JITEL 2017
XIII Jornadas de Ingeniería Telemática
27-29 Septiembre 2017, Valencia (España)

Actas de las XIII Jornadas de Ingeniería Telemática (JITEL 2017), Valencia (España), 27-29 de Septiembre de 2017

ISBN: 978-84-9048-595-8
DOI: http://dx.doi.org/10.4995/JITEL2017.2017.6558

Sistemas de gestión de contenido web: Uso y estudio comparativo inicial de su seguridad

Antonio-José Aledo-Hernández[1], Antonio Guillen-Pérez[1], Jose-Manuel Martinez-Caro[1], Ramón Sánchez-Iborra[2], María-Dolores Cano[1]

[1]Dpto. Tecnologías de la Información y las Comunicaciones

[1]Universidad Politécnica de Cartagena

[2]Dpto. Ingeniería de la *Información* y las *Comunicaciones*

[2]Universidad de Murcia

antonio.aledo@alu.upct.es, agp4@alu.upct.es, jmmc0@alu.upct.es, ramonsanchez@um.es, mdolores.cano@upct.es

***Resumen*- Los Sistemas de Gestión de Contenido Web (*Web Content Management Systems,* WCMS) han ganado mucha popularidad debido a la facilidad que aportan a la hora de crear páginas o portales web, *sites* de comercio electrónico, etc. En este trabajo se explica de forma resumida cómo es el manejo los WCMS y qué se puede lograr con su uso. Para ello, trabajaremos con tres de los más populares WCMS de tipo *open-source* empleados hoy en día, Joomla, Wordpress y Drupal, y veremos las ventajas e inconvenientes de trabajar con cada uno de ellos. Con este fin, crearemos tres web iguales en requisitos y funcionalidades, una con cada WCMS, y se analizará cualitativamente la complejidad de cada uno de ellos. Finalmente, realizaremos un análisis básico de seguridad de las webs creadas, informando de sus posibles vulnerabilidades, explicando cómo mejorar su seguridad, qué fallos no debemos cometer y qué WCMS es inicialmente más seguro/vulnerable.**

***Palabras Clave*- sistemas de gestión de contenidos web, seguridad, Joomla, Wordpress, Drupal**

I. Introducción

Los Sistemas de Gestión de Contenido Web (*Web Content Management Systems*, WCMS) se utilizan generalmente en entornos donde es necesario crear un portal online de contenidos web sin necesidad de disponer de cocimientos sobre programación web y donde además se permite la creación de gran variedad de roles de usuario [1-3]. Un ejemplo de lo citado sería la redacción de un periódico, donde están interesados en lanzar su edición electrónica online. En este caso los editores sólo tienen conocimientos informáticos de ofimática, no de programación web para crear su portal, y es en este punto donde harían uso de los gestores de contenido web para poder llevar a cabo tal fin.

Los Open Source WCMS, también denominados WCMS de segunda generación, son plataformas de código abierto (*open-source*) desarrolladas a menudo en PHP y alimentadas con sus propias comunidades de usuarios, que aportan tanto soluciones como funcionalidades a estos [4]. La estructura básica de un WCMS la componen las siguientes partes: i) los archivos propios del gestor de contenidos, ii) un proveedor de hospedaje para almacenar en él los archivos propios del WCMS, y iii) una base de datos (p.e.: MySQL) para vincularla y almacenar en ella la información de nuestro futuro *site.* Los WCMS ofrecen una zona de administración o desarrollo, donde podremos añadir artículos, funcionalidades o dotar de un aspecto concreto conocida como *back-end.* Por otro lado la parte visible para el visitante del *site* es conocida como *front-end.*

En este artículo, ampliación del trabajo realizado en [5], se explica cómo es el manejo los WCMS y qué se puede lograr con su uso. Para ello, trabajaremos con tres WCMS populares y con buenas prestaciones [6]: Joomla [7], Wordpress [8] y Drupal [9]. Aunque Joomla y Drupal han ido perdiendo protagonismo en los últimos años frente a otras propuestas su todavía amplio uso nos ha hecho optar por incluirlos en esta comparativa [10]. Veremos también las ventajas e

inconvenientes de trabajar con cada uno de ellos en la actualidad. Con este fin, crearemos tres web iguales, una con cada WCMS, y se analizará la complejidad de cada uno de ellos. Finalmente, realizaremos un análisis preliminar de seguridad de los *sites* creados, informando de sus posibles vulnerabilidades, explicando cómo dotarlos de seguridad, qué fallos no debemos cometer y qué WCMS es más seguro/vulnerable.

El resto del artículo se distribuye de la siguiente forma. La sección II describe brevemente los trabajos relacionados en la temática. En la sección II se detallan los WCMS con los que trabajaremos. La sección IV proporciona una guía y una comparación de cómo crear un *site* utilizando los WCMS bajo estudio. En la sección V se presenta un análisis de seguridad básico. El documento finaliza con las conclusiones más relevantes de este trabajo.

II. Trabajos Relacionados

En la literatura especializada podemos encontrar algunos trabajos que también abarcan este tema. En [11] los autores presentan un estudio comparativo de siete WCMS, incluyendo los tres empleados en este trabajo. No obstante, la comparativa se basó únicamente en la implementación de un *site* con cada WCMS y en determinar si alguna de las funcionalidades establecidas como requisitos previos estaban disponibles o no. Por otro lado, [12-13] son trabajos muy interesantes que introducen y analizan la seguridad en en los WCMS. En [12], se identifican las vulnerabilidades y ataques a los que están expuestos los WCMS, así como posibles medidas de respuesta. Como caso práctico, experimentaron con Joomla y Drupal con versiones previas a las empleadas en este trabajo y realizaron algunos test de penetración sencillos usando las herramientas WebScarab [14] y TamperData [15]. Como principal resultado los autores indican que a pesar de disponer de mecanismos de seguridad, ambos WCMS pueden ser víctimas fáciles de ataques. En [13], los autores presentan un estudio en profundidad de los sistemas de gestión de contenidos, aunque desde una perspectiva puramente cualitativa, incluyendo otros aspectos como el efecto de los servidores web (i.e, Apache, Nginx, etc.) en base a estadísticas disponibles. Como veremos en las siguientes secciones, este trabajo actualiza el llevado a cabo en [12] y contribuye de forma práctica a los resultados obtenidos en [13].

III. Sistemas de Gestión de Contenido Web bajo estudio

Joomla! [7] es uno de los WCMS más populares para crear portales web dinámicos [10]. En la fecha de preparación de este trabajo, tenía su compatibilidad limitada únicamente a bases de datos de tipo MySQL. Su característica fundamental es que ofrece la gama más alta de funcionalidades, como pueden ser galerías de imágenes, foros, chats, blogs, deslizadores de imágenes, noticias y un largo etc. Por su parte, Drupal [9] también está destinado a la creación de portales dinámicos. Nos permite una compatibilidad con gran diversidad de tipos de bases de datos. Su característica principal es su seguridad, rapidez de carga y la diversidad en los roles de usuario, ya que nos permite por ejemplo limitar el acceso de cierto usuario hasta el punto de sólo poder éste modificar las propiedades de una determinada funcionalidad e incluso sólo de ciertos parámetros de esta funcionalidad. Wordpress [8] es un WCMS destinado sobre todo a la creación de *blogs*, y ha evolucionado hasta proporcionar soluciones de aplicaciones web y comercio electrónico. Tal y como ocurre con Joomla! su compatibilidad está limitada (en la fecha de realización de este trabajo) a la base de datos MySQL. Su característica principal es su gran posicionamiento SEO (*Search Engine Optimization*), ya que una web Wordpress dispone de numerosos *plugins* para facilitar una rápida aparición en los motores de búsqueda en comparación con otros WCMS [16]. Además nos permite crear un *blog* sencillo de forma gratuita sin necesidad de tener contratado un hospedaje, bajo la extensión de un subdominio propio (*.wordpress.com*). De forma detallada las características fundamentales de los tres WCMS a estudio para creación de portales web se muestra en la Tabla I obtenida a partir de [4-11, 16].

Cabe destacar algunos aspectos de la Tabla I como pueden ser que Wordpress ofrece muchas extensiones para realizar una funcionalidad concreta pero no tiene un amplio abanico de funcionalidades como puede ser el caso de Joomla!. También decir que Wordpress ofrece la posibilidad de añadir extensiones integradas en su entorno de administración, y que el más complejo de los tres WCMS para llevar a cabo su extensibilidad es Drupal, ya que una funcionalidad concreta puede tener dependencias de librerías u otras funcionalidades relacionadas, lo que hace que este proceso de extensibilidad pueda ser en ocasiones largo y tedioso.

Tabla I
Algunas características fundamentales de Joomla!, Wordpress y Drupal

Característica	Joomla!	Drupal	Wordpress
Tipo principal de contenido	Web *sites*, online apps	Blog	Blog, e-commerce, online apps
Disponibilidad de extensiones	Alta	Media	Alta
Variedad de funcionalidades	Alta	Media	Alta
Repositorio de extensiones	Distribuido	Centralizado	Distribuido
Documentación	Muy buena	Buena	Muy buena
Comunidad de usuarios	Muy activa	Limitada	Muy activa
Facilidad de uso	Sencilla	Compleja	Sencilla
Personalización de roles de usuario	Media	Muy alta	Media
Posicionamiento SEO manual	Sí	Sí	Sí
Posicionamiento SEO automático	Extensiones	Modules	Plugins and tools

IV. Creación de un portal web con Joomla!, Drupal y Wordpress

A la hora de comparar los tres WCMS, se va a crear un portal web con cada uno de ellos. Este portal deberá tener las siguientes funcionalidades (algunas de ellas se muestran en la Fig. 1) y además se desea que las funcionalidades estén dispuestas siguiendo la distribución de la Fig. 2.:

- Deslizador de imágenes: *Slider* o *banner* de imágenes en movimiento para la página principal del *site*, basado en Javascript
- Módulo de acceso o login: Permitirá crear zonas privadas en el *site* y el registro de usuarios al mismo
- Integración redes sociales: Twitter y Facebook
- Módulo multilenguaje: Traducción basada en el traductor de Google para ofrecer la posibilidad de traducir los contenidos
- Módulo buscar: Para búsqueda de contenido indexado por parte del visitante del *site*
- Formulario de contacto
- Vídeos
- Mapa
- Descargas: Descargas multiusuario personalizadas
- Boletín de noticias y eventos: Dotar de la posibilidad al usuario de estar informado de novedades acerca del *site* por e-email y por otro lado situar noticias importantes en la página principal de nuestra página

Fig. 1. Algunos elementos que deseamos tener en nuestras 3 versiones

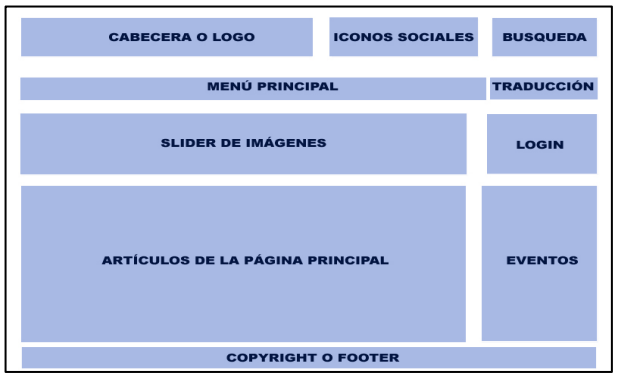

Fig. 2. Disposición de las funcionalidades

Para comenzar, veremos el proceso de instalación de un WCMS. Independientemente de con cuál estemos trabajando, se suelen seguir siempre las siguientes pautas:

1. Contratar un proveedor de hospedaje (por ejemplo 1and1.es) que incluya un sistema de base de datos (por ejemplo MySQL)
2. Crear una base de datos para el CMS
3. Descargar el paquete de instalación del CMS desde la web oficial y descomprimir en el directorio virtual proporcionado por el proveedor de hospedaje
4. Instalar el CMS usando el asistente que proporciona y que enlaza con la base de datos

Por otra parte, el proceso de configuración y personalización de un WCMS suele seguir el siguiente patrón:

1. Descargar (o diseñar con el software creador de plantillas conocido como Artisteer [17]) la plantilla base del sitio
2. Asignar la plantilla al sitio web a partir del gestor de plantillas
3. Buscar las extensiones (funcionalidades) necesarias
4. Activar extensiones dentro del sitio web y situarlas en las posiciones deseadas (ver bloques o posiciones disponibles de plantilla previamente en la configuración de esta)
5. Generar contenidos por parte de los editores, para ello los WCMS incorporan su propio editor para incluir imágenes, videos, personalizar textos, etc.

De forma resumida, la Tabla II incluye las diferencias y similitudes detectadas a la hora de implementar el *site* con Joomla, Drupal y Wordpress. Obsérvese que las plantillas seleccionadas disponen de un *framework* propio que les permite cambiar la disposición de los bloques y su tamaño, el ancho de la hoja, las fuentes, colores, etc. Algo que nos permite dar el aspecto que deseemos al *site* además de asemejarse en las tres versiones a realizar. Referente a Artisteer [17] decir que es un software muy intuitivo en su utilización que nos permite plasmar él aspecto, disposición y estilos que deseemos para la plantilla de un *site* y exportarlo en un archivo .zip listo para ser instalado en nuestro gestor de contenido web. En cuanto a la cabecera y pie de página, el módulo Artical para Joomla! convierte un artículo creado con el editor en un módulo para situarlo en la posición que deseemos del *site*. Respecto al *slider* de Drupal, comentar que éste necesita una librería adiccional llamada *jquery.cycle.all.js* para funcionar correctamente y que en el caso de Wordpress para mostrar el *slider* tenemos que incluir dentro de una página o entrada el código que nos devuelve este en su configuración. En la configuración de sendos *sliders* cabe destacar la elección de las dimensiones para estos y la elección de

Tabla II
SIMILITUDES Y DIFERENCIAS DETECTADAS A LA HORA DE IMPLEMENTAR LAS FUNCIONALIDADES DEL *SITE*

Funcionalidad	Joomla!	Drupal	Wordpress
Diseñador de plantillas propio	Artisteer		
Plantilla usada	Yoo Downtown	AT commerce	Yoo Downtown
Cabecera y pie de página	Módulo Artical	Incluido en plantilla	Incluido en plantilla
Slider de imágenes	Nivo	Incluido en plantilla	
Redes sociales	ITPsocialbuttons	Web de Linksalpha nos proporciona el código social	
Traducción	GTranslate		
Eventos	JNews	Bloque de contenido reciente	Widget enlaces permanentes
Gestor de descargas	Jdownloads	CMS Mollify	
Boletín de noticias	Web de Mailchimp		
Formulario de contacto	CKForms	Ya incluido en núcleo	CformsII
Creador de módulos	Jumi	-	-
Editor de artículos	JCE	Propio	Propio
Youtube, Twitter y Mapa	Mediante inserción del código que nos devuelve la webs oficiales de cada una en páginas nuevas o artículo		
Busqueda y login	En el núcleo		

la ruta donde se encuentran las imágenes que queremos que estos contengan. En cuanto a los botones sociales, hay páginas web que nos devuelven el código necesario para insertarlo en un nuevo bloque en nuestra web. Cuando un visitante del *site* haga clic sobre uno de estos botones enlazará con su propia cuenta de la red social asociada al botón para que este visitante pueda publicar como recomendado el enlace de nuestro *site*. Respecto a la disponibilidad de un visitante de cambiar el idioma de todo el contenido del *site* se puede decir que Google se encarga de ello disponiendo de un módulo donde prácticamente sólo tenemos que indicar en su configuración qué idiomas queremos ofrecer (con banderas de elección para el visitante) y el idioma por defecto del sitio.

Para llevar a cabo la gestión de eventos en la versión de Drupal se ha hecho uso del bloque de contenido reciente para indicar qué páginas creadas situaremos como eventos. Para la versión Joomla!, se ha configurado el módulo usado para este fin indicándole que identificadores de artículos queremos resaltar como los más importantes, y por último en la de Wordpress se ha empleado el *widget* de enlaces permanentes para pegar en él los enlaces de las páginas que nos interesen para eventos. Importante reseñar que cuando creamos una página Wordpress ésta nos devuelve un enlace permanente.

La funcionalidad de gestión de descargas multiusuario no estaba disponible en Drupal y Wordpress (en la fecha de desarrollo de este trabajo), por lo que se hizo uso del gestor de archivos Mollify. Para el boletín de noticias se utilizó la web oficial de Mailchimp [18] en las tres versiones. Esta web devuelve un código para insertar en nuestra web para que los usuarios puedan subscribirse y estar por tanto informados vía email de las novedades que deseamos ofrecer a nuestros visitantes.

Una de las funcionalidades que más configuración y personalización requiere es la del formulario de contacto ya que se deben crear las etiquetas, cajas de texto simples, áreas de texto, botones de envío y reseteo que deseemos que éste contenga y además indicar en su configuración parámetros como pueden ser el email destino de las consultas. Se cambiará el editor por defecto de Joomla! por otro más completo que sí permite inserción de fotos, videos, etc., y por último indicar que la página principal de cada una de las versiones son entradas o artículos seleccionados para formar la presentación de la página principal del *blog*.

V. ANÁLISIS PRELIMINAR DE SEGURIDAD

Se ha llevado a cabo un estudio básico de las vulnerabilidades de las implementacions hechas con Joomla! y Drupal, ya que están destinadas a la creación del mismo tipo de portales web. Por este motivo hemos desestimado realizar el de la versión de Wordpress (destinada principalmente para realización de *sites* de tipo *blog*). Antes de comenzar, citamos algunas recomendaciones de seguridad cuando trabajamos con WCMS:

- Hacer copias de seguridad periódicas tanto de los archivos de nuestro WCMS como de la base de datos (exportarla)
- Contratar proveedores de hospedaje profesionales, así además estaremos más seguros contra ataques del tipo SQL (como explicaremos más adelante)
- Usar la versión más reciente del WCMS y de los *plugins* instalados en él
- Usar *plugins* específicos para seguridad como puede ser JHackGuard para Joomla!, para dotar de seguridad extra
- Restringir el acceso a archivos y carpetas de administración
- Eliminar el *script* de instalación (por ejemplo *install.php* en Drupal o la carpeta llamada *installation* en el caso de Joomla!)
- Modificar las contraseñas por defecto y definir roles de usuario seguros
- Habilitar CAPTCHA para usuarios no registrados, de este modo evitaremos el spam
- Activar URLs amigables
- Cambiar la configuración por defecto en los parámetros globales del *site*
- Cambiar el prefijo por defecto para las tablas de la base de datos durante la instalación si el WCMS en concreto nos lo permite
- Evitar mostrar por error información sensible sobre el WCMS en la parte visible al usuario (*front-end*)

Los ataques más comunes sufridos por los WCMS son del tipo SQL *injection*, que consiste básicamente en que un usuario accede al *site* pudiendo alterar la base de datos, y de tipo *Cross-site scripting* (conocido como XSS), que se produce cuando un usuario mal intencionado encuentra la forma de insertar un fragmento de código malicioso en nuestra web [12]. Más específicamente, *SQL injection* es una técnica en la que el atacante inserta caracteres o palabras clave (*keywords*) en una sentencia SQL mediante parámetros de entrada de usuario sin restricciones para cambiar la lógica de la consulta deseada [19]. Por su parte, un ataque XSS consiste en inyectar una secuencia de comandos maliciosa en un sitio web de confianza que se ejecuta en el navegador de un visitante sin el conocimiento del visitante y, por lo tanto, permite al atacante acceder a datos de usuario sensibles, como *tokens* de sesión y *cookies* almacenados en el navegador [20]. Mientras que la inyección de SQL se dirige a la función de consulta que interactúa con la base de datos, XSS explota la función de salida HTML que envía datos al navegador.

Para llevar a cabo el análisis de vulnerabilidades de nuestras dos versiones Joomla! y Drupal vamos a hacer uso de una herramienta diseñada para tal fin denominada Acunetix [21]. Tras realizar las pruebas de *SQL injection* y XSS con Acunetix sobre nuestras dos versiones del *site* obtenemos los resultados que se muestran en la Tabla III. Lo primero que debemos decir acerca de los resultados es que respecto a inyección SQL y XSS ambos son seguros, en contraposición a lo ocurrido en [12]. Todas las alertas son de nivel de riesgo bajo y de índole similar en todos los casos. No hay que alarmarse tras ver las alertas que aparecen ya que la gran mayoría de éstas son referentes a contenido o extensiones que han sido borradas o desinstaladas de nuestros WCMS durante el desarrollo del proyecto, y por este motivo aparece contenido sin indexar, para el cual nos recomienda Acunetix borrar manualmente.

Por otro lado nos muestra una alerta relacionada con el módulo de *login* en las cuatro pruebas, ya que tenemos activada la opción de autocompletar, y nos recomienda desactivarla. Por último en este apartado citar que en el caso de las pruebas sobre Joomla! nos indica que a través de la extensión *Jdownloads* se permite la subida de archivos por parte de usuarios, algo peligroso aunque en este caso somos conscientes de ello pues se trata de un requisito de diseño.

Tabla III
ALERTAS TRAS MEDICIONES CON ACUNETIX

	Joomla!	Drupal
SQL injewction	Total de alertas 133 Riesgo de nivel 1 o bajo	Total de alertas 78 Riesgo de nivel 1 o bajo
XSS	Total de alertas 122 Riesgo de nivel 1 o bajo	Total de alertas 9 Riesgo de nivel 0

Tanto Joomla! como Drupal tienen una comunidad extensa de usuarios y desarrolladores muy activa y siempre pendiente de incluir mejoras de las versiones disponibles en concepto de seguridad y por supuesto de reportar los fallos y las posibles soluciones a estos. Por supuesto las versiones posteriores corrigen los fallos o vulnerabilidades que presentaban las anteriores. Ambos deberían de tener más recomendaciones durante la instalación con respecto a carpetas y archivos importantes que hay que proteger a posteriori, y sobre todo deberían indicarnos que no es seguro usar el nombre de usuario de administración por defecto y los prefijos por defecto para las bases de datos. Por el contrario, sólo nos avisan de que borremos los archivos de instalación tras realizar esta. En ambos WCMS tenemos módulos adicionales desarrollados por terceros para dotar de seguridad extra al *site* como puede ser *Taxonomy Access Control* que se basa en el uso de roles de usuario para dotar de seguridad extra a un *site* Drupal o el módulo Marco's SQL *Injection* que permite proteger contra inyección SQL e inclusión de ficheros en un *site* Joomla!.

Por otro lado ambos WCMS están programados e incorporan directivas de protección contra ataques de tipo *SQL injection* y XSS, además de estar dotados de un sistema de reconocimiento interno para comprobar las terminaciones y las extensiones de los archivos que sean subidos mediante sendos gestores. Además los componentes de terceros que puedan ser configurados para permitir subidas de archivos por parte de un visitante, como puede ser una galería de imágenes, un formulario de contacto o un gestor de descargas, disponen de medidas propias de seguridad en su configuración, que van desde el uso de *captcha* hasta filtrar por IP a los usuarios que accedan al *site*.

Por último comentar que en el caso de Drupal debemos desactivar las alertas desde su propia configuración para no facilitar el trabajo a un usuario mal intencionado, ya que de no hacerlo estaríamos mostrando nuestras debilidades al atacante. La Tabla IV resume el análisis básico de seguridad realizado.

VI. CONCLUSIONES

Los WCMS permiten distribución de contenidos online para usuarios con escasos conocimientos informáticos y son relativamente sencillos de utilizar y gestionar. En cuestión de funcionalidades y apariencia podemos conseguir prácticamente cualquier resultado que deseemos para una web. En nuestro caso, hemos conseguido realizar tres *sites* prácticamente iguales en apariencia y aspecto mediante el uso de diferentes WCMS, aunque hemos requerido del uso de otro WCMS para dotar de la funcionalidad de gestión de archivos multiusuario en las versiones de Wordpress y Drupal debido a que las funcionalidades que estos ofrecían en el momento de realización de este trabajo no era tan extensa como en el caso de Joomla!. Todos los WCMS son bajo licencia libre y se pueden obtener en sus páginas web oficiales, necesitan de una base de

Tabla IV
RESUMEN DEL ANÁLISIS BÁSICO DE SEGURIDAD

	Joomla!	Drupal
¿Tiene comunidad?	Sí	Sí
¿Tiene informes de vulnerabilidad?	Sí	Sí
¿Hay sugerencias de seguridad en la instalación?	Sí. Se nos indicará que borremos la carpeta de instalación.	Sí
¿Existen módulos o componentes a terceros para dotar de seguridad extra?	Sí	Sí
Existen contramedidas para *SQL injection*	Sí	Sí
Warnings cuando vas a usar módulos de terceros	Siempre	Siempre
Comprueba las terminaciones de los archivos en subidas	Sí	Sí
Apis contra XSS	Sí	Sí
Protegido contra XSS	Sí	Sí
Usuario por defecto de administración	Inseguro	Inseguro
Protección contra *spam*	Básica. Suelen tener ambos puntos débiles en este campo en el registro de nuevos usuarios.	Básica. Es recomendable usar *Captcha*
Versiones recientes que solucionan problemas de versiones anteriores	Sí	Sí
Avisos de nuevas actualizaciones de componentes y módulos	Sí	Sí
Incluyen archivos que es necesario proteger mediante permisos	Sí. En ambos CMS debemos proteger de forma extra algunos archivos claves	Sí
Warnings peligrosos en *front-end*	No	Sí. Desactivarlos

datos para almacenar en ellas la información y se instalan a través del navegador web a través de un asistente.

En base al trabajo realizado, concluimos que en cuanto a la administración para gestión de contenidos y funcionalidades Joomla! ofrece el entorno más intuitivo, Drupal el más complicado de gestionar, y Wordpress tiene la ventaja de poder buscar estas funcionalidades desde su propio *back-end*. Respecto al importante apartado de dotar de aspecto a un *site*, los tres gestores disponen de un creador de plantillas Artisteer que además es muy sencillo de utilizar, esto nos otorga la posibilidad de conseguir prácticamente el resultado estético deseado.

Por otra parte las características más destacables de los WCMS analizados son las siguientes. Joomla! ofrece un enorme número de funcionalidades comparado con los otros WCMS estudiados y posee una comunidad de usuarios altamente activa en comparación con Drupal y Wordpress, lo que permite tener siempre disponible una ayuda importante ante cualquier dificultad que pueda aparecernos en el camino. Drupal posee el concepto de seguridad y de roles de usuario más potente y Wordpress es el mejor en términos de posicionamiento SEO, dando además la posibilidad de obtener un alojamiento gratuito. Por último comentar que tras dotar de seguridad extra a los *sites* y llevar a cabo el estudio de seguridad podemos afirmar que tanto Joomla! como Drupal son robustos frente a los ataques más comunes en la web como son SQL *injection* y *Cross-site scripting* (XSS). No obstante, pensamos que la seguridad de este tipo de sistemas seguirá siendo objeto de estudio. Desde el uso de *phising* hasta *malware* específico para desarrollos en PHP, nuevas amenazas surgen constantemente. Lo que implicará una continua actualización y aplicación de medidas de prevención, detección y recuperación especificas para este tipo de sistemas.

AGRADECIMIENTOS

This research was supported by the AEI/FEDER, UE project grant TEC2016-76465-C2-1-R (AIM).

REFERENCIAS

[1] W. F. Cody, J.T. Kreulen, V. Krishna, W. S. Spangler, "The integration of business intelligence and knowledge management", IBM Systems Journal, Vol. 41 (4), pp. 697-713, 2002.

[2] S. Bergstedt, S. Wiegreffe, J. Wittmann, D. Moller, "Content management systems and e-learning systems -a symbiosis?", Proc. 3rd IEEE International Conference on Advanced Technologies, pp. 155-159, 2003.

[3] R. McDaniel, J. R. Fanfarelli, R. Lindgren, "Creative Content Management: Importance, Novelty, and Affect as Design Heuristics for Learning Management Systems", IEEE Transactions on Professional Communication, Vol. PP (99), pp. 1-18, 2017.

[4] D. Barker, "Web Content Management: Systems, Features, and Best Practices", O'Reilly Media, 1st Ed, 2016. ISBN 978-1491908129.

[5] A. J. Aledo Hernández, "Sistemas de gestión de contenidos web: uso y estudio comparativo de su seguridad". Proyecto Fin de Carrera, I.T.T. especialidad Telemática. Directora: María Dolores Cano Baños. Octubre 2015

[6] S. K. Patel, V. R. Rathod, J. B. Prajapati, "Performance Analysis of Content Management Systems - Joomla, Drupal and Wordpress", International Journal of compuer Applications, vol. 21 (4), pp. 39-43, 2011.

[7] Joomla. Disponible en: <http://www.joomla.org>. Último acceso mayo 2017.

[8] Wordpress. Disponible en: <http://www.wordpress.com>. Último acceso mayo 2017.

[9] Drupal. Disponible en: <http://www.drupal.org>. Último acceso mayo 2017.

[10] W3Techs. Web Technology Surveys. Disponible en: https://w3techs.com/technologies/overview/content_management/all. Último acceso julio 2017.

[11] A. Mirdha, A. Jain, K. Shah, "Comparative Analysis of Open Source Content Management Systems", Proc. IEEE International Conference on Computational Intelligence and Computing Research, pp. 1-4, 2014.

[12] M. Meike, J. Sametinger, A. Wiesauer, "Security in Open source Web Content management systems", Internet Security & Privacy, Vol. 7 (4), pp. 44-51, 2009.

[13] H. Jerković, P. Vranešić, S. Dadić, "Securing web content and services in open source content management systems", Proc. 39th International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO), pp. 1402 – 1407, 2016.

[14] Proyecto Webscarab OWASP. Disponible en: < https://www.owasp.org/index.php/Proyecto_WebScarab_OWASP>. Último acceso julio 2017.
[15] A. O'Donell, "Tamper Data: The Firefox Add-on", Lifewire, abril 2017. Disponible en: < https://www.lifewire.com/firefox-addon-that-hackers-dont-want-you-to-know-about-2487289>. Último acceso julio 2017.
[16] S. K. Shivakumar, "Enterprise Content and Search Management for Building Digital Platforms", Wiley-IEEE Press, 2017. ISBN 9781119206842
[17] Artisteer. Disponible en: <http://www.artisteer.com>. Último acceso mayo 2017.
[18] Mailchimp. Disponible en: <http://www.mailchimp.com>. Último acceso mayo 2017.
[19] L. Khin Shar and H. Beng Kuan Tan, "Defeating SQL Injection", Computer, vol. 46 (3), pp. 69-77, 2013.
[20] I. Yusof and A. K. Pathan, "Mitigating Cross-Site Scripting Attacks with a Content Security Policy", vol. 49 (3), pp. 56-63, Computer, 2016.
[21] Acunetix. Disponible en: <http://www.acunetix.com>. Último acceso mayo 2017.