servicios web

1.5.2. Objetivos específicos

- Implementar el algoritmo en lenguaje PHP, para la protección de puerta trasera contra ataques de autentificación en sistemas de bases de datos expuestos a servicios web.

- Demostrar el uso del algoritmo mediante el desarrollo de un prototipo capaz de proteger la autentificación en sistemas de bases de datos expuestos a servicios web.

- Demostrar el cumplimiento de estándar de codificación del sistema de protección de puerta trasera.

1.6. Justificación e importancia

El presente estudio de investigación se justifica por las siguientes razones:

- Debido a los códigos y consultas libres que existen en la web, se hace cada vez vulnerable los sistemas.
- Debido a la programación literal (conexión simple a consultas sql), sin restricción a inyecciones, ingresan al servidor de bases de datos.
- Algoritmos maliciosos por programadores expertos para Vulnerar el sistema Login.

1.7. Limitaciones

El presente estudio de investigación solo está basado para aquellos sistemas que están expuestos sus bases de datos a servicios web y con mayor caso sobre servidor BD PhpMyadmin y páginas Joomla de 1.5 a 3.0.

CAPITULO II

MARCO TÓRICO

2.1. ANTECEDENTES DEL PROBLEMA

2.1.1. Antecedentes a nivel mundial

Según la investigación realizada por ***e- securing*** *(seguridad banca por internet) y Redactado por: Mauro Maulini R.* muestra las diez *vulnerabilidades más comunes de las bases de datos.*[1]

Aunque el tema no está relacionado directamente con las aplicaciones web, es importante para su sano funcionamiento, proteger de forma correcta los datos que estas utilizan y definitivamente proteger las bases de datos no es una tarea fácil, sobretodo porque a menudo los ataques que van tras las más simples vulnerabilidades son lo que tienen más éxito.

De acuerdo con Alex Rothacker, gerente AppSec'sTeam SHATTER (Security Heuristics of ApplicationTestingTechnologyfor Enterprise Research), su equipo ha encontrado 10 tipos comunes de vulnerabilidades base de datos por las que las organizaciones padecen ataques una y otra vez.

El hilo común de esta lista es que rara vez las bases de datos salen al mercado "securityready", y su configuración de seguridad no es un trabajo de "hacer y olvidar" para los administradores de base de datos.

Las organizaciones deben evaluar continuamente los paquetes de su software de base de datos para determinar si son realmente

necesarios y desactivar los que no son necesarios para reducir las superficies de ataque. Tienen que ser cuidadosos en controlar los campos en las búsquedas para prevenir inyecciones y tener conciencia de la debilidad en las credenciales de inicio de sesión. Y lo más importante, es necesario aplicar los parches de actualización de la casa matriz con regularidad.[2]

Alrededor de la mitad de las vulnerabilidades nombradas por Rothacker y su equipo están directa o indirectamente relacionadas con las prácticas flojas de gestión de parches en el entorno de base de datos. Ese es un pensamiento aterrador considerando sólo el 38% de los administradores aplican los ajustes de seguridad en sus bases de datos Oracle en el ciclo de revisión inicial de tres meses. Y casi un tercio de ellos toman un año o más para aplicar el primer parche.

A continuación se muestra los diez tipos de vulnerabilidades de bases de datos más comunes:

A. Nombre de usuario/password en blanco, por defecto o débil.

No es nada raro conseguir en el día a día pares de usuario/password como sa/1234, está en la primera línea de defensa y un punto fundamental de la armadura de nuestras bases de datos. Es importante hacer revisiones periódicas de credenciales.

B. Inyecciones SQL.

Cuando la plataforma de base de datos falla para desinfectar las entradas, los atacantes son capaces de ejecutar las inyecciones SQL de forma similar a como lo hacen en los ataques basados en Web, lo que les permite elevar sus privilegios y obtener acceso a una amplia gama de funcionalidades. Muchos de los proveedores han dado a conocer soluciones para evitar estos problemas, pero no servirá de mucho si los parches no se aplican o no se toman los correctivos correspondientes.[3]

C. Preferencia de privilegios de usuario por privilegios de grupo.

Las organizaciones necesitan garantizar que los privilegios no se les den a los usuarios por asignación directa quien finalmente los recogerá como conserjes recogen las llaves en sus llaveros. En cambio, Rothacker recomienda que los usuarios sólo reciban privilegios por parte de grupos o funciones y sean manejados colectivamente. De esta forma será más fácil eliminar derechos a un usuario con simplemente eliminarlo del grupo, sin que queden derechos ocultos u olvidados asignados a dicho usuario.[4]

D. Características de base de datos innecesariamente habilitadas.

Cada instalación de base de datos viene con paquetes adicionales de todas las formas y tamaños que en su mayoría rara vez son utilizados por una sola organización. Dado que el nombre del juego en materia de seguridad de base de datos es el de reducir las superficies de ataque, las empresas necesitan buscar los paquetes

que no utilizan y desactivarlos. Esto no sólo reduce los riesgos de ataques (0)day a través de estos vectores, sino que también simplifica la gestión de parches.

E. Configuración de seguridad ineficiente.

Del mismo modo, las bases de datos tienen una gran cantidad opciones de configuración y consideraciones diferentes a disposición de los administradores para ajustar el rendimiento y funcionalidades mejoradas. Las organizaciones necesitan conseguir y desactivar aquellas configuraciones inseguras que podrían estar activadas por defecto para mayor comodidad de los DBA o desarrolladores de aplicaciones. Las configuraciones de bases de datos en producción y desarrollo deben ser radicalmente diferentes.

F. Desbordamientos de búfer.

Otro favorito de los piratas cibernéticos, las vulnerabilidades de desbordamiento de búfer, son explotadas por las inundaciones de las fuentes de entrada con valores diferentes o muy superiores a los que aplicación espera - por ejemplo, mediante la adición de 100 caracteres en un cuadro de entrada pidiendo un número de Seguro Social. Los proveedores de bases de datos han trabajado duro para solucionar los problemas técnicos que permiten estos ataques se produzcan. Esta es otra razón por la cual los parches son tan importantes.[5]

G. Escalada de privilegios

Del mismo modo, las bases de datos con frecuencia exponen vulnerabilidades comunes que permiten a un atacante escalar privilegios en una cuenta de privilegios bajos hasta tener acceso a los derechos de un administrador. A medida que estas vulnerabilidades son descubiertas, los proveedores las corrigen y los administradores deben mantener las actualizaciones y parches actualizados.

H. Ataque de denegación de servicio

El caso del SQL Slammer es siempre un ejemplo muy esclarecedor de cómo los atacantes pueden utilizar las vulnerabilidades de los DBMS para derribar los servidores de base de datos a través de un alto flujo de tráfico. Aún más ilustrativo es el hecho de que cuando el Slammer atacó en 2003, un parche ya estaba por ahí que se dirigió a corregir la vulnerabilidad por la que se generó su ataque. Hoy en día siete años más tarde, SQL Slammer todavía está dando dolores de cabeza en los servidores no actualizados.[6]

I. Bases de datos sin actualizar.

Esto podría sonar repetitivo, pero vale la pena repetirlo. Los administradores de base de datos a veces no aplican un parche en el momento oportuno porque tienen miedo de este dañe sus bases de datos. Pero el riesgo de ser hackeado hoy es mucho más alto que el riesgo de aplicar un parche que descomponga la base de datos. Además existen ante esos temores los backups y las réplicas. Quizás este punto pudo haber sido válido hace cinco años,

pero los proveedores ahora sin encriptar los datos sensibles en reposo y en movimiento [7].

J. Datos sensibles sin cifrar, tanto en reposo como en movimiento.

Tal vez sea una obviedad, pero las organizaciones no deben almacenar los datos sensibles en texto plano en una tabla. Y todas las conexiones a la base de datos siempre que manejen datos sensibles deben utilizar el cifrado [6].

2.1.2. Antecedentes a nivel nacional

Según el estudio realizado por la escuela de Informática de la Universidad Nacional de Trujillo – Trujillo – Perú; titulado "ATAQUE Y SEGURIDAD A LA BASE DE DATOS", menciona que La gran mayoría de los datos sensibles del mundo están almacenados en sistemas gestores de bases de datos comerciales tales como Oracle, Microsoft SQL Server entre otros, y atacar una bases de datos es uno de los objetivos favoritos para los criminales.

Esto puede explicar por qué 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.

Para empeorar las cosas, según un estudio publicado en febrero de 2012 TheIndependent Oracle UsersGroup (IOUG), casi la mitad de todos los usuarios de Oracle tienen al menos dos parches sin aplicar en sus manejadores de bases de datos.

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.

A continuación mencionaremos los puntos importantes de los estudios realizados

A. ATAQUE Y SEGURIDAD DE SESSION

SessionHijacking (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, aunque en este artículo nos centraremos en las que tienen que ver con las vulnerabilidades de las sesiones y en algunas técnicas para mitigarlas.

Como pudimos ver en el artículo sobre el funcionamiento de las sesiones, 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.

a). 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:

- 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.

- El historial del navegador.

- El referrer, que es un encabezado que envían muchos navegadores a las páginas 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 instrucción:

ini_set('session.use_only_cookies', 1);

b). Robo en servidor compartido:

Si tenemos nuestra página web alojada en un servidor compartido, los archivos físicos de las sesiones se guardan, por defecto, en un directorio común para todas las páginas web del servidor. Esto quiere decir que todas las personas que tengan su página web en ese mismo servidor, tienen acceso a todos los archivos de sesiones. Dado que el nombre de los archivos es "sess_" más el identificador de sesión, cualquier atacante tendrá una lista de identificadores de sesión válidos con sólo leer la lista de archivos del directorio común.

Esta vulnerabilidad se puede combatir de dos formas:

- Usando la función session_save_path para guardar los archivos de sesión de la página web en un directorio dentro de su cuenta al que sólo pueda acceder PHP (ya sea por estar fuera del directorio web o con un .htaccess con la instrucción denyfromall). Este método no es demasiado fiable, ya que el resto de usuarios seguirá pudiendo leer en ese directorio, sólo tienen que averiguar su localización.

- Guardando las sesiones en base de datos en lugar de en archivos. Esto se consigue fácilmente usando la función session_set_save_handler. Esta solución es la más segura ya que la página web será la única que tendrá acceso a la base de datos y, por tanto, a las sesiones.

c). Robo por 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 evitando los ataques XSS) haciendo 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:

ini_set('session.cookie_httponly', 1);

Métodos de prevención generales:

Además de los métodos de prevención concretos vistos para cada tipo de ataque, vamos a ver algunas técnicas de prevención que ayudan a evitar el robo de las sesiones:

- Limitar tiempo de inactividad: eliminar la sesión si está cierto tiempo sin ser usada (de 5 a 30 minutos, según el nivel de seguridad de la página web).
- Cambiar el identificador de sesión: cada cierto tiempo o después de cada acción, cambiar el identificador de la sesión por otro distinto y eliminar la sesión antigua.
- Sistema de logout: dar a los usuarios una forma de salir de su cuenta y destruir la sesión.
- Verificación doble: usar un segundo método para intentar reconocer al usuario de la sesión. Esto puede hacerse guardando cabeceras como HTTP_USER_AGENT (navegador del usuario) o REMOTE_ADDR (IP del usuario) cuando se crea la sesión, de esta forma:

```
$_SESSION['fingerprint'] =
md5($_SERVER['HTTP_USER_AGENT']);
```

La IP del usuario es más significativa que su navegador, pero es más problemática ya que hay usuarios a los que les cambia la IP habitualmente (IP dinámica, proxies,...).
Con este sistema, un atacante tendría que robar la sesión a un usuario y, además, enviar la misma cabecera de navegador para poder usarla.
Hasta aquí el artículo sobre el robo de sesiones. En próximos artículos veremos más tipos de ataque contra las sesiones y formas de prevenirlos.

B. ATAQUE POR INYECCION DE CODIGO

La inyección SQL consiste en la modificación del comportamiento de nuestras consultas mediante la introducción de parámetros no deseados en los campos a los que tiene acceso el usuario.
Este tipo de errores puede permitir a usuarios malintencionados acceder a datos a los que de otro modo no tendrían acceso y, en el peor de los casos, modificar el comportamiento de nuestras aplicaciones.
Vamos a ver con un ejemplo que significa eso de "Inyección de código":
Supongamos que tenemos una aplicación Web (realizada en ASP por sencillez) en la que el acceso a ciertas secciones está restringido. Para restringir ese acceso creamos una tabla de usuarios y contraseñas y sólo los usuarios que se validen contra esa tabla podrán acceder a esos contenidos.
Una manera de que los usuarios se validen será colocar un par de cuadros de texto en nuestra página Web (por ejemplo txtUsuario y txtPassword) donde puedan introducir su nombre y su contraseña y enviar ese par usuario/contraseña a la base de datos para comprobar si es válido.

Primero creamos la tabla que vamos a usar y la rellenamos con datos:

Use web

-Nuestra base de datos se llama web

go

- Creamos una tabla para almacenar los pares usuario/contraseña

create table usuarios (Usuariovarchar (50) not null primary key,

Passwordvarchar (50))

go

- Introducimos un par de datos de prueba

insert into usuarios (Usuario, Password) values ('Admin', '1234')

insertinto usuarios (Usuario, Password) values ('Usuario', 'abcd')

Ahora veamos el código de las páginas que forman parte del proceso de login.

index.htm

```
<form action="login.asp" method="post">
Nombre: <input type="text" name="txtUsuario"><br>
Password: <input type="password" name="txtPassword"><br>
<input type="submit">
</form>
```

Esta primera página es sencilla. Simplemente

Los dos cuadros de texto mencionados que enviarán los datos a la página de login.

Login.asp

```
<%
Dim Usuario, Password, RS, SSQL
```

```
Usuario = Request.Form("txtUsuario")
Password = Request.Form("txtPassword")
SSQL = "SELECT count(*) FROM Usuarios WHERE Usuario = '"
& Usuario&
"' AND password='" & Password & "'"
Set RS = Server.CreateObject("ADODB.Recordset")
RS.Open SSQL, "Cadena de conexion"
if (RS.EOF) Then
Response.Write "Accesodenegado."
Else
Response.Write "Te has identificado como " &RS("Usuario")
End If
Set RS = Nothing
%>
```

Y en esta segunda página creamos dinámicamente una sentencia SQL que enviamos a la base de datos para la validación.

Si el usuario escribe Admin y 1234 la sentencia creada será:

```
"SELECT Count(*) FROM Usuarios WHERE Usuario='Admin' AND
Password='1234'"
```

Y como esta sentencia nos devuelve un registro, dejaremos que el usuario entre en la Web. Si el usuario escribe por ejemplo 'Admin' y de contraseña cualquier otra cosa, la sentencia no nos devolverá registros y no permitiremos entrar a esa persona. Pero ¿qué ocurre si el usuario escribe ' or '1'='1 como usuario y lo mismo de contraseña? En este caso la variable Consulta contendrá la cadena:

```
"SELECT Count(*) FROM Usuarios WHERE Usuario = '' or '1'='1'
AND
password = '' or '1'='1'"
```

Y obviamente esta sentencia nos devuelve registros con lo que el usuario entrará en nuestra Web sin tener permiso.
Pero esto no es lo peor. Lo peor será el usuario utilice estos trucos de inyección de SQL para ejecutar código arbitrario en nuestro servidor. Sentencias DDL, cambiar permisos, utilizar procedimientos almacenados y un largo etcétera. Qué ocurriría si alguien escribiese de contraseña cosas como:
' exec master..xp_cmdshell 'net user test /ADD' – Como evitarlo:

Y ahora lo más importante, ¿qué podemos hacer para evitar estos errores?
Pues hay varios sistemas para evitarlo. Por ejemplo podemos filtrar las entradas de los usuarios reemplazando la aparición de ' por '' (dos comillas simples) e incluso evitando que los usuarios puedan pasar caracteres como \ / " ' o cualquier otro que se nos ocurra que puede causar problemas. Estos filtros pueden ser tan sencillos como utilizar la sentencia replace de Visual Basic:

```
SSQL= "SELECT count(*) FROM Usuarios WHERE Usuario = '" &
Replace
txtUsuario.Text, "'", "''") & "' AND password='" & Replace
(txtPassword.Text, "'", "''") & "'"
```

Otro factor importante en cuanto a la seguridad es limitar al máximo los permisos del usuario que ejecuta estas sentencias para evitar posibles problemas. Por ejemplo utilizando un usuario distinto para las sentencias SELECT, DELETE, UPDATE y asegurándonos que cada ejecución de una sentencia ejecute una sentencia del tipo permitido. Por supuesto utilizar el usuario 'sa' o uno que pertenezca al rol 'db_owner' para ejecutar las sentencias de uso habitual de la base de datos debería quedar descartado.

Una solución definitiva sería trabajar con procedimientos almacenados.

El modo en el que se pasan los parámetros a los procedimientos almacenados evita que la inyección SQL pueda ser usada. Por ejemplo utilizando el siguiente procedimiento almacenado:

```
CREATE Procedure Validar @usuario varchar(50), @passwordvarchar(50)
AS
If (SELECT Count(*) FROM Usuarios WHERE Usuario=@Usuario and Password=@password)>0
Return 1
Return 0
```

También deberíamos validar los datos que introduce el usuario teniendo en cuenta por ejemplo la longitud de los campos y el tipo de datos aceptados. Esto lo podemos hacer en el cliente con los RegularExpressionValidator o con los CustomValidators del VB.NET. De todos modos si la seguridad es importante todas estas validaciones hay que repetirlas en el servidor.

Por último, y ya que estamos pensando en entornos Web, podemos programar en ASP.NET y utilizar siempre que sea posible las clases System.Web.Security.FormsAuthentication para que los usuarios entren en nuestras aplicaciones Web.

C. ATAQUE POR INYECCION DE COMANDOS EN PHP

En esta ocasión hablaremos de una gran problemática que existe al momento de desarrollar con php y se utilizan funciones como system, exec, etc.

Estas funciones tiene la facultad de ejecutar desde php comandos en sistema operativo, el problema es que si no tenemos las

validaciones correctas, nos pueden hacer una inyección de código y ejecutar comandos arbitratiamente en el servidor.

Un ejemplo práctico:

Este programa lo que hace es tomar una variable por médotoget (también aplica para post con los cambios necesarios) y crea un directorio

```
<?
if (trim($_GET['directorio'])!=NULL) {
system ("mkdir {$_GET['directorio']}");
echo "directorio creado {$_GET['directorio']}";
}
else
echo "directorio vacio";
?>
```

NOTA IMPORTANTE: Se actúa en el supuesto de que el directorio donde se encuentra el script inicial tiene permisos 777 o pertenece al usuario que ejecuta el servicio de apache (generalmente apache), para poder crear los directorios.

En la url del navegador se escribirá algo como sigue:

http://misitio.net/dirs/script.php?directorio=directorionuevo

("directorionuevo" es el directorio recién creado)

En la línea del código:

system ("mkdir {$_GET['directorio']}");

Se ejecuta un comando de sistema operativo con la función system para crear el directorio.

Si sustituimos la ejecución del script desde el navegador en la función system tendremos:

mkdirdirectorionuevo

Nosotros sabemos que en linux/unix, podemos ejecutar más de un comando en una misma línea, por ejemplo:

mkdirdirectorionuevo; pwd

Bien, ahora nosotros podemos hacer lo mismo desde la url del navegador:
http://misitio.net/dirs/script.php?directorio=directorionuevo;%20pwd

Con esto nos creará el directorio "directorionuevo" como era lo esperado, y además, nos dará la ruta desde donde se está ejecutando el script (comando pwd).
/var/www/html/dirs/

Nota: El "%20" representa un espacio en blanco, en algunos navegadores ya no es necesario poner éste código, sino simplemente se debe colocar el espacio en blanco.

Además podemos hacer otras cosas como buscar directorios o archivos con permisos 777:
http://misitio.net/dirs/script.php?directorio=directorionuevo;%20find%20-perm%20777

Ver el archivo de /etc/password, que a futuro nos puede servir para saber que usuarios válidos tiene y hacer un ataque bruteforce:
http://misitio.net/dirs/script.php?directorio=directorionuevo;%20cat%20/etc/passwd

Lo más preocupante es que se puede descargar malware a nuestro servidor, para hacerlo parte de una botnet para atacar a otros equipos o simplemente mandar spam:
http://misitio.net/dirs/script.php?directorio=directorionuevo;%20wget%20http://sitiomalo.org/malware.txt

El archivo malware.txt se descargará y posteriormente podrá ser ejecutado.

Solución:

Lo anterior es de gran preocupación porque se pueden hacer varias cosas más como borrar archivos, nuestro server puede ser víctima de un ataque DOS, etc.

Aquí presentamos algunas soluciones:

•En este caso específico, usar la funcionmkdir de php, en vez del comando system para ejecutar un comando de sistema operativo

•No usar:

funcionesshell_exec,exec,system,readfile,passthru,escapeshellcmd, proc_open,posix_uname,posix_getuid,posix_geteuid, posix_getgid,getcwd para comandos en sistema operativo

•Si se administran host virtuales, en el archivo php.ini agregar las funciones anteriores en la directiva "disable_functions"

•Revisar en access_logcomportamnientos extraños y por medio de fw bloquear las ip's

•También para evitar la inyección de wget, GET, curl, y demás comandos para descargar software se puede agregar un script en el php.ini en la directiva, auto_prepend_file para que cuando se detecten en la url de cualquier script php, se omita la ejecución para no descargar el software.

```
<?
foreach ($_GET as $variable => $valor)
if (eregi("wget |curl |GET ", $valor)) {
exit
}
?>
```

Lo que hace el script es recorrer el arreglo GET (todas sus variables), a fin de que se detecten esos comandos y se evite la descarga del malware.

Hay software del que ya se conocen vulnerabilidades de este tipo como Mambo y PhpNuke, por lo que es recomendable no usarlos.

.

2.2. Conceptos fundamentales

Protección de puerta trasera es defecto en un software o página web que permite ingresar a un recurso que usualmente está restringida a un usuario ajeno. No siempre es un defecto, también puede ser una entrada secreta de los programadores o webmasters con diversos fines. Ej: una página web tiene clave de acceso para ingresar, pero existe una determinada dirección que nos permite entrar saltando la misma.

2.3. Marco situacional

En la actualidad, se está haciendo cada vez más importante entre los usuarios la conciencia respecto a que la seguridad en una cuestión relevante para las aplicaciones web; basta consultar la página de Preguntas Frecuentes de una aplicación típica, donde se alienta al usuario a considerar que se trata de una aplicación segura; por ejemplo: "*Este sitio es absolutamente seguro. Ha sido diseñado para utilizar tecnología 128-bit SSL a fin de evitar que usuarios no autorizados visualicen su información. Ud. puede utilizar este sitio con la tranquilidad de que sus datos están seguros con nosotros.*". Como vemos, las aplicaciones web declaran ser seguras debido a que utilizan SSL (*Secure Socket Layer*); asimismo, se les propone a los usuarios verificar el certificado del sitio, informarse sobre los protocolos de criptografía avanzada que se emplean, y a partir de todo ello, se los alienta a confiarles su información personal. Pero de hecho, las mayoría de las aplicaciones web son inseguras debido a condiciones que no

tienen ninguna relación con SSL (de esto, no se tiene que inferir que es innecesario para la seguridad de dichas aplicaciones, ya que utilizado adecuadamente, provee una efectiva defensa contra varios ataques significativos). Diferentes fuentes indican que, dado los resultados de testeos sobre cientos de aplicaciones web realizados en los dos últimos años, porcentajes elevados de ellas están afectadas por algunas categorías comunes de vulnerabilidades, las cuales se describen brevemente a continuación:

a). *Quiebre de la autenticación* (67%). Esta categoría de vulnerabilidad comprende diferentes defectos dentro del mecanismo de inicio de sesión de la aplicación, lo cual puede permitir que un atacante adivine contraseñas débiles, lance un ataque de fuerza bruta, o eluda por completo el proceso de inicio de sesión.

b). *Quiebre de los controles de acceso* (78%). Esto comprende los casos en que la aplicación fracaza en proteger adecuadamente el acceso a sus datos y sus funcionalidades, potencialmente permitiendo que un atacante visualice datos sensibles de otro usuario hospedados en el servidor, o lleve a cabo acciones privilegiadas.

c). *Inyección de SQL* (36%). Esta vulnerabilidad permite que un atacante envíe una entrada con un contenido tal que provoque una interferencia en la interacción de la aplicación con las bases de datos del back-end, y así ser capaz de recuperar datos arbitrarios vía la propia aplicación, interferir en la lógica de la misma, o ejecutar comandos sobre el servidor de base de datos.

d). *Cross-site scripting* (91%). Esta vulnerabilidad permite que usuarios de la aplicación sean blancos de un atacante, el que logra acceder a los datos de dichos usuarios, ejecutando acciones no

autorizadas en nombre de los mismos, o desencadenando ataques contra ellos.

e). ***Fuga de información*** **(81%).** Comprende los casos en que una aplicación divulga información sensible, la que es utilizada por un atacante para desencadenar un ataque contra la aplicación, mediante el manejo de un error derivado de un defecto.

En el ítem mencionado anteriormente muestra los tipos de vulnerabilidad en el acceso a la web. Para ello tratare de resolverlos teniendo en consideración el ítem mencionado y pruebas correspondientes con hechos reales

2.4. Definición de términos

2.4.1. Servicio Web:

Es una tecnología que utiliza un conjunto de protocolos y estándares que sirven para intercambiar datos entre aplicaciones. Distintas aplicaciones de software desarrolladas en lenguajes de programación diferentes, y ejecutadas sobre cualquier plataforma, pueden utilizar los servicios web para intercambiar datos en redes de ordenadores como Internet

2.4.2. Puerta Trasera:

En un sistema informático, es una secuencia especial dentro del código de programación, mediante la cual se pueden evitar los sistemas de seguridad del algoritmo (autentificación) para acceder al sistema. Aunque estas puertas pueden ser utilizadas para fines maliciosos y espionaje no siempre son un error, pueden haber sido diseñadas con la intención de tener una entrada secreta

2.4.3. Base de Datos:

Son los datos que se almacenan en tablas, donde cada fila identifica unívocamente a un elemento distinto. Para acceder a ellas se realizan mediante consultas de SQL (Lenguaje estructurado de consultas).

2.4.4. Algoritmo:

Es un conjunto prescrito de instrucciones o reglas bien definidas, ordenadas y finitas que permite realizar una actividad mediante pasos sucesivos que no generen dudas a quien deba realizar dicha actividad. Dados un estado inicial y una entrada, siguiendo los pasos sucesivos se llega a un estado final y se obtiene una solución

2.4.5. Inyección SQL

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 consultas a una base de datos.

El origen de la vulnerabilidad radica en el incorrecto chequeo y/o filtrado de las variables utilizadas en un programa que contiene, o bien genera, código SQL. Es, de hecho, un error de una clase más general de vulnerabilidades que puede ocurrir en cualquier lenguaje de programación o script que esté embebido dentro de otro.

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.

CAPITULO III

MARCO METODÓLOGICO

3.1. Tipo y nivel de investigación

Fundamentando los propósitos de la investigación y la naturaleza importante de las necesidades y problemas detectados, la investigación es de Acción exploratorio y Desarrollo, el mismo que se constituye como un método de estudio y acción de tipo cualitativo, que busca obtener resultados fiables y útiles para mejorar situaciones colectivas, basando la investigación en la exploratoria. Cabe mencionar según Roberto Hernández Sampieri, afirma que los tipos de estudios que no puede establecer hipótesis son los exploratorios [9]. Por otra parte la investigación exploratorio es aquella que se efectúa sobre un tema u objeto desconocido o poco estudiado, por lo que sus resultados constituyen una visión aproximada de dicho objeto, es decir, un nivel superficial de conocimientos [10].

3.2. Métodos de investigación

El proyecto utilizará dos tipos de metodología, la primera relacionada a la investigación y la segunda al desarrollo del algoritmo.

3.2.1. Investigación

En el estudio se tomó en cuenta páginas gubernamentales de Huánuco que utilizan CMS Joomla desde la versión 1.5 en adelante.

3.2.1.1. Arquitectura MVC de Joomla

Joomla! es un CMS (sistema de gestión de contenidos) famoso en todo el mundo que utiliza la arquitectura MVC (Modelo-Vista-Controlador) para construir sus componentes. Ha estado ahí desde hace muchos años y ha pasado por muchos cambios para encajar en diferentes tipos de necesidades. Hace muy poco, los desarrolladores Joomla! tuvieron la idea de hacer esta arquitectura más flexible para superar los retos del futuro. Como resultado de esto, este nuevo MVC se introduce en el Joomla!.

En las versiones de 2.5 en adelante ha surgido algunos cambios en el MVC, por el tema de mejoramiento y optimización de códigos y sobre todo de fácil administración del CMS.

3.2.1.1.1. Primer vistazo al Nuevo MVC

La Estructura de MVC de Joomla, aunque en las versiones anteriores no difiere mucho en el reordenamiento, la idea se mantiene.

Nueva estructura de carpetas

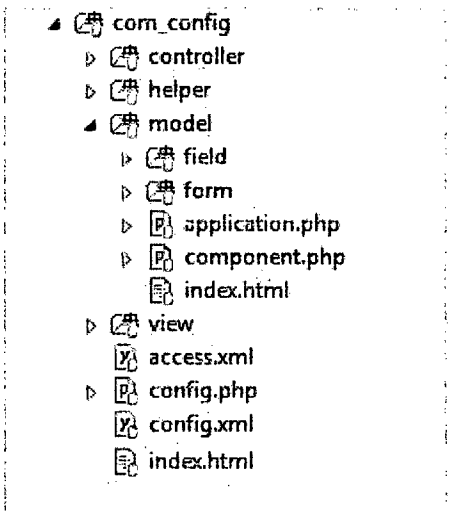


Imagen 02: Estructura de Carpetas de Joomla

Es prácticamente igual a la arquitectura del MVC existente. Pero los nombres de carpetas son singulares. Así que en el nuevo MVC se espera que los usuarios utilicen nombres de carpetas distintos.
Fichero único para el controlador.

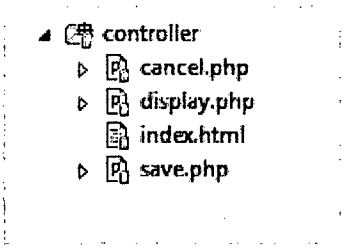


Imagen 03: Archivos del controlador de Joomla

Este es un cambio importante en el nuevo MVC. El nuevo MVC tiene un archivo independiente para cada controlador. Hay ventajas especiales en este diseño que serán discutidas en este artículo más adelante. En el MVC existente sólo hay un archivo que contiene todas estas tareas de control.
Método único para el controlador

```
class ConfigControllerApplicationSave extends JControllerBase
{
    /**
     * Method to save global configuration.
     *
     * @return  bool    True on success.
     *
     * @since   3.2
     */
    public function execute()
}
```

Imagen 04: Código del controlador de Joomla

Dentro de un controlador, la estructura es muy simple. Solo hay un método, llamado "ejecutar". La funcionalidad completa de dicho controlador se agrega a ese método "ejecutar". Este cambio hace de los controladores algo muy fácil de entender y de acceder.

Nuevas librerías para MVC

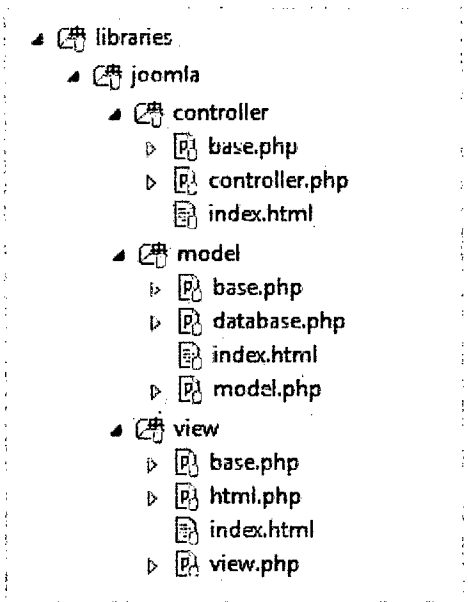


Imagen 05: librerías nuevas de MVC de Joomla

Para el nuevo MVC, las librerías/carpetas Joomla contiene 3 nuevas subcarpetas: controlador, modelo y vista. Cada carpeta contiene su respectiva clase Base, que debe ser utilizada en la implementación de las clases del controlador, el modelo y la vista de los componentes. Actualmente éstos se incluyen dentro de la última release del CMS. Estas clases a nivel de librería son la base de cualquier nuevo componente MVC [11]. La principal ventaja de esto es reducir la redundancia de código en las clases heredadas, que se utilizan en el MVC existente.

Conforme pasa el tiempo, la clase más reutilizable se añadirá a las librerías. Puedes ver algunos ejemplos de posibles controladores reutilizables en la carpeta del controlador: com_config[12] o la carpeta del cms: com_cache-com_checkinrepository[13]. Gracias a la carga automática de clases, en algunos casos, los componentes no requerirán ningún controlador propio o no uno por clase. Además verás que las clases de la Vista también se vuelven muy simples y

las clases del Modelo son similares a las existentes en el actual MVC. En el nuevo MVC la clase del Modelo no tiene por qué ser hija de la clase JModelBase, pero también puede ser una clase de tabla en función de la exigencia.

3.2.1.1.2. Ventajas del nuevo MVC

Al crear un componente Joomla! usando el nuevo MVC, como desarrollador detectarás los siguientes puntos:

Fact	Existing MVC	New MVC
SupportFor Web Services	No	Yes
Learnability	Hard	Easy
Controllers, Models and Views	FairlyComplex	Very Simple
Extendusing	Clases Legacy	Clases Base
Number of class files	Menos	Varía en el componente

Cuadro 01: Componentes del MVC

En este momento los siguientes componentes se han transferido a un nuevo MVC y están esperando en la cola de peticiones de extracción.

- com_config
- com_checkin
- com_cache

3.2.1.1.3. Administración del Front-end

Ahora que se ha mostrado la importancia del nuevo MVC. Ahora es posible que se vea algunos logros especiales de este cambio. La administración del front-end es una de esas cosas que fácilmente

se puede implementar con el nuevo MVC. A través de la administración del front-end, los Usuarios Joomla! tendrán la oportunidad de hacer varios cambios sin acceder al backend en absoluto. Creo que esto sería muy útil para atraer a los usuarios novatos, ya que el back-end (vista administrativa) contiene paneles de configuración muy complejos.

Como paso inicial, crear con éxito la vista de administración del front-end para la Configuración Global y el Gestor de plantillas. A continuación se encuentran algunas de las características que ofrecen los componentes de administración del front-end:

A. Cambiar el nombre del sitio
B. Cambiar el logo del sitio
C. Cambiar el color de la plantilla
D. Cambiar el color de fondo
E. Poner el sitio online/offline
F. Añadir Meta descripciones

Con el tiempo habrá más funcionalidades que se añadan a la administración de front-end.

3.2.1.2. La arquitectura del sistema Joomla

Joomla se basa en una serie de elementos que se podrá crear, editar y configurar en el Panel de Control de la administración del sitio y que tienen reflejo visual en el Frontend del sitio.

La instalación básica de Joomla, la que se realiza, se muestra alguna de estas opciones. Si se navega por los menús del Panel de Control se puede ver los términos Artículos, Componentes, Menús, Plugins, Módulos o Plantillas. ¿Qué significa todo esto y dónde están en realidad en mi sitio?

3.2.1.2.1. La arquitectura del sistema Joomla

Son el elemento fundamental de Joomla. Joomla es un gestor de contenidos y el artículo es el contenido estrella. Incluye un título, texto y/o elementos multimedia diversos (imágenes, reproductores flash, enlaces a documentos, vídeos, presentaciones...) y diversos parámetros de configuración.

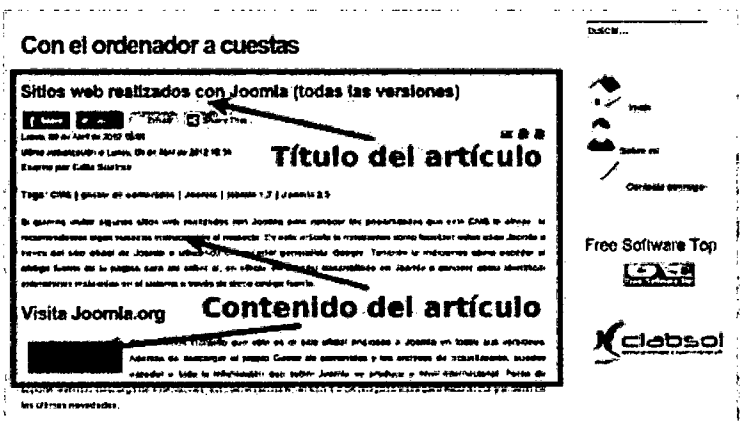


Imagen 06: Gestor de contenidos de Joomla

No vamos a entrar en demasiados detalles por ahora, pero te adelantamos que Joomla puede mostrar un articulo

individualmente, un listado de artículos, puedes configurar que el artículo esté publicado o no (como guardado en la recámara hasta un momento mejor). Puedes hacer que un artículo sea destacado para que se muestre en la portada del sitio o página principal o no, archivarlos, enviarlos a la papelera, recuperarlos más adelante, reutilizarlos, copiarlos, moverlos de sitio, etc.

3.2.1.2.2. Componentes

La arquitectura de Joomla, aunque simple, puede convertirse en extremadamente compleja. Cuando se instale el sistema lo realiza con lo más elemental, pero Joomla está preparado para que se le añada nuevas funcionalidades instalando pequeñas aplicaciones que así lo permitan. Son las llamadas extensiones. Una de estas extensiones es, precisamente, ésta de la que hablamos, los componentes.

Todos los componentes instalados se listan en el menú Componentes. En la instalación básica de Joomla ya vienen preinstalados unos cuantos que sirven a los propósitos generales de cualquier página web: un buscador, un sistema de enlaces, feeds RSS...

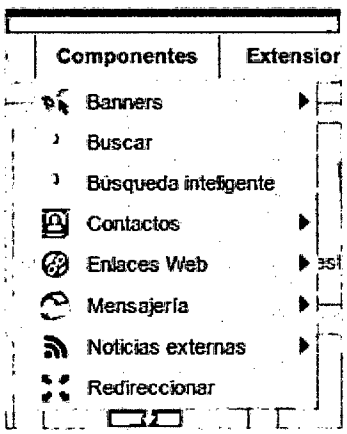


Imagen 07: Componente en gestor de contenidos de Joomla

Pero existen cientos de componentes que se podrá instalar en el sitio para multitud de propósitos: desde componentes sencillos que permiten visualizar imágenes o vídeos hasta componentes complejos con los que se podrá montar un auténtico portal de servicios, un portal de ventas o incluso montar una propia red social. Recuerda que se puede localizar físicamente en la carpeta components de Joomla 2.5

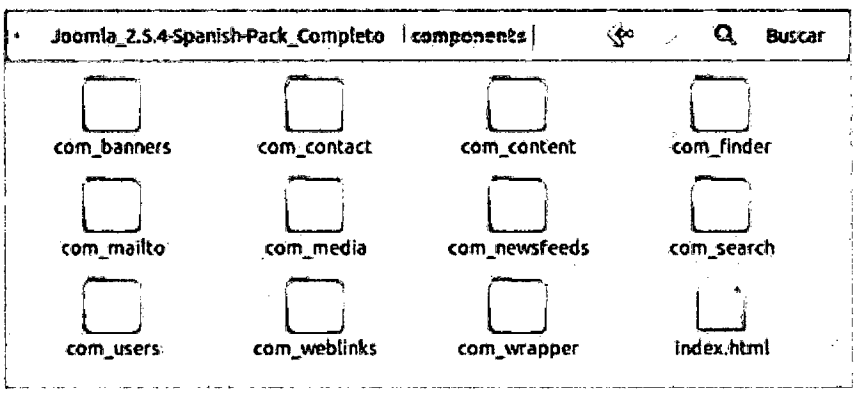


Imagen 08: Componentes de Joomla

3.2.1.2.3. Módulos

Si te fijas la mayoría de los sitios Web realizados con Joomla tienen una estructura muy similar. En la zona central se muestra el contenido principal (los artículos de un blog, las últimas noticias...) y, a su alrededor, multitud de cajitas o contenedores, mostrando información adicional como el formulario de ingreso para los usuarios o el menú de navegación por las distintas páginas y secciones de la página.

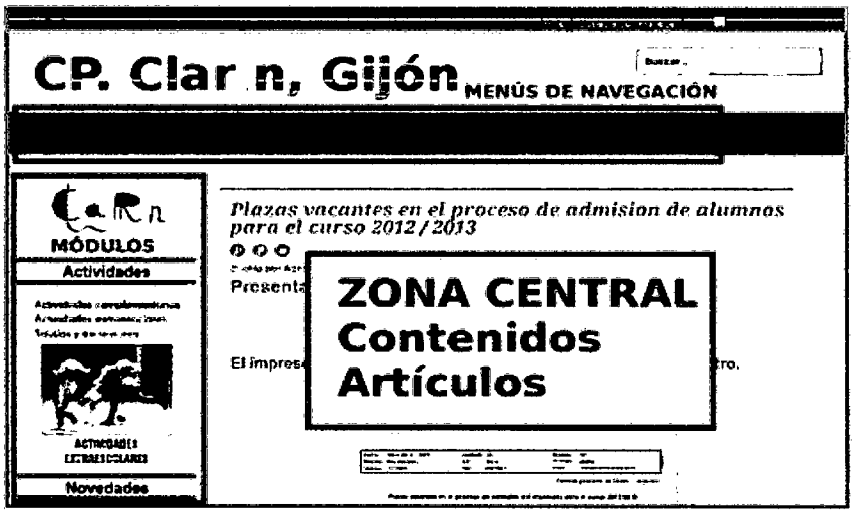


Imagen 09: páginas realizadas en Joomla

Todos estos contenedores son los módulos. Son, por tanto, bloques de contenido independientes que pueden ser colocados de manera flexible a lo largo del portal usando las posiciones que se muestran predefinidas en la plantilla que se está utilizando. Ya sé que, de momento, estos son conceptos un tanto complicados de entender a estas alturas, pero la idea de que en el Panel de Control se podrás crear, editar y configurar diversos módulos y que estos módulos tendrán su reflejo en el Frontend del sitio en esa especie de contenedores o cajones que se muestran alrededor de la zona central.

Todos los módulos se administran desde la sección **Extensiones – Gestor de módulos**:

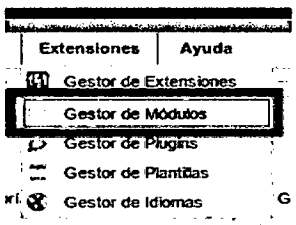


Imagen 10: Gestor de módulos en Joomla

Tal y como ocurre con los componentes, Joomla ya viene con una cantidad concreta de módulos disponibles en su instalación básica pero también se podrá instalar numerosos módulos disponibles para múltiples funcionalidades. Recuerda que se puede localizar físicamente en la carpeta modules de Joomla 2.5

Si a todo ello se le añade la flexibilidad de poder ubicar los módulos en diversos lugares de la página, entenderás que Joomla 2.5, ofrece enormes posibilidades a la hora de poder diseñarla, incluso a posteriori, cuando el sitio ya esté en marcha. Y todo sin tener apenas conocimientos informáticos y sin necesidad de tocar código. ¡Extremadamente sencillo!

3.2.1.2.4. Plugins

Son pequeños scripts dedicados a hacer tareas en el sistema, en principio, sencillas. Por ejemplo, un plugin puede dedicarse a crear automáticamente imágenes en miniatura en un artículo que enlazan a la versión completa de la imagen, en su tamaño original. En este mismo sitio con el ordenador a cuestas se tienes ejemplos concretos de esto. Pero también existen plugins muy complejos como el plugin de caché, que permite mejorar el rendimiento de la página web.

Todos los plugins se administran desde el menú **Extensiones – Gestor de plugins**:

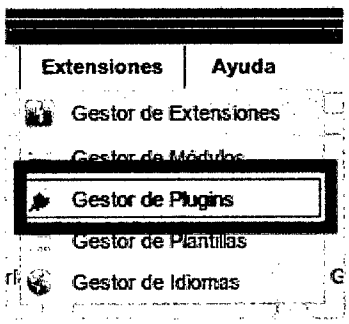


Imagen 11: Gestor de Plugins en Joomla

Y tal y como sucede con Componentes y Módulos, la instalación básica de Joomla ya incluye varios plugins preinstalados, pero se podrá instalar muchísimos más según diferentes propósitos. Recuerda que físicamente los plugins se encuentran en la carpeta plugins de Joomla 2.5.

3.2.1.2.5. Las plantillas

Las plantillas, diseños, themes o templates son el esqueleto estético de la Web. Son las responsables del diseño estético del sitio: los colores, los formatos de fuente, los lugares en los que puedes ubicar los módulos (las posiciones de la plantilla)... todo está definido en la plantilla que se esté utilizando en cada momento [14].

Joomla es un gestor de contenidos que se preocupa precisamente de eso, de los contenidos. El diseño del sitio, su aspecto, es relativamente importante [15], tanto es así que no se tendrá que ocupar de eso, si no se desea. El diseño va a depender de la plantilla que se esté utilizando en cada momento, y se podrá cambiarlo, eso sí con ciertas precauciones y restricciones, siempre que lo desees. Un sitio Joomla, con los mismos contenidos, componentes y módulos instalados puede parecer diferente simplemente desactivando la plantilla actual y habilitando una nueva. ¡Así de sencillo!

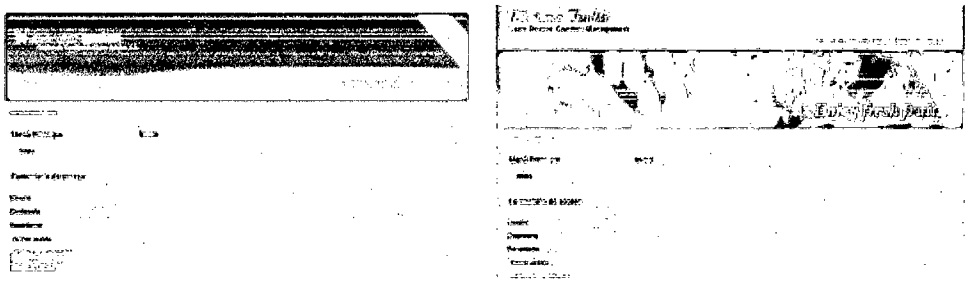


Imagen 12: Páginas de Joomla en forma tradicional

Mismos contenidos – distintas plantillas en uso – distintos diseños del sitio

La instalación básica de Joomla incluye un número muy limitado de estas plantillas pero, no hay que preocuparse, se encontrará cientos de plantillas diferentes, gratuitas y de pago, en la red que se podrá instalar en el sitio para que el aspecto de la webweb sea espectacular. Recuerda que todas las plantillas se encuentran físicamente en la carpeta templates de Joomla 2.5.

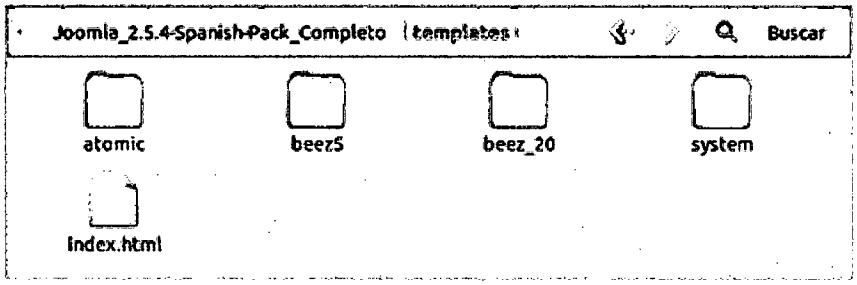


Imagen 13: plantillas de Joomla en el pack gratuito

Y si a ello se le añade distintas plantillas en diferentes páginas o secciones de la web, se entenderá la enorme flexibilidad que le proporciona Joomla también en cuestiones de diseño.

Todas las plantillas se administran desde el menú **Extensiones – Gestor de plantillas**:

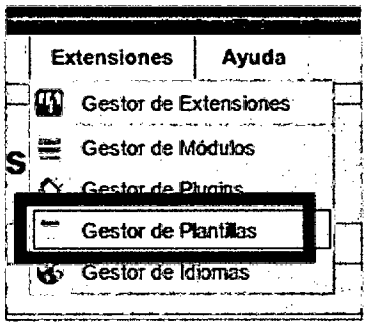


Imagen 14: Gestor de plantillas en sistema Joomla

Desde este Gestor se podrá habilitar la plantilla en uso para el sitio web, pero también se podrá configurarla según las necesidades.

3.2.1.3. La estructura de carpetas y archivos

Entender pues, la estructura que Joomla posee y cuáles son sus principales ficheros y directorios puede ayudarte a una mejor comprensión de su funcionamiento. Le indicamos algunas de sus particularidades que se tendrá que tener en consideración más adelante, cuando inicies el trabajo con este gestor de contenidos.
Y, a continuación, vamos a visualizar la estructura de Joomla 2.5:

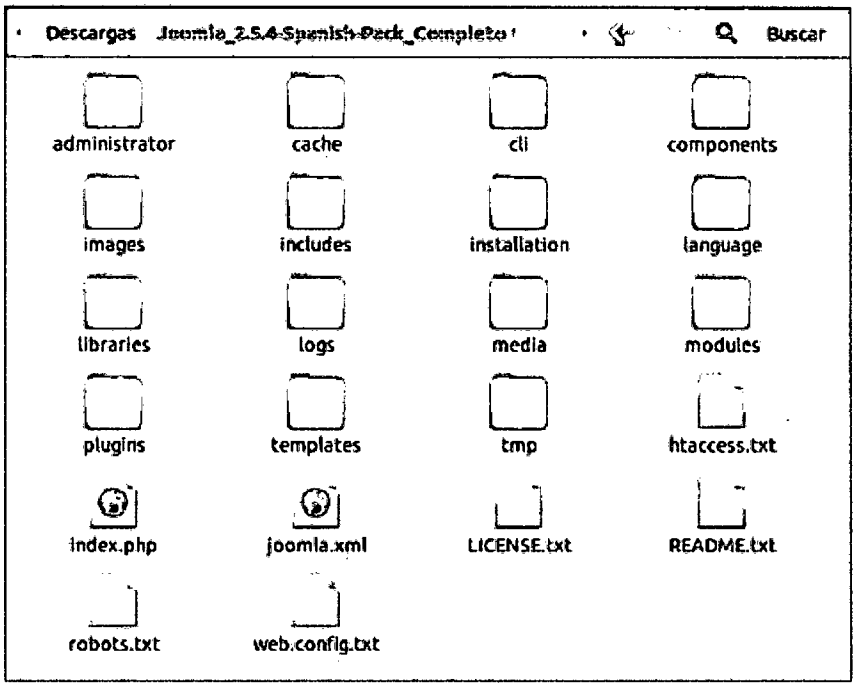


Imagen 15: Contenido completo de Joomla 2.5

3.2.1.3.1. Carpeta Administrator

Incluye básicamente todos los ficheros y directorios necesarios para el correcto funcionamiento del Backend de Joomla. El contenido de esta carpeta es accedido cuando accedes al administrador desde el navegador.

3.2.1.3.2. Carpeta Components

En esta carpeta puedes ver subdirectorios diferentes según los componentes instalados en el sistema. Como puedes apreciar la instalación básica de Joomla 2.5 ya incluye una gran variedad de componentes básicos instalados. Son los responsables de la creación, configuración y publicación de artículos (componente com_content), de la creación, configuración y gestión de banners publicitarios (componente com_banners), de las búsquedas en el sistema (componente com_search), de la gestión de los archivos multimedia (componente com_media), etc.

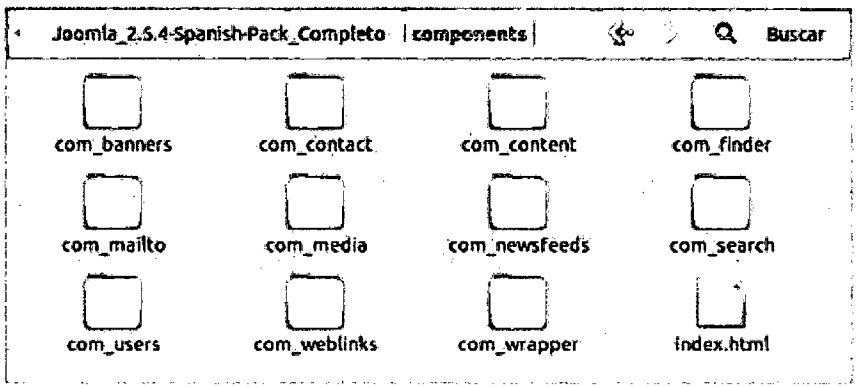


Imagen 16: Componentes de Joomla 2.5

Más adelante, cuando instales nuevos componentes en el sistema podrás ver aquí (y también en otros directorios de Joomla, dependiendo de la complejidad del componente instalado) los distintos ficheros y carpetas necesarios para cada uno de ellos. Identificarás rápidamente la carpeta asociada al componente instalado por el nombre del directorio, idéntico al del componente.

3.2.1.3.3. Carpeta images

Incluye algunas de las imágenes necesarias para el sistema como los banners, las imágenes de ejemplo o los logos de Joomla. Pero

sobre todas las cosas, lo fundamental es que **esta carpeta es la predeterminada por Joomla 2.5 para ser utilizada por el Gestor Multimedia del sistema.** En esta carpeta crearás subcarpetas específicas que puedan alojar tus propias imágenes. Una buena clasificación en ella te evitará muchos quebraderos de cabeza cuando desees enlazar imágenes en tus artículos, por ejemplo.

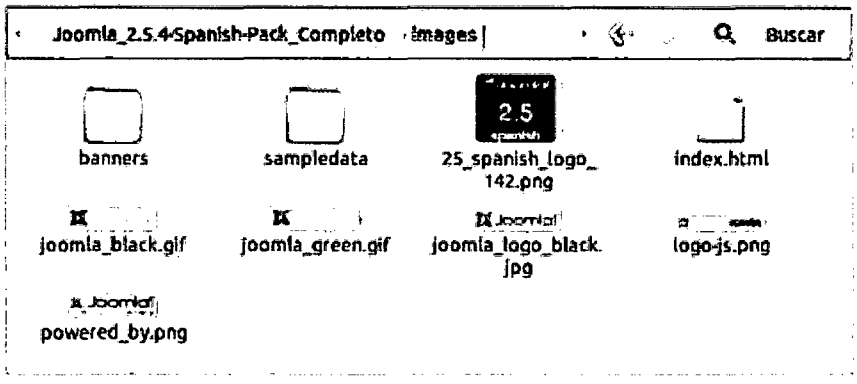


Imagen 17: Carpeta de imágenes de Joomla 2.5

Para que comprendas básicamente, de momento, su funcionamiento, te indicamos que desde el Gestor multimedia puedes crear carpetas específicas y subir a ellas las imágenes que utilices en los artículos. También esto mismo hará Joomla cuando añadas imágenes directamente en los artículos que crees mediante el botón Imagen: las incluirá de forma predeterminada en esta carpeta (aunque este botón también te permitirá crear subcarpetas donde alojarlas).

El reflejo, en cualquier caso, en la estructura de Joomla es la creación de dichos directorios en esta carpeta images o la presencia de dichas imágenes en esta carpeta.

Atención

Cuando instalas extensiones nuevas en Joomla 2.5 que trabajan con imágenes como galerías de fotografías o visores y

reproductores de imágenes tienes que considerar que los programadores - desarrolladores de la extensión utilizan la carpeta images en su configuración. ¡No lo olvides: no borres esta carpeta o te arriesgas a que estas extensiones no funcionen!

3.2.1.3.4. Carpeta modules

Se encuentran aquí los directorios y ficheros necesarios para el funcionamiento de cada uno de los módulos instalados en el sistema. Cada módulo dispone de su propia carpeta diferenciada. Puedes identificarla por su nombre.

Joomla 2.5 ya incluye preinstalados una gran variedad de módulos, que iremos viendo a lo largo de los diferentes tutoriales dedicados a la utilización de Joomla, pero también podrás instalar muchos otros en el futuro. Por supuesto, estas nuevas instalaciones crearán nuevas subcarpetas en este directorio.

3.2.1.3.5. Carpeta plugins

Similar a la anterior. Se encuentran aquí los directorios y ficheros necesarios para el funcionamiento de cada uno de los plugins instalados en el sistema. Al igual que ocurría con los anteriores, cada plugin tiene su propia carpeta diferenciada y cuando instales nuevos plugins de terceros aquí habrá nuevas carpetas específicas de cada uno de ellos.

3.2.1.3.6. Carpeta templates

Esta carpeta es la carpeta que incluye todos los archivos necesarios para las distintas plantillas que puedes utilizar en el sistema, tanto las relativas al Frontend como al Backend del sistema. De la misma forma, la instalación de nuevas plantillas en el sistema se corresponde con la aparición de nuevas carpetas en este directorio. Observa que cada carpeta se llama igual que la plantilla correspondiente. Veremos más en profundidad esta

carpeta, y sus subcarpetas, cuando te mostremos como crear, configurar, editar, instalar o desinstalar nuevas plantillas para tu sitio Joomla.

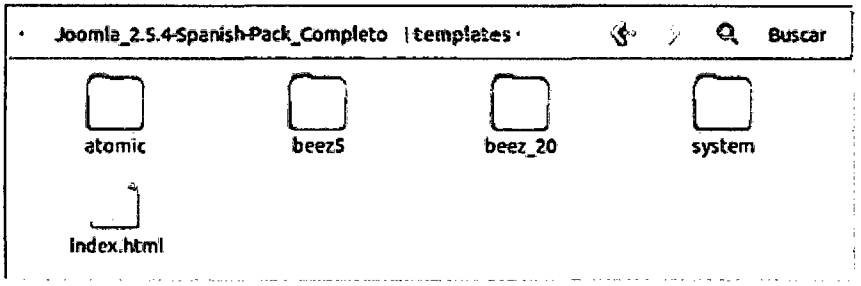


Imagen 18: Carpeta de plantillas de Joomla 2.5

3.2.1.4. El frontend y el Backend

La primera cuestión que se tiene que tener en cuenta es que, cuando se trabajas con Joomla, se debe considerar que vas a acceder a dos espacios diferentes accesibles vía web, dos espacios diferentes pero estrechamente relacionados, como verás; es decir, dispones de dos puertas de acceso a tu sitio.

3.2.1.4.1. El sitio o Frontend de Joomla

Por un lado se encuentra tu sitio, tal y como lo puede ver cualquier usuario que navegue hasta él. Es la apariencia que se muestra en pantalla cuando el usuario escribe la dirección URL del sitio en el campo Dirección del navegador. E incluye los contenidos que Google indexará.

Por ejemplo, cuando escribes en el navegador

http://postgrado.unheval.edu.pe

Accedes a este sitio que está realizado con Joomla.

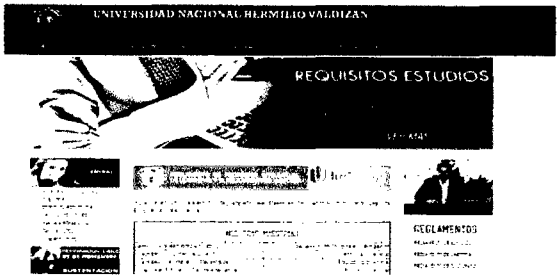


Imagen 19: Carpeta de plantillas de Joomla 2.5

En el caso del sitio Joomla instalado en el servidor local podrás ver la apariencia del sitio que se construye al inicio si se escribes la dirección siguiente en el navegador:

http://localhost/portal

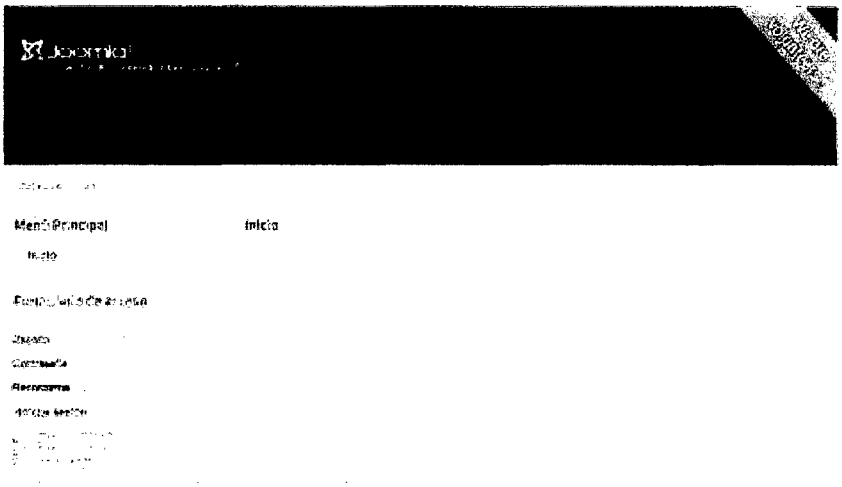


Imagen 20: vista de la plantillas de Joomla 2.5

En realidad, lo que el usuario está viendo en pantalla es el llamado Frontend del sitio Joomla. Cualquier visitante puede verlo. Este gestor de contenidos implementa un sistema de privilegios basado en identidades acreditadas (compuestas por un nombre de usuario y una clave o contraseña) mediante las cuales, y dependiendo de

los privilegios de esta acreditación, se podrá acceder o no a ciertos contenidos e interactuar con ellos según los permisos que se tenga concedidos.

3.2.1.4.2. La administración del sitio o Backend de Joomla

Es la puerta de acceso privada al sitio. En la administración se gestionan los contenidos, los usuarios que pueden acceder al sistema, la estética o apariencia del sitio y, en general, todo el funcionamiento del mismo: desde cambiarle el nombre al portal o añadir un nuevo artículo a la portada, hasta instalar una galería de fotografías o un foro completo destinado a la participación y consulta de los visitantes.

En el caso de este sitio Joomla para acceder a la administración no hay más que escribir en el campo Dirección del navegador, la dirección URL del sitio más el término administrator, es decir,

http://postgrado.unheval.edu.pe/Administrator

Nos lleva a la página de acceso a la administración del sitio a la que sólo podrás acceder si dispones de los datos de acceso que te acrediten como usuario con acceso al sistema.

En el caso de Joomla instalado en el servidor local la dirección URL que te da acceso al Backend del sitio Joomla es idéntica, es decir, la misma que la del Frontend seguida de /administrator/, es decir

http://localhost/portal/administrator

El resultado será la aparición en pantalla de una página en la que tienes que introducir un nombre de usuario y una contraseña (datos que algún administrador del sistema te habrá facilitado o bien que conoces por ser tú mismo el administrador).

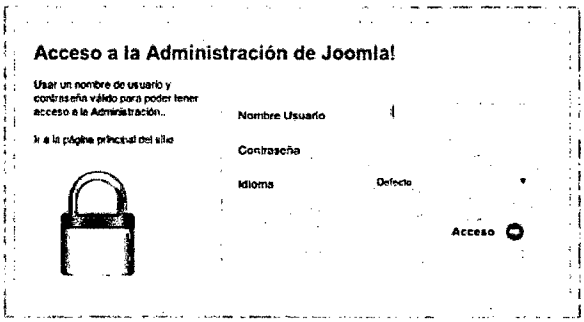


imagen 21: sistema del logeo de Joomla 2.5

Si el sistema te reconoce como usuario dado de alta en el mismo, sea cual sea la finalidad que tengas (escritor de contenidos, editor del sitio, administrador, o simple usuario registrado.) te permitirá acceder; en caso contrario, no:

Por tanto, el Backend es una página oculta al visitante del sitio. Engloba el área de administración. En ella sólo pueden entrar el administrador del sistema, es decir, tú, y aquellos usuarios a los que se les haya acreditado esta posibilidad, previo inicio de sesión en el sistema, con sus credenciales personalizadas.

Cuando se realiza la instalación de Joomla 2.5 se creó un único usuario acreditado ante el sistema, el usuario superadministrador, con privilegios absolutos en el sistema, con el nombre de usuario admin y la contraseña aquella que estableciste en su momento cuando se realiza la instalación de Joomla. Con estos dos datos se podrá acceder como administrador absoluto a Joomla de tal manera que se podrá añadir contenido nuevo, editar contenido existente, cambiar el aspecto del sitio, crear menús de navegación, e incluso borrar y eliminar todo el sitio, tal como se muestra en la imagen siguiente:

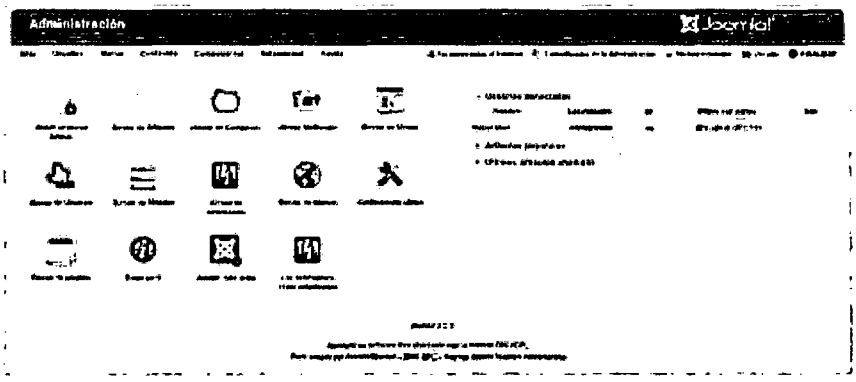


Imagen 22: sistema administrador (panel de control) Joomla 2.5

El Panel de Control de Joomla con un montón de botones y menús que te permitirán acceder a todas las funciones y posibilidades de este CMS.

Nota

Este no es el único acceso a la administración de Joomla. También es posible entrar en el sistema directamente desde el propio Frontend de tu sitio. Vas a comprobarlo. Haz clic en la pestaña del navegador donde se muestra tu sitio, la pestaña Inicio, y localiza en la página el apartado **Formulario de acceso**.

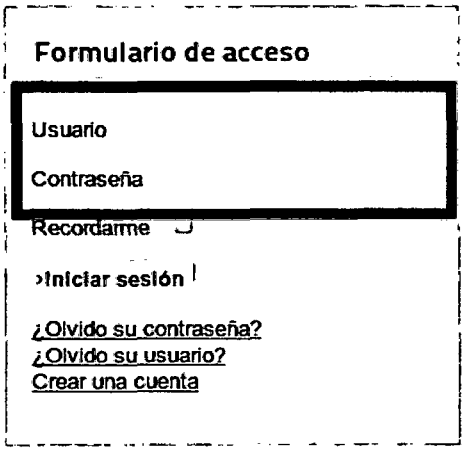


Imagen 23: Formulario de acceso a través de la web

Escribe el mismo nombre de usuario, admin, y la contraseña en los campos correspondientes y pulsa la tecla Intro de tu teclado o el botón Iniciar sesión. ¡Un mensaje te da la bienvenida y te reconoce como superadministrador!

Parece que no puedes hacer nada más, al menos de momento... Haz clic en el botón**Finalizar sesión** para salir del sistema.

En efecto, otra cuestión a tener en cuenta. Cierra siempre correctamente Joomla. Si estás en el Frontend de tu sitio con este botón Finalizar sesión y, si en cambio, estás en el Backend utiliza el enlace **FINALIZAR** que localizarás en la parte superior derecha de la página.

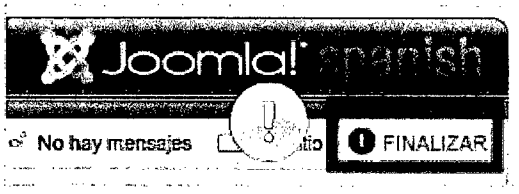


Imagen 24: Finalización del panel de control

3.2.1.5. Vulnerabilidad de Joomla

3.2.1.5.1. Escalamiento de privilegios

Muestra como subir rango de user simple a useradministrator en joomla y defacear[40] eso se llama como escalar privilegios.

Una manera de escalar privilegios es a través de los dorks, lo cual a través de esto resultaría fácil y sencillo y también defacear. A continuación se muestra los Dork de joomla para las versiones 1.5 a 3.0

a. component/users/?view=registration
b. /index.php/component/user/?view=registration (v1.5)
c. /index.php?option=com_users&view=registration (v.2.5)
d. /index.php/component/users/?view=registr-ation (v.2.5,3.0)

Para poder utilizar estos dorks, es necesario saber la versión de Joomla, o en todo caso probar cuál de estos dorks aceptan en el CMS Joomla.

Por ejemplo, a manera de prueba lo realizaremos a la página de una universidad reconocida: UNCP

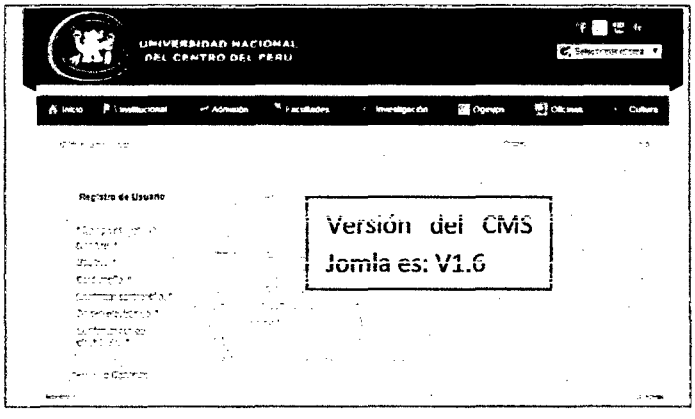


Imagen 25: Acceso de registro de usuario.

3.2.1.5.2. Script: hackeojoomla

Mediante el explorador chrom de google, realizamos inspección de elemento para insertar código html.

El script se insertará tal como se muestra en la siguiente imagen.

```
<input value="7" name="jform[groups][]" />
```

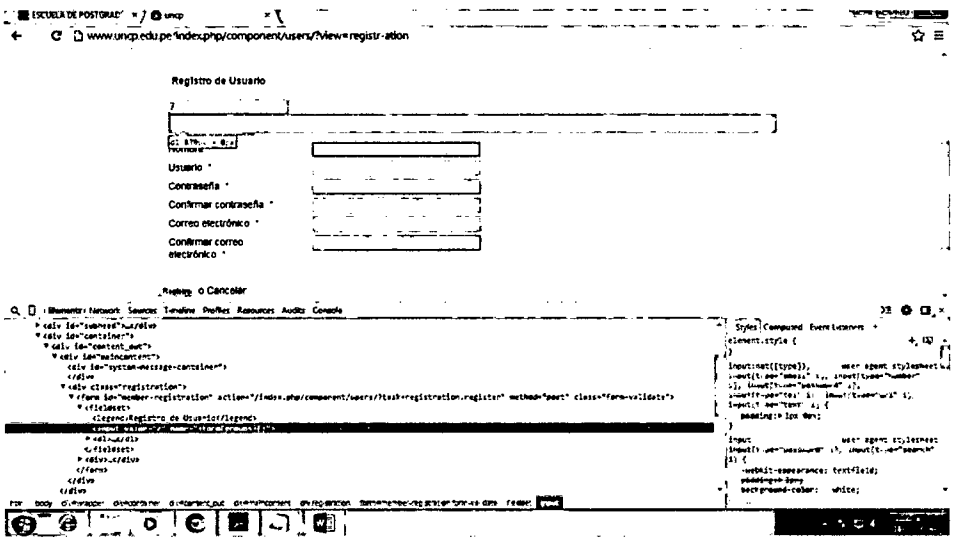


Imagen 26: Realizando la inyección a página de UNCP.

Una vez puesto el script y quedando tal como se muestra en la imagen, es necesario introducir los datos correctos a excepto en contraseña, que se tiene que introducir lo correcto y lo erróneo, pues en la confirmación se tiene que introducir los correcto, luego saldrá un mensaje de usuario registrado. Para activar la cuenta para joomla se tiene que confirmar en el correo una vez hecho esto ya podemos entrar a través de administrator. Ver imagen siguiente.

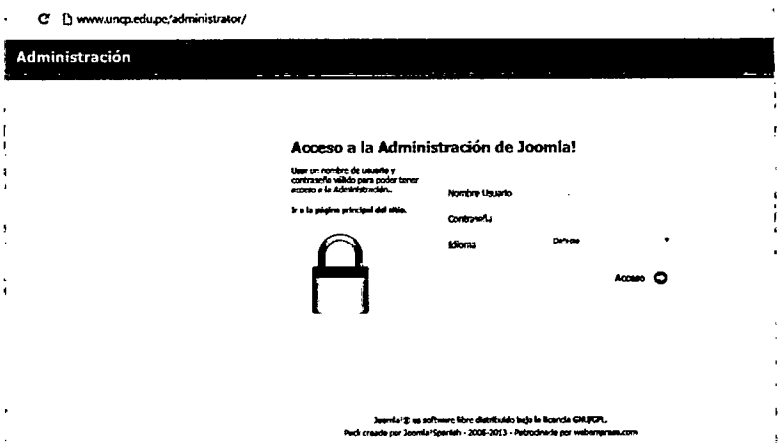


Imagen 27: Sistema Logeo UNCP.

Introducimos el usuario y la contraseña y quedará como se ve en la imagen siguiente.

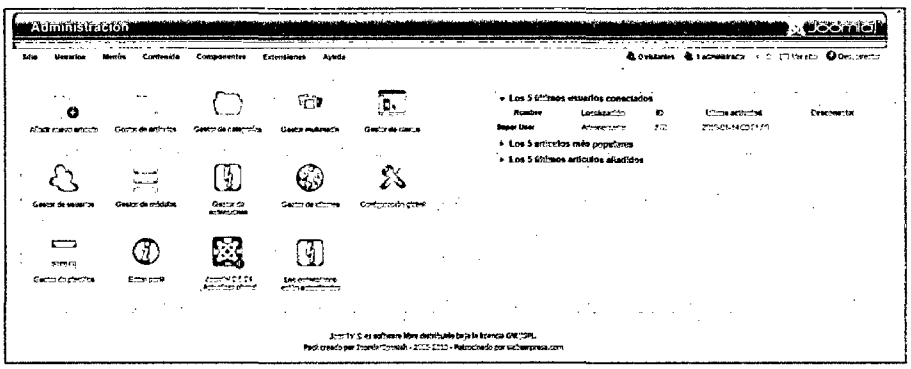


Imagen 28: Ingreso al panel de control de UNCP.

La página está vulnerable para realizar cualquier modificación o eliminación inclusive de la base de datos, teniendo el control total de página de joomla.

CAPITULO IV

RESULTADOS

Para obtener el resultado se realizó el desarrollo del algoritmo y aplicación final.

4.1. DESARROLLO

En la etapa de desarrollo se aplicó el modelo de prototipos, debido a que se implementará una aplicación con el algoritmo que mejor prestación presente, para lo cual en primer lugar se comprendió el modelo de administración de Joomla por defecto, para posterior presentar el modelo que será "Modelo final", aplicación final del algoritmo.

4.1.1.1. Modelo del Sistema administración de Joomla

La funcionalidad del sistema de Administración de Joomla, está determinado tal como se muestra en la imagen siguiente.

Sistema de administración de Joomla

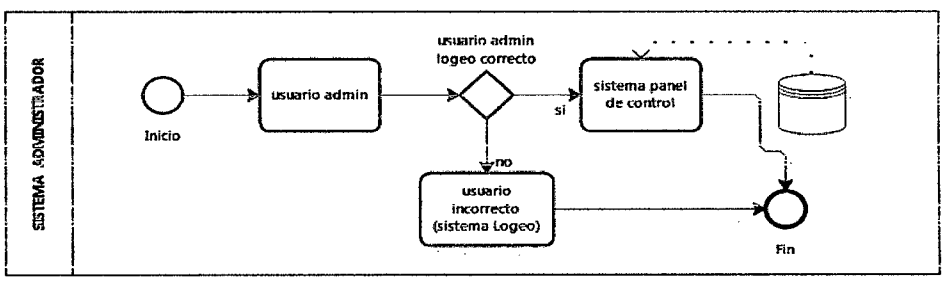


Figura 01: Sistema de administración de Joomla

Fuente elaboración propia.

En el figura 01, muestra que el usuario ingresa al sistema de administración de Joomla, luego se logea, y si todo resulta correcto ingresa al sistema de panel de control.

A continuación se presenta el prototipo de la administración de Joomla, figura 02(sistema de Logeo) y figura 03(Sistema de panel de control).

Funcionalidad de Joomla gráficamente

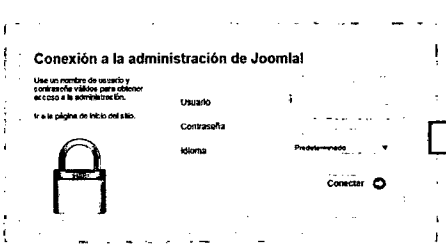


Figura 02: sistema Login de Joomla

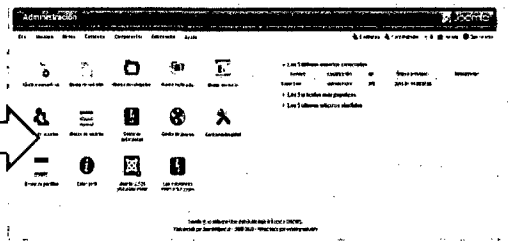


Figura 03: sistema de panel de control de Joomla

4.1.1.2. Sistema propuesto

En el sistema que se propuso, se realizó la doble seguridad para el sistema de administración, de esta manera no puede fácilmente facear la página de joomla.

Modelo del sistema propuesto

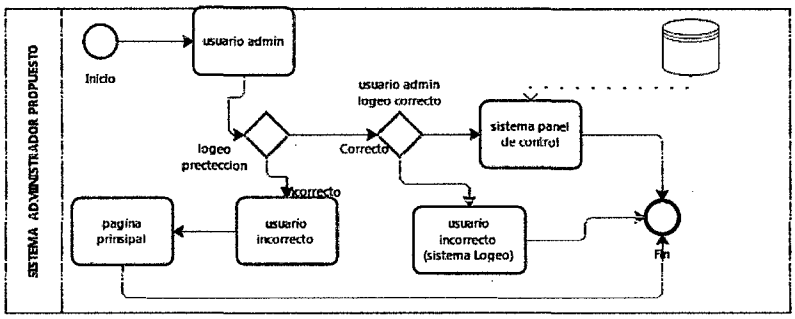


Figura 04: Modelo del sistema propuesto
Fuente elaboración propia

En el modelo propuesto, se puede apreciar que el usuario antes de poder ingresar al sistema de administración de Joomla, tiene que registrarse con sus usuario y contraseña correctamente ver figura 04, en parte "logeo protección", una vez legeado correctamente, ingresa al logeo del administrador de Joomla.

A continuación se presenta el prototipo que será como la aplicación final ver figura 05 y también los sistemas que ya existen (figura 06 y 07)

Funcionalidad propuesto gráficamente

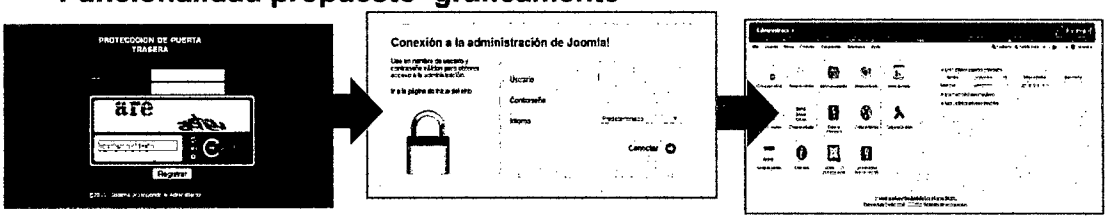


Figura 05: sistema propuesto *Figura 06: sistema login* *Figura 07: Administración*

Figura5: muestra la seguridad del administrador de joomla (*figura 6*).

Para conseguir la aplicación final, se realizó el desarrollo del algoritmo ver figura 08, prosiguiendo la lectura del algoritmo ver página 62, para la comprensión del algoritmo y el pesocódigo del algoritmo figura 09, así de esta manera poder obtener el resultado deseado que es la Figura 05.

A continuación se llevó acabo el desarrollo de la aplicación de protección.

4.1.1.3. Desarrollo del algoritmo para la protección de puerta trasera

Para el desarrollo del algoritmo se tuvo en cuenta:

Situación problemática con el Joomla:

- Usuarios que se realizaban escalamiento de users (hackeo del sistema)
- Teniendo la cuenta de administrador, fácilmente podían ingresar al panel de control y vulnerar la base de datos.

Por otro lado también se consideró aspectos para la seguridad: sólo ingresado por usuarios (mano humana), mas no por robots informáticos.

- A través del **reCAPTCHA**
- Usuario sin encriptaciones
- Contraseña encriptado

En cuanto a las encriptaciones se consideró el sistema MD5 de Mysql.

4.1.1.4. Propuesta del algoritmo

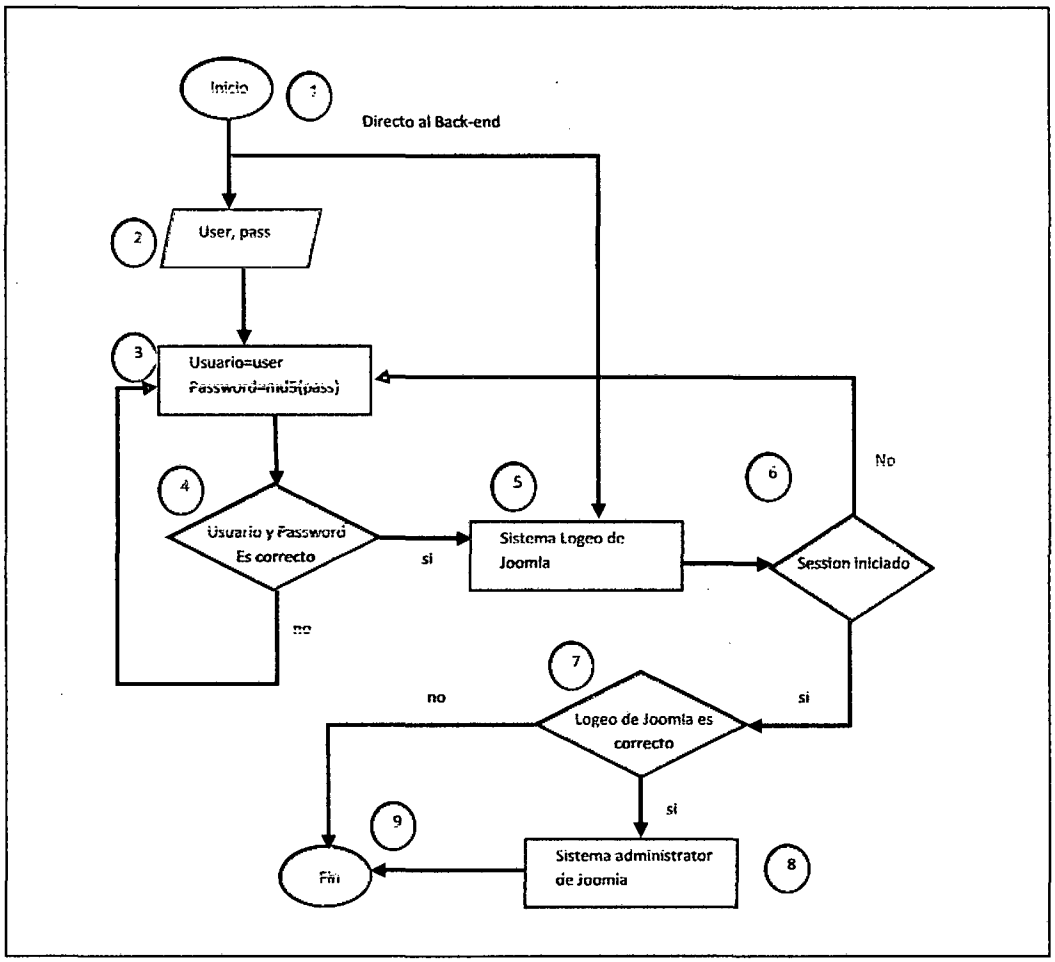


Figura 08. Diagrama de flujo (representación del algoritmo)

La simbología del sistema del diagrama de flujo se encuentra en el anexo N° 02.

Lectura del algoritmo

1. Iniciamos la sesión en el sistema propuesto
2. Definimos las variables para usuario y contraseña

3. Asignamos las variables (usuario y contraseña) bajo la encriptación de MD5 de lenguaje php.
4. Si el usuario y contraseña son válidos, entonces ingresa al administrador de joomla; de lo contrario imprimirá lo incorrecto que fue el logeo (5).
6. Una vez ingresado compará los usuarios y contraseñas son válidas, con respecto al sistema de joomla y verificando la sesión iniciado y correcto.
7. Si todo es correcto ingresará al sistema administrator de Joomla como administrador del sistema.
8. Si la sesión iniciado es incorrecto, entonces volverá al inicio del sistema, para luego proseguir los pasos del 1° al 5°.
9. Finalización de toda la administración del sistema

Pseocódigo del algoritmo

```
1  inicio
2  $usuario,$password;
3
4  $usuario=$_GET["user"];
5  $password==MD5($_GET["pass"]);
6  SI
7    ($usuario==usuario_joomla) and ($password=password_joomla)
8     SI
9      (SESSION!=NULL)
10      entonces:
11      ingresa al sistema administrador de joomla
12        SI
13         ($usuario==usuario_joomla) and ($password=password_joomla)
14          Entonces ingresa al administrador de joomla
15        SINO
16          emprime: usuario incorrecto
17     SINO
18       Vuelve al inicio
19    FIN SI
20  SINO
21   imprime usuario incorrecto
22  FIN NSI
23
24  FIN
```

Figura 09. PSeudocódigo del algoritmo propuesto

Resultados obtenidos:

- Se logró implementar el algoritmo(Figura 08) para la protección de puerta trasera, en lenguaje php y como resultado final se muestra en la siguiente figura.

Prototipo del sistema de protección de puerta Trasera

Figura 10: aplicación sistema de seguridad

Descripción:

Para el sistema se incluyó dos cajas de texto, el uno para poner el usuario y el otro para poner la contraseña y el recuadro de color rojo es el sistema **recaptcha,** que es un sistema para ser ingresado por mano humana las letras que aparecen, en este caso **are arshev,** cuya letra están en formato imagen, por lo cual el software inteligente no podría detectar la imagen de las letras para poder ingresar en caja de texto donde indica "Introduzca el texto". Haciendo que esta aplicación sea capaz de proteger de algún atentado informático.

El **recaptcha,** es una librería que esta codificado en lenguaje php, cuyos creadores son de la empresa Google. Y para incluir en el proyecto solo basta poner en el lugar adecuado como se muestra en la siguiente figura 11, previo inclusión de la librería

recaptchalib.php como se muestra en la línea 2 de la codificación.

Código de funcionamiento del recaptcha

```
1  <?php
2  require_once('recaptchalib.php');
3  // Get a key from https://www.google.com/recaptcha/admin/create
4  $publickey = "6LeGlAATAAAAAGv9z5M2dVMUcNgqkKAMnfW02NQt";
5  $privatekey = "6LeGlAATAAAAAPknSWMFxHqQJbwwwMgMsEZw0NH_";
6  # the response from reCAPTCHA
7  $resp = null;
8  # the error code from reCAPTCHA, if any
9  $error = null;
10 # was there a reCAPTCHA response?
11 if ($_POST["recaptcha_response_field"]) {
12         $resp = recaptcha_check_answer ($privatekey,
13                                         $_SERVER["REMOTE_ADDR"],
14                                         $_POST["recaptcha_challenge_field"],
15                                         $_POST["recaptcha_response_field"]);
16
17         if ($resp->is_valid) {
18                 echo "You got it!";
19         } else {
20                 # set the error code so that we can display it
21                 $error = $resp->error;
22         }
23 }
24
```

Figura 11: codificación de recaptcha

- Se logró proteger ataques de fuerza bruta (probar todas las combinaciones hasta encontrar aquella que permite el acceso) automáticos con el Recaptcha.

Figura 12: pruebas sobre ataques de fuerza bruta.

Descripción:

El Recaptcha permite comparación de texto que se muestra en la imagen con la ingresada en la textbox donde indica "Introdusca el texto", esto permite bloquear ataques de fuerza bruta. El ataque de fuerza bruta normalmente utiliza todas las combinaciones posibles hasta encontrar el usuario y la contraseña correcta, mediante software inteligente bajo plataforma de java y para ello el sistema recaptcha lo detiene, porque tiene ser ingresados por intervención humana las letras que aparacen en formato imagen. Ya que un programa inteligente no podría realizar esa tarea, porque tendría que interpretarlo la que está en la imagen

- Se logró proteger el sistema back-end de joomla filtrando doble seguridad bajo el algoritmo implantado.

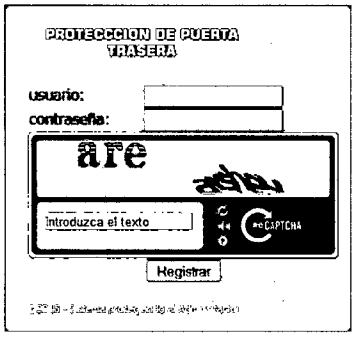


Figura 13: sistema de protección Del login de Joomla

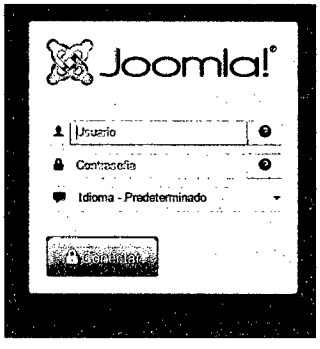


Figura 14: sistema Login de Joomla

Descripción:

Si el sistema propuesto (*figura 13*) es validado correctamente y iniciado la sesión correctamente, entonces ingresa al sistema administrator de joomla (figura 14) y prosiguiendo al panel del administrador ver figura 5. De lo contrario no podría ingresar al

panel de control de joomla tal como lo indica el algoritmo propuesto ver figura 6.

- El sistema está codificado teniendo en cuenta el estándar de codificación de W3C.

Sistema que verificador de codificación estándar

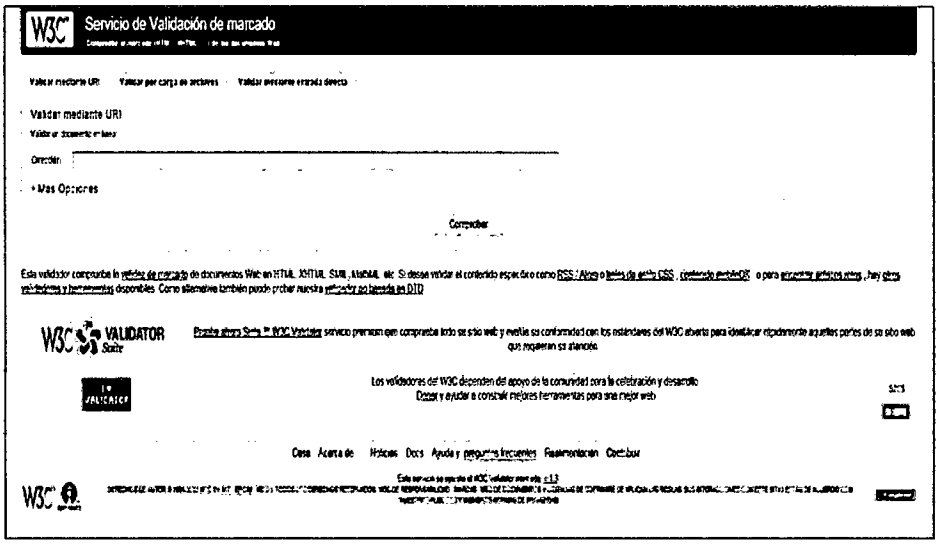


Figura 15: verificar del estándar de codificación

Descripción:

La codificación del sistema está respaldado par el estándar de codificación internacional como la W3C, tal es así que sólo basta poner la dirección web de la aplicación donde indica "Dirección", para la comprobación y luego nos arrojará el resultado línea por línea, si es que tenemos errores para posterior corrección.

CONCLUSIONES

1. Se ha logrado implementar el algoritmo que protege a sistemas web de los ataques a la autentificación en puerta trasera.

2. En la demostración del uso de prototipo se muestra que el acceso de administración de Joomla es redirigida a doble seguridad, dando protección sobre los hackers.

3. La implementación del algoritmo es una de las soluciones posibles contra la autentificación insuficiente.

4. Con el algoritmo se detuvo autentificación automático incluyéndolo el recaptcha

RECOMENDACIONES

1. Explorar los dorks en otros gestores de contenidos y páginas propietarios.

2. En el algoritmo implementado es una de las soluciones de protección, como también puede ser mejorado debido al creciente conocimiento de hackers y vulneración de los sietmas web.

3. Realizar periódicamente copias de seguridad de todo el sistema web, para no sufrir de Hackers.

4. Realizar el monitoreo usuarios privilegiados, que se encuentran en el sistema de Joomla y que cuenten con accesos permitidos.

REFERENCIA BIBLIOGRAFÍCA

1 Jeffrey, V. (2001). *Arte y ciencia del diseño Web*. Madrid: Pearson Alhambra.

2 Pavón,J. (2008). *PHP y MySQL*. Argentina:Alfaomega.

3 Lopez,J.(2008). *Domine PHP 5* .Argentina: Alfaomega.

4 Minera, F. (2008). *PHP Master*. Argentina: USERS.

5 Minera, F. (2010). *PHP 6*. Argentina: USERS.

6 Holzner,S.(2009). *Manual de referencia PHP*. México: Mc Graw Hill

7 Minera, F. (2008). *Curso de Programación PHP*. Argentina: USERS.

8 Niederts,J.(2008). *Creación y Diseño Web profesional*. México:O´Reilly

9 Hernández,R. (1997*). Metodología de la investigación*. Colombia: Panamericana Formas e Impresos S.A. p.97.

10 Arias,G.. (2011). Metodología de la investigación en las ciencias aplicadas al deporte: un enfoque cuantitativo. Noviembre 20,2014, de Universidad Central de Venezuela Sitio web: http://www.efdeportes.com/efd157/investigacion-en-deporte-enfoque-cuantitativo.htm.

10 Budd, A. (2007). *CSS Manual Avanzado*.España: Anaya.

11 Schmitt, C. (2007). Curso de CSS. México: Anaya / O´Reilly

12 HTML La guía completa Musciano y Kennedy, Editorial Mc Graw Hill. 1999.

13 Picouto,F.&Llorente,I. (2007). *HACKING Y SEGURIDAD EN INTERNET*.España: RA-MA EDITORIAL

14 Lynch,p.(2004). *Manual de Estilo Web*.México: Gustavo Gili.

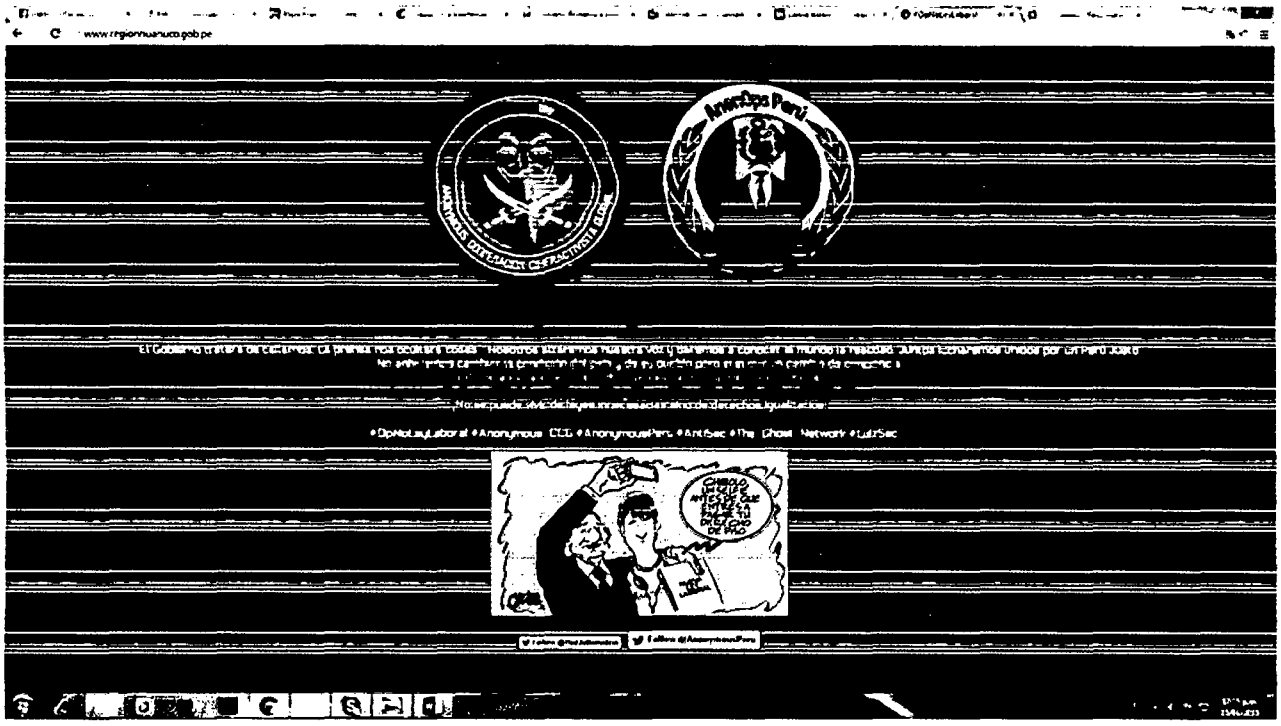
WEBGRAFÍA

- www.desarrolloweb.com
- www.programacion.com
- www.webtaller.com.ar
- www.webexperto.com
- www.phpya.com.ar
- www.htmlya.com.ar
- www.cssya.com.ar
- www.php.net
- http://www.e-securing.com/novedad.aspx?id=58
- http://ataquebd.blogspot.com/
- http://docs.joomla.org/Model-View-Controller
- http://docs.joomla.org/J3.1:Developing_a_MVC_Component/Introduction
- http://joomla.github.io/joomla-platform/?chapters/packages/mvc.md
- https://github.com/joomla/joomla-cms/pull/2063
- https://github.com/joomla/joomla-cms/pull/1989

ANEXOS

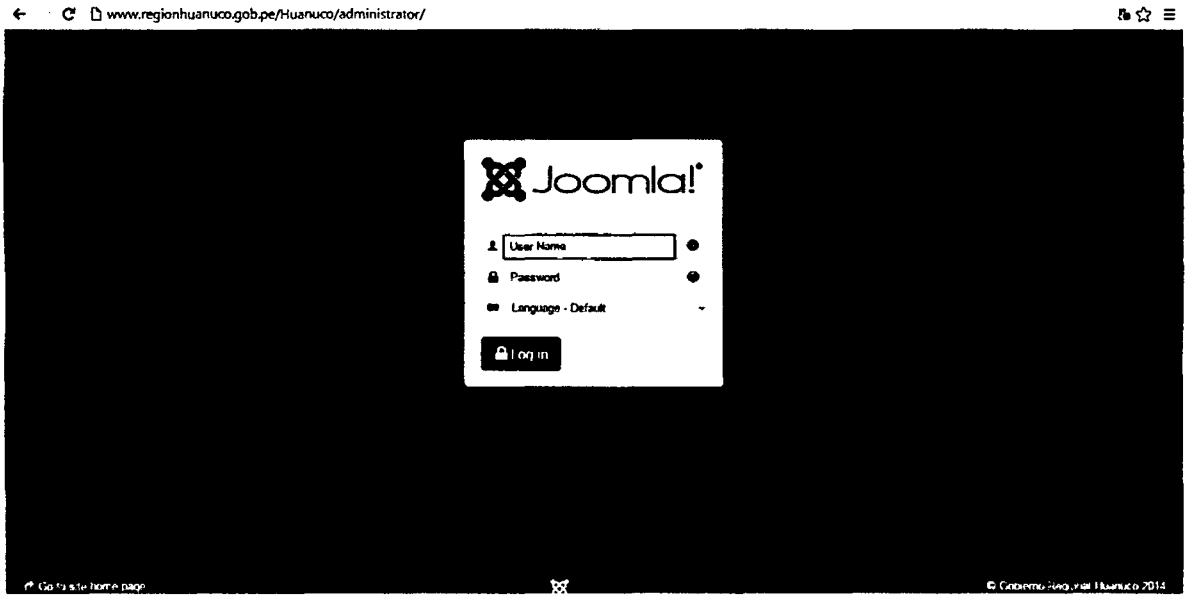
ANEXO N° 01

PAGINA GUBERNAMENTAL HACKEADO: GOREHCO



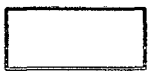
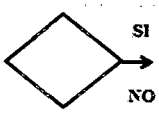
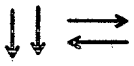
Página web del Gobierno regional Huánuco fue hackeado el 15 de Enero del 2015.

La página utilizaba el CMS Joomla 3.0 tal como se ve en la imagen siguiente



ANEXO N° 02

Elementos del Algoritmo

SIMBOLOGIA	PSEUDOCODIGO	FUNCION
	Inicio o Fin	Se utiliza para empezar y terminar un programa
	Conocer o pedir	Utilizar para pedir datos por teclado estos pueden ser números, textos o datos alfanuméricos
	Evaluar o Asignación	Sirve para evaluar operaciones aritméticas como formular y también para asignar.
	Preguntas de decisión	Sirve para hacer decisiones.
	Imprimir	Imprime el Resultado
	Flechas de Dirección	Sirven para guardar la dirección del flujo del programa

Simbologia del diagrama de flujo

"Año de la Promoción de la Industria Responsable y del Compromiso Climático"

UNIVERSIDAD NACIONAL "HERMILIO VALDIZÁN" HUÁNUCO – PERÚ
FACULTAD DE INGENIERÍA INDUSTRIAL Y DE SISTEMAS

RESOLUCIÓN Nº 0511-2014-UNHEVAL/FIIyS-D.

Huánuco, 16 de octubre de 2014.

CONSIDERANDO:

Que mediante Resolución Nº 054-2013-UNHEVAL-CEU, del 15.MAY.2013, se RECONOCE, la elección de la M.Sc. Guadalupe RAMÍREZ REYES, como Decana Titular de la Facultad de Ingeniería Industrial y de Sistemas a partir del 18.MAY.2013 al 18.MAY.2016;

Que con Resolución Nº 0452-2014-UNHEVAL/FIIS-D, de fecha 01.SET.2014, se nombra los Miembros Revisores adhoc de Plan de Tesis para optar Titulo Profesional de Ingeniero de Sistemas, titulado: ***"ALGORRITMO DE PROTECCIÓN DE PUERTA TRASERA CONTRA ATAQUES DE AUTENTIFICACIÓN EN SISTEMAS DE BASES DE DATOS EXPUESTOS A SERVICIOS WEB"***, presentado por el Bachiller en Ingeniería de Sistemas: **Wilmer OCHOA ALVARADO;**

Que mediante Informes de los señores docentes: Dr. Milton Pérez Solís e Ing. Elmer Cghuquiyauri Saldivar, en calidad de Miembros Revisores Ad-hoc del Proyecto de Tesis, hacen llegar la CONFORMIDAD y solicitan se realicen los trámites para la continuación de la Tesis;

Que revisado el expediente y en mérito al Capítulo IV de la Modalidad de Tesis Art. 15° y 16º, se resuelve emitir una Resolución **APROBANDO** el Proyecto de Tesis, titulado: ***"ALGORRITMO DE PROTECCIÓN DE PUERTA TRASERA CONTRA ATAQUES DE AUTENTIFICACIÓN EN SISTEMAS DE BASES DE DATOS EXPUESTOS A SERVICIOS WEB"*** presentado por el Bachiller en Ingeniería de Sistemas: **Wilmer OCHOA ALVARADO,** debiendo el recurrente proceder a desarrollar su Proyecto de Tesis en un tiempo mínimo de sesenta (60) días hábiles. Si no lo desarrollara en un plazo de un año, debe presentar un nuevo Proyecto de Tesis;

Estando a las atribuciones conferidas al Decano de la Facultad por la Ley Universitaria, por el Estatuto de la UNHEVAL y por la Resolución Nº 054-2013-UNHEVAL-CEU;

SE RESUELVE:

1° APROBAR el Proyecto de Tesis Titulado: ***"ALGORRITMO DE PROTECCIÓN DE PUERTA TRASERA CONTRA ATAQUES DE AUTENTIFICACIÓN EN SISTEMAS DE BASES DE DATOS EXPUESTOS A SERVICIOS WEB"*** presentado por el Bachiller en Ingeniería de Sistemas: **Wilmer OCHOA ALVARADO.**

2° RATIFICAR la Resolución N° 0452-2014-UNHEVAL/FIIyS-D, de fecha 01.SET.2014, con el cual se nombra Asesor de Tesis al Ing. Aicides Bernardo Tello.

3° ESTABLECER que el recurrente deberá proceder al desarrollo de su Proyecto de Tesis en un tiempo mínimo de sesenta (60) días hábiles.

4° DAR A CONOCER a los órganos internos y al interesado.

Regístrese, Comuníquese y Archívese.

DECANA
M.Sc. Guadalupe Ramírez Reyes
DECANA FIIS

Distribución:
CGyT/FIIS
Interesado
Asesor
Archivo

"Año de la Diversificación Productiva y del Fortalecimiento de la Educación"

UNIVERSIDAD NACIONAL "HERMILIO VALDIZÁN" HUÁNUCO – PERÚ

FACULTAD DE INGENIERÍA INDUSTRIAL Y DE SISTEMAS

RESOLUCIÓN Nº 0125-2015-UNHEVAL/FIIS-D.

Huánuco, 09 de abril de 2015.

CONSIDERANDO:

Que mediante Resolución Nº 054-2013-UNHEVAL-CEU, del 15.MAY.2013, se RECONOCE, la elección de la M.Sc. Guadalupe RAMÍREZ REYES, como Decana Titular de la Facultad de Ingeniería Industrial y de Sistemas a partir del 18.MAY.2013 al 18.MAY.2016;

Que con Resolución Nº 0511-2014-UNHEVAL/FIIS-D, del 16.OCT.2014, se aprueba el Proyecto de Tesis Titulado: ***"ALGORRITMO DE PROTECCIÓN DE PUERTA TRASERA CONTRA ATAQUES DE AUTENTIFICACIÓN EN SISTEMAS DE BASES DE DATOS EXPUESTOS A SERVICIOS WEB"***, presentado por el Bachiller en Ingeniería de sistemas **Wilmer OCHOA ALVARADO;**

Que con solicitud de fecha 31.MAR.2015, el Bachiller en Ingeniería de Sistemas: Wilmer OCHOA ALVARADO, se dirige a la Decana de la FIIS y solicita el nombramiento de jurados para la sustentación de tesis titulado: ***"ALGORRITMO DE PROTECCIÓN DE PUERTA TRASERA CONTRA ATAQUES DE AUTENTIFICACIÓN EN SISTEMAS DE BASES DE DATOS EXPUESTOS A SERVICIOS WEB"***;

Que en cumplimiento al Reglamento de Grados y Títulos de la UNHEVAL, Art. 17º al 19º, mi Despacho considera procedente nombrar los jurados de Tesis para el otorgamiento de Titulo Profesional de Ingeniero Industrial, integrado por cuatro (04) profesores ordinarios: tres (03) serán titulares y un (01) accesitario. El Jurado se compone de: Presidente, Secretario y Vocal. De los cuales dos son especialistas del tema de la Tesis y un especialista en metodología de la investigación. El de mayor categoría y precedencia preside dicho jurado. El Jurado tendrá la responsabilidad de ***dictaminar en un plazo que no exceda los quince (15) días hábiles, acerca de la suficiencia del trabajo.*** Si el trabajo fuera declarado insuficiente, lo devolverá para que el tesista lo corrija en un plazo que no exceda los treinta (30) días hábiles;

Estando a las atribuciones conferidas al Decano de la Facultad por la Ley Universitaria, por el Estatuto de la UNHEVAL y por la Resolución Nº 054-2013-UNHEVAL-CEU;

SE RESUELVE:

1º NOMBRAR los Jurados de Tesis, titulado: : ***"ALGORRITMO DE PROTECCIÓN DE PUERTA TRASERA CONTRA ATAQUES DE AUTENTIFICACIÓN EN SISTEMAS DE BASES DE DATOS EXPUESTOS A SERVICIOS WEB"***, presentado por el Bachiller en Ingeniería de Sistemas: **Wilmer OCHOA ALVARADO,** integrado por los siguientes docentes:

Dr. Milton Pérez Solis	-	PRESIDENTE
Ph.D. Marcelino Reynaga Martinez	-	SECRETARIO
Ing. Luis Meza Ordoñez	-	VOCAL
M.Sc. *Guadalupe Ramírez Reyes*	-	ACCESITARIO

2º ESTABLECER que el Jurado tendrá la responsabilidad de dictaminar en un plazo que no exceda los quince (15) días hábiles, acerca de la suficiencia del trabajo.

3º DAR A CONOCER a los órganos internos y a los Interesados.

Regístrese, comuníquese y archívese.

M.Sc. Guadalupe Ramírez Reyes
DECANA FIIS

Distribución:
CGyT.FIIS/Jurados/Interesado/Archivo