APLICACIÓN DE LA METODOLOGÍA DE PRUEBAS OWASP (OPEN WEB APPLICATION SECURITY PROJECT) PARA MEJORAMIENTO DE LA SEGURIDAD EN EL SISTEMA E-COMMERCE SIEMBRAVIVA.COM

MANUEL FERNANDO MARULANDA AGUIRRE
JACOBO DIAZ MONTES

UNIVERSIDAD NACIONAL ABIERTA Y A DISTANCIA - UNAD
ESCUELA DE CIENCIAS BASICAS TECNOLOGIAS E INGENIERIAS
ESPECIALIZACION EN SEGURIDAD INFORMÁTICA
MANIZALES
2018

APLICACIÓN DE LA METODOLOGÍA DE PRUEBAS OWASP (OPEN WEB APPLICATION SECURITY PROJECT) PARA MEJORAMIENTO DE LA SEGURIDAD EN EL SISTEMA E-COMMERCE SIEMBRAVIVA.COM

MANUEL FERNANDO MARULANDA AGUIRRE
JACOBO DIAZ MONTES

Proyecto aplicado de seguridad informática para optar al título de especialista en seguridad informática.

Asesores de Proyecto
Ing. Jorge Hernando Peña Hidalgo

UNIVERSIDAD NACIONAL ABIERTA Y A DISTANCIA - UNAD
ESCUELA DE CIENCIAS BASICAS TECNOLOGIAS E INGENIERIAS
ESPECIALIZACION EN SEGURIDAD INFORMATICA
MANIZALES
2018

Nota de Aceptación

_____

_____

_____

_____

Presidente del Jurado

_____

Jurado

_____

Jurado

Manizales, 11 de agosto de 2018.

DEDICATORIA

Dedicamos este trabajo principalmente a Dios, por habernos permitido el haber llegado hasta este momento tan importante en nuestra formación profesional.

A nuestros padres y hermanos que fueron pilares incondicionales de apoyo para lograr todas las metas propuestas, inculcando siempre pensamientos de lucha por ser mejores profesionales en nuestras áreas de acción.

A todos nuestros compañeros y amigos, que compartimos juntos durante nuestro proceso de formación como especialistas en seguridad informática.

Finalmente, a los docentes y tutores que nos apoyaron durante esta etapa de formación.

AGRADECIMIENTOS

Agradecemos al ingeniero y especialista José Hernando Peña Hidalgo por su acompañamiento durante el desarrollo de este proyecto aplicado, en el cual se comprometió a guiarnos durante todo el desarrollo del proyecto de grado para optar al título como especialistas.

A la Universidad Nacional Abierta y a Distancia, CCAV Dosquebradas, por permitirnos culminar nuestros estudios de postgrado.

CONTENIDO

Pág.

LISTA DE FIGURAS

Pág.

LISTA DE TABLAS

Pág.

INTRODUCCIÓN

En la actualidad la problemática que aqueja principalmente al mundo informático y empresarial es la deficiente seguridad informática que está siendo implementada ya sea en un equipo de escritorio, en un sistema de información o una solución robusta de procesamiento de datos.

Solo en el contexto local se estima que hay pérdidas económicas de 500 millones de dólares, lo cual corresponde al 0.14% del PIB. Las empresas en Colombia no están teniendo en cuenta los baches de seguridad que existen para sus sistemas, por esta razón no se están tomando medidas a tiempo y esto permite que los ataques sean mucho más constantes y que generen mayores pérdidas económicas.

Una de las preocupaciones de la seguridad informática es el comercio electrónico, las páginas que están destinadas a ofrecer este tipo de servicios manejan transacciones bancarias y que deben tener una protección especial de los datos del cliente para que esa información no caiga en manos inescrupulosas.

Este proyecto está dirigido a conocer las vulnerabilidades del sitio de comercio electrónico Siembraviva.com, este sitio busca una manera de ofrecer facilidades para la adquisición de productos de primera necesidad de tal manera que pueden ser adquiridos en línea. Se busca realizar un análisis de vulnerabilidades basándose en la metodología OWASP versión 3 para identificar el nivel de seguridad empleado y las falencias que pueden existir en el sitio web, posteriormente se plantearán los resultados del análisis determinando el nivel de vulnerabilidad del sitio.

1. DEFINICION DEL PROBLEMA

1.1. ANTECEDENTES DEL PROBLEMA

Dado al crecimiento vertiginoso de la web, esta ha originado mayores posibilidades de ser vulnerables a cualquier tipo de ataque informático, desde accesos no autorizados, robo de credenciales o sustracción de información confidencial de clientes y servicios.

De esta forma, la seguridad de los sitios web se ha convertido en un tema de mucha importancia, de tal manera que se requieren aplicaciones web con altos niveles de seguridad, integridad, confiabilidad, confidencialidad y disponibilidad, ya que por una mala administración se puede ver comprometido el funcionamiento del sitio web, recursos de tiempo y dinero. En el caso de SiembraViva el análisis de vulnerabilidades se va a efectuar para prevenir posibles ataques y establecer en términos de seguridad en que niveles esta la aplicación Web, hasta el momento no se han tenido reportes de ataques en la página web.

1.2. FORMULACIÓN DEL PROBLEMA

¿Cómo los resultados de las pruebas de la metodología OWASP ayudarán a disminuir los problemas de seguridad y mejorará la administración de la seguridad, podrán mejorar la seguridad en el sistema e-commerce SiembraViva.com?

1.3. DESCRIPCIÓN DEL PROBLEMA

Respecto a la seguridad de los sistemas e-commerce, varios de los problemas de seguridad que más se evidencian son los relacionados a problemas de accesos no autorizados de usuarios externos al sistema de información, sustracción de credenciales de acceso y robo de información personal y financiera de los usuarios. Todas estas fallas son originadas por una posible mala administración en los niveles de usuarios y respectivos permisos según la jerarquía en el sitio.

Como consecuencia de los accesos no autorizados y de la mala administración del sistema se tienen fugas de información, lentitud de respuesta de las funcionalidades de la página y fallas autenticación dentro de la aplicación; al aplicar este proyecto se van a detectar que módulos se deben reestructurar y mejorar el sistema, agregar confiabilidad como resultado y ser más estable la aplicación para los usuarios.

En cuanto la fuga de información, aplicando la metodología OWASP 3.0 se puede mitigar los riesgos existentes para la aplicación. Los usuarios también tienen parte de responsabilidad debido a la baja fortaleza de las contraseñas y como y donde se accede a la aplicación, estaciones de trabajo que no son confiables y que contribuyen a que el sistema colapse. La finalidad de este análisis al sitio web siembraviva.com es realizar un diagnóstico de las debilidades con las que cuenta el sistema para poder sugerir medidas que mejores las falencias encontradas.

La seguridad del sitio se encuentra en buenas condiciones, pero es importante realizar los análisis pertinentes para proteger a los usuarios y a la compañía.

2. JUSTIFICACIÓN

El tráfico en la red se ha venido incrementando con el pasar del tiempo. Una cantidad enorme de operaciones se hacen en la red, trámites, operaciones bancarias, intercambios de información y otras tareas que son llamativas para robar información crítica para las personas o empresas. Los accesos a la red se pueden dar desde cualquier lugar y esto permite que no haya un control de quien o cuando pueda acceder a un recurso de la red.

Toda aplicación o servicio web que disponga de altos niveles de seguridad es un elemento clave y diferenciador en el nivel de generar confianza y valor agregado de la empresa para todos sus clientes. De esta forma, el aumento en el uso de las TICS exige que dichos sistemas se adapten a las nuevas tendencias y con ellas mejoras en su seguridad.

Los recursos informáticos con los que cuentan las empresas deben tener un nivel de seguridad aceptable que de un respaldo frente los problemas que surgen cada día en términos de seguridad informática; una aplicación o un sitio web que cuente con la seguridad apropiada es un elemento de confianza y genera valor para la empresa.

Las empresas que tomen medidas y que inviertan recursos para mejorar la seguridad informática a futuro disminuirán su preocupación por la pérdida de información y por la pérdida de recursos a nivel financiero.

La importancia radica en el aumento de la confiabilidad al usar la aplicación por parte de los usuarios y da a la empresa una garantía de que el servicio prestado va a tener la disponibilidad requerida por parte del consumidor; tanto usuario como empresa verán reflejado en sus bolsillos que la aplicación sea auditada y se corrijan los errores a tiempo.

Los beneficios que puede traer a la empresa la aplicación de este proyecto son:

- Identificar las amenazas potenciales que tiene el sistema e-commerce SiembraViva.com.
- Conocer los niéveles de seguridad que presenta el sistema e-commerce SiembraViva.com

- Tener una base de las medidas que se deben tomar para poder quitar o disminuir los baches de seguridad de la página web.
- Tomar medidas efectivas que disminuyan la pérdida de recursos financieros.

3. OBJETIVOS

3.1. OBJETIVO GENERAL

Realizar un análisis de riesgos a la seguridad de la información en el sistema e-commerce SiembraViva.com, haciendo uso de guía de pruebas de la metodología OWASP versión 3.0.

3.2. OBJETIVOS ESPECÍFICOS

- Investigar el proyecto OWASP y el manual de pruebas de pentesting propuesta en su versión 3.0, para determinar el plan de pruebas que se aplicaran sobre el sistema e-commerce SiembraViva.com bajo el enfoque de análisis de seguridad en las aplicaciones web.
- Aplicar las pruebas de la metodología OWASP 3.0 de seguridad de aplicaciones web y analizar los resultados de estas para determinar el nivel de seguridad del sistema e-commerce SiembraViva.com.
- Determinar los controles de acuerdo con los resultados de las pruebas de la metodología OWASP que permitan disminuir el impacto de las vulnerabilidades y mejorar la seguridad del sistema e-commerce SiembraViva.com.
- Elaborar un informe sobre el resultado de las pruebas realizadas, nivel de seguridad identificado y los controles propuestos de acuerdo con los resultados de las pruebas de la metodología OWASP del sistema e-commerce SiembraViva.com.

4. ALCANCE

El alcance del proyecto busca aplicar los procesos de análisis y pruebas de la metodología de pentesting OWASP (Open Web Application Security Project) para determinar el nivel de seguridad que posee el sistema e-Commerce SiembraViva.com ubicada en la ciudad de Medellín, la cual proporciona productos sin agrotóxicos por medio de un sistema de gestión de suministros permitiendo realizar compras y pagos por internet.

El proyecto busca aplicar una auditoria web de pentesting utilizando una de las metodologías más famosas por ser gratuita y abierta, OWASP (Open Web Application Security Project) siendo un proyecto de código abierto dedicado a determinar y combatir las causas que hacen que una aplicación web sea insegura.

Dentro de los procesos involucrados en el testing del sistema web e-commerce, se cubren las pruebas específicas de vulnerabilidades, como por ejemplo, A1-Inyección, A2-Secuencia de Comandos en Sitios Cruzados (XSS), A3-Perdida de Autenticación de y Gestión de Sesiones, A4-Referencia Directa Insegura de Objetos, A5-Falsificación de Petición de Sitios Cruzados (CSRF), A6-Configuración Defectuosa de Seguridad, A7-Almacenamiento Criptográfico Inseguro, A8-Falla de restricción de acceso URL, A9-Protección Insuficiente de la Capa de Transporte, A10-Redirecciones y Destinos No Validos.

5. MARCO REFERENCIAL

5.1. ANTECEDENTES

Las pruebas de pentesting son un elemento fundamental para hacer una revisión de cómo está la seguridad de un sitio web que necesita ser seguro tanto como para los usuarios como para la empresa. Es legal y cuenta con la autorización de las partes implicadas, brinda pruebas al administrador del sistema o a la alta gerencia de que pasaría si se detecta la vulnerabilidad y no es corregida de manera oportuna. Esta técnica pentesting analiza la vida real frente al sistema.

Por medio de las pruebas conocidas como caja blanca se puede hacer un test para determinar la debilidad que existe en el sistema, se muestra como una perspectiva de un usuario real; teniendo en cuenta el objetivo. También se tienen las pruebas caja negra que difiere con las de caja blanca que estas no tienen claridad del objetivo del ataque.

Para realizar la evaluación del sistema se cuenta con tres elementos: testing, examen y entrevista. Examen se estudian o analizan puntos que deben ser evaluados buscando incrementar el detalle del resultado. Testing se evalúan los objetivos que deben ser cumplidos y comparar como es su comportamiento real. Por medio de la entrevista se realizan reuniones personales para determinar problemas y facilitar el entendimiento de procesos de la organización.

El pentesting es una técnica ampliamente usada para el análisis de sitios web o aplicaciones, muchas empresas como DragonJar[1] en el contexto local están dedicadas a realizar este análisis y al mismo tiempo guiar sobre las acciones requeridas que se deben tomar para que ya no existan estas vulnerabilidades. Las técnicas y metodologías para realizar este proceso ya están documentadas y probadas en múltiples sitios web del ámbito global y nacional.

[1] DragonJAR, Restrepo, J. A., & Tibaquirá, J. (20 de marzo de 2016). *DragonJAR Soluciones y Seguridad Informática S.A.S.* Obtenido de Comunidad DragonJAR: http://www.dragonjar.org/

5.1.1. Web App Security: Django and the OWASP Top 10.

El Top Ten de OWASP proporciona un documento de la conciencia de gran alcance para la seguridad de aplicaciones web. La lista representa un amplio consenso acerca de lo que los defectos más críticos de seguridad de aplicaciones web son.

- OWASP es llegar a los desarrolladores, no sólo la comunidad de seguridad de aplicaciones
- El Top 10 es acerca de la gestión del riesgo, no sólo evitando vulnerabilidades
- Para gestionar estos riesgos, las organizaciones necesitan un programa de gestión de riesgos, no es suficiente la formación de la conciencia en los usuarios; es importante aplicar medidas que fortalezcan las seguridad interna y perimetral de aplicativo o página web.

Además, el sistema de autenticación de usuario integrada de Django no permite fuera de las instalaciones redirigir enlaces como parámetros de URL. Tenga en cuenta que todavía desea auditar el código para cualquier uso manual de enlaces de redireccionamiento de URL en los parámetros.

5.1.2. OWASP Top Ten Series: Sensitive Data Exposure

La seguridad es una tarea multifacética. Como se indica en el artículo de la serie de OWASP sobre Mala configuración de seguridad, hay muchas maneras de exponer las vulnerabilidades en las aplicaciones web. Definitivamente, usted debe leer y pensar en el consejo en ese puesto y los demás artículos de OWASP. En referencia específica a la prevención de la exposición de información confidencial, tenemos que:

- El top ten contribuye a la protección de datos personales.
- La exposición de fallas de datos personales dentro de la página puede afectar a los usuarios como a la compañía.

5.2. MARCO CONTEXTUAL

5.2.1. Reseña Histórica SiembraViva

La empresa SiembraViva nace como una idea de negocio en septiembre del año 2011, idea de tres amigos profesionales en las ramas de Ingeniería de Sistemas y un ingeniero agronómico, que buscaban desarrollar una solución informática

innovadora que se diferenciara de las actuales empresas de desarrollo de software, basándose en la experiencia de estos profesionales en el mercado nacional e internacional. A pesar de este inconveniente se hicieron esfuerzos individuales de los tres por continuar madurando esta idea de negocio. En agosto de 2012, se reúnen nuevamente dos de ellos, para definir el futuro de la empresa. Es así como deciden continuar formalmente y deciden constituirse como SiembraViva.com. [2]

Acorde a las áreas de desarrollo profesional se detectó una necesidad, la adquisición de productos de la canasta familiar se hace tediosa y en ocasiones la confiabilidad de los productos no es la esperada. El concepto de Siembraviva se mantuvo como una empresa de e-commerce que facilita la venta y promoción de productos agrícolas, naturales y con precios cómodos para los usuarios. Inicialmente se planeó el lanzamiento en Medellín para probar la efectividad del negocio, los resultados fueron satisfactorios y la empresa se extendiendo a otras localidades cercanas de Medellín. (SiembraViva, 2014).

5.2.2. Misión SiembraViva

En el 2020 SiembraViva será una organización de productos para la siembra con enfoque empresarial, ofertando al mercado productos de calidad. Se busca satisfacer necesidades del cliente a través de mejores alternativas para la siembra, brindándoles a nuestros clientes una atención esmerada y oportuna con responsabilidad.

5.2.3. Visión SiembraViva

Ser una empresa de un producto para sembrar en el cual se va a pensar en la comodidad de un sector específico el cual son los agricultores cuando necesite un producto y servicio de calidad, entregando responsabilidad y logremos tener la confianza de las personas que adquiera nuestro producto.

---

[2] SiembraViva. 2014. Acerca de SiembraViva.com. [En línea] 2014. [Citado el: 07 de febrero de 2017.] https://siembraviva.com/home/quienes-somos/.

5.2.4. Organigrama

En la figura 1 podemos observar el organigrama estructural de la empresa SiembraViva, que incluyen las estructuras departamentales o áreas, o las personas encargadas que dirigen dichas áreas.

Figura 1. Organigrama SiembraViva

Fuente: SiembraViva.com

5.2.5. Activos Informáticos

Tabla 1. Activos informáticos SiembraViva

Tipos de activos	Descripción
Activo de información	Bases de datos remota de los clientes y de sus productos.
Software o aplicación	Ofimática Office, software de contabilidad Apolo. Photoshop

Tipos de activos	Descripción
	Adobe, entre otros.
Hardware	Servidores en la nube, tres portátiles marca Toshiba y un Mac desktop.
Red	Router Cisco, swicth 3com.
Equipamiento auxiliar	UPS.
Instalación	Montaje de oficinas, cableado de las instalaciones.
Servicios	Mesa de ayuda, conectividad cableada y WIFI.
Personal	Soporte a usuario final, administrativo y desarrolladores.
Fuente: Activos SiembraViva.com	

5.2.6. Activos De Información

Tabla 2. Activos de información SiembraViva

Tipos de activos	Descripción
Sistema e-commerce	SiembraViva.com
Gestor de contenidos	Wordpress 4.2.2
Lenguaje de programación	PHP 5.6.18
Servidor	Apache 2.4.6
Sistema operativo	CentOS
OpenSSl	1.0.1e-fips
Motor de bases de datos	MySQL
Métodos de pagos	Permite realizar pagos online con el sistema PSC, tarjetas de crédito VISA, American Express, MasterCard
Seguridad	Utilizan sistemas de cifrado Secure Socket Layer (SSL) que codifica la información durante la compra.
Certificado de seguridad	Certificado de seguridad emitido por COMODO RSA bajo el protocolo TLS 1.2 con intercambio de claves (ECDHE_RSA), sistema de cifrado fuerte (AES_128_GCM).

Tipos de activos	Descripción
Plugins externos	Yoast SEO, Google Analytics, Twitter Emoji, Facebook, Clicky,
Fuente: Análisis Wappalyzer	

5.2.7. Funcionalidades Del Sistema E-Commerce

En el sitio web SiembraViva.com existente los siguientes módulos que garantizan la funcionalidad del sistema e-commerce:

5.2.7.1. Catálogo de productos y servicios.

Es algo más que un listado de los productos y servicios que el sistema e-commerce ofrece a sus clientes, estos catálogos esta ordenados de tal forma que su presentación tiene una lógica de impacto, con su descripción y graficas representativas.

5.2.7.2. Cesta o carrito de la compra.

Es una forma para controlar lo que vamos comprando y de esta forma poder añadir, eliminar o modificar cualquier producto sin tener que dirigirnos a otras páginas para ello. Por eso, es muy importante la manipular del carrito de compras desde la misma página donde estemos y que siempre esté visible.

5.2.7.3. Auto-promo y gestión de ofertas.

El precio es fundamental y decisivo a la hora de comprar; por ello, siempre debe esta visible. Además, una plataforma de ventas online debe tener la opción de gestionar y mostrar claramente las promociones u ofertas que estén en ese momento disponibles.

5.2.7.4. Registro de usuarios y panel de control de usuarios.

En la mayoría de los casos, antes de poder comprar en una tienda online, el comprador debe registrarse en el sistema e-commerce y poder registrar con los datos personales los datos de contacto y envió.

5.2.7.5. Motor de búsquedas.

Es una manera rápida que el cliente puede utilizar para buscar directamente el producto por el que está interesado, conocer su disponibilidad y cantidad.

5.2.7.6. Proceso de compra.

Es corto y claro. El comprador no puede estar perdido en el proceso de compra porque podría suponer un abandono. El comprador tiene a su disposición un carrito de compras donde anexa todos los productos que desea comprar para luego proceder con el método de pago y despacho hasta la puerta de su casa.

5.2.7.7. Métodos de pago.

Es un momento crítico y cuando se producen más abandonos. Se debe ser flexible en cuanto a la forma de pago y en las opciones de entrega del envío.

5.2.7.8. Servicio posventa.

Hay tantas posibilidades como plataformas o canales de comunicación. Como las redes sociales Facebook, Twitter, Instagram, Pinterest, WhatsApp, el correo electrónico, teléfono y los puntos de entregas SiembraViva.

5.3. MARCO TEÓRICO

OWASP es un proyecto que se dedica principalmente a ampliar, adquirir y a conservar aplicaciones que son confiables para los dispositivos en los que se ejecute, en el 2013 se lanzó el “top 10” que permite establecer un margen de seguridad en las aplicaciones por medio de la identificación y exaltación de los riesgos más críticos que desafían la seguridad de las organizaciones.

Para establecer el proyecto top 10 se llevó a cabo un estudio por medio de fuentes bibliográficas, libros y diferentes herramientas, se tomaron datos estandarizados incluyendo PCI, DISA, FTC entre otras; el top 10 fue denominado con este nombre debido a la conmemoración número 10 del proyecto, teniendo en cuenta la importancia de adquirir seguridad en las aplicaciones web.

Las técnicas Owasp permiten identificar por medio de una comparación si las pruebas de penetración son efectivas al momento de encontrar la vulnerabilidad en el estado de seguridad de las aplicaciones web, ayudando a mitigar los riesgos de privacidad dentro de las aplicaciones.

5.3.1. A1 – Inyección SQL

La inyección directa de los comandos SQL es una habilidad maliciosa donde el atacante daña los comandos SQL exhibiendo dentro de ellos datos etéreos y

ocultos ubicando así peligrosos comandos en la base de datos, donde el atacante logra ajustar de forma directa los parámetros definidos en el momento que se realiza una consulta SQL, tomando como punto de partida ejemplos reales donde la inyección directa de SQL es efectiva. Existe un acoplamiento de caracteres que tienen un significado específico para el manejador de la base de datos teniendo en cuenta que estos caracteres hacen parte de dicho lenguaje del manejador; entre ellos encontramos algunos ejemplos claros de los ataques de la inyección SQL:

- ‘o “(caracteres que indican cadenas de texto)
- # (línea simple comentada)
- /* … */ (varias líneas comentadas)
- + (suma o concatenación)
- % (indicador para varios caracteres)

Por medio de estos caracteres un atacante puede acceder a la manipulación total de la información de las aplicaciones enviando dichos comandos mencionados anteriormente, llevándolo a la realidad encontramos como la inyección SQL accede a los comandos del sistema operativo en los equipos accediendo directamente a las bases de datos, donde se almacena la mayor cantidad de información del equipo.

En la figura 2 podemos observar el ejemplo de una consulta que podría originar el ataque en el sistema operativo que hospeda la base de datos (Servidor MYSQL).

Figura 2. Consulta SQL en PHP

```php
<?php
$consulta  = "SELECT * FROM products WHERE id LIKE '%$prod%'";
$resultado = mssql_query($consulta);
?>
```

Fuente: El autor

En la figura 3 podemos observar que cuando el atacante envía el valor a%’ exec master xp_ cmdshell ‘net user test testpass /ADD’ – a $pord, la $consultaserà;

Figura 3. Consulta SQL en PHP vulnerable

```
<?php
$consulta  = "SELECT * FROM products
              WHERE id LIKE '%a%'
              exec master..xp_cmdshell 'net user test testpass /ADD' --%'";
$resultado = mssql_query($consulta);
?>
```

Fuente: El autor

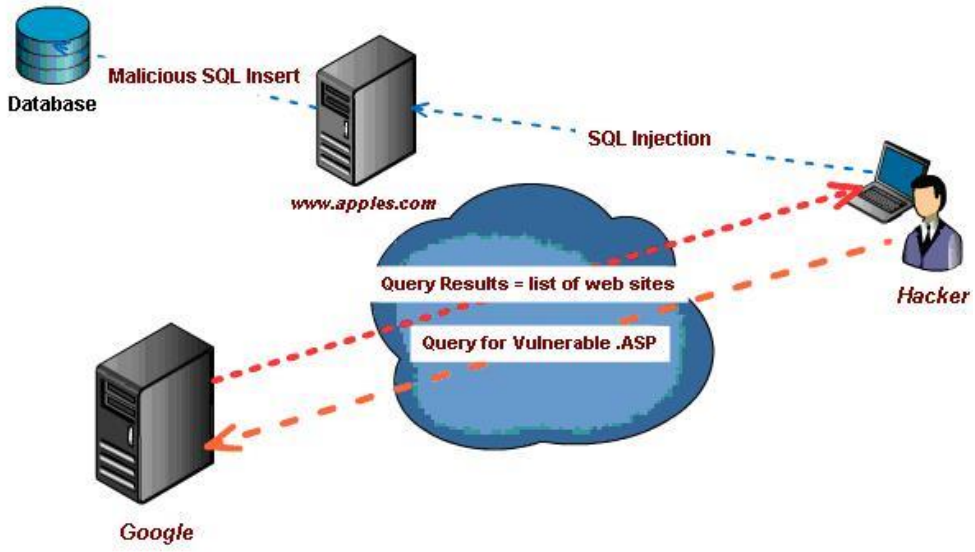
En este ejemplo el servidor MYSQL elabora la sentencia SQL en el lote estableciendo un comando para insertar un nuevo usuario a la base de datos, teniendo en cuenta que si esta aplicación se estuviera efectuando como (sa) y el servicio MYSQLSERVER se estuviera produciendo con la libertad suficiente el atacante poseería de dicho modo una cuenta sigilosa con la cual accede al equipo y hurta la información que allí se encuentra tendiendo total control de la misma.

5.3.1.1. Protección contra Blind SQL Injection.

Esta actividad se realiza evaluando la seguridad en las aplicaciones realizando un análisis contra la inyección de código, los impulsadores de los lenguajes de programación y ejecución de las aplicaciones web ofrecen mejores técnicas para la protección de la inyección de código, estableciendo varias recomendaciones para evitar un posible ataque, teniendo en cuenta que los datos pueden ser modificados al principio o al final, se realiza la ejecución de tratamiento para los posibles casos de ataques maliciosos, es recomendable usar códigos que permitan la ejecución de consultas pre-compiladas y así evitar que los atacantes extraigan la información que esta almacenada en las aplicaciones; es trascendental examinar y poseer control sobre todas las posibilidades que generan error, fiscalizando esta acción por medio de un procedimiento seguro y confiable así se evita que el atacante logre tener control y manejo de la información de las aplicaciones. De esta forma, teniendo un registro total de la margen de error establecida se procede a auditar los errores de la aplicación para evitar un posible ataque.

En la figura 4 podemos observar la estructura del *Ataque Blind SQL Injection* recreado por Panda Security bajo un entorno posible.

Figura 4. Ataque Blind SQL Injection, Autor: Panda Security



Fuente: SiteSecurity

5.3.2. A2 - Pérdida de autenticación y gestión de sesiones

Se basa en la mala ejecución de técnicas de autenticación o gestión de sesiones que pueden comprometer los elementos utilizados para encontrar el acceso a las herramientas como las contraseñas, llaveros cifrados entre otros.

Encontrar vulnerabilidades en este método se puede realizar de la siguiente manera dependiendo del tipo de usuario si es interno o externo con la ejecución de las herramientas adecuadas para su identificación, la seguridad en la web es poco importante para algunos usuarios lo que conlleva a la vulnerabilidad, en relación con la perdida de autenticación y gestión de sesiones las aplicaciones Web se vuelven indefensas cuando un atacante accede a sus herramientas, ya que el atacante posee control de realizar suplantaciones de usuarios para llegar a estropear los registros de la aplicación, debido a este tipo de acciones se originan accesos no autorizados a la información que se encuentra entrelazada y recopilada en el servidor y los servicios de las aplicaciones que han sido afectadas.

Este tipo de ataque se encuentra principalmente en la gestión de las contraseñas o en el momento de cerrar las sesiones de las aplicaciones, este ataque está enfocado a la recuperación automática de las contraseñas en nuestras cuentas.

Un ejemplo claro de este tipo de ataque se puede evidenciar en la figura 5.

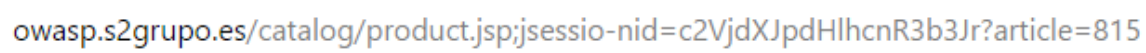
Figura 5. Reescritura de URL poniendo los ID de sesión en la dirección

Fuente: El autor

En el ejemplo se puede observar que un usuario quiere compartir la oferta de vuelos a sus compañeros y por medio del correo electrónico envía el enlace anterior a sus amigos sin tener conocimientos de que ha enviado su ID de sesión, al momento que sus compañeros acceden al enlace, se abre la sesión y podrán utilizar la tarjeta de crédito del usuario.

En la figura 6, se evidencia este ataque por medio de una aplicación que tiene una URL en el que el usuario que se encuentra autenticado realiza diferentes búsquedas de tipo o categoría, el usuario encuentra un artículo que le llama la atención y quiere compartirlo a sus amigos y accede a la URL que se encuentra en su navegador.

Figura 6. Ejemplo de ataque



Fuente: El autor

El usuario cierra el navegador y publica el anterior enlace en una red social para que sus amigos accedan a ella, el usuario no cerró la sesión y por consiguiente el servidor en el que se encuentra la búsqueda realizada por el usuario tampoco, sus amigos pueden acceder al enlace como el usuario autenticado y por consiguiente a su información.

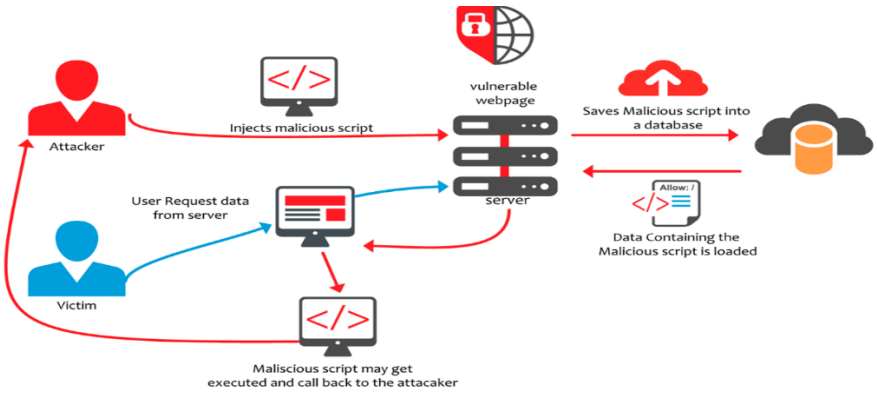
5.3.3. A3 - Secuencia de comandos en sitios cruzados XSS

En esencia la secuencia de comandos en sitios cruzados es un tipo de inyección HTML, este tipo de ataque es el más perjudicial en cuanto a la seguridad que requieren las aplicaciones, esta acción maliciosa se efectúa en el momento en que una aplicación adquiere información producida por un usuario y es enviada a un navegador web sin ser aprobada o validada por su contenido.

Los atacantes tienen el poder de realizar ejecuciones en cuanto a las secuencias de comandos en un navegador Web perteneciente al usuario que en este caso es la víctima del ataque, XSS permite al atacante realizar ciertos cambios en sus aplicaciones ya que puede modificar datos, realizar cambios en diferentes sitios web que con gran frecuencia utiliza la víctima, también pueden colocar dentro de las aplicaciones contenido incompatible con la información que allí se tenga registrada, por medio de comandos maliciosos el atacante toma el control total de las búsquedas que se realizan en el equipo de la víctima por medio de los navegadores, los atacantes utilizan JavaScript para sus ataques aunque existen varios comandos que son usados para los ataques.

En términos generales esta acción ocurre cuando el atacante recurre a una aplicación web para enviar un código malicioso por medio de un Script dentro del navegador a la víctima; es difícil identificar si el Script contiene contenido malicioso porque la victima puede pensar que este proviene de un sitio que es confiable y lo ejecuta, cuando el script malicioso es ejecutado este obtiene acceso a los tokens de sesión, a las cookies entre otras como se evidencia en la figura 7.

Figura 7. Ataque Cross Site Scripting -XSS

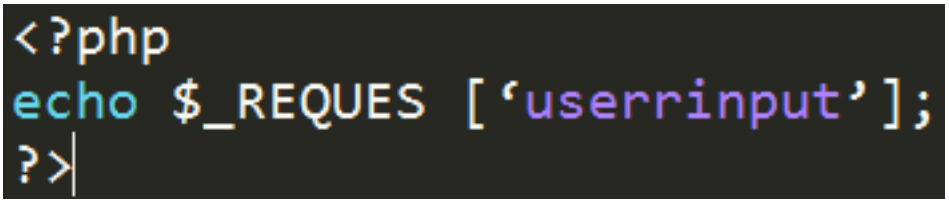


Fuente: Security-EC.com

No todas las herramientas de la web están protegidas contra los ataques, aunque estas se encuentren automatizadas es posible encontrar problemas XSS. En cuanto a las aplicaciones web cada una genera páginas de salida diferentes a su contenido del lado del navegador como ActiveX, JavaScript, Silverlight entre otras, este tipo de acciones hace que sea más ardua la detección automatizada.

En la figura 8 podemos observar una vulnerabilidad que podría ocasionar un posible ataque:

Figura 8. Ataque XSS



Fuente: El autor

En este ataque XSS coloca información incompatible y es almacenada en una base de datos, después esta información pasa a ser mostrada sin ningún filtrado, esto afecta especialmente a los sistemas de administración como los foros o los blogs ya que dentro de ellos otros usuarios acceden a la información y se registra posteriormente la de ellos.

5.3.3.1. Protección contra XSS.

En cuanto a la protección contra XSS se llega a la conclusión que por medio de la combinación de validación de listas blancas se permite verificar la dirección de los ataques y estas listas tienen la capacidad de prevenir los ataques maliciosos.

Una de las formas para prevenir los ataques en las aplicaciones es estandarizar un mecanismo de validación de entrada que permita revisar la información que es ingresada o almacenada y omitir la información maliciosa teniendo en cuenta aquellos mensajes de errores; se debe realizar la codificación fuente de salida que permite que toda la información que el usuario suministra sea debidamente codificada por medio de XML o HTML teniendo en cuenta el mecanismo de salida de esta.

También se debe realizar un proceso de especificación de la codificación de salida como (UTF8) y denegar al atacante que coloque esta codificación en el nombre de los usuarios, es importante no utilizar la validación de lista negra (blacklist) ya que los atacantes reemplazan los caracteres (“<” “>”) o las frases como script y les es posible atacar sigilosamente.

5.3.4. A4 - Referencia directa insegura a objetos (Direct Object Reference)

Esta acción sucede cuando un programador muestra una referencia destinada a un punto interno de la aplicación, como por ejemplo al registro de la base de datos o a la URL, los atacantes acceden a este tipo de herramientas para sacar información confidencial sin autorización del usuario, como medida de prevención se recomienda colocar un control de accesos para que no se efectúe dicho ataque.

Como es bien sabido las aplicaciones muestran a sus usuarios objetos internos y el atacante puede acceder a estos objetos internos y realizar modificaciones, saltándose un control de accesos malintencionadamente, en el siguiente caso se establece que el código utiliza la entrada del usuario para modificar y cambiar nombres de fichero, el atacante ingresa a otros recursos y secciones más privadas de la aplicación obteniendo toda su información.

Figura 9. Ejemplo de referencia insegura

```
<select name="language"><option value="fr">Francais</option></select>
...
require_once ($_REQUEST['language']."lang.php");
```

Fuente: El autor

En la figura 9 y 10 es posible evidenciar un ejemplo claro en donde la aplicación usa datos no verificados en una llamada SQL y posteriormente accede a la información de la cuenta:

Figura 10. Ejemplo de datos no verificados

```
String query = "SELECT * FROM accts WHERE account =?";
PreparedStatement pstmt = connecCon.prepareStatement(query,…);
pstmt.setString(1,request.getparameter("acct"));
ResultSetresults = pstmt.executeQuery();
```

Fuente: El autor

En la figura 11, el atacante cambia el parámetro (acct) en el navegador para acceder a los números de cuenta que desee, en el ejemplo se evidencia el cambio:

Figura 11. Ejemplo de datos modificables a través de URL

Fuente: El autor

5.3.4.1. Prevención del ataque.
Se debe proceder a usar referencias indirectas por usuario y por las sesiones a las que ingrese, esta acción le impide al atacante acceder a la información y a los recursos no autorizados,

Un ejemplo claro es que en vez de usar la clave del recurso de bases de datos se procede a utilizar los números de acceso y así revelar el valor que el usuario eligió, entonces la aplicación debería realizar una similitud entre la clave de la base de datos y la referencia indirecta, después se procede a comprobar el acceso, ya que si no hay una previa comparación de control de acceso para verificar si el usuario está registrado, es posible que exista vulnerabilidad y por consiguiente un posible ataque.

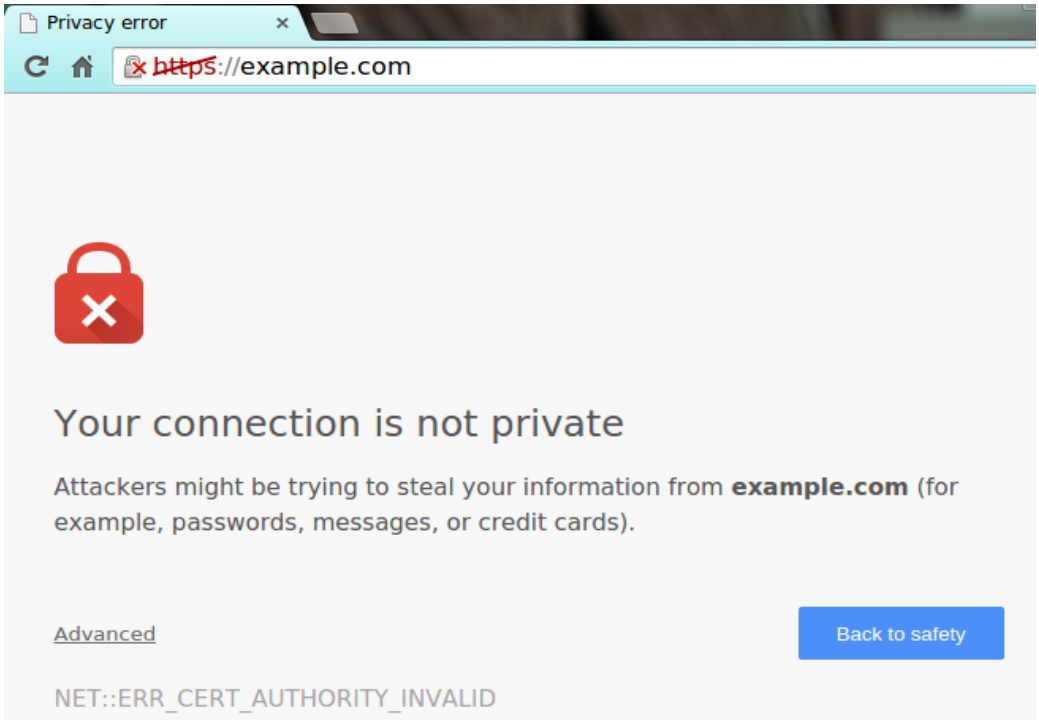
5.3.5. A5- Configuración de seguridad incorrecta
Se define como una mala configuración de seguridad en lo que compete a las aplicaciones web, ya que va desde la aplicación hasta el servidor, si no se efectúa una correcta configuración de la base de datos de la aplicación, de los marcos de

trabajo y del servidor de la aplicación este se encuentra vulnerable a los ataques, por lo que se recomienda actualizar el software seguidamente.

Por consiguiente, las aplicaciones usan el nombre real de las mismas para crear páginas web en cuanto a su promoción, las aplicaciones se confían de la poca seguridad que hay en ellas y no confirman si el usuario que accede está autorizado para realizar dicha acción, este es un punto de referencia para los atacantes ya que por ahí pueden establecer los diferentes métodos de ataque a las aplicaciones, obteniendo los valores de los parámetros y conseguir los códigos de acceso a las aplicaciones.

En la figura 12 es posible identificar un certificado SSL con una configuración incorrecta o mal parametrizado.

Figura 12. Configuración incorrecta certificado SSL



Fuente: Security-EC.com

5.3.5.1. Prevención del ataque.

Es importante realizar periódicamente escaneos y hacer auditorias para encontrar las falencias de las aplicaciones y prevenir posibles ataques, puertos abiertos o servicios innecesarios.

5.3.6. A6 – Exposición de datos sensibles

Se encuentran falencias en las aplicaciones ya que dentro de su manejo operacional no se resguardan datos sensibles como las credenciales de autenticación o los números de las tarjetas de crédito de los usuarios, cuando los atacantes detectan este tipo de falencias en las aplicaciones ingresan sigilosamente y modifican los datos para obtener varias identidades y así efectuar millonarios robos; para proteger los datos sensibles es primordial utilizar el cifrado de datos y ciertas reservas o mesuras cuando se produce un intercambio de información por medio de los navegadores.

Las aplicaciones almacenan su información con un cifrado inseguro, ya que carece de algoritmos o simplemente se asignan contraseñas débiles por parte de los usuarios, los cuales no están cambiando las claves. Las aplicaciones tienden a confiar en la seguridad perimetral que se implementen confiando en su fortaleza, también tienen deficiencias en la capa de transporte y por consiguiente en la protección de del tráfico de peticiones y no poseen un adecuado uso de SSL/TLS.

Dentro de las aplicaciones se encuentran diferentes fallas debido a la exposición de datos sensibles ya que estos datos deberían estar protegidos, una posible solución para la protección de los datos almacenados es utilizar módulos criptográficos autorizados como FIPS 140, otra de las viables soluciones es deshabilitar el cacheado de las páginas web que almacenen datos sensibles.

Un ejemplo claro donde se presenta este tipo de casos es el siguiente:

Una aplicación utiliza el cifrado automático para los números de tarjetas de crédito en la base de datos local, lo que nos da a entender que automáticamente se descifran cuando se rescatan los datos accediendo por este medio a una posible vulnerabilidad donde se produce un ataque de inyección SQL permitiendo al atacante obtener específicamente la información de las tarjetas de crédito; la aplicación debería cifrar los números y acceder luego por medio de procedimiento de descifrado utilizando una clave privada.

5.3.7. A7 - Ausencia de control de accesos a las funciones

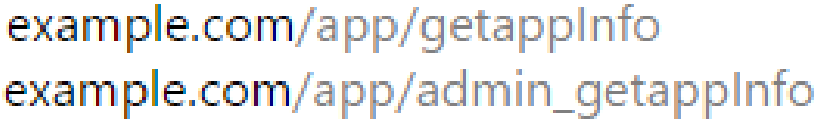
La ausencia de control, de accesos surge debido a que las aplicaciones no realizan los procedimientos de seguridad de manera eficiente, a pesar de que estas confirman los derechos de acceso a nivel de orden o función en la zona de comunicación con el usuario es necesario ratificar el control de acceso que se produce en el servidor al momento de cada función; si este procedimiento no se realiza adecuadamente el atacante establecerá control en las autorizaciones y manejara a su antojo la aplicación; en consecuencia la acción del atacante es sacar datos sensibles de las aplicaciones y enviarlas a páginas que se encuentran ocultas, ya que la información ejecutada no es validada en el procesamiento de la URL.

Los atacantes pueden utilizar maniobras para conseguir su objetivo enviando una petición como un usuario anónimo y así tener ingreso a la funcionalidad de privilegios en las aplicaciones, lo que hace el atacante es modificar el URL a una función de privilegios, como no se tiene la suficiente seguridad cualquier usuario anónimo puede acceder a funciones privadas en las aplicaciones que no se encuentran protegidas ante las amenazas de los posibles ataques.

La protección dentro de las aplicaciones es fundamental porque estas en algunos casos no protegen de la mejor manera las funcionalidades establecidas dentro de las mismas, lo que ocasiona un punto de referencia para que el atacante pueda realizar los cambios que desee, en algunos casos se llega a observar que el sistema se encuentra configurado incorrectamente y no se registran chequeos por código lo que ocasiona más vulnerabilidad; es fundamental establecer cuáles son las (URLs) que son afectadas en gran medida por los atacantes para tener un control sobre ellas, posiblemente los atacantes irán directamente a las funciones administrativas de las aplicaciones.

En la figura 13 se muestra como un atacante puede forzar los browsers para apuntar la dirección URL de la aplicación; por consiguiente, la URL necesita ser autenticada, así mismo se necesitan los derechos del administrador para ingresar a la página admin_getappInfo.

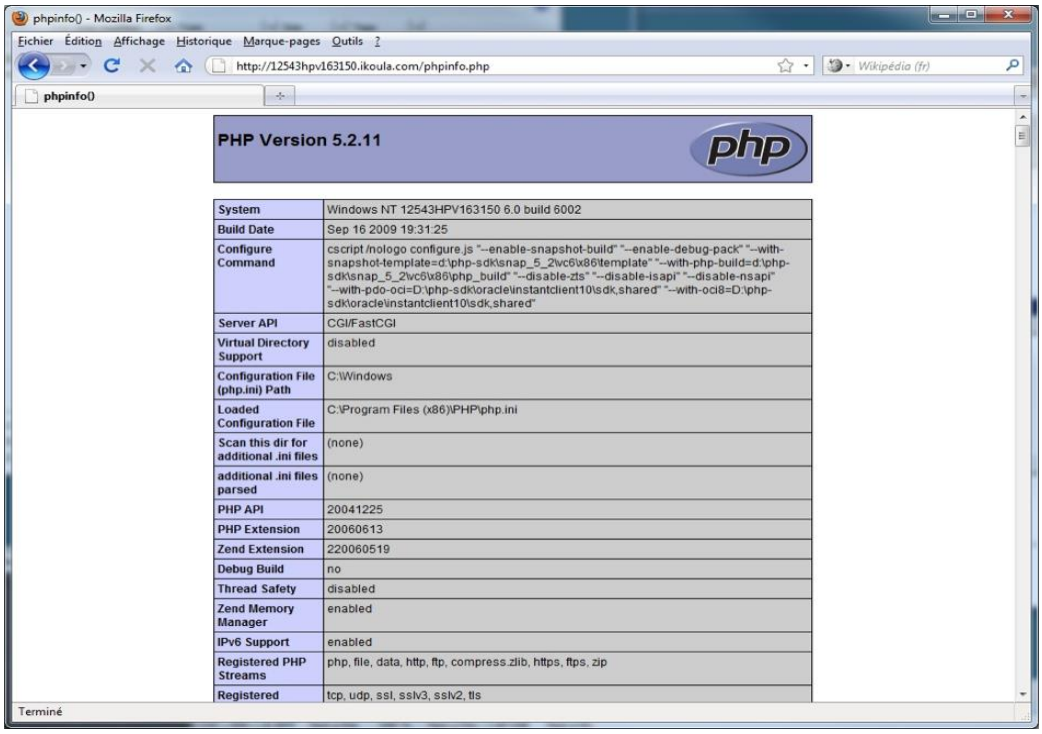
Figura 13. Ejemplo de ausencia de control



Fuente: El autor

Los usuarios pueden ingresar a cualquier página; pero si un usuario que no se encuentra registrado como administrador ingresa a la admin_getPHPInfo (figura 14) y ver toda la información sensible, este procedimiento se convierte en un defecto y da las posibilidades de realizar un ataque de administración.

Figura 14. Visualización de PHPInfo públicamente



Fuente: LibrosWeb.ces

5.3.8. A8 - Falsificación de peticiones en sitios cruzados CSRF

Los ataques CSRF exige realizar al navegador de la víctima el envío de una petición HTTP, pero esta viene falsificada entrelazando información del usuario y es enviada a una página web que se encuentra vulnerable, esta acción permite al atacante exigir al navegador del usuario victima realizar pedidos como peticiones legitimas del usuario víctima.

Este ataque se fundamenta en la familiaridad que posee la web en el usuario víctima del ataque, estas acciones hacen que la víctima realice operaciones maliciosas sin ser consciente de lo que está haciendo, obteniendo datos por medio de la modificación de estos, en ese momento el atacante accede a observar ese tipo de cambios y es donde toma control de la aplicación y por consiguiente de la transferencia de información.

Los servicios de las aplicaciones web contienen las credenciales de los usuarios registrados y autorizados a través de la IP, credenciales de dominio, entre otras; es posible que si la aplicación no tiene manera de anular CSRF le es muy difícil diferenciar cuales son las peticiones legitimas del usuario víctima y las peticiones CSRF que vendrían siendo un ataque para la aplicación.

5.3.8.1. Prevención del ataque.

Para prevenir estos posibles ataques se deben realizar métodos de autenticación agregados para todas las peticiones que se realicen así es posible identificar cuáles son las peticiones que no se encuentran autorizadas, existen varias técnicas para encontrar este tipo de ataques como Synchronizer token pattern este tipo de técnica contiene un token secreto para identificar el tipo de petición, por medio del servidor el token se genera y es comprobado satisfactoriamente, así el atacante no podrá tener acceso al valor indicado de la petición y no puede efectuar su ataque.

5.3.9. A9 – Uso de componentes con vulnerabilidades conocidas

Específicamente los atacantes encuentran un componente que se encuentre vulnerable por medio de una búsqueda manual, donde se incorpora el exploit y es ejecutada la acción de ataque, en los sistemas operativos encontramos diferentes librerías que para su funcionamiento necesitan de la mayoría de privilegios que tiene la aplicación y si en este tipo de componentes el atacante encuentra cierto tipo de vulnerabilidad accede en el servidor y por consiguiente los datos almacenados se perderían concediéndole dominio total al atacante y debilitando la aplicación.

Esta acción se produce a consecuencia de que en conjunto la mayoría de las aplicaciones no resguardan la seguridad en sus componentes y se confían aun sabiendo que hay atacantes a la vista; como ya se mencionó anteriormente las bibliotecas son las más afectadas principalmente porque el usuario no actualiza la aplicación y sigue usando versiones anteriores por donde se pueden filtrar los atacantes, el impacto que puede causar el atacante con su inyección va desde lo más mínimo a lo más alto, sigilosamente el atacante va ingresando hasta

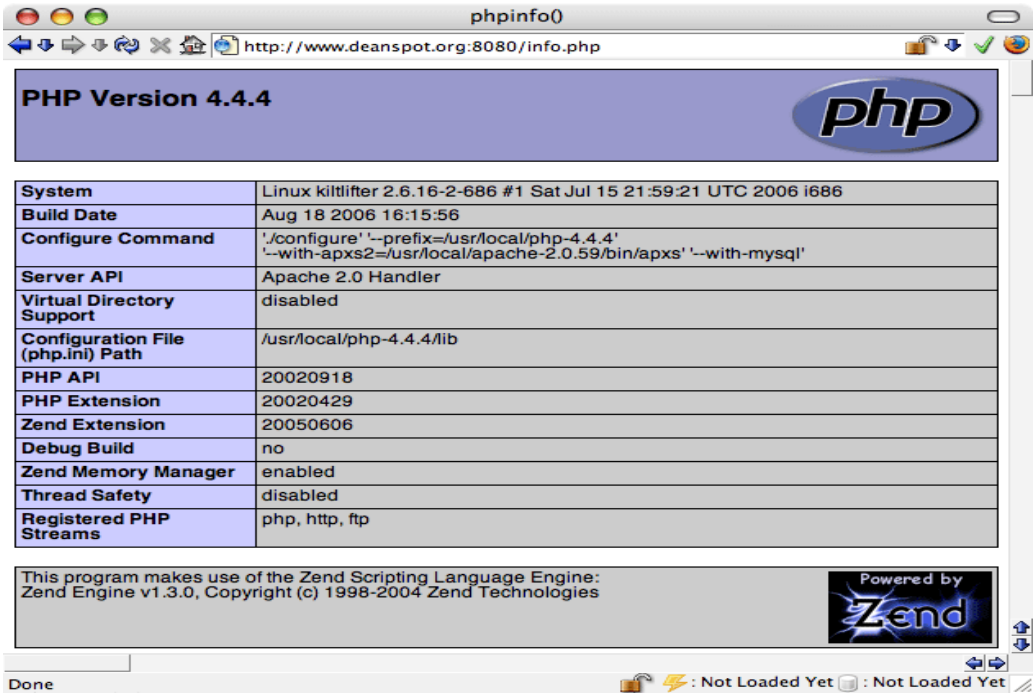
apoderarse de toda la información que quiera sacar, una posible solución para contrarrestar este tipo de daños es no utilizar aquellos componentes que no se encuentren codificados ya que si el componente no tiene o crea un parche por ahí se puede filtrar el atacante, el usuario debe estar revisando periódicamente la seguridad en las bases de datos, igualmente en las listas de correos y actualizarlos.

En el siguiente ejemplo se evidencia esta falencia que tienen las aplicaciones

La función phpinfo() de PHP tal como lo muestra la figura 15, da a conocer la información sobre el lenguaje y aquellos módulos que están habilitados en el servidor.

ntitle:phpinfo() + PHP Versión 4.4.1.

Figura 15. Phpinfo 4



Fuente: php.net

5.3.10. A10 - Redirecciones y reenvíos no validos

Es común que las aplicaciones web por medio de links envían a los usuarios a otras páginas web con su información, pero este procedimiento acarrea datos que no son netamente confiables para poder acceder al destino de la página web, ya que si la validación no es adecuada el procedimiento que utilizan los atacantes es enviar a los usuarios a otros sitios como malware y por consiguiente realizan reenvíos ingresando a páginas que no son confiables.

Los atacantes son muy hábiles en cuanto a las técnicas que utilizan ya que pueden tergiversar la información haciéndole creer a los usuarios que realicen varias peticiones a su aplicación web siendo estas de carácter malicioso. Hay que tener en cuenta que los códigos HTML al que el usuario accede pensando que es confiable podría ser atacado, en ese momento el código malicioso roba los datos del usuario y sus contraseñas, esta acción de ataque permite el acceso total al equipo de la víctima.

Una posible solución a estos ataques es que los valores de los parámetros de destino sean valores de mapeo, también se recomienda utilizar ESAPI para rectificar el método sendRedirect() posteriormente el usuario puede asegurarse que los destinos son confiables.

La figura 16 es un ejemplo de dicho ataque, una aplicación posee una página web llamada “redirect.jsp” y esta recibe un parámetro denominado “URL”, la acción del atacante es volver esa URL maliciosa y lo que hace es llevar a los usuarios a otra aplicación donde phishing que instala un código malicioso en el equipo de la víctima.

Figura 16. Ejemplo de redirecciones no validas

Fuente: El autor

5.4. MARCO LEGAL

5.4.1. LEY 527 DE 1999

Por medio de la cual se define y reglamenta el acceso y uso de los mensajes de datos, del comercio electrónico y de las firmas digitales, se establecen las entidades de certificación y se dictan otras disposiciones.[3]

5.4.2. LEY 1273 DE 2009

Por medio de la cual se modifica el Código Penal, se crea un nuevo bien jurídico tutelado - denominado "de la protección de la información y de los datos"- y se preservan integralmente los sistemas que utilicen las tecnologías de la información y las comunicaciones, entre otras disposiciones.[4]

Creada en el congreso de Republica el 5 de enero de 2009. Se crea un bien jurídico “Protección de la información y de los datos”. Se clarifico que es considerado como delitos cuando se trata de manejo de datos personales.

En la actualidad la delincuencia busca obtener información ilícitamente información personal como datos bancarios, infiltrarse a sistemas de cómputo para manipular programas o sustraer información crítica de la organización o datos personales para extorsionar a las personas; las empresas son ampliamente afectadas por estas prácticas.

Las penas van desde 4 años de cárcel, como mínimo, o una compensación de 100 salarios mínimos legales vigentes. *Esta ley está dividida en dos partes:*

Primera parte:

---

[3] Descripción de la ley 527 de 1999. (agosto 18) Disponible en http://www.alcaldiabogota.gov.co/sisjur/normas/Norma1.jsp?i=4276

[4] Descripción de la ley 1273 de 2009. (enero 05) Disponible en Descripción http://www.alcaldiabogota.gov.co/sisjur/normas/Norma1.jsp?i=34492

Artículo 269A: Acceso abusivo a un sistema informático. El que, sin autorización o por fuera de lo acordado, acceda en todo o en parte a un sistema informático protegido o no con una medida de seguridad, o se mantenga dentro del mismo en contra de la voluntad de quien tenga el legítimo derecho a excluirlo, incurrirá en pena de prisión de cuarenta y ocho (48) a noventa y seis (96) meses y en multa de 100 a 1.000 salarios mínimos legales mensuales vigentes.

Artículo 269B: Obstaculización ilegítima de sistema informático o red de telecomunicación. El que, sin estar facultado para ello, impida u obstaculice el funcionamiento o el acceso normal a un sistema informático, a los datos informáticos allí contenidos, o a una red de telecomunicaciones, incurrirá en pena de prisión de cuarenta y ocho (48) a noventa y seis (96) meses y en multa de 100 a 1000 salarios mínimos legales mensuales vigentes, siempre que la conducta no constituya delito sancionado con una pena mayor.

Artículo 269C: Interceptación de datos informáticos. El que, sin orden judicial previa intercepte datos informáticos en su origen, destino o en el interior de un sistema informático, o las emisiones electromagnéticas provenientes de un sistema informático que los transporte incurrirá en pena de prisión de treinta y seis (36) a setenta y dos (72) meses.

Artículo 269D: Daño Informático. El que, sin estar facultado para ello, destruya, dañe, borre, deteriore, altere o suprima datos informáticos, o un sistema de tratamiento de información o sus partes o componentes lógicos, incurrirá en pena de prisión de cuarenta y ocho (48) a noventa y seis (96) meses y en multa de 100 a 1.000 salarios mínimos legales mensuales vigentes.

Artículo 269E: Uso de software malicioso. El que, sin estar facultado para ello, produzca, trafique, adquiera, distribuya, venda, envíe, introduzca o extraiga del territorio nacional software malicioso u otros programas de computación de efectos dañinos, incurrirá en pena de prisión de cuarenta y ocho (48) a noventa y seis (96) meses y en multa de 100 a 1.000 salarios mínimos legales mensuales vigentes.

Artículo 269F: Violación de datos personales. El que, sin estar facultado para ello, con provecho propio o de un tercero, obtenga, compile, sustraiga,

ofrezca, venda, intercambie, envíe, compre, intercepte, divulgue, modifique o emplee códigos personales, datos personales contenidos en ficheros, archivos, bases de datos o medios semejantes, incurrirá en pena de prisión de cuarenta y ocho (48) a noventa y seis (96) meses y en multa de 100 a 1000 salarios mínimos legales mensuales vigentes.

Artículo 269G: Suplantación de sitios web para capturar datos personales. El que con objeto ilícito y sin estar facultado para ello, diseñe, desarrolle, trafique, venda, ejecute, programe o envíe páginas electrónicas, enlaces o ventanas emergentes, incurrirá en pena de prisión de cuarenta y ocho (48) a noventa y seis (96) meses y en multa de 100 a 1.000 salarios mínimos legales mensuales vigentes, siempre que la conducta no constituya delito sancionado con pena más grave.

Artículo 269H: Circunstancias de agravación punitiva: Las penas imponibles de acuerdo con los artículos descritos en este título, se aumentarán de la mitad a las tres cuartas partes si la conducta se cometiere:

1. Sobre redes o sistemas informáticos o de comunicaciones estatales u oficiales o del sector financiero, nacionales o extranjeros.

2. Por servidor público en ejercicio de sus funciones.

3. Aprovechando la confianza depositada por el poseedor de la información o por quien tuviere un vínculo contractual con este.

4. Revelando o dando a conocer el contenido de la información en perjuicio de otro.

5. Obteniendo provecho para sí o para un tercero.

6. Con fines terroristas o generando riesgo para la seguridad o defensa nacional.

7. Utilizando como instrumento a un tercero de buena fe.

8. Si quien incurre en estas conductas es el responsable de la administración, manejo o control de dicha información, además se le impondrá hasta por tres años, la pena de inhabilitación para el ejercicio de profesión relacionada con sistemas de información procesada con equipos computacionales.[5]

Segunda parte:

Artículo 269I: Hurto por medios informáticos y semejantes. El que, superando medidas de seguridad informáticas, realice la conducta señalada en el artículo 239 manipulando un sistema informático, una red de sistema electrónico, telemático u otro medio semejante, o suplantando a un usuario ante los sistemas de autenticación y de autorización establecidos, incurrirá en las penas señaladas en el artículo 240 de este Código.

Artículo 269J: Transferencia no consentida de activos. El que, con ánimo de lucro y valiéndose de alguna manipulación informática o artificio semejante, consiga la transferencia no consentida de cualquier activo en perjuicio de un tercero, siempre que la conducta no constituya delito sancionado con pena más grave, incurrirá en pena de prisión de cuarenta y ocho (48) a ciento veinte (120) meses y en multa de 200 a 1.500 salarios mínimos legales mensuales vigentes. La misma sanción se le impondrá a quien fabrique, introduzca, posea o facilite programa de computador destinado a la comisión del delito descrito en el inciso anterior, o de una estafa.

Si la conducta descrita en los dos incisos anteriores tuviere una cuantía superior a 200 salarios mínimos legales mensuales, la sanción allí señalada se incrementará en la mitad.[6]

La importancia de esta ley es que además les da un marco jurídico e importancia a las nuevas tecnologías agregándolas en la legislación y evitando así la

---

[5] Descripción de la ley 1273 de 2009. (enero 05) Disponible en Descripción http://www.alcaldiabogota.gov.co/sisjur/normas/Norma1.jsp?i=34492

[6] Ibídem., capitulo III

impunidad en delitos tan frecuentes en nuestro país tales como accesos no autorizados a correos electrónicos, espionaje de chats, redes sociales, interceptación de comunicaciones, instalación de software malicioso o de keylogger y en especial el hurto o transferencias indebidas de dinero a través de medios informáticos.

5.4.3. LEY ESTATUTARIA 1266 DEL 31 DE DICIEMBRE DE 2008

"*Por la cual se dictan las disposiciones generales del Hábeas Data y se regula el manejo de la información contenida en bases de datos personales, en especial la financiera, crediticia, comercial, de servicios y la proveniente de terceros países y se dictan otras disposiciones*."[7]

5.4.4. LEY ESTATUTARIA 1581 DE 2012

"Entró en vigor la Ley 1581 del 17 de octubre 2012 de PROTECCIÓN DE DATOS PERSONALES, sancionada siguiendo los lineamientos establecidos por el Congreso de la República y la Sentencia C-748 de 2011 de la Corte Constitucional."

[7] Marco legal de Seguridad de la Información en Colombia, (enero 23 2010). Disponible en http://seguridadinformacioncolombia.blogspot.com.co/2010/02/marco-legal-de-seguridad-de-la.html

6. METODOLOGÍA

6.1. METODOLOGÍA DE LA INVESTIGACIÓN

El enfoque de la investigación es cuantitativo ya que se pretende medir los niveles de seguridad del sistema e-commerce SiembraViva.com de acuerdo con las pruebas de pentesting planteadas por la metodología del proyecto OWASP.

El tipo de investigación es exploratoria, descriptiva y explicativa de acuerdo con la metodología planteada en el desarrollo del proyecto.

- Exploratoria puesto que se va a analizar la problemática que está afectando la empresa, basado en este análisis se van a buscar métodos o técnicas para la resolución del problema detectado. En este caso concreto de investigación es detectar que vulnerabilidades está afectando al sistema y que técnica aplicar para resolver estas vulnerabilidades.
- Descriptiva debido a que se busca documentar las vulnerabilidades y las pruebas paso a paso para analizar los resultados de estas.
- Explicativa porque se unen los métodos analíticos y sintéticos para explicar la razón de la investigación. Se aborda cada vulnerabilidad y se describe cómo funcionan.

6.2. METODOLOGÍA DE DESARROLLO A SEGUIR

Paso 1: Revisar la metodología que se eligió para analizar el sitio web y determinar cuáles son los pasos por seguir.

- Consultar fuentes confiables sobre la metodología que brinden documentación sobre el tema.
- Consolidar las ideas con la información encontrada y determinar los pasos a seguir.

Paso 2: Someter al análisis el sistema e-commerce SiembraViva.com basados en la metodología seleccionada.

- Encontrar las vulnerabilidades que tiene el sitio web y agrupar según la gravedad que se tiene.
- Seguir las fases de la metodología OWASP para asegurar la efectividad del proceso.

Paso 3: Documentar las vulnerabilidades encontradas en el proceso tal como se detalla en la figura 17.

- Ordenar según el tipo de vulnerabilidad para facilitar el proceso de documentación.
- Consultar el CVE sobre la vulnerabilidad y nivel de riesgo en el sistema.

Figura 17. Metodología en el proceso de análisis

Fuente: OWASP DB IMG

7. EJECUCIÓN DE LA METODOLOGÍA DE PRUEBAS OWASP

7.1. PRUEBA DE INTRUSIÓN SIEMBRAVIVA.COM

A continuación, se describe la metodología OWASP para la realización de las pruebas de intrusión en aplicaciones web, y se explica cómo realizar la comprobación de cada una de las vulnerabilidades.

En la metodología OWASP se tienen 2 fases para la implementación:

- Fase 1: Modo pasivo, en esta fase el pentester ingresa a la aplicación Web y analiza cómo es su funcionamiento. La manera más adecuada es ir explorando cada una de las funcionalidades de la aplicación, complementario a esto se identifican los formularios y las paginas donde se pueda acceder de un punto A dentro de la aplicación a un punto B.

- Fase 2: Modo activo, la aplicación es sometida a las pruebas que son sugeridas por la metodología.

Durante la fase 2, se han dividido el conjunto de pruebas en 9 subcategorías de 66 procesos distribuidas de la siguiente forma:

Tabla 3. Conjunto de pruebas en 9 subcategoría

Categoría	Numero de Referencia	Nombre de la Prueba	Vulnerabilidad
Recopilación de información	OWASP-IG-001	Spiders, Robots y Crawlers	N.A.
	OWASP-IG-002	Descubrimiento/Reconocimiento mediante motores de búsqueda	N.A.
	OWASP-IG-003	Identificación de puntos de entrada de la aplicación	N.A.
	OWASP-IG-004	Pruebas de firma digital de aplicaciones web	N.A.

Categoría	Numero de Referencia	Nombre de la Prueba	Vulnerabilidad
	OWASP-IG-005	Descubrimiento de Aplicaciones	N.A.
	OWASP-IG-006	Análisis de Códigos de Errores	Exposición de información
Pruebas de Gestión de configuración	OWASP-CM-001	Pruebas SSL/TLS (SSL Versión, Algoritmos, longitud de Claves, Validez de Certificado Digital)	Débil implementación de SSL
	OWASP-CM-002	Prueba de DB Listener	Debilidad de DB Listener
	OWASP-CM-003	Prueba de Gestión de Configuración de Infraestructura	Debilidad de gestión de configuración de infraestructura
	OWASP-CM-004	Prueba de Gestión de Configuración de Aplicación	Debilidad en gestión de configuración de aplicación
	OWASP-CM-005	Prueba del Gestor de Extensión de Ficheros	Gestor de extensión de ficheros
	OWASP-CM-006	Antiguo, backup y ficheros no referenciados	Antiguo, backup y ficheros no referenciados
	OWASP-CM-007	Interfase de Administración de Aplicación e Infraestructura	Acceso a Interfaces de Administración
	OWASP-CM-008	Prueba de métodos HTTP y XST	Métodos HTTP habilidades, XST permitidos, HTTP Verbos
Pruebas de autenticación	OWASP-AT-001	Transporte de Credenciales sobre canal cifrado	Transporte de Credenciales sobre canal cifrado

Categoría	Numero de Referencia	Nombre de la Prueba	Vulnerabilidad
	OWASP-AT-002	Prueba para Enumeración de usuarios	Enumeración de usuarios
	OWASP-AT-003	Prueba de detección de Cuentas de Usuario Adivinables (Diccionario)	Cuentas de usuario adivinables
	OWASP-AT-004	Prueba de Fuerza Bruta	Credenciales débiles
	OWASP-AT-005	Prueba para evitar los esquemas de autenticación	Evitar esquema de autenticación
	OWASP-AT-006	Prueba de recordatorio de contraseña y restablecimiento	Vulnerabilidad de Recordatorio de contraseña y debilidad de restablecimiento de contraseña
	OWASP-AT-007	Prueba de Cierre de Sesión y Gestión de Cache de Navegación	Función de Cierre de Sesión no implementada correctamente, debilidad en la cache de navegación
	OWASP-AT-008	Prueba de CAPTCHA	Debilidad en la implementación de Captcha
	OWASP-AT-009	Prueba de Autenticación de Múltiple Factores	Debilidad de Autenticación de Múltiple Factores
	OWASP-AT-010	Prueba de Condiciones de Carrera	Vulnerabilidad de Condiciones de Carrera
Gestión de sesiones	OWASP-SM-001	Prueba del Esquema de Gestión de Sesión	Bypassing Session Management Schema, Weak Session Testigo
	OWASP-SM-002	Prueba de atributos de Cookies	Cookies son definidos como no ‘HTTP Only’, ‘Secure’, y sin tiempo de validez
	OWASP-SM-	Prueba de Fijación de Sesión	Fijación de Sesión

Categoría	Numero de Referencia 003	Nombre de la Prueba	Vulnerabilidad
	OWASP-SM-004	Prueba de Variables de Sesión Expuestas	Variables sensibles de Sesión expuestas
	OWASP-SM-005	Prueba de CSRF	CSRF
Pruebas de automatización	OWASP-AZ-001	Prueba de Ruta Transversal	Ruta Transversal
	OWASP-AZ-002	Prueba para Evitar Esquema de Autorización	Evitar esquema de autorización
	OWASP-AZ-003	Prueba de escalada de privilegios	Escalado de Privilegios
Pruebas de Lógica de negocio	OWASP-BL-001	Prueba de Lógica de Negocio	Evitar lógica de negocio
Pruebas de validación de datos	OWASP-DV-001	Prueba de XSS Reflejado	Reflejado XSS
	OWASP-DV-002	Prueba de XSS Almacenado	Almacenado XSS
	OWASP-DV-003	Prueba de XSS basado en DOM	DOM XSS
	OWASP-DV-004	Prueba de XSS basado en Flash	XSS basado en Flash
	OWASP-DV-005	Inyección SQL	Inyección SQL
	OWASP-DV-006	Inyección LDAP	Inyección LDAP
	OWASP-DV-	Inyección ORM	Inyección ORM

Categoría	Numero de Referencia 007	Nombre de la Prueba	Vulnerabilidad
	OWASP-DV-008	Inyección XML	Inyección XML
	OWASP-DV-009	Inyección SSI	Inyección SSI
	OWASP-DV-010	Inyección XPath	Inyección XPath
	OWASP-DV-011	Inyección IMAP/SMTP	Inyección IMAP/SMTP
	OWASP-DV-012	Inyección de Código	Inyección de Código
	OWASP-DV-013	Inyección de Ordenes del Sistema Operativo	Inyección de Ordenes del OS
	OWASP-DV-014	Desbordamiento de *buffer*	Desbordamiento de buffer
	OWASP-DV-015	Prueba de Vulnerabilidad incubada	Vulnerabilidad Incubada
	OWASP-DV-016	Prueba de HTTP Splitting/Smuggling	HTTP *Splitting, Smuggling*
Pruebas de Denegación de Servicio	OWASP-DS-001	Prueba de Ataques a través de Comodines SQL	Vulnerabilidad de los Comodines SQL
	OWASP-DS-002	Bloqueo de Cuentas de Usuarios	Bloqueo de Cuentas de Usuarios
	OWASP-DS-003	Pruebas de DoS mediante Desbordamiento de Buffer	Desbordamientos de Buffer
	OWASP-DS-	Asignación de Objeto de Usuario	Asignación de

Categoría	Numero de Referencia	Nombre de la Prueba	Vulnerabilidad
	004	Especificado	Objeto de Usuario Especificado
	OWASP-DS-005	Entrada de usuario como un contador de bucle	Entrada de usuario como un contador de bucle
	OWASP-DS-006	Prueba de Escritura en Disco de data provista por Usuario	Escritura en Disco de Data provista por Usuario
	OWASP-DS-007	Fallo en Liberar Recursos	Fallo en Liberar recursos
	OWASP-DS-008	Almacenamiento de demasiados datos en Sesión	Almacenamiento de demasiados datos en Sesión
Pruebas de servicios web	OWASP-WS-001	Recopilación de Información de WS	N.A.
	OWASP-WS-002	Prueba de WSDL	Debilidad de WSDL
	OWASP-WS-003	Prueba en la Estructura del XML	Debilidad en la estructura del XML
	OWASP-WS-004	Prueba del XML a nivel de contenido	XML a nivel de contenido
	OWASP-WS-005	Prueba de REST/parámetros HTTP GET	Parámetros WS HTTP GET/REST
	OWASP-WS-006	Adjuntos SOAP maliciosos	WS SOAP adjuntos maliciosos
	OWASP-WS-007	Prueba de Repetición	Prueba de Repetición de WS
Prueba de Ajax	OWASP-AJ-001	Vulnerabilidades Ajax	N. A
	OWASP-AJ-002	Pruebas Ajax	Debilidad Ajax

Categoría	Numero de Referencia	Nombre de la Prueba	Vulnerabilidad
Fuente: Guía de pruebas de OWASP versión 3.0, 2008. https://www.owasp.org/images/8/80/Gu%C3%ADa_de_pruebas_de_OWASP_ver_3.0.pdf			

7.2. RECOPILACIÓN DE INFORMACIÓN

Es la primera fase en la evaluación de seguridad de una aplicación web, se centra en recoger toda la información como sea posible sobre la aplicación web objetivo y conocer la manera cómo opera. La recopilación de información es un paso fundamental y necesario en una prueba de intrusión. Esta prueba de reconocimiento pasivo de información se realiza sin tocar los servidores o que estos generen logs de seguridad. Esta tarea se puede llevar a cabo de muchas formas.

Utilizando herramientas de acceso público motores de búsqueda, scanners, enviando peticiones HTTP simples, o peticiones especialmente diseñadas, es posible forzar a la aplicación a filtrar información al exterior con mensajes de error devueltos, o revelar las versiones y tecnología en uso por la aplicación web.

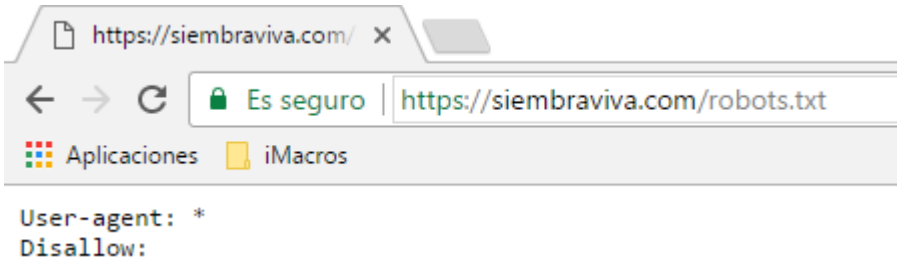
7.2.1. Spiders, Robots, y Crawlers (OWASP-IG-001).

Esta fase del proceso de recopilación de información consiste en navegar y capturar los recursos relacionados con la aplicación web, como la configuración robots.txt, la cual es de uso frecuente por los motores de búsqueda para categorizar los enlaces internos.

7.2.1.1. Robots.txt.

Los crawlers/robots/spiders web se encargan de inspeccionar los sitios web, y sucesivamente acceden para obtener su contenido. Su comportamiento está estrictamente controlado por el protocolo de exclusión de robots del fichero robots.txt almacenado en la raíz del sitio web, ver figura 18.

Figura 18. Configuración archivo robots.txt



Fuente: El autor, SiembreViva.com/robots.txt

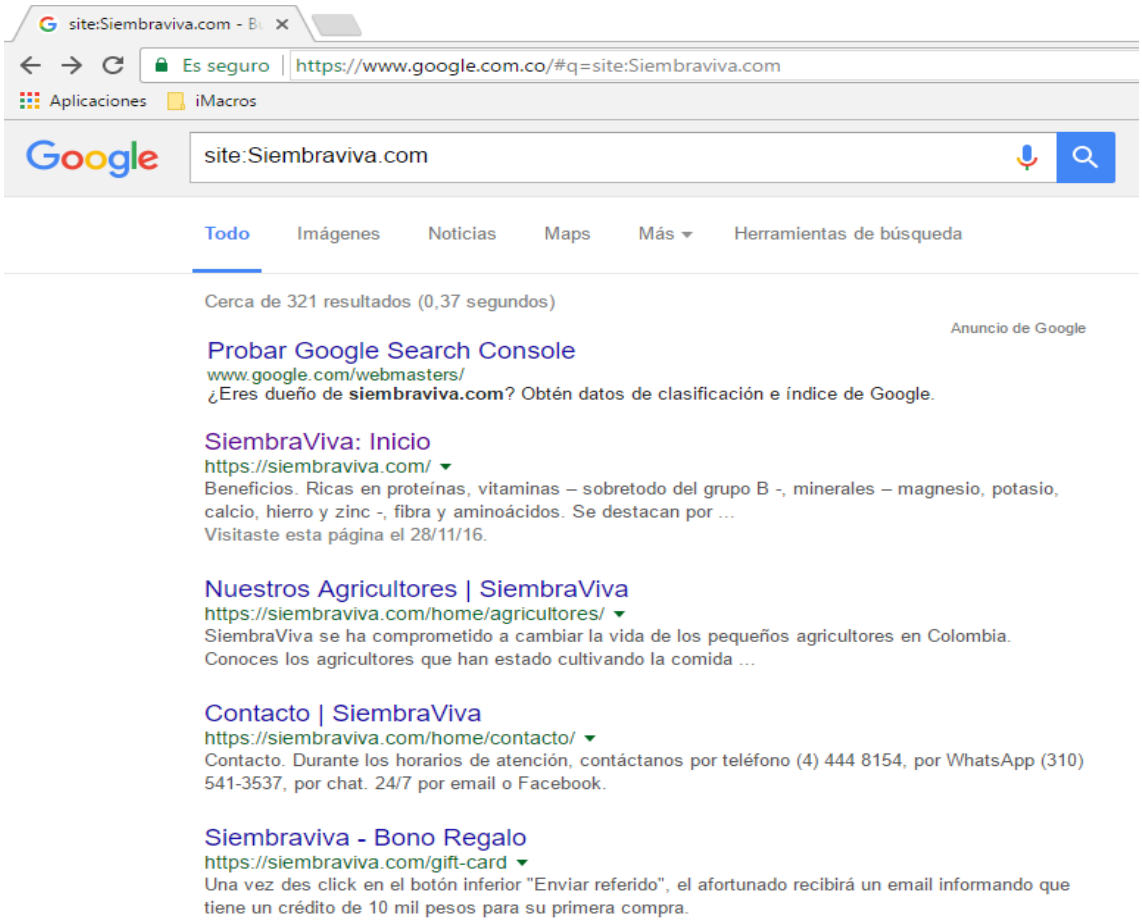
La directiva "User-agent: *" hace referencia de aplicar a todos los crawlers/robots/spiders las reglas de seguimiento. La directiva "Disallow" especifica al robot que no debe visitar cualquier página del sitio web.

Actualmente la directiva no prohíbe ningún acceso a directorios, por consiguiente, todos los crawlers/robots/spiders pueden rastrear el sitio web por completo.

7.2.2. Reconocimiento mediante motores de búsqueda (OWASP-IG-002).
Una vez los robots han completado el proceso de crawling, inicia la indexación del contenido de las páginas web basándose en el contenido y los atributos más relevantes para la búsqueda.

7.2.2.1. Site:SiembreViva.com.
Devuelve todas las paginas indexadas en Google.com.co y Bing.com tal como se muestra en la figura 19 y 20 respectivamente.

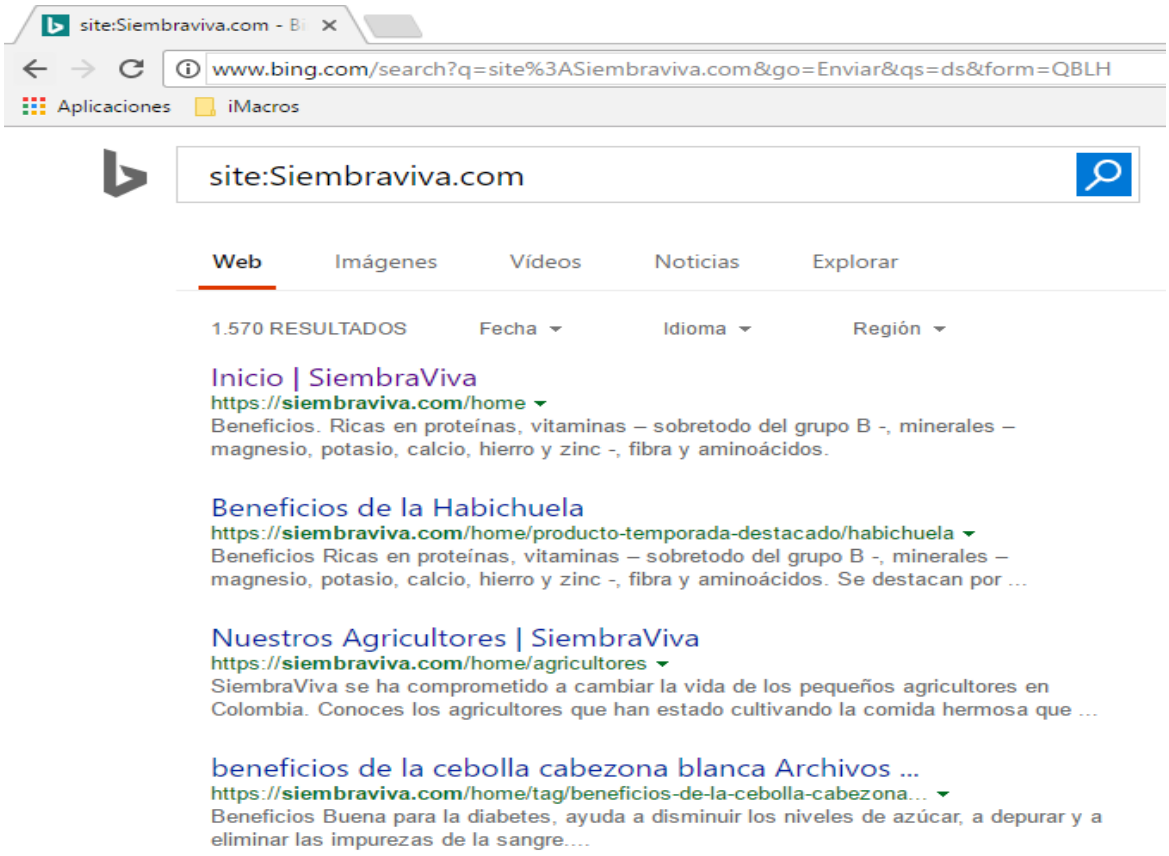
Figura 19. Site:SiembreViva.com Google.com.co



Fuente: El autor, Google.com.co/#q=site:siembraviva.com

En la figura anterior se puede evidenciar que hay cerca de 321 páginas indexadas por el motor de búsqueda Google.

Figura 20. Site: SiembreViva.com Bing.com



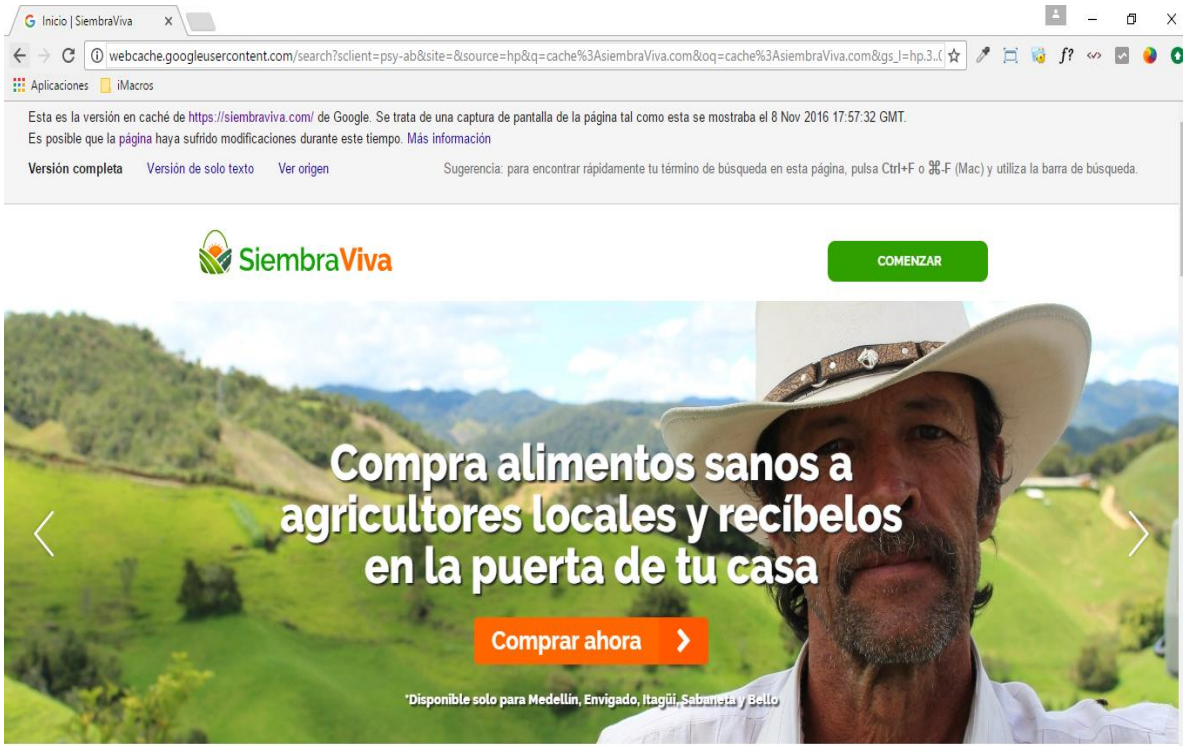
Fuente: El autor

En la figura anterior se puede evidenciar que hay cerca de 1570 resultados sobre el sitio web SiembraViva.com en el motor de búsquedas web operado por Microsoft, Bing.com.

7.2.2.2. Cache: SiembraViva.com.

En la figura 21 se muestra la última captura de pantalla del sitio web por parte de los motores de búsqueda.

Figura 21. Cache: SiembraViva.com Google.com.co



Fuente: El autor

La cache del sitio web corresponde al 8 noviembre de 2016 17:57:32 GMT, último día en que Google rastreo el sitio web www.SiembreViva.com.

7.2.2.3. Archive.org

El sitio web Archive.org nos permite acceder a una biblioteca digital con las capturas de pantalla de los sitios web públicos, recursos multimedia, metadatos, etc.

En la figura 22 se muestran los resultados del 20 de febrero 2014 a octubre 30 del 2016.

Figura 22. Archive.org SiembraViva.com

Fuente: El autor, https://web.archive.org/web/*/siembraviva.com

Para el análisis del sitio web SiembraViva.com, tenemos que este empezó cerca del mes de febrero del año 2014 con una redirección al dominio QuieroSiembraViva.com, Archive.org ha capturado cerca de 23 modificaciones en la línea de tiempo desde el 20 de febrero de 2014 hasta el 30 octubre del 2016.

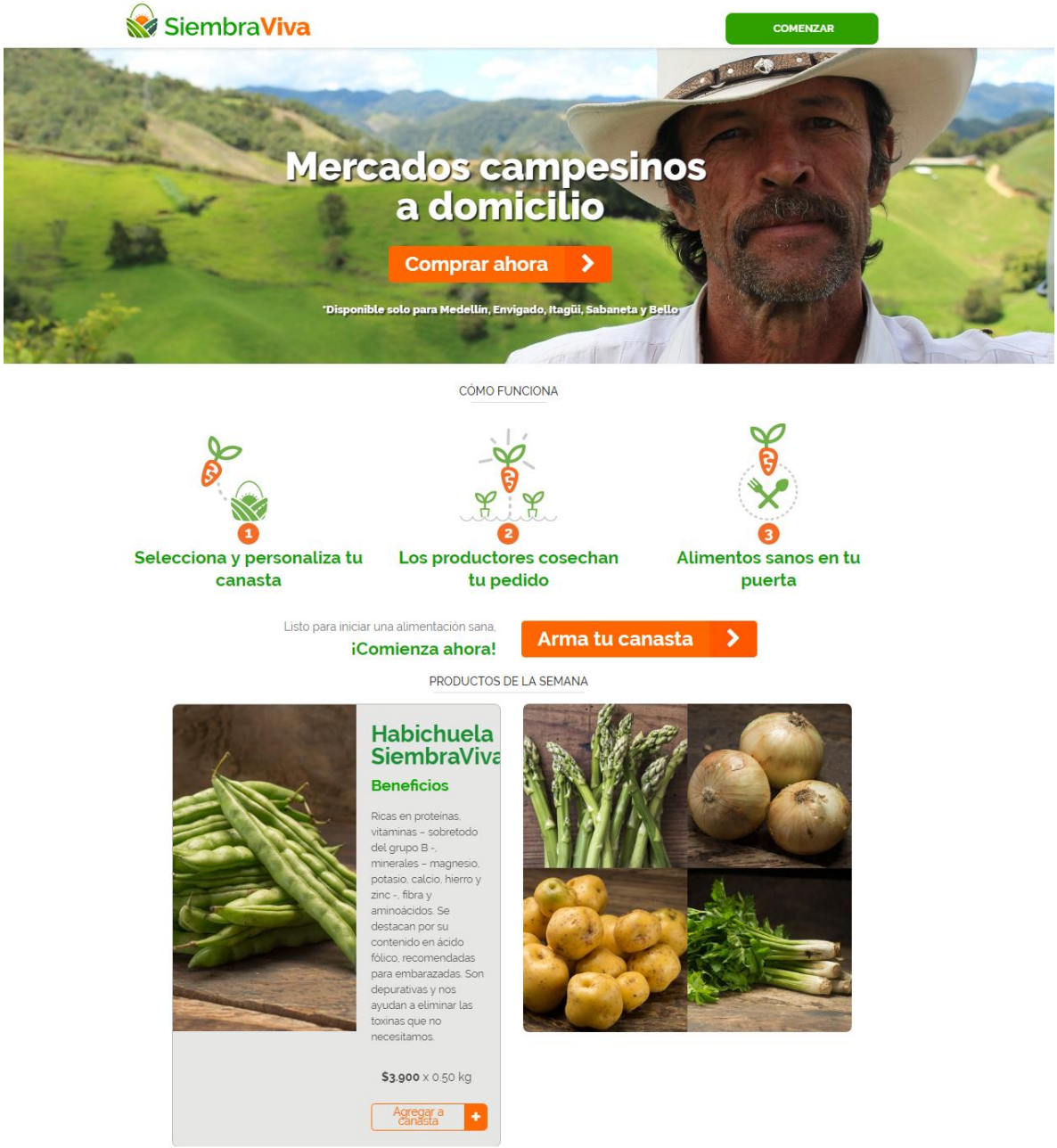
En la figura 23 se muestra como era el diseño del sitio web de SiembraViva.com para el 20 de febrero 2014.

Figura 23.SiembraViva.com 20 de febrero 2014

Fuente: El autor,

https://web.archive.org/web/20140220214517/http://www.quierosiembraviva.com/

Figura 24. SiembraViva.com 30 de noviembre 2016



Fuente: El autor,

https://web.archive.org/web/20160302075523/http://siembraviva.com/home/

7.2.2.4. Whois

Whois es el protocolo vía TCP basado en petición/respuesta que se utiliza para realizar consultas a las bases de datos de las entidades registradoras de nombres de dominios a nivel mundial. Estas consultas WHOIS se realizan para identificar los datos públicos del dominio en cuestión como, registrador, fecha de registro, fecha de vencimiento, contacto, dirección, etc.

Tabla 4. Registro Whois SiembreViva.com

Siembraviva.com Registro a whois
Domain Name: SIEMBRAVIVA.COM Registrar: GODADDY.COM, LLC Sponsoring Registrar IANA ID: 146 Whois Server: whois.godaddy.com Referral URL: http://www.godaddy.com Name Server: NS29.DOMAINCONTROL.COM Name Server: NS30.DOMAINCONTROL.COM Status: clientDeleteProhibited https://icann.org/epp#clientDeleteProhibited Status: clientRenewProhibited https://icann.org/epp#clientRenewProhibited Status: clientTransferProhibited https://icann.org/epp#clientTransferProhibited Status: clientUpdateProhibited https://icann.org/epp#clientUpdateProhibited Updated Date: 05-feb-2016 Creation Date: 04-feb-2013 Expiration Date: 04-feb-2017
Fuente: Whois, http://www.whois.com/whois/siembraviva.com

Tabla 5. Registrador Whois SiembraViva.com

Siembraviva.com Registrador a whois
Domain Name: SIEMBRAVIVA.COM Registry Domain ID: 1778342970_DOMAIN_COM-VRSN Registrar WHOIS Server: whois.godaddy.com Registrar URL: http://www.godaddy.com Update Date: 2016-02-05T08:37:06Z Creation Date: 2013-02-04T18:55:47Z Registrar Registration Expiration Date: 2017-02-04T18:55:47Z Registrar: GoDaddy.com, LLC Registrar IANA ID: 146 Registrar Abuse Contact Email: abuse@godaddy.com

Siembraviva.com Registrador a whois
Registrar Abuse Contact Phone: +1.4806242505 Domain Status: clientTransferProhibited http://www.icann.org/epp#clientTransferProhibited Domain Status: clientUpdateProhibited http://www.icann.org/epp#clientUpdateProhibited Domain Status: clientRenewProhibited http://www.icann.org/epp#clientRenewProhibited Domain Status: clientDeleteProhibited http://www.icann.org/epp#clientDeleteProhibited Registry Registrant ID: Not Available From Registry Registrant Name: Diego Benitez Registrant Organization: Registrant Street: Calle 64 no 50 67 Registrant City: Medellin Registrant State/Province: Antioquia Registrant Postal Code: 00001 Registrant Country: CO Registrant Phone: +57-310-405-7330 Registrant Phone Ext: Registrant Fax: Registrant Fax Ext: Registrant Email: dbenitezv@hotmail.com Registry Admin ID: Not Available From Registry Admin Name: Diego Benitez Admin Organization: Admin Street: Calle 64 no 50 67 Admin City: Medellin Admin State/Province: Antioquia Admin Postal Code: 00001 Admin Country: CO Admin Phone: +57-310-405-7330 Admin Phone Ext: Admin Fax: Admin Fax Ext: Admin Email: dbenitezv@hotmail.com Registry Tech ID: Not Available From Registry Tech Name: Diego Benitez Tech Organization: Tech Street: Calle 64 no 50 67 Tech City: Medellin Tech State/Province: Antioquia

Siembraviva.com Registrador a whois
Tech Postal Code: 00001 Tech Country: CO Tech Phone: +57-310-405-7330 Tech Phone Ext: Tech Fax: Tech Fax Ext: Tech Email: email@hotmail.com Name Server: NS29.DOMAINCONTROL.COM Name Server: NS30.DOMAINCONTROL.COM DNSSEC: unsigned URL of the ICANN WHOIS Data Problem Reporting System: http://wdprs.internic.net/ >>> Last update of WHOIS database: 2016-11-30T05:00:00Z <<<
Fuente: Whois, http://www.whois.com/whois/siembraviva.com

Tabla 6. Siembraviva.com Registro whois Sitio web

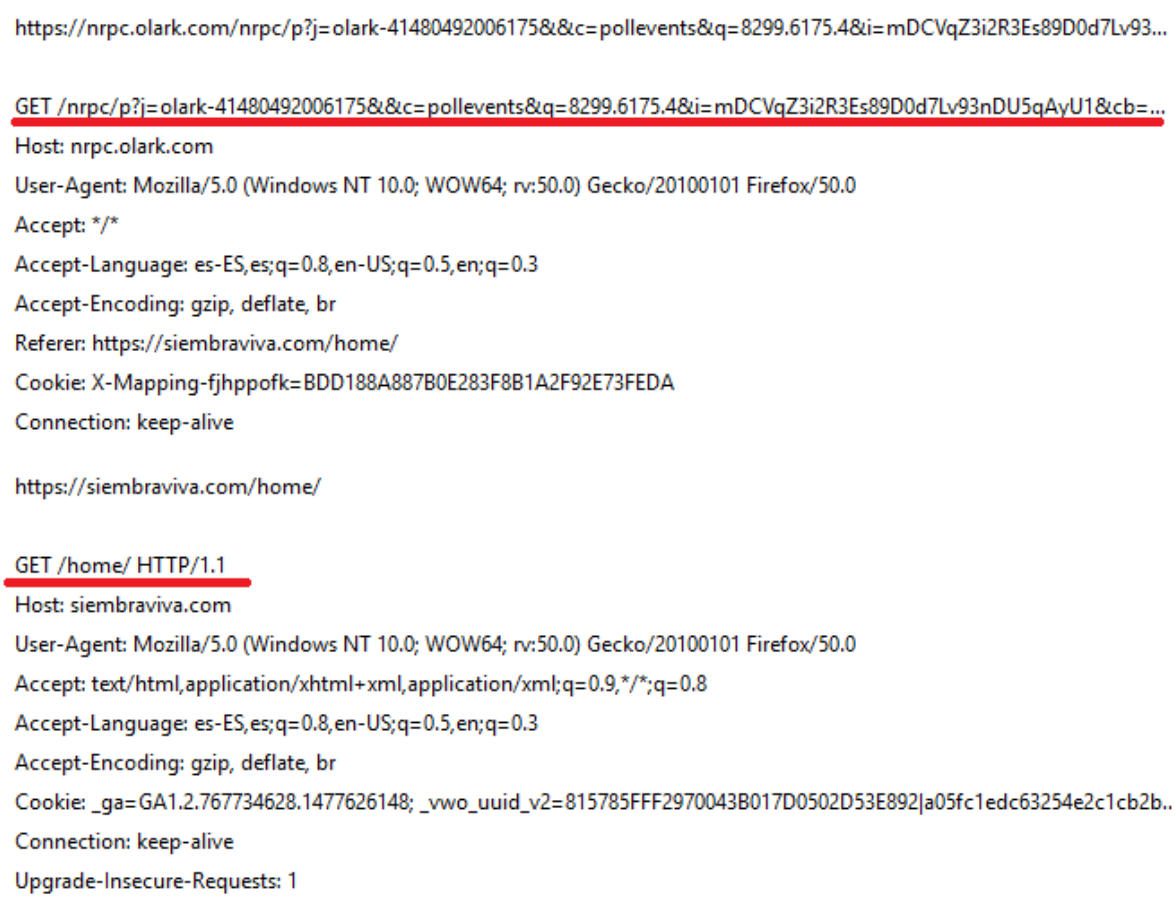
Siembraviva.com Registro whois Sitio web
Website Title: Inicio | SiembraViva Server Type: Apache/2.4.6 (CentOS) OpenSSL/1.0.1e-fips PHP/5.6.18 Response Code: 200 SEO Score: 69% Terms: 613 (Unique: 309, Linked: 123) Images: 22 (Alt tags missing: 6) Links: 62 (Internal: 50, Outbound: 9) IP Address: 69.64.46.243 is hosted on a dedicated server IP Location: United States - Missouri - Saint Louis - Server4you Inc. ASN: United States AS30083 SERVER4YOU, US (registered Jul 16, 2003) Domain Status: Registered And Active Website
Fuente: Whois, http://whois.domaintools.com/siembraviva.com

7.2.3. Identificación de puntos de entrada de la aplicación (OWASP-IG-003).
Este componente es importante porque permite conocer cómo funciona la aplicación, como es la interacción de los usuarios y el navegador. Las peticiones GET y POST son un componente importante dentro de las páginas Web, se debe prestar mucha atención cuando se usa GET y cuando POST dentro de la aplicación. Las peticiones GET son usadas cuando es información sensible,

cuando se hace una petición POST se complementa usándola por GET. Es común que dentro de las páginas Web se envié información sin que el usuario se dé cuenta.

Se utiliza el complemento HTTPS headers disponible para Mozilla Firefox (ver figura 25), al momento de verificar las solicitudes al sitio web SiembraViva.com, se encuentra que utilizan los dos tipos de peticiones; GET y POST.

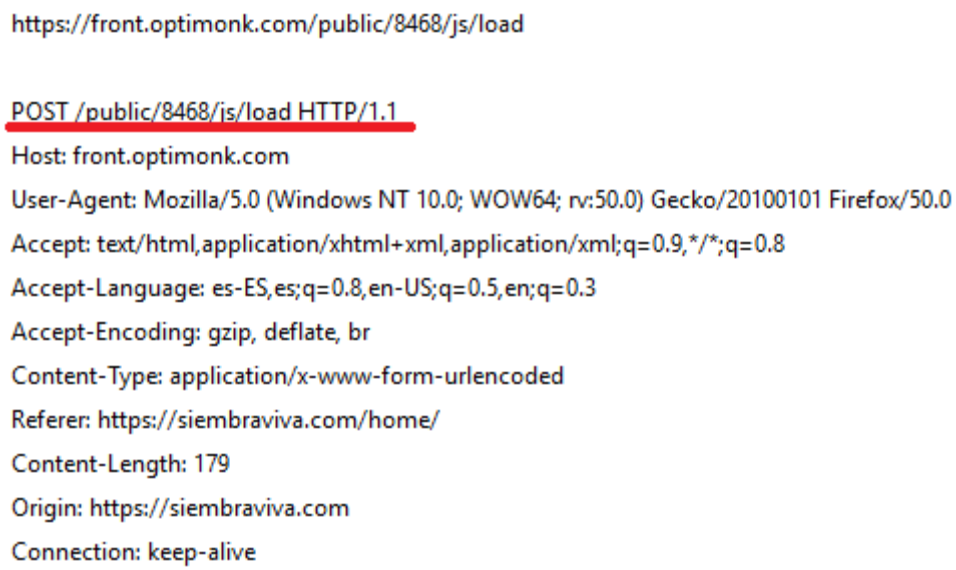
Figura 25. HTTPS Header petición GET



Fuente: El autor, HTTP Headers

Se debe rescatar también la información de las cookies del navegador, además de los elementos que componen los componentes de la petición que se acabó de hacer a la página.

Figura 26. HTTPS Header petición POST



Fuente: El autor, HTTP Headers

El sitio web SiembraViva.com usa los tipos de peticiones, dependiendo de la solicitud que se desee implementar, para el manejo de información sensible son usadas las peticiones tipo POST, y GET para peticiones que no son sensibles.

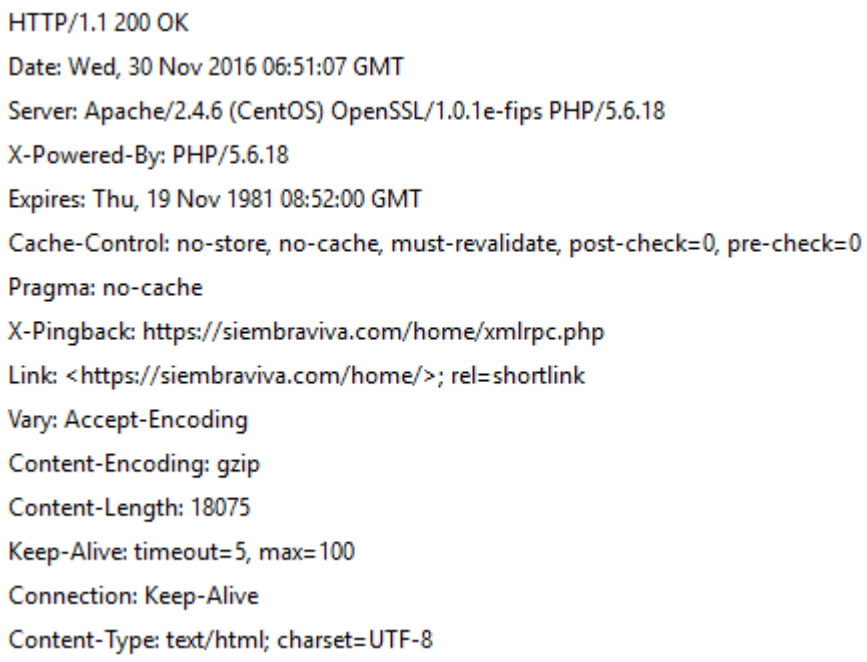
7.2.4. Pruebas para encontrar firmas de Aplicaciones Web (OWASP-IG-004)

Saber el tipo exacto y versión del servidor web nos ayuda considerablemente el proceso de análisis empleado en el sitio web SiembraViva.com, nos permite determinar vulnerabilidades conocidas en dichas versiones y los exploits apropiados a usar durante la prueba, puede cambiar el curso de las pruebas.

Con la herramienta netcat (figura 25), comprobamos en el campo Server en la cabecera de respuesta HTTP.

Nc 69.64.46.243 80

Figura 25. Cabecera HTTP

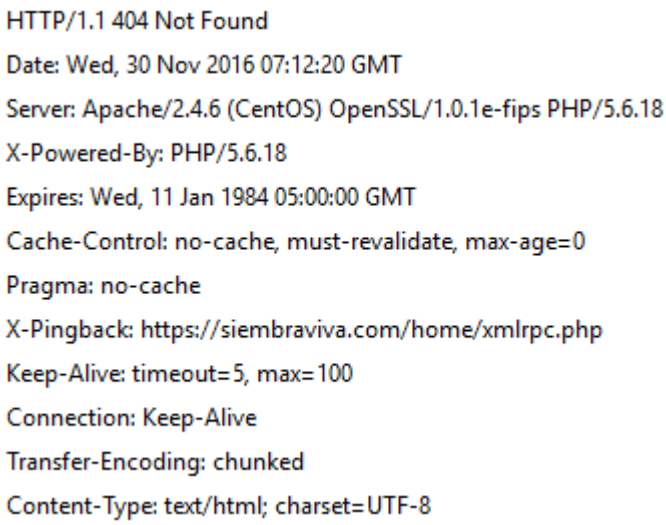


Fuente: El autor, netcat

En el campo Server logramos identificar que el servidor es un Apache versión 2.4.6 ejecutándose sobre el sistema operativo CentOS, OpenSSL 1.0.1e-fips y PHP 5.6.18.

Para estar un poco más seguros sobre la veracidad de los campos de los header HTTP, realizamos peticiones mal formadas con páginas web no existentes en el servidor para verificar las respuestas de los header HTTP y comparar los resultados.

Figura 26. Petición mal formada Cabecera HTTP



Fuente: El autor, netcat

La respuesta en el campo Server de la cabecera HTTP es la misma en ambas peticiones.

7.2.5. Descubrimiento de aplicaciones (OWASP-IG-005).

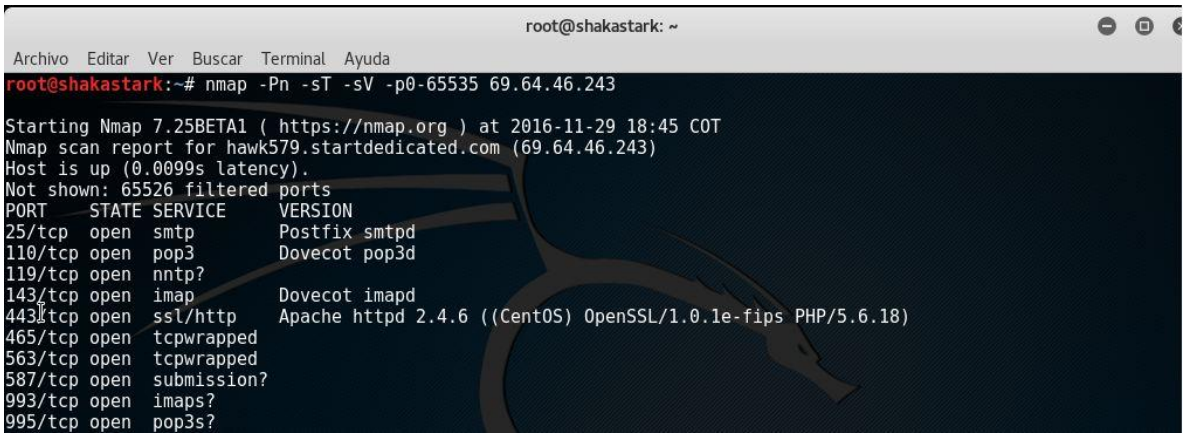
En la actualidad es común que a una dirección IP se le asignen múltiples dominios, al hacer el análisis de los puertos se encuentra que algunos servicios a pesar de estar instalados en el servidor no son visibles para los usuarios de la página web o para quien quiera hacer un análisis de los servicios instalados. Un ejemplo de ello es el puerto 80, este puerto es usado generalmente para el servicio http y al ingresar externamente al servidor para conocer el estado de este servicio se muestra un mensaje de error para notificar que no se tiene instalado.

Es una práctica común que se suministren las direcciones IP y sus respectivos nombres, pero esos nombres en ocasiones pueden tener asignados otros nombres que no se especifican en el momento. Las URL de las aplicaciones web también contienen errores y no relacionan el sitio con el contenido publicado, este tipo de errores esta dado a ocurrir por una incorrecta configuración del servidor o son links de configuración del servidor. Para verificar la correcta configuración y conocer que servicios se encuentran activos o instalados dentro del servidor se realizan las siguientes pruebas al sitio web que se está analizando:

- Verificar los servicios instalados dentro del servidor:

En la figura 27 se evidencia la ejecución del siguiente comando para conocer los servicios instalados en el servidor donde está alojada la página que está siendo analizada:

Figura 27. Consulta servicios configurados



Fuente: El autor, nmap

El comando anterior se realizó en la distribución de Linux Kali Linux y se puede analizar lo siguiente:

1. En el puerto 25 se encuentra instalado el servicio smtp el cual corresponde a un servicio de correo electrónico.
2. En el puerto 110 se encuentra el servicio pop3 el cual descarga el contenido del servidor de correo y lo elimina.
3. En el puerto 143 se encuentra el servicio imap que se encarga de descargar el contenido de los servidores de correo electrónico.
4. En el puerto 443 se encuentra configurado un servidor https Apache.
5. Los demás servicios no se conoce su función, no están especificados.
6. En el puerto 465 y 563 se configurado una medida de seguridad que se incluye dentro de los servidores Linux, los tcpwrappers agrupan los servicios de un servidor y permite el acceso a ellos dependiendo del grado

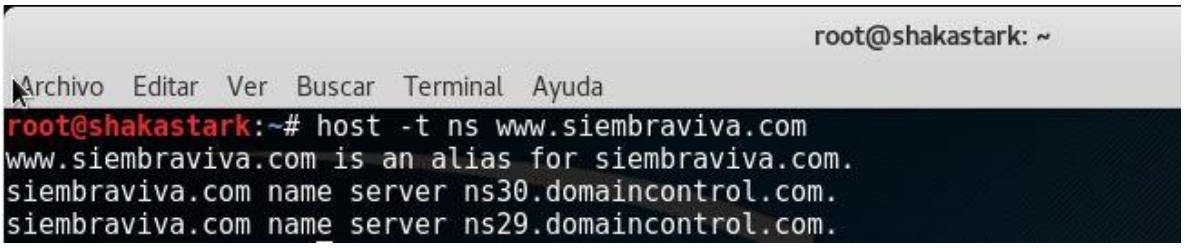
de privilegios que le sea asignado a un usuario en particular. Esta medida es complementaria a las IPTABLES configuradas dentro del servidor.

- Identificación de nombres DNS asociados a una dirección IP

Transferencias de zona DNS

Teniendo en cuenta la figura 28, se consultan los servidores de nombres asociados a la dirección IP de la página de Siembra viva, el siguiente comando permite consultar la información solicitada:

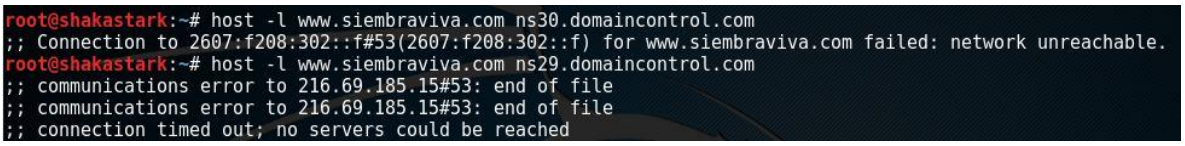
Figura 28. Consulta servicios configurados



Fuente: El autor, host

Después de este paso se hace una petición de transferencia de zona en las URL que fueron visualizadas en el paso anterior, en la figura 29 se puede apreciar el resultado al ejecutar el comando.

Figura 29. Consulta servicios configurados



Fuente: El autor, transferencia DNS

Se realizó la prueba para comprobar la efectividad del comando, pero el resultado obtenido no fue satisfactorio y la razón de esto es que los servidores tienen bloqueado este tipo de operaciones para no perjudicar el servicio de la página web.

De haberse dado un proceso exitoso se muestran todos los enlaces conocidos asignados a la URL que se está realizando el proceso de transferencia DNS.

7.2.6. Análisis de códigos de error (OWASP-IG-006).

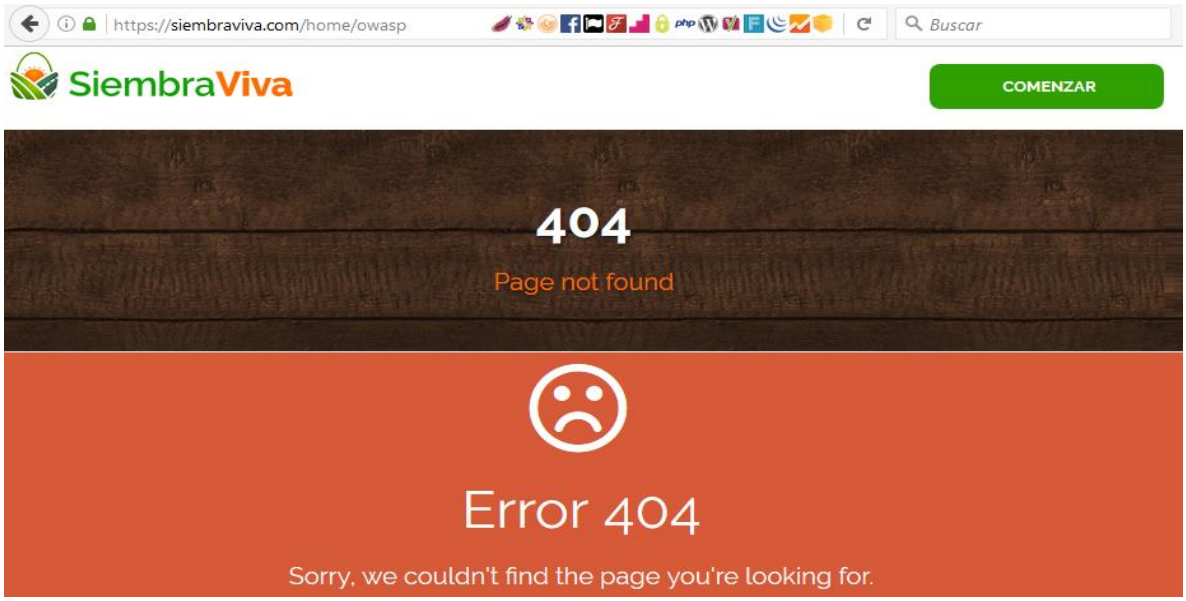
Durante las pruebas y análisis de las aplicaciones web, suelen encontrarse alertas y códigos de error generados por el servidor web. De esta forma mediante peticiones específicas, es posible obtener resultados útiles para el proceso de recopilación de información del sitio web SiembreViva.com.

Estos códigos son de gran utilidad durante las pruebas, porque revelan información sobre el servidor, bases de datos, bugs y otros componentes tecnológicos directamente relacionados con la aplicación web que nos permitan realizar una evaluación de seguridad categorizada según los mensajes de error, esta información será insumo para las fases siguientes del análisis de vulnerabilidades del sitio web SiembraViva.com.

7.2.6.1. HTTP 404 Not Found.

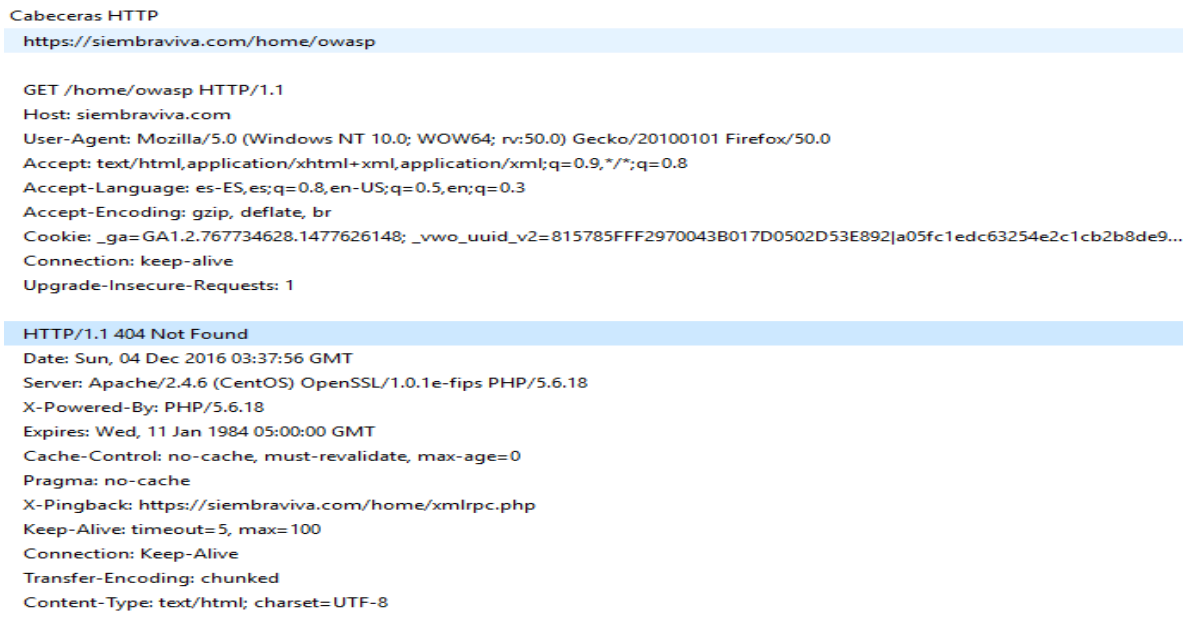
En la figura 27 se puede observar el error 404, un error muy común que se puede encontrar durante las búsquedas o cuando se accede manualmente a las URL del sitio web, este código proporciona detalles sobre el servidor web que ejecuta la aplicación y componentes asociados.

Figura 27. Error 404 not found



Fuente: El autor

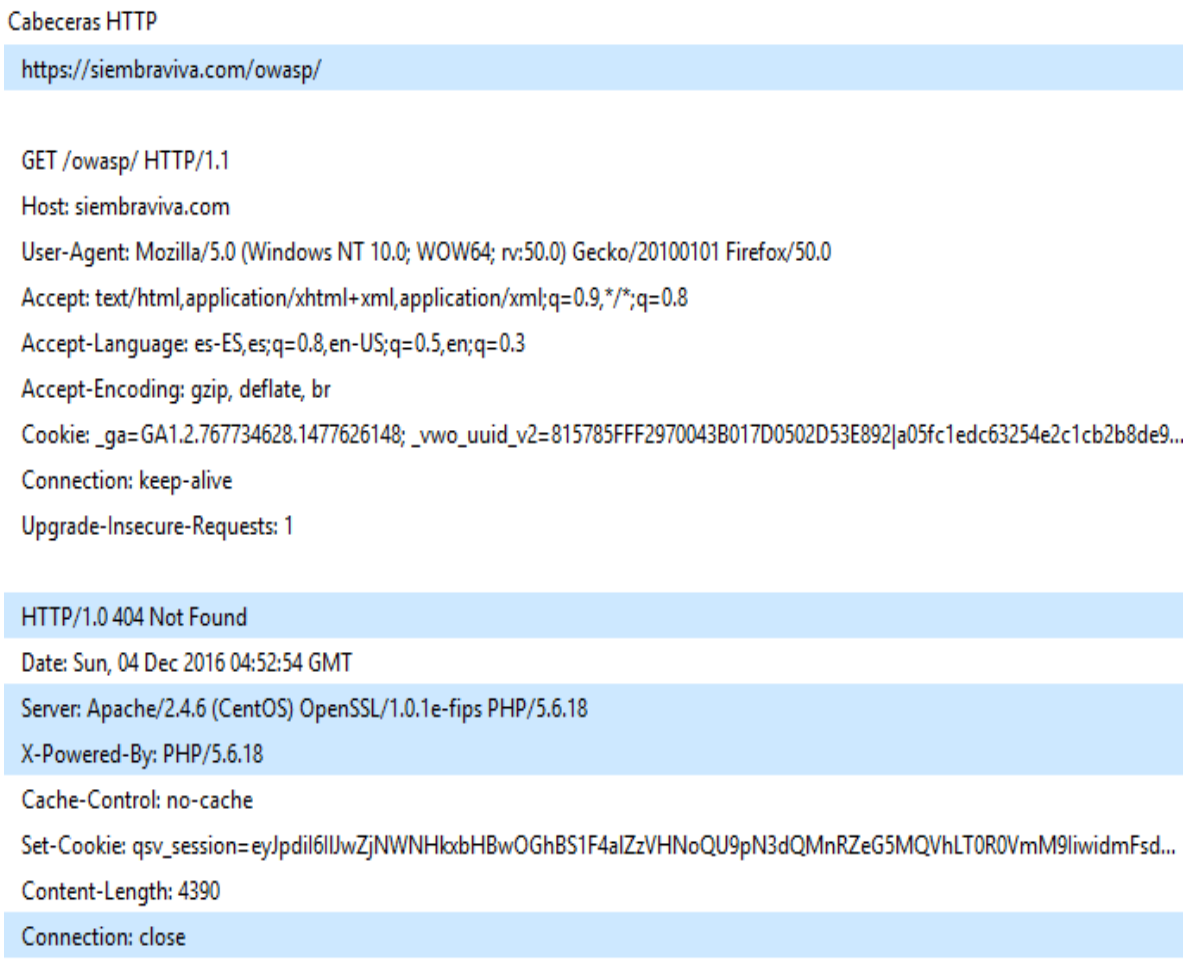
Figura 28. Cabecera HTTP error 404



Fuente: El autor

En la figura 28, se puede evidenciar que la URL https://siembraviva.com/home/owasp no existe, su mensaje de error es 404 no found, aparece la información del servidor web, sistema operativo, servidor web Apache, OpenSSL y PHP. Esta información es muy importante desde el punto de vista de la identificación del sistema operativo y versión de aplicaciones.

Figura 29. Cabecera HTTP error 404 - SiembraViva.com/owasp



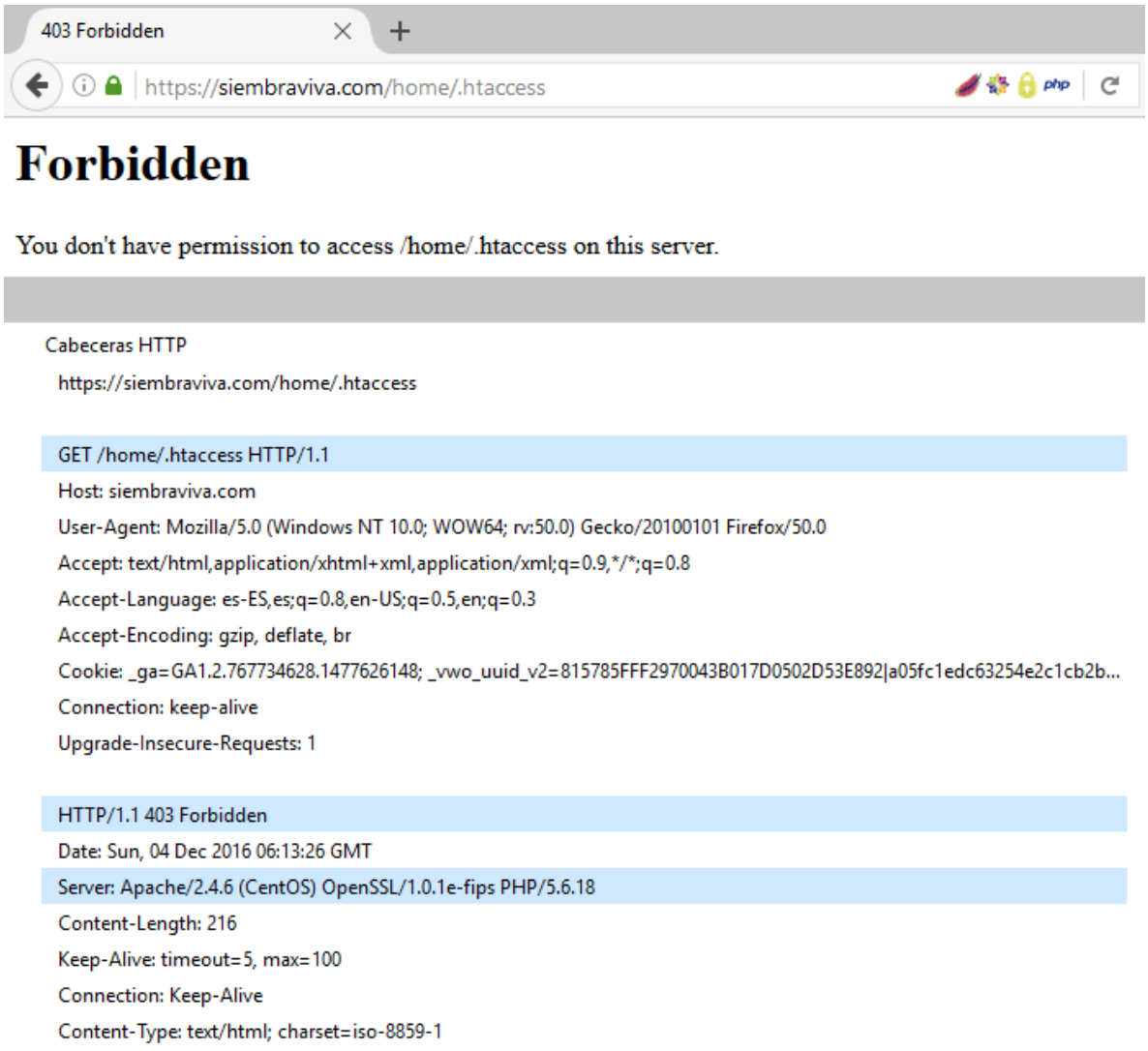
Fuente: El autor

Al realizar la prueba al directorio del sitio http://siembraviva.com/owasp nos muestra el siguiente mensaje de error, “Whoops, looks like something went wrong”, dicho mensaje de error nos devuelve en la cabecera HTTP el “error 404 not found” y la conexión al servidor es cerrada.

7.2.6.2. HTTP 403 Forbidden

Este error es originado por un intento de acceso a recursos limitados por el servidor, en otras palabras, el servidor ha prohibido el acceso y devuelve a la petición la respuesta "acceso denegado" tal como lo muestra la figura 30.

Figura 30. HTTP 403 Forbidden



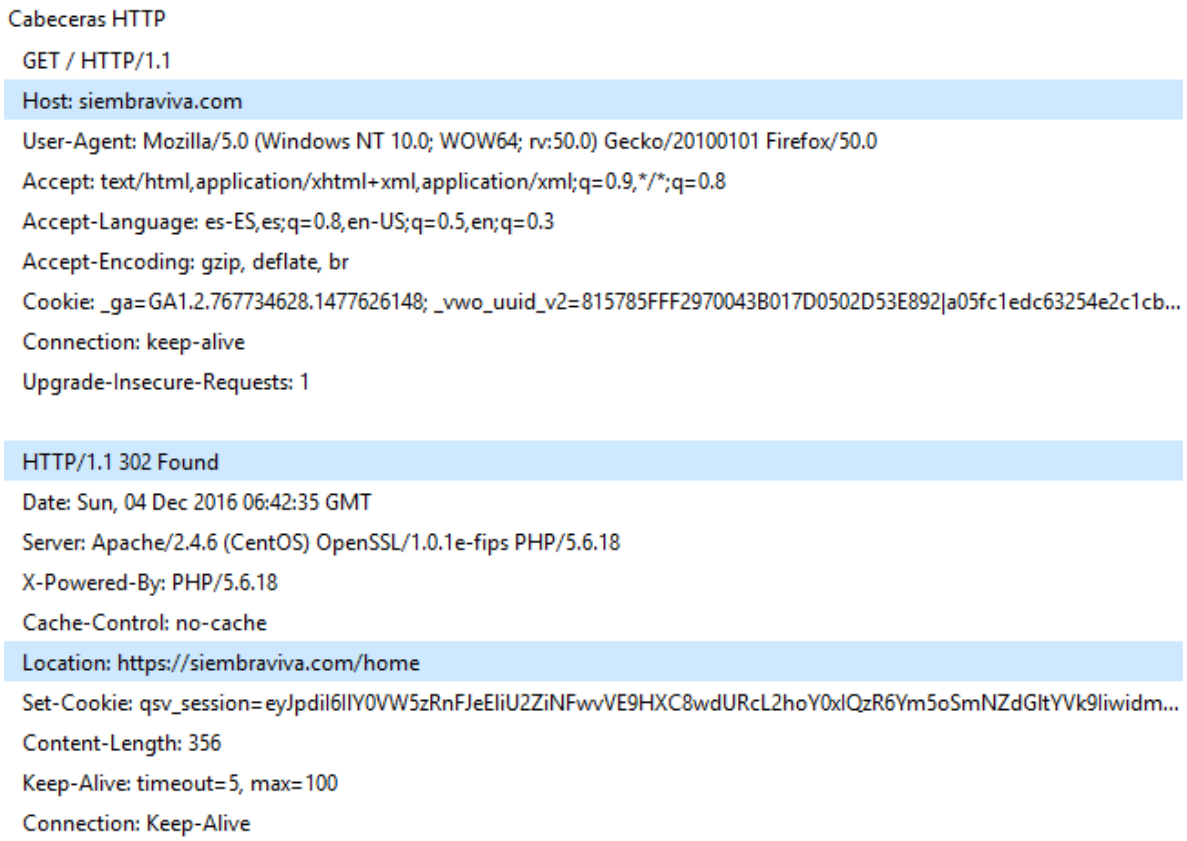
Fuente: El autor

7.2.6.3. HTTP 301 Moved Permanently

El siguiente mensaje de error es originado debido a que el servidor no logro comunicarse con el archivo solicitado, pero este ha sido movido a una dirección permanente, la respuesta suele ser redireccionada a la ubicación correcta.

Al ingresar desde el navegador al sitio web https://siembraviva.com devuelve en el HTTP header error 301 y es redireccionado al directorio https://siembraviva.com/home/ automáticamente.

Figura 31. HTTP 301 Moved Permanently



Fuente: El autor

7.3. PRUEBAS DE GESTIÓN DE CONFIGURACIÓN DE LA INFRAESTRUCTURA

Los análisis de infraestructura o topología de la arquitectura web del servidor suelen revelar datos importantes sobre la aplicación web. Los datos sensibles por obtener son el código fuente cliente, métodos HTTP permitidos, funciones

permitidas, métodos de autenticación y respuestas, validaciones de datos y hasta configuraciones de infraestructura tecnológica.

7.3.1. Pruebas de SSL/TLS (OWASP-CM-001).

Las pruebas SSL/TLS hacen parte del análisis de los dos protocolos que proporcionan soporte criptográfico a los certificados web, como principal canal seguro de protección y confidencialidad sobre la información que se transmite entre el cliente y el servidor.

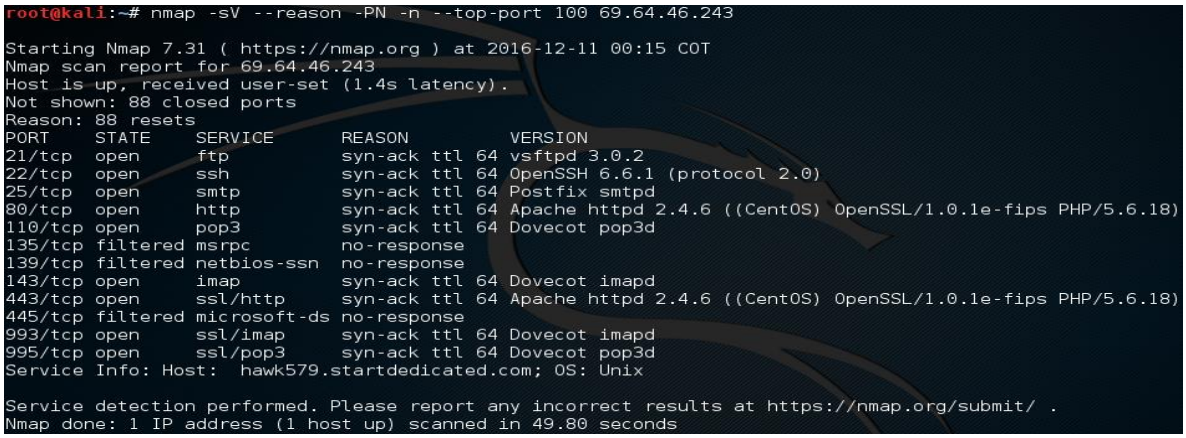
Por considerarse críticas estas implementaciones de seguridad, es importante verificar la utilización de un algoritmo de cifrado fuerte, especificaciones y su correcta implementación.

7.3.1.1. Prueba de caja negra SSL/TLS.

Para detectar el posible soporte a cifrados débiles en SSL/TLS, se identifican los puertos asociados a los servicios de SSL/TLS. (Por defecto 443), aunque estos puertos pueden varias ya que el administrador puede modificar dicha configuración y ajustarla a puertos no estándar.

En la figura 32 se realiza un escaneo con la herramienta nmap, parámetro -sV que puede identificar los servicios asociados a los puertos SSL/TLS. nmap -sV –reason -PN -n 69.64.46.243.

Figura 32. Reconocimiento de servicios SSL



Fuente: El autor, nmap

Los puertos ssl/http 443, ssl/imap 993, ssl/pop3 995 se encuentras abiertos, los cuales son usados para realizar la comprobación de la información del certificado, cifrados débiles y SSLv2 mediante nmap.

Se ejecuta el script SSL para validar las conexiones SSLv3/TLS, esta valida el sistema de cifrado y compresor durante la conexión si un host la acepta o rechaza.

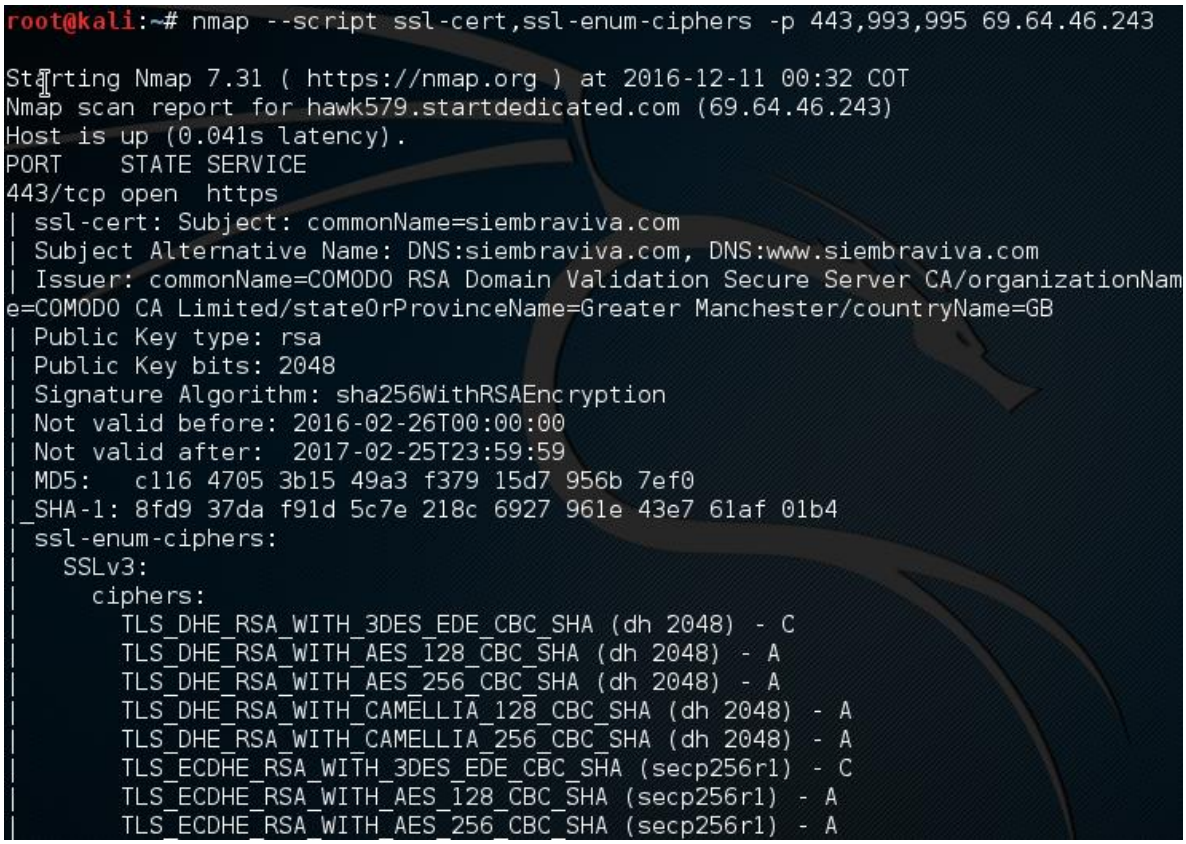
Cada conjunto de cifrado se muestra con un grado de la letra (A a F) que indica la fuerza de la conexión. La calificación se basa en la potencia criptográfica del intercambio de claves y del cifrado de flujo. La elección algoritmo de integridad de mensaje (hash) no es un factor. La línea de salida comienza con “Least strength” y muestra la fuerza del cifrado más débil.

SSLv3/TLSv1 requiere más esfuerzo para determinar qué sistemas de cifrado y métodos de compresión del servidor admite SSLv2. Se enumeran los sistemas de cifrado y compresión capaz de soportar, y el servidor responde con un solo sistema de cifrado elegido, o una notificación de rechazo.

En la figura 33 y tabla 7 se pueden observar la ejecución del Script y sus resultados, este script advertirá sobre ciertas configuraciones incorrectas de SSL, como certificados firmados con MD5, parámetros DH efímeros de baja calidad y la vulnerabilidad POODLE.

Este script es intrusivo, ya que inicia muchas conexiones al servidor, y por lo tanto es bastante ruidoso.

Figura 33. Script ssl-enum-ciphers



Fuente: El autor, nmap

Tabla 7. Resultado completo Script ssl-enum-ciphers

Resultado análisis Script ssl-enum-ciphers
root@kali:~# nmap --script ssl-cert,ssl-enum-ciphers -p 443,993,995 69.64.46.243 Starting Nmap 7.31 (https://nmap.org) at 2016-12-11 00:32 COT Nmap scan report for hawk579.startdedicated.com (69.64.46.243) Host is up (0.041s latency). PORT STATE SERVICE 443/tcp open https | ssl-cert: Subject: commonName=siembraviva.com | Subject Alternative Name: DNS:siembraviva.com, DNS:www.siembraviva.com | Issuer: commonName=COMODO RSA Domain Validation Secure Server

```
CA/organizationName=COMODO CA Limited/stateOrProvinceName=Greater
Manchester/countryName=GB
| Public Key type: rsa
| Public Key bits: 2048
| Signature Algorithm: sha256WithRSAEncryption
| Not valid before: 2016-02-26T00:00:00
| Not valid after:  2017-02-25T23:59:59
| MD5:   c116 4705 3b15 49a3 f379 15d7 956b 7ef0
|_SHA-1: 8fd9 37da f91d 5c7e 218c 6927 961e 43e7 61af 01b4
| ssl-enum-ciphers:
|   SSLv3:
|     ciphers:
|       TLS_DHE_RSA_WITH_3DES_EDE_CBC_SHA (dh 2048) - C
|       TLS_DHE_RSA_WITH_AES_128_CBC_SHA (dh 2048) - A
|       TLS_DHE_RSA_WITH_AES_256_CBC_SHA (dh 2048) - A
|       TLS_DHE_RSA_WITH_CAMELLIA_128_CBC_SHA (dh 2048) - A
|       TLS_DHE_RSA_WITH_CAMELLIA_256_CBC_SHA (dh 2048) - A
|       TLS_ECDHE_RSA_WITH_3DES_EDE_CBC_SHA (secp256r1) - C
|       TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA (secp256r1) - A
|       TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA (secp256r1) - A
|       TLS_ECDHE_RSA_WITH_RC4_128_SHA (secp256r1) - C
|       TLS_RSA_WITH_3DES_EDE_CBC_SHA (rsa 2048) - C
|       TLS_RSA_WITH_AES_128_CBC_SHA (rsa 2048) - A
|       TLS_RSA_WITH_AES_256_CBC_SHA (rsa 2048) - A
|       TLS_RSA_WITH_CAMELLIA_128_CBC_SHA (rsa 2048) - A
|       TLS_RSA_WITH_CAMELLIA_256_CBC_SHA (rsa 2048) - A
|       TLS_RSA_WITH_RC4_128_SHA (rsa 2048) - C
|     compressors:
|       NULL
|     cipher preference: client
|     warnings:
|       64-bit block cipher 3DES vulnerable to SWEET32 attack
|       Broken cipher RC4 is deprecated by RFC 7465
|       CBC-mode cipher in SSLv3 (CVE-2014-3566)
|   TLSv1.0:
|     ciphers:
|       TLS_DHE_RSA_WITH_3DES_EDE_CBC_SHA (dh 2048) - C
|       TLS_DHE_RSA_WITH_AES_128_CBC_SHA (dh 2048) - A
|       TLS_DHE_RSA_WITH_AES_256_CBC_SHA (dh 2048) - A
|       TLS_DHE_RSA_WITH_CAMELLIA_128_CBC_SHA (dh 2048) - A
|       TLS_DHE_RSA_WITH_CAMELLIA_256_CBC_SHA (dh 2048) - A
|       TLS_ECDHE_RSA_WITH_3DES_EDE_CBC_SHA (secp256r1) - C
|       TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA (secp256r1) - A
```

```
|       TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA (secp256r1) - A
|       TLS_ECDHE_RSA_WITH_RC4_128_SHA (secp256r1) - C
|       TLS_RSA_WITH_3DES_EDE_CBC_SHA (rsa 2048) - C
|       TLS_RSA_WITH_AES_128_CBC_SHA (rsa 2048) - A
|       TLS_RSA_WITH_AES_256_CBC_SHA (rsa 2048) - A
|       TLS_RSA_WITH_CAMELLIA_128_CBC_SHA (rsa 2048) - A
|       TLS_RSA_WITH_CAMELLIA_256_CBC_SHA (rsa 2048) - A
|       TLS_RSA_WITH_RC4_128_SHA (rsa 2048) - C
|     compressors:
|       NULL
|     cipher preference: client
|     warnings:
|       64-bit block cipher 3DES vulnerable to SWEET32 attack
|       Broken cipher RC4 is deprecated by RFC 7465
|   TLSv1.1:
|     ciphers:
|       TLS_DHE_RSA_WITH_3DES_EDE_CBC_SHA (dh 2048) - C
|       TLS_DHE_RSA_WITH_AES_128_CBC_SHA (dh 2048) - A
|       TLS_DHE_RSA_WITH_AES_256_CBC_SHA (dh 2048) - A
|       TLS_DHE_RSA_WITH_CAMELLIA_128_CBC_SHA (dh 2048) - A
|       TLS_DHE_RSA_WITH_CAMELLIA_256_CBC_SHA (dh 2048) - A
|       TLS_ECDHE_RSA_WITH_3DES_EDE_CBC_SHA (secp256r1) - C
|       TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA (secp256r1) - A
|       TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA (secp256r1) - A
|       TLS_ECDHE_RSA_WITH_RC4_128_SHA (secp256r1) - C
|       TLS_RSA_WITH_3DES_EDE_CBC_SHA (rsa 2048) - C
|       TLS_RSA_WITH_AES_128_CBC_SHA (rsa 2048) - A
|       TLS_RSA_WITH_AES_256_CBC_SHA (rsa 2048) - A
|       TLS_RSA_WITH_CAMELLIA_128_CBC_SHA (rsa 2048) - A
|       TLS_RSA_WITH_CAMELLIA_256_CBC_SHA (rsa 2048) - A
|       TLS_RSA_WITH_RC4_128_SHA (rsa 2048) - C
|     compressors:
|       NULL
|     cipher preference: client
|     warnings:
|       64-bit block cipher 3DES vulnerable to SWEET32 attack
|       Broken cipher RC4 is deprecated by RFC 7465
|   TLSv1.2:
|     ciphers:
|       TLS_DHE_RSA_WITH_3DES_EDE_CBC_SHA (dh 2048) - C
|       TLS_DHE_RSA_WITH_AES_128_CBC_SHA (dh 2048) - A
|       TLS_DHE_RSA_WITH_AES_128_CBC_SHA256 (dh 2048) - A
|       TLS_DHE_RSA_WITH_AES_128_GCM_SHA256 (dh 2048) - A
```

```
|       TLS_DHE_RSA_WITH_AES_256_CBC_SHA (dh 2048) - A
|       TLS_DHE_RSA_WITH_AES_256_CBC_SHA256 (dh 2048) - A
|       TLS_DHE_RSA_WITH_AES_256_GCM_SHA384 (dh 2048) - A
|       TLS_DHE_RSA_WITH_CAMELLIA_128_CBC_SHA (dh 2048) - A
|       TLS_DHE_RSA_WITH_CAMELLIA_256_CBC_SHA (dh 2048) - A
|       TLS_ECDHE_RSA_WITH_3DES_EDE_CBC_SHA (secp256r1) - C
|       TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA (secp256r1) - A
|       TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA256 (secp256r1) - A
|       TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256 (secp256r1) - A
|       TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA (secp256r1) - A
|       TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA384 (secp256r1) - A
|       TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384 (secp256r1) - A
|       TLS_ECDHE_RSA_WITH_RC4_128_SHA (secp256r1) - C
|       TLS_RSA_WITH_3DES_EDE_CBC_SHA (rsa 2048) - C
|       TLS_RSA_WITH_AES_128_CBC_SHA (rsa 2048) - A
|       TLS_RSA_WITH_AES_128_CBC_SHA256 (rsa 2048) - A
|       TLS_RSA_WITH_AES_128_GCM_SHA256 (rsa 2048) - A
|       TLS_RSA_WITH_AES_256_CBC_SHA (rsa 2048) - A
|       TLS_RSA_WITH_AES_256_CBC_SHA256 (rsa 2048) - A
|       TLS_RSA_WITH_AES_256_GCM_SHA384 (rsa 2048) - A
|       TLS_RSA_WITH_CAMELLIA_128_CBC_SHA (rsa 2048) - A
|       TLS_RSA_WITH_CAMELLIA_256_CBC_SHA (rsa 2048) - A
|       TLS_RSA_WITH_RC4_128_SHA (rsa 2048) - C
|     compressors:
|       NULL
|     cipher preference: client
|     warnings:
|       64-bit block cipher 3DES vulnerable to SWEET32 attack
|       Broken cipher RC4 is deprecated by RFC 7465
|_  least strength: C
993/tcp open  imaps
| ssl-cert: Subject: commonName=hawk579.startdedicated.com/countryName=DE
| Issuer: commonName=hawk579.startdedicated.com/countryName=DE
| Public Key type: rsa
| Public Key bits: 1024
| Signature Algorithm: sha1WithRSAEncryption
| Not valid before: 2016-03-11T16:32:45
| Not valid after:  2017-03-11T16:32:45
| MD5:   ce6f 1868 c083 7a39 25cd d4fa 4797 015d
|_SHA-1: 1335 2220 84d7 e1b7 13a3 1711 328b 134a bb3b 16e8
| ssl-enum-ciphers:
|   SSLv3:
|     ciphers:
```

```
|       TLS_DHE_RSA_WITH_3DES_EDE_CBC_SHA (dh 1024) - D
|       TLS_DHE_RSA_WITH_AES_128_CBC_SHA (dh 1024) - A
|       TLS_DHE_RSA_WITH_AES_256_CBC_SHA (dh 1024) - A
|       TLS_DHE_RSA_WITH_CAMELLIA_128_CBC_SHA (dh 1024) - A
|       TLS_DHE_RSA_WITH_CAMELLIA_256_CBC_SHA (dh 1024) - A
|       TLS_DHE_RSA_WITH_SEED_CBC_SHA (dh 1024) - A
|       TLS_ECDHE_RSA_WITH_3DES_EDE_CBC_SHA (secp384r1) - D
|       TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA (secp384r1) - A
|       TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA (secp384r1) - A
|       TLS_ECDHE_RSA_WITH_RC4_128_SHA (secp384r1) - D
|       TLS_RSA_WITH_3DES_EDE_CBC_SHA (rsa 1024) - D
|       TLS_RSA_WITH_AES_128_CBC_SHA (rsa 1024) - A
|       TLS_RSA_WITH_AES_256_CBC_SHA (rsa 1024) - A
|       TLS_RSA_WITH_CAMELLIA_128_CBC_SHA (rsa 1024) - A
|       TLS_RSA_WITH_CAMELLIA_256_CBC_SHA (rsa 1024) - A
|       TLS_RSA_WITH_IDEA_CBC_SHA (rsa 1024) - A
|       TLS_RSA_WITH_RC4_128_MD5 (rsa 1024) - D
|       TLS_RSA_WITH_RC4_128_SHA (rsa 1024) - D
|       TLS_RSA_WITH_SEED_CBC_SHA (rsa 1024) - A
|     compressors:
|       NULL
|     cipher preference: client
|     warnings:
|       64-bit block cipher 3DES vulnerable to SWEET32 attack
|       64-bit block cipher IDEA vulnerable to SWEET32 attack
|       Broken cipher RC4 is deprecated by RFC 7465
|       CBC-mode cipher in SSLv3 (CVE-2014-3566)
|       Ciphersuite uses MD5 for message integrity
|       Weak certificate signature: SHA1
|   TLSv1.0:
|     ciphers:
|       TLS_DHE_RSA_WITH_3DES_EDE_CBC_SHA (dh 1024) - D
|       TLS_DHE_RSA_WITH_AES_128_CBC_SHA (dh 1024) - A
|       TLS_DHE_RSA_WITH_AES_256_CBC_SHA (dh 1024) - A
|       TLS_DHE_RSA_WITH_CAMELLIA_128_CBC_SHA (dh 1024) - A
|       TLS_DHE_RSA_WITH_CAMELLIA_256_CBC_SHA (dh 1024) - A
|       TLS_DHE_RSA_WITH_SEED_CBC_SHA (dh 1024) - A
|       TLS_ECDHE_RSA_WITH_3DES_EDE_CBC_SHA (secp384r1) - D
|       TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA (secp384r1) - A
|       TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA (secp384r1) - A
|       TLS_ECDHE_RSA_WITH_RC4_128_SHA (secp384r1) - D
|       TLS_RSA_WITH_3DES_EDE_CBC_SHA (rsa 1024) - D
|       TLS_RSA_WITH_AES_128_CBC_SHA (rsa 1024) - A
```

```
|       TLS_RSA_WITH_AES_256_CBC_SHA (rsa 1024) - A
|       TLS_RSA_WITH_CAMELLIA_128_CBC_SHA (rsa 1024) - A
|       TLS_RSA_WITH_CAMELLIA_256_CBC_SHA (rsa 1024) - A
|       TLS_RSA_WITH_IDEA_CBC_SHA (rsa 1024) - A
|       TLS_RSA_WITH_RC4_128_MD5 (rsa 1024) - D
|       TLS_RSA_WITH_RC4_128_SHA (rsa 1024) - D
|       TLS_RSA_WITH_SEED_CBC_SHA (rsa 1024) - A
|     compressors:
|       NULL
|     cipher preference: client
|     warnings:
|       64-bit block cipher 3DES vulnerable to SWEET32 attack
|       64-bit block cipher IDEA vulnerable to SWEET32 attack
|       Broken cipher RC4 is deprecated by RFC 7465
|       Ciphersuite uses MD5 for message integrity
|       Weak certificate signature: SHA1
|   TLSv1.1:
|     ciphers:
|       TLS_DHE_RSA_WITH_3DES_EDE_CBC_SHA (dh 1024) - D
|       TLS_DHE_RSA_WITH_AES_128_CBC_SHA (dh 1024) - A
|       TLS_DHE_RSA_WITH_AES_256_CBC_SHA (dh 1024) - A
|       TLS_DHE_RSA_WITH_CAMELLIA_128_CBC_SHA (dh 1024) - A
|       TLS_DHE_RSA_WITH_CAMELLIA_256_CBC_SHA (dh 1024) - A
|       TLS_DHE_RSA_WITH_SEED_CBC_SHA (dh 1024) - A
|       TLS_ECDHE_RSA_WITH_3DES_EDE_CBC_SHA (secp384r1) - D
|       TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA (secp384r1) - A
|       TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA (secp384r1) - A
|       TLS_ECDHE_RSA_WITH_RC4_128_SHA (secp384r1) - D
|       TLS_RSA_WITH_3DES_EDE_CBC_SHA (rsa 1024) - D
|       TLS_RSA_WITH_AES_128_CBC_SHA (rsa 1024) - A
|       TLS_RSA_WITH_AES_256_CBC_SHA (rsa 1024) - A
|       TLS_RSA_WITH_CAMELLIA_128_CBC_SHA (rsa 1024) - A
|       TLS_RSA_WITH_CAMELLIA_256_CBC_SHA (rsa 1024) - A
|       TLS_RSA_WITH_IDEA_CBC_SHA (rsa 1024) - A
|       TLS_RSA_WITH_RC4_128_MD5 (rsa 1024) - D
|       TLS_RSA_WITH_RC4_128_SHA (rsa 1024) - D
|       TLS_RSA_WITH_SEED_CBC_SHA (rsa 1024) - A
|     compressors:
|       NULL
|     cipher preference: client
|     warnings:
|       64-bit block cipher 3DES vulnerable to SWEET32 attack
|       64-bit block cipher IDEA vulnerable to SWEET32 attack
```

```
|       Broken cipher RC4 is deprecated by RFC 7465
|       Ciphersuite uses MD5 for message integrity
|       Weak certificate signature: SHA1
|   TLSv1.2:
|     ciphers:
|       TLS_DHE_RSA_WITH_3DES_EDE_CBC_SHA (dh 1024) - D
|       TLS_DHE_RSA_WITH_AES_128_CBC_SHA (dh 1024) - A
|       TLS_DHE_RSA_WITH_AES_128_CBC_SHA256 (dh 1024) - A
|       TLS_DHE_RSA_WITH_AES_128_GCM_SHA256 (dh 1024) - A
|       TLS_DHE_RSA_WITH_AES_256_CBC_SHA (dh 1024) - A
|       TLS_DHE_RSA_WITH_AES_256_CBC_SHA256 (dh 1024) - A
|       TLS_DHE_RSA_WITH_AES_256_GCM_SHA384 (dh 1024) - A
|       TLS_DHE_RSA_WITH_CAMELLIA_128_CBC_SHA (dh 1024) - A
|       TLS_DHE_RSA_WITH_CAMELLIA_256_CBC_SHA (dh 1024) - A
|       TLS_DHE_RSA_WITH_SEED_CBC_SHA (dh 1024) - A
|       TLS_ECDHE_RSA_WITH_3DES_EDE_CBC_SHA (secp384r1) - D
|       TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA (secp384r1) - A
|       TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA256 (secp384r1) - A
|       TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256 (secp384r1) - A
|       TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA (secp384r1) - A
|       TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA384 (secp384r1) - A
|       TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384 (secp384r1) - A
|       TLS_ECDHE_RSA_WITH_RC4_128_SHA (secp384r1) - D
|       TLS_RSA_WITH_3DES_EDE_CBC_SHA (rsa 1024) - D
|       TLS_RSA_WITH_AES_128_CBC_SHA (rsa 1024) - A
|       TLS_RSA_WITH_AES_128_CBC_SHA256 (rsa 1024) - A
|       TLS_RSA_WITH_AES_128_GCM_SHA256 (rsa 1024) - A
|       TLS_RSA_WITH_AES_256_CBC_SHA (rsa 1024) - A
|       TLS_RSA_WITH_AES_256_CBC_SHA256 (rsa 1024) - A
|       TLS_RSA_WITH_AES_256_GCM_SHA384 (rsa 1024) - A
|       TLS_RSA_WITH_CAMELLIA_128_CBC_SHA (rsa 1024) - A
|       TLS_RSA_WITH_CAMELLIA_256_CBC_SHA (rsa 1024) - A
|       TLS_RSA_WITH_IDEA_CBC_SHA (rsa 1024) - A
|       TLS_RSA_WITH_RC4_128_MD5 (rsa 1024) - D
|       TLS_RSA_WITH_RC4_128_SHA (rsa 1024) - D
|       TLS_RSA_WITH_SEED_CBC_SHA (rsa 1024) - A
|     compressors:
|       NULL
|     cipher preference: client
|     warnings:
|       64-bit block cipher 3DES vulnerable to SWEET32 attack
|       64-bit block cipher IDEA vulnerable to SWEET32 attack
|       Broken cipher RC4 is deprecated by RFC 7465
```

```
|       Ciphersuite uses MD5 for message integrity
|       Weak certificate signature: SHA1
|_  least strength: D
995/tcp open  pop3s
| ssl-cert: Subject: commonName=hawk579.startdedicated.com/countryName=DE
| Issuer: commonName=hawk579.startdedicated.com/countryName=DE
| Public Key type: rsa
| Public Key bits: 1024
| Signature Algorithm: sha1WithRSAEncryption
| Not valid before: 2016-03-11T16:32:45
| Not valid after:  2017-03-11T16:32:45
| MD5:   ce6f 1868 c083 7a39 25cd d4fa 4797 015d
|_SHA-1: 1335 2220 84d7 e1b7 13a3 1711 328b 134a bb3b 16e8
| ssl-enum-ciphers:
|   SSLv3:
|     ciphers:
|       TLS_DHE_RSA_WITH_3DES_EDE_CBC_SHA (dh 1024) - D
|       TLS_DHE_RSA_WITH_AES_128_CBC_SHA (dh 1024) - A
|       TLS_DHE_RSA_WITH_AES_256_CBC_SHA (dh 1024) - A
|       TLS_DHE_RSA_WITH_CAMELLIA_128_CBC_SHA (dh 1024) - A
|       TLS_DHE_RSA_WITH_CAMELLIA_256_CBC_SHA (dh 1024) - A
|       TLS_DHE_RSA_WITH_SEED_CBC_SHA (dh 1024) - A
|       TLS_ECDHE_RSA_WITH_3DES_EDE_CBC_SHA (secp384r1) - D
|       TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA (secp384r1) - A
|       TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA (secp384r1) - A
|       TLS_ECDHE_RSA_WITH_RC4_128_SHA (secp384r1) - D
|       TLS_RSA_WITH_3DES_EDE_CBC_SHA (rsa 1024) - D
|       TLS_RSA_WITH_AES_128_CBC_SHA (rsa 1024) - A
|       TLS_RSA_WITH_AES_256_CBC_SHA (rsa 1024) - A
|       TLS_RSA_WITH_CAMELLIA_128_CBC_SHA (rsa 1024) - A
|       TLS_RSA_WITH_CAMELLIA_256_CBC_SHA (rsa 1024) - A
|       TLS_RSA_WITH_IDEA_CBC_SHA (rsa 1024) - A
|       TLS_RSA_WITH_RC4_128_MD5 (rsa 1024) - D
|       TLS_RSA_WITH_RC4_128_SHA (rsa 1024) - D
|       TLS_RSA_WITH_SEED_CBC_SHA (rsa 1024) - A
|     compressors:
|       NULL
|     cipher preference: client
|     warnings:
|       64-bit block cipher 3DES vulnerable to SWEET32 attack
|       64-bit block cipher IDEA vulnerable to SWEET32 attack
|       Broken cipher RC4 is deprecated by RFC 7465
|       CBC-mode cipher in SSLv3 (CVE-2014-3566)
```

```
|       Ciphersuite uses MD5 for message integrity
|       Weak certificate signature: SHA1
|   TLSv1.0:
|     ciphers:
|       TLS_DHE_RSA_WITH_3DES_EDE_CBC_SHA (dh 1024) - D
|       TLS_DHE_RSA_WITH_AES_128_CBC_SHA (dh 1024) - A
|       TLS_DHE_RSA_WITH_AES_256_CBC_SHA (dh 1024) - A
|       TLS_DHE_RSA_WITH_CAMELLIA_128_CBC_SHA (dh 1024) - A
|       TLS_DHE_RSA_WITH_CAMELLIA_256_CBC_SHA (dh 1024) - A
|       TLS_DHE_RSA_WITH_SEED_CBC_SHA (dh 1024) - A
|       TLS_ECDHE_RSA_WITH_3DES_EDE_CBC_SHA (secp384r1) - D
|       TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA (secp384r1) - A
|       TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA (secp384r1) - A
|       TLS_ECDHE_RSA_WITH_RC4_128_SHA (secp384r1) - D
|       TLS_RSA_WITH_3DES_EDE_CBC_SHA (rsa 1024) - D
|       TLS_RSA_WITH_AES_128_CBC_SHA (rsa 1024) - A
|       TLS_RSA_WITH_AES_256_CBC_SHA (rsa 1024) - A
|       TLS_RSA_WITH_CAMELLIA_128_CBC_SHA (rsa 1024) - A
|       TLS_RSA_WITH_CAMELLIA_256_CBC_SHA (rsa 1024) - A
|       TLS_RSA_WITH_IDEA_CBC_SHA (rsa 1024) - A
|       TLS_RSA_WITH_RC4_128_MD5 (rsa 1024) - D
|       TLS_RSA_WITH_RC4_128_SHA (rsa 1024) - D
|       TLS_RSA_WITH_SEED_CBC_SHA (rsa 1024) - A
|     compressors:
|       NULL
|     cipher preference: client
|     warnings:
|       64-bit block cipher 3DES vulnerable to SWEET32 attack
|       64-bit block cipher IDEA vulnerable to SWEET32 attack
|       Broken cipher RC4 is deprecated by RFC 7465
|       Ciphersuite uses MD5 for message integrity
|       Weak certificate signature: SHA1
|   TLSv1.1:
|     ciphers:
|       TLS_DHE_RSA_WITH_3DES_EDE_CBC_SHA (dh 1024) - D
|       TLS_DHE_RSA_WITH_AES_128_CBC_SHA (dh 1024) - A
|       TLS_DHE_RSA_WITH_AES_256_CBC_SHA (dh 1024) - A
|       TLS_DHE_RSA_WITH_CAMELLIA_128_CBC_SHA (dh 1024) - A
|       TLS_DHE_RSA_WITH_CAMELLIA_256_CBC_SHA (dh 1024) - A
|       TLS_DHE_RSA_WITH_SEED_CBC_SHA (dh 1024) - A
|       TLS_ECDHE_RSA_WITH_3DES_EDE_CBC_SHA (secp384r1) - D
|       TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA (secp384r1) - A
|       TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA (secp384r1) - A
```

```
|       TLS_ECDHE_RSA_WITH_RC4_128_SHA (secp384r1) - D
|       TLS_RSA_WITH_3DES_EDE_CBC_SHA (rsa 1024) - D
|       TLS_RSA_WITH_AES_128_CBC_SHA (rsa 1024) - A
|       TLS_RSA_WITH_AES_256_CBC_SHA (rsa 1024) - A
|       TLS_RSA_WITH_CAMELLIA_128_CBC_SHA (rsa 1024) - A
|       TLS_RSA_WITH_CAMELLIA_256_CBC_SHA (rsa 1024) - A
|       TLS_RSA_WITH_IDEA_CBC_SHA (rsa 1024) - A
|       TLS_RSA_WITH_RC4_128_MD5 (rsa 1024) - D
|       TLS_RSA_WITH_RC4_128_SHA (rsa 1024) - D
|       TLS_RSA_WITH_SEED_CBC_SHA (rsa 1024) - A
|     compressors:
|       NULL
|     cipher preference: client
|     warnings:
|       64-bit block cipher 3DES vulnerable to SWEET32 attack
|       64-bit block cipher IDEA vulnerable to SWEET32 attack
|       Broken cipher RC4 is deprecated by RFC 7465
|       Ciphersuite uses MD5 for message integrity
|       Weak certificate signature: SHA1
|   TLSv1.2:
|     ciphers:
|       TLS_DHE_RSA_WITH_3DES_EDE_CBC_SHA (dh 1024) - D
|       TLS_DHE_RSA_WITH_AES_128_CBC_SHA (dh 1024) - A
|       TLS_DHE_RSA_WITH_AES_128_CBC_SHA256 (dh 1024) - A
|       TLS_DHE_RSA_WITH_AES_128_GCM_SHA256 (dh 1024) - A
|       TLS_DHE_RSA_WITH_AES_256_CBC_SHA (dh 1024) - A
|       TLS_DHE_RSA_WITH_AES_256_CBC_SHA256 (dh 1024) - A
|       TLS_DHE_RSA_WITH_AES_256_GCM_SHA384 (dh 1024) - A
|       TLS_DHE_RSA_WITH_CAMELLIA_128_CBC_SHA (dh 1024) - A
|       TLS_DHE_RSA_WITH_CAMELLIA_256_CBC_SHA (dh 1024) - A
|       TLS_DHE_RSA_WITH_SEED_CBC_SHA (dh 1024) - A
|       TLS_ECDHE_RSA_WITH_3DES_EDE_CBC_SHA (secp384r1) - D
|       TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA (secp384r1) - A
|       TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA256 (secp384r1) - A
|       TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256 (secp384r1) - A
|       TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA (secp384r1) - A
|       TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA384 (secp384r1) - A
|       TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384 (secp384r1) - A
|       TLS_ECDHE_RSA_WITH_RC4_128_SHA (secp384r1) - D
|       TLS_RSA_WITH_3DES_EDE_CBC_SHA (rsa 1024) - D
|       TLS_RSA_WITH_AES_128_CBC_SHA (rsa 1024) - A
|       TLS_RSA_WITH_AES_128_CBC_SHA256 (rsa 1024) - A
|       TLS_RSA_WITH_AES_128_GCM_SHA256 (rsa 1024) - A
```

```
|       TLS_RSA_WITH_AES_256_CBC_SHA (rsa 1024) - A
|       TLS_RSA_WITH_AES_256_CBC_SHA256 (rsa 1024) - A
|       TLS_RSA_WITH_AES_256_GCM_SHA384 (rsa 1024) - A
|       TLS_RSA_WITH_CAMELLIA_128_CBC_SHA (rsa 1024) - A
|       TLS_RSA_WITH_CAMELLIA_256_CBC_SHA (rsa 1024) - A
|       TLS_RSA_WITH_IDEA_CBC_SHA (rsa 1024) - A
|       TLS_RSA_WITH_RC4_128_MD5 (rsa 1024) - D
|       TLS_RSA_WITH_RC4_128_SHA (rsa 1024) - D
|       TLS_RSA_WITH_SEED_CBC_SHA (rsa 1024) - A
|     compressors:
|       NULL
|     cipher preference: client
|     warnings:
|       64-bit block cipher 3DES vulnerable to SWEET32 attack
|       64-bit block cipher IDEA vulnerable to SWEET32 attack
|       Broken cipher RC4 is deprecated by RFC 7465
|       Ciphersuite uses MD5 for message integrity
|       Weak certificate signature: SHA1
|_  least strength: D
```

Fuente: El autor, nmap

Se puede observar en la tabla anterior, que el certificado fue emitido por Comodo CA el 26 de febrero de 2016, con fecha de vencimiento para el 25 de febrero del 2017. La clave pública es de tipo RSA a 2048 bits. Su algoritmo de firma digital es sha256 con encriptación RSA.

- El servidor de SiembraViva puede ser vulnerable al ataque de POODLE. Grado cubierto C. (CVE-2014-3566)[8]. El protocolo SSL3.0, tal como se utiliza en OpenSSL a través de 1.0.1i y otros productos, utiliza un relleno de CBC no determinista, lo que facilita a los atacantes del tipo “man-in-the-

[8] Common Vulnerabilities Exposures, Ataque de POODLE CVE-2014-3566, [disponible] https://cve.mitre.org/cgi-bin/cvename.cgi?name=cve-2014-3566

middle” obtener datos de texto claro a través de un ataque “padding-oracle”.

- El servidor de SiembraViva puede ser vulnerable a la vulnerabilidad de Oracle “OpenSSL Padding” (CVE-2016-2107)[9] e inseguro. Grado establecido en F.

  La implementación de AES-NI en OpenSSL antes de 1.0.1t y 1.0.2 antes de 1.0.2h no considera la asignación de memoria durante una comprobación de relleno determinada, lo que permite a los atacantes remotos obtener información de texto sensible a través de un ataque de padding-oracle contra una sesión AES CBC. Esta vulnerabilidad existe debido a una actualización incorrecta para CVE-2013-0169.

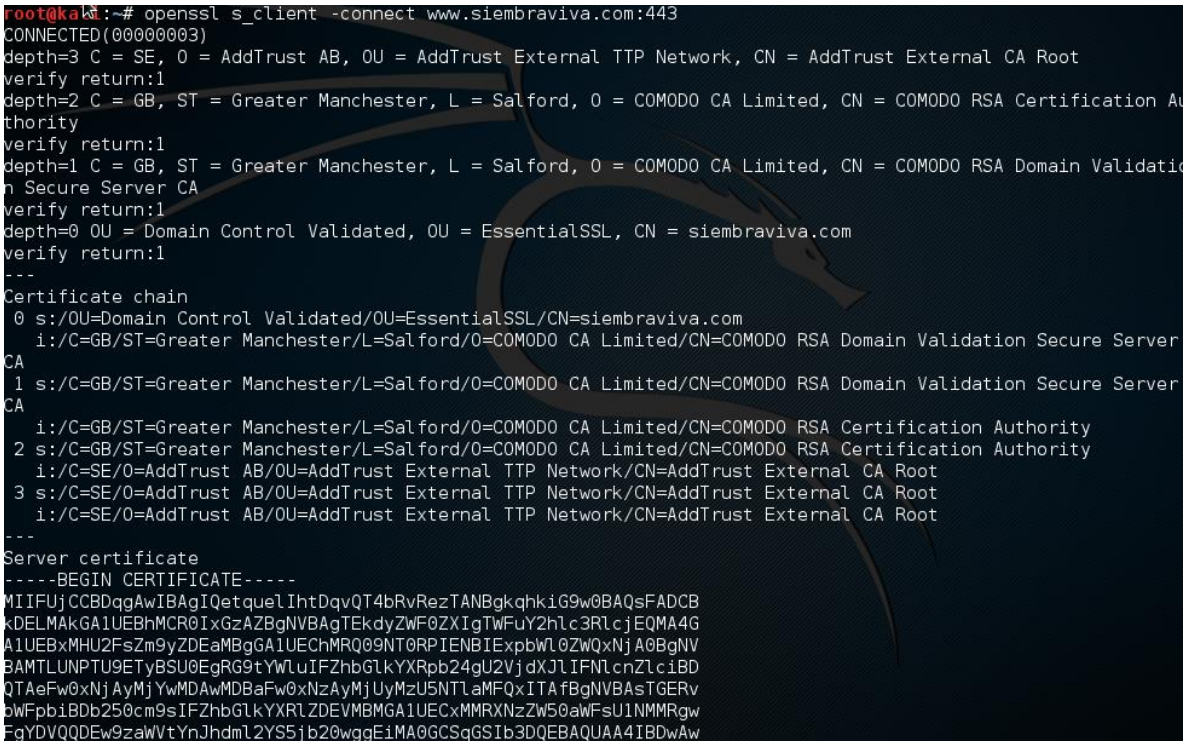
- El servidor acepta cifrado RC4[10], pero sólo con versiones de protocolo anteriores. Grado cubierto B.

- El servidor no admite el secreto directo con los navegadores de referencia.[11]

Auditoría manual de niveles de cifrado SSL débiles mediante OpenSSL. Conexión mediante consola a SiembraViva.com con SSLv2.
openssl s_client -connect www.siembraviva.com:443

[9] Common Vulnerabilities Exposures, Vulnerabilidad Oracle OpenSSL CBC Ciphersuites CVE-2014-3566, [disponible] https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2016-2107
[10] SSL Labs, Ivan Ristic RC4 in TLS is Broken: Now What? [disponible] https://blog.qualys.com/ssllabs/2013/03/19/rc4-in-tls-is-broken-now-what
[11] SSL Labs, Ivan Ristic. SSL Labs: Deploying Forward Secrecy [disponible] https://blog.qualys.com/ssllabs/2013/06/25/ssl-labs-deploying-forward-secrecy

Figura 34. Auditoria cifrado SSL débiles mediante OpenSSL



Fuente: El autor, nmap

En la figura 34 está la auditoría de cifrado SSL débil mediante OpenSSL, el probador intenta iniciar una renegociación por el cliente [m] conectándose al servidor con openssl. El probador escribe la primera línea de una solicitud HTTP y escribe "R" en una nueva línea. A continuación, espera la renegociación y la finalización de la solicitud HTTP y comprueba si se admite la renegociación segura mirando la salida del servidor. Utilizando estas solicitudes manuales es posible ver si la Compresión está habilitada para TLS, para cifrado y otras vulnerabilidades.

Tabla 8. Resultado completo auditoria cifrado SSL débiles mediante OpenSSL

Auditoria SSL débiles mediante OpenSSL
root@kali:~# openssl s_client -connect www.siembraviva.com:443 CONNECTED(00000003)

depth=3 C = SE, O = AddTrust AB, OU = AddTrust External TTP Network, CN = AddTrust External CA Root
verify return:1
depth=2 C = GB, ST = Greater Manchester, L = Salford, O = COMODO CA Limited, CN = COMODO RSA Certification Authority
verify return:1
depth=1 C = GB, ST = Greater Manchester, L = Salford, O = COMODO CA Limited, CN = COMODO RSA Domain Validation Secure Server CA
verify return:1
depth=0 OU = Domain Control Validated, OU = EssentialSSL, CN = siembraviva.com
verify return:1

Certificate chain
 0 s:/OU=Domain Control Validated/OU=EssentialSSL/CN=siembraviva.com
   i:/C=GB/ST=Greater Manchester/L=Salford/O=COMODO CA Limited/CN=COMODO RSA Domain Validation Secure Server CA
 1 s:/C=GB/ST=Greater Manchester/L=Salford/O=COMODO CA Limited/CN=COMODO RSA Domain Validation Secure Server CA
   i:/C=GB/ST=Greater Manchester/L=Salford/O=COMODO CA Limited/CN=COMODO RSA Certification Authority
 2 s:/C=GB/ST=Greater Manchester/L=Salford/O=COMODO CA Limited/CN=COMODO RSA Certification Authority
   i:/C=SE/O=AddTrust AB/OU=AddTrust External TTP Network/CN=AddTrust External CA Root
 3 s:/C=SE/O=AddTrust AB/OU=AddTrust External TTP Network/CN=AddTrust External CA Root
   i:/C=SE/O=AddTrust AB/OU=AddTrust External TTP Network/CN=AddTrust External CA Root
---
Server certificate
-----BEGIN CERTIFICATE-----
MIIFUjCCBDqgAwIBAgIQetquelIhtDqvQT4bRvRezTANBgkqhkiG9w0BAQsFADCB
kDELMAkGA1UEBhMCR0IxGzAZBgNVBAgTEkdyZWF0ZXIgTWFuY2hlc3RlcjEQMA4G
A1UEBxMHU2FsZm9yZDEaMBgGA1UEChMRQ09NT0RPIENBIExpbWl0ZWQxNjA0BgNV
BAMTLUNPTU9ETyBSU0EgRG9tYWluIFZhbGlkYXRpb24gU2VjdXJlIFNlcnZlciBD
QTAeFw0xNjAyMjYwMDAwMDBaFw0xNzAyMjUyMzU5NTlaMFQxITAfBgNVBAsTGERv
bWFpbiBDb250cm9sIFZhbGlkYXRlZDEVMBMGA1UECxMMRXNzZW50aWFsU1NMMRgw

FgYDVQQDEw9zaWVtYnJhdml2YS5jb20wggEiMA0GCSqGSIb3DQEBAQUAA4I
BDwAw
ggEKAoIBAQDSIjKendmUHrRJ7gLIRnKgGshlp4zXx7G+OhTWk9faf7z28BbEImR
F
tA4Kmy+QpYgxQRbuWnpqJN5s1V/aLCqE0Tag+e1kHDFkR6nZtqkqamTz/P0ci/cx
+QcEMPW4p+sfLDzygwCU9qyrTIUigM2RY+a2aWQHIZAVtgDoGG99ZvxbukCv2
Qk2
UeEBx3u0+LT7b99caEIWK7ybgzmGkg4NmjPl6cx3DL3jzTuvZrU+1Q5gxpmdoT+
R
DdS7joYiWQ9t2Zaj25m5GhzlTsQxdF3n9uZ2He84bxRX/gecxgEgNTNB6WwyKxv
R
VVLkIw2qQOF76AYuSnwhd/2irjE6+beLAgMBAAGjggHhMIIB3TAfBgNVHSMEGD
AW
gBSQr2o6lFoL2JDqElZz30O0Oija5zAdBgNVHQ4EFgQUcA3yu65SoNT5nMteqjV
e
2mgqTpwwDgYDVR0PAQH/BAQDAgWgMAwGA1UdEwEB/wQCMAAwHQYDVR
0lBBYwFAYI
KwYBBQUHAwEGCCsGAQUFBwMCME8GA1UdIARIMEYwOgYLKwYBBAGyMQ
ECAgcwKzAp
BggrBgEFBQcCARYdaHR0cHM6Ly9zZWN1cmUuY29tb2RvLmNvbS9DUFMwCA
YGZ4EM
AQIBMFQGA1UdHwRNMEswSaBHoEWGQ2h0dHA6Ly9jcmwuY29tb2RvY2EuY2
9tL0NP
TU9ET1JTQURvbWFpblZhbGlkYXRpb25TZWN1cmVTZXJ2ZXJDQS5jcmwwgYU
GCCsG
AQUFBwEBBHkwdzBPBggrBgEFBQcwAoZDaHR0cDovL2NydC5jb21vZG9jYS5jb
20v
Q09NT0RPUlNBRG9tYWluVmFsaWRhdGlvblNlY3VyZVNlcnZlckNBLmNydDAkBg
gr
BgEFBQcwAYYYaHR0cDovL29jc3AuY29tb2RvY2EuY29tMC8GA1UdEQQoMCa
CD3Np
ZW1icmF2aXZhLmNvbYITd3d3LnNpZW1icmF2aXZhLmNvbTANBgkqhkiG9w0BA
QsF
AAOCAQEAfoql4GO8WldMNIZD4XPfv89lDxwF5yjAyLFbi8oqiTSJgDNTJpFhbtWJ
Pkmw3LJaHzz9OGhz19EC6GxE9OMuac1XI7J3gDQVGsswlNmx2z79/Rq6mu2qJ
tpc
Xiz/iODmrGBd5n8w2BsbXdHHO0a/xIukfCNPafLg8P5yc3SPT/ZmAV+yQTwNct5E
2ega55ma1RANOW+smc7qhp3vhJwdmrHFtAtceJD/tYyfGru9TOS+rCntCgmDPzc
t
Xx7kzawHFgpNwd9NiFw89gL7M0o45C8qKgxLP2TWqDkKkpiR660itqAIvrqEvie/
ejOECFB4zC38Lrqdr/VHJs6blONeag==
-----END CERTIFICATE-----
subject=/OU=Domain Control Validated/OU=EssentialSSL/CN=siembraviva.com

```
issuer=/C=GB/ST=Greater Manchester/L=Salford/O=COMODO CA
Limited/CN=COMODO RSA Domain Validation Secure Server CA
---
No client certificate CA names sent
Peer signing digest: SHA512
Server Temp Key: ECDH, P-256, 256 bits
---
SSL handshake has read 6096 bytes and written 441 bytes
---
New, TLSv1/SSLv3, Cipher is ECDHE-RSA-AES256-GCM-SHA384
Server public key is 2048 bit
Secure Renegotiation IS supported
Compression: NONE
Expansion: NONE
No ALPN negotiated
SSL-Session:
    Protocol: TLSv1.2
    Cipher: ECDHE-RSA-AES256-GCM-SHA384
    Session-ID:
207B4E1703C98C631FEFDE244CD0D1ECF77E171E25805B8661C9AAB6B2AC
433D
    Session-ID-ctx:
    Master-Key:
97C20D6DDEDC6F377F864136FED435AF78A0BAA987AB3D7C86D0DC238698
4BC3A616C16519F6D2710464CCEBFBCA88D5
    Key-Arg: None
    PSK identity: None
    PSK identity hint: None
    SRP username: None
    TLS session ticket lifetime hint: 300 (seconds)
    TLS session ticket:
    0000 - ec ca b0 1b fa c8 8e 76-58 83 c4 79 b0 a3 6f 5c   .......vX..y..o\
    0010 - 20 8e fa 1d 0d 93 18 b3-10 e4 2b 75 68 0f e2 50    .........+uh..P
    0020 - 26 27 bc 16 31 41 4f 94-85 e3 a0 2b 1b d8 20 e9   &'..1AO....+.. .
    0030 - eb 82 23 0e 4b 85 7c 4b-d5 dd 10 a8 18 72 f1 80   ..#.K.|K.....r..
    0040 - ed 18 a3 1e 61 61 fd 3d-55 06 c1 3b ab 43 1d 42   ....aa.=U..;.C.B
    0050 - 2f 4e 6e 99 91 51 6a a6-74 79 3d d7 65 39 2e 18   /Nn..Qj.ty=.e9..
    0060 - df 30 66 9f 0b f5 0a 73-d0 10 bd e1 28 65 74 01   .0f....s....(et.
    0070 - f7 40 9c 2d cf c9 e9 75-b4 22 52 0e 38 40 ff 3b   .@.-...u."R.8@.;
    0080 - 8b c1 20 ee fe 02 53 9b-ed 2d 4a 0b e1 7b 3f 15   .. ...S..-J..{?.
    0090 - 9a 11 a4 ab 95 b0 0e ad-03 a8 20 c2 97 8a 9f 00   .......... .....
    00a0 - 7d 85 07 20 76 e3 0b f5-1e bc 58 f2 a0 da fe 2f   }.. v.....X..../
    00b0 - 34 ec 02 84 52 59 ba a8-b6 e1 73 38 87 64 f8 53   4...RY....s8.d.S
```

Start Time: 1481439272 Timeout: 300 (sec) Verify return code: 0 (ok)
Fuente: El autor, nmap

7.3.2. Pruebas del receptor de escucha de la BD (OWASP-CM-002).

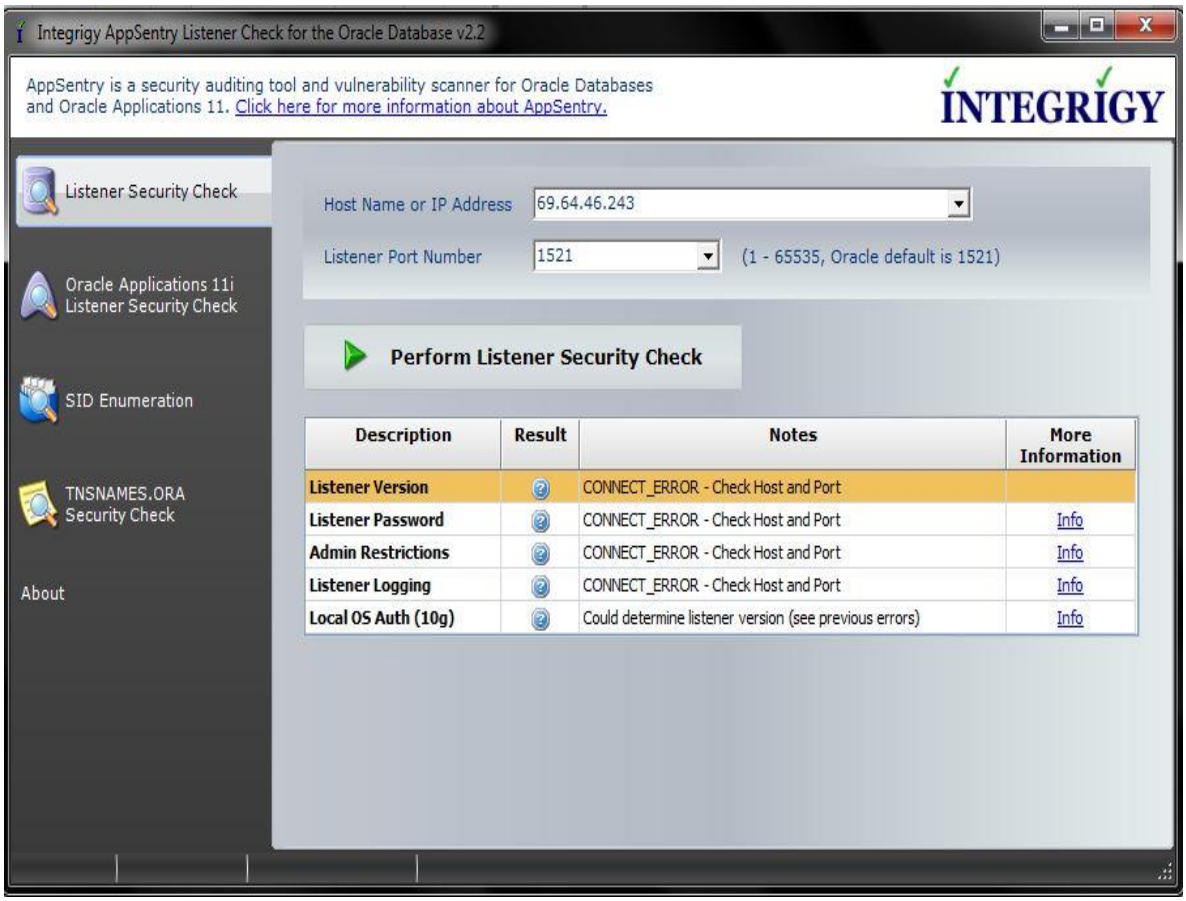
Los servidores de bases de datos cuentan con maneras de ser administrados de manera remota que facilitan el proceso, en el caso de Oracle se cuenta con un puerto de escucha para su administración. El puerto que se tiene por defecto es el 1521, con la reciente actualización se tiene que el puerto cambio a ser el 2483 y el 2484 con SSL.

La función principal de la aplicación Integrigy es atender las peticiones externas de administración y estar constantemente a la espera de que dichas peticiones se den. De tener éxito la conexión remota se pueden dar los siguientes ataques:

- Proceder a hacer un ataque de denegación de servicio.
- Secuestrar la base de datos, se cambia la contraseña y no se notifica al administrador de dicho cambio.
- Revelar la información de la base de datos.
- Obtener la información del servidor, sobre el receptor y las bases de datos que se encuentren almacenada.

Para evaluar que el receptor está o no asequible para el usuario se descarga del programa *integrigy* y se hace una evaluación de si el receptor se encuentra habilitado:

Figura 35. Prueba de receptor en base de datos



Fuente: El autor, Integrigy AppSentry Listener Checkfor Oracle

En la figura 35 se realizó la prueba de receptor en base de datos, analizando la información arrojada por la aplicación se encuentra que no se puede tener acceso al receptor de la base de datos, no hay acceso de manera remota si no se cuenta con los datos necesarios y los permisos por parte del administrador para realizar este tipo de procedimiento en el servidor.

De esta manera se puede concluir que la página web tiene controles de seguridad correctamente establecidos en dicho aspecto.

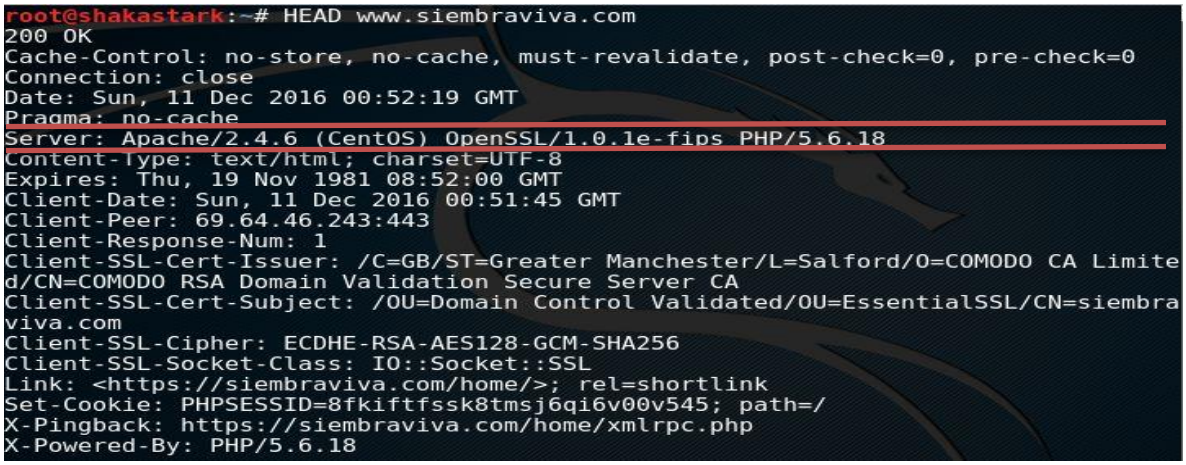
7.3.3. Pruebas de gestión de configuración de la infraestructura (OWASP-CM-003).

La infraestructura bajo la cual está implementada el servidor Web es un factor importante a tener en cuenta para que no haya vulnerabilidades que puedan afectar el sistema; no importa el número de vulnerabilidades de darse alguna dentro del sistema esto pone en entredicho las medidas de seguridad tomadas.

Las aplicaciones dentro del servidor, las bases de datos contenidas dentro del servidor y servidores que contribuyan a la autenticación, si estos componentes de la infraestructura no cuentan con las medidas de seguridad pertinentes pueden incrementar el riesgo a los diferentes problemas que afectan el sistema.

Para establecer la información relevante de las aplicaciones y medidas de seguridad que están siendo implementadas dentro de una máquina de prueba se ejecuta el siguiente comando, cabe aclarar que debe ser un entorno Linux para poder llevar a cabo el proceso:

Figura 36. Comando HEAD



Fuente: El autor

Como resultado del análisis anterior (figura 36), se detallan las siguientes características:

- Servidor Apache/2.4.6

- El sistema operativo instalado es un Linux, la distribución es CentOS
- Para la comunicación remota con el servidor se tiene instalado OpenSSL 1.0.1
- La versión de PHP es la 5.6.18

Con los datos anteriores se pueden buscar vulnerabilidades que puedan afectar el sistema, los hallazgos también contribuyen a mejorar la seguridad y encontrar medidas para mitigar los riesgos o usar actualizaciones para que no haya más problemas en el sistema.

Un comando que ayuda a consultar que versión se tiene instalada del servicio ftp y que software lo maneja es el comando FTP, ver los resultados del comando en la figura 37.

Figura 37. vsFTPd 3.0.2

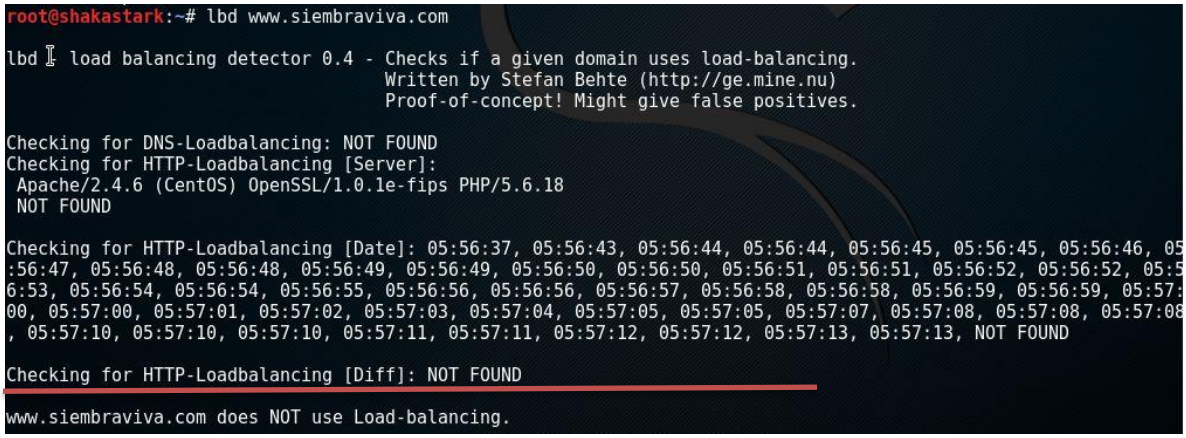
Fuentes. El autor

Conociendo esta información se puede buscar en internet las vulnerabilidades que afecten este servicio, conociendo la versión es más sencillo indagar que bugs de seguridad aun no cuentan con la solución y de darse esta situación las personas sin escrúpulos pueden explotar estos fallos y afectar el sistema.

Como una medida de seguridad importante es necesario configurar el servidor para que oculte la mayor cantidad de información a las personas que la solicite sin autorización, por ejemplo, es importante ocultar las versiones de los aplicativos instalados dentro del servidor y ocultar los servidores y los puertos que usan.

Otro procedimiento es consultar el uso de nivelador de cargas del servidor, en un sistema Linux se puede realizar con el procedimiento LBD ejecutado en la figura 38.

Figura 38. Comando Balanceador de Cargas



Fuente. Autor.

Se evidencia un hallazgo muy importante, el sitio web SiembraViva.com no cuenta con un balanceador de cargas. Los balanceadores de cargas son de tres tipos y ninguno de los tipos se encuentra implementado.

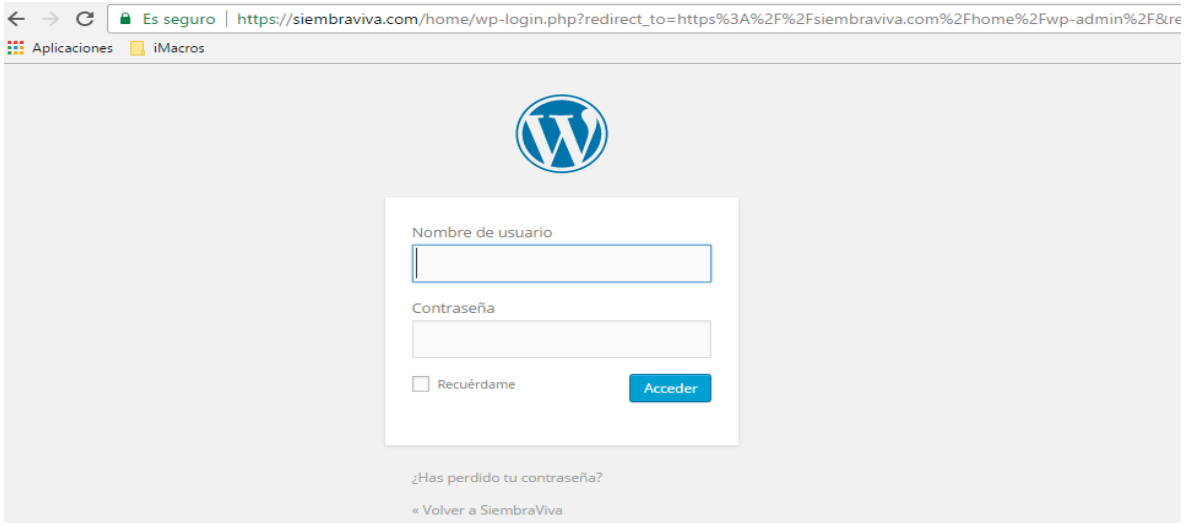
EL análisis que se realizó al servidor de la aplicación arrojo que corre una distribución CentOS de Linux, no se sabe con certeza la versión del sistema operativo, pero esto no evita que se indague cuáles son las vulnerabilidades vigentes que afectan el sistema y también consultar cuales han salido últimamente que puedan afectar el sistema.

En la revisión de las interfaces de administración empleadas para gestionar las diferentes partes de la arquitectura del sitio web, se encuentra que el sitio web SiembraViva.com trabaja bajo el sistema de gestión de contenidos Wordpress en su versión tentativa 4.2.2, según los datos arrojados por la herramienta Wappalyzer de la figura 39 y 40.

Figura 39. Sistema de gestión de contenidos Wordpress

Fuente: Autor, Wappalyzer.

Figura 40. Panel de administración wp-admin



Fuente: Autor.

7.3.4. Pruebas de gestión de configuración de la aplicación (OWASP-CM-004).

La configuración adecuada de todos y cada uno de los elementos que componen la arquitectura de una aplicación es importante, de cara a evitar errores que podrían comprometer la seguridad de la arquitectura al completo.

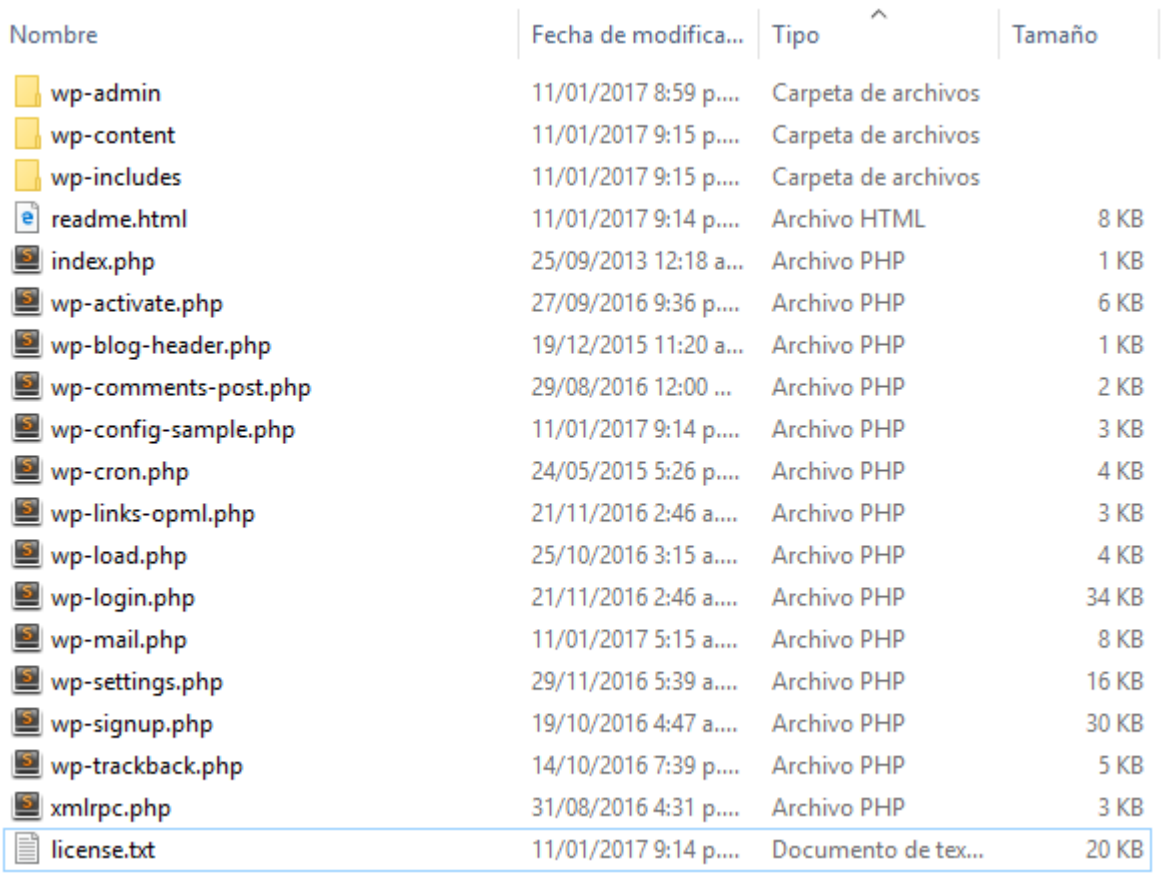
La revisión y prueba de configuraciones es una tarea crítica a la hora de crear y mantener un tipo de arquitectura así, ya que generalmente se disponen muchos sistemas diferentes con configuraciones genéricas que podrían no estar adecuadas a la tarea que realizarán en el sitio web específico en que están instalados. Aunque la instalación típica de servidores web y de aplicaciones incluye muchas funcionalidades (como ejemplos de aplicación, documentación y páginas de prueba), aquello que no es esencial para la aplicación debería ser eliminado antes de la implementación, para evitar explotaciones post-instalación.

7.3.4.1. Archivos y directorios conocidos.

Muchos servidores web y aplicaciones proporcionan información de una instalación predeterminada, archivos y aplicaciones de muestra que son provistas para beneficio del desarrollador, y con el fin de comprobar que el servidor está funcionando adecuadamente tras la instalación.

Los Scanners de CGI incluyen una lista detallada de archivos conocidos y directorios de muestra que son provistos por diferentes servidores web o aplicaciones, es un método rápido para determinar si estos archivos están presentes. No obstante, la única manera de estar realmente seguro es una revisión completa de los contenidos del servidor web o de la aplicación y determinar si están relacionados a la propia aplicación.

Figura 41. Estructura de ficheros y directorios Wordpress



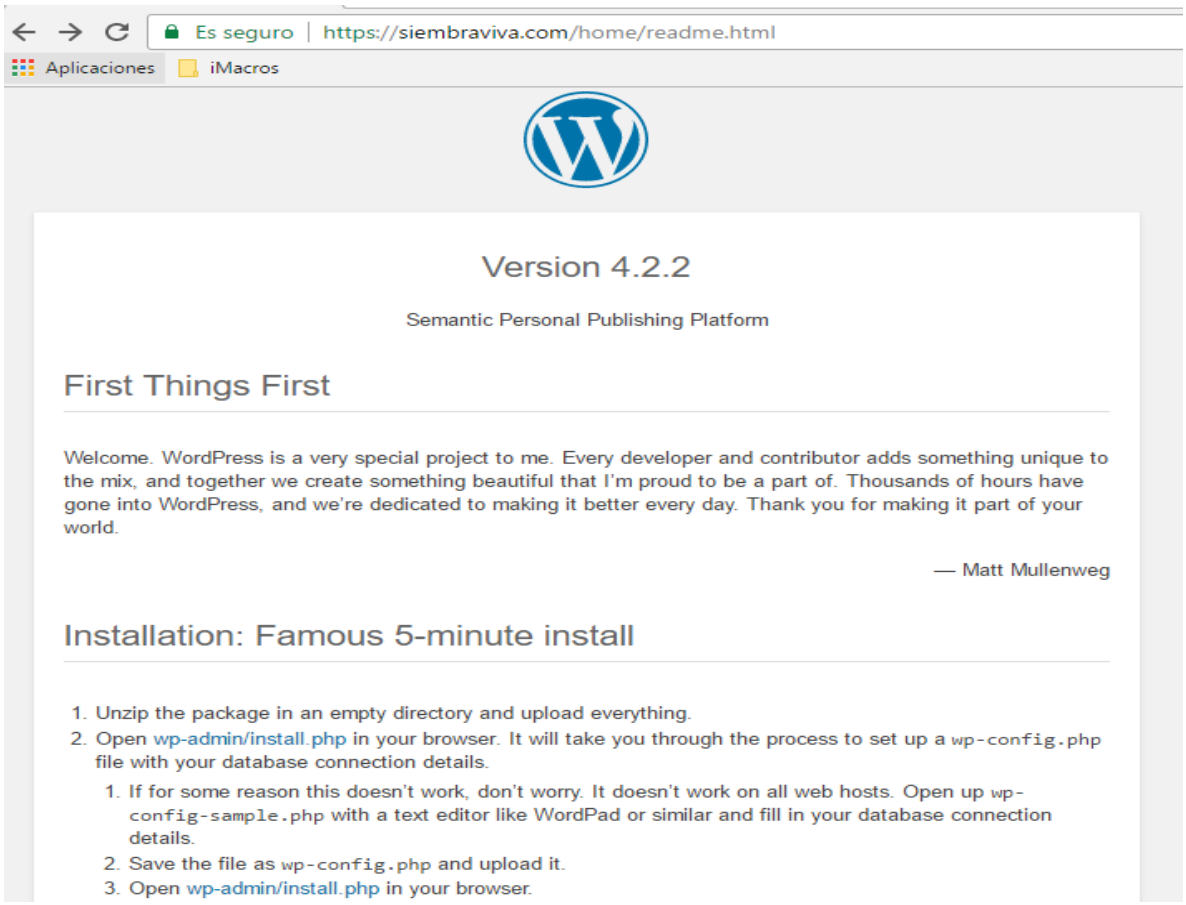
Nombre	Fecha de modifica...	Tipo	Tamaño
wp-admin	11/01/2017 8:59 p....	Carpeta de archivos	
wp-content	11/01/2017 9:15 p....	Carpeta de archivos	
wp-includes	11/01/2017 9:15 p....	Carpeta de archivos	
readme.html	11/01/2017 9:14 p....	Archivo HTML	8 KB
index.php	25/09/2013 12:18 a...	Archivo PHP	1 KB
wp-activate.php	27/09/2016 9:36 p....	Archivo PHP	6 KB
wp-blog-header.php	19/12/2015 11:20 a...	Archivo PHP	1 KB
wp-comments-post.php	29/08/2016 12:00 ...	Archivo PHP	2 KB
wp-config-sample.php	11/01/2017 9:14 p....	Archivo PHP	3 KB
wp-cron.php	24/05/2015 5:26 p....	Archivo PHP	4 KB
wp-links-opml.php	21/11/2016 2:46 a....	Archivo PHP	3 KB
wp-load.php	25/10/2016 3:15 a....	Archivo PHP	4 KB
wp-login.php	21/11/2016 2:46 a....	Archivo PHP	34 KB
wp-mail.php	11/01/2017 5:15 a....	Archivo PHP	8 KB
wp-settings.php	29/11/2016 5:39 a....	Archivo PHP	16 KB
wp-signup.php	19/10/2016 4:47 a....	Archivo PHP	30 KB
wp-trackback.php	14/10/2016 7:39 p....	Archivo PHP	5 KB
xmlrpc.php	31/08/2016 4:31 p....	Archivo PHP	3 KB
license.txt	11/01/2017 9:14 p....	Documento de tex...	20 KB

Fuente: Autor, Wordpress.org.

La estructura de ficheros y directorios de Wordpress maneja un estándar para almacenar los archivos de multimedia (ver figura 41), plugins, temas y lenguajes. Esta información suele proporcionar detalles que pueden ser utilizados en la recolección de información e identificación del sistema de gestión de contenidos. Una forma rápida es verificando el fichero conocido como readme.html que proporciona la versión de la página de Wordpress, este archivo se distribuye como parte de los archivos de instalación del núcleo de Wordpress y puede ser observado en la figura 42.

<meta name="generator" content="WordPress 4.2.2" />

Figura 42. Archivo Readme febrero 2018



Fuente: Autor, SiembraViva.com/home/readme.html.

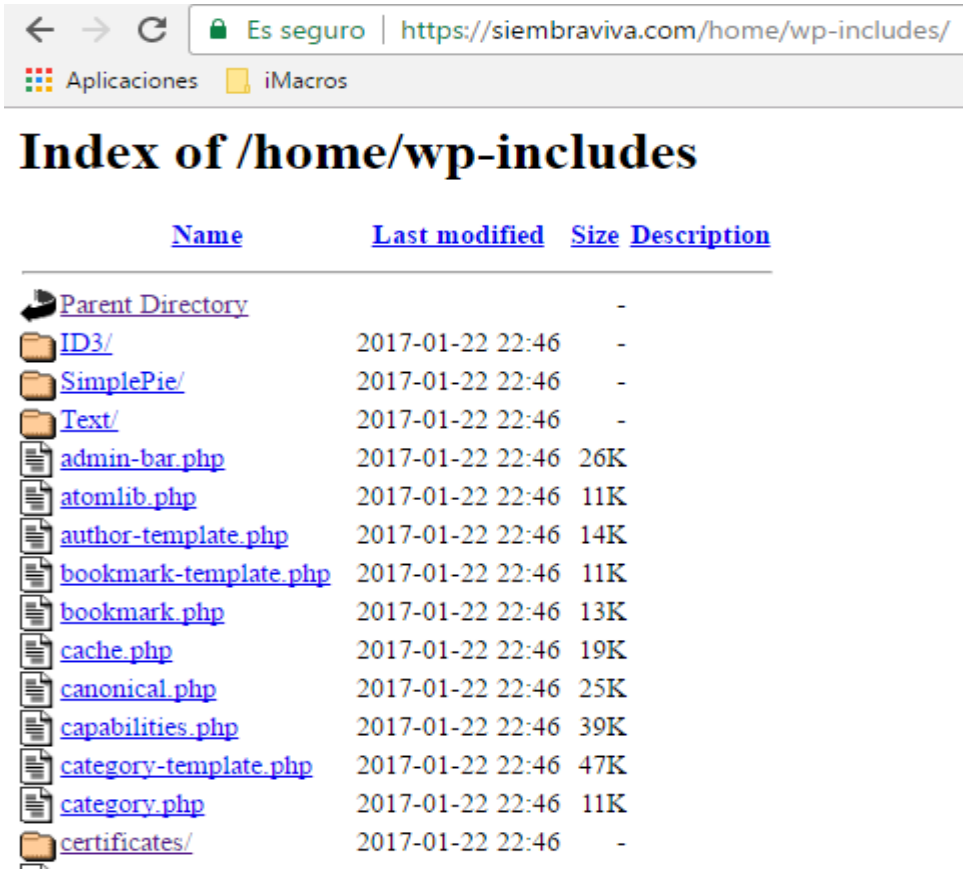
7.3.4.2 Indexación de directorios

La indexación de directorios es una función del servidor web que permite ver el contenido de ciertos directorios en la ruta de acceso web desde el navegador, en la figura 43 es posible ver el contenido de un directorio, en específico permite que un usuario no autorizado recopile gran cantidad de información de la instalación, tales como plugins, themes, lenguajes, archivos multimedia y demás información que podría ser sensible.

Al navegar por los diferentes directorios se logró tener respuesta de los siguientes índices con información sensible:

/wp-content/languages/
/wp-content/languages/plugins/
/wp-content/languages/themes/
/wp-content/plugins/contact-form-7/
/wp-includes/
/wp-includes/certificates/
/wp-includes/fonts/
/wp-includes/css/
/wp-includes/images/
/wp-admin/css/
/wp-admin/images/
/wp-admin/js/
/wp-admin/maint/

Figura 43. Indexación directorio wp-includes



Fuente: Autor, SiembraViva.com/home/wp-includes/

7.3.4.3. Revisión de comentarios.

Es muy común e incluso recomendado en los programadores, incluir los comentarios en el código fuente que permita a otros programadores comprender mejor el código cuando se requiere realizar alguna corrección o implementación de cambios. Sin embrago, los cometarios incluidos en el código HTML pueden revelar información interna sensible, que un atacante potencial podría utilizar para vulnerar dicho sistema.

La revisión de comentarios se realiza para determinar si está siendo visible cualquier tipo de información a través de los comentarios del código fuente del sitio web. Esta revisión se realiza minuciosamente mediante un análisis del contenido estático y dinámico del sitio web, y a través de la búsqueda de archivos, directorios y enlaces.

Se realiza la revisión del código fuente de la página principal https://siembraviva.com/home/ encontrando los siguientes hallazgos:

En la revisión del código fuente de la aplicación web siembraviva.com se observa que no se encuentra visible ningún tipo de información, solo se encuentran comentarios de los plugin usados en la implementación. Comentarios sensibles como contraseñas de las aplicaciones, contraseñas internas de la aplicación como administradores, la aplicación no cuenta con comentarios sensibles para la seguridad.

En cuento a los registros de logs no se puede establecer como es el manejo interno que se le da, esta información según el desarrollador es sensible y no debe conocerse como están siendo administrados dentro del servidor y en qué lugar se almacenan para su procesamiento.

Comentarios con versión de plugins.

En la figura 44, línea 15 del código fuente de la página principal, se encuentra un comentario HTML con información del plugin Yoast SEO, en este comentario se evidencia la versión actual del plugin versión 2.3.5, siendo una información generada automáticamente por el plugin y puede ser sensible para la obtención de información de los diversos complementos que funcionan en el sitio web SiembraViva.com.

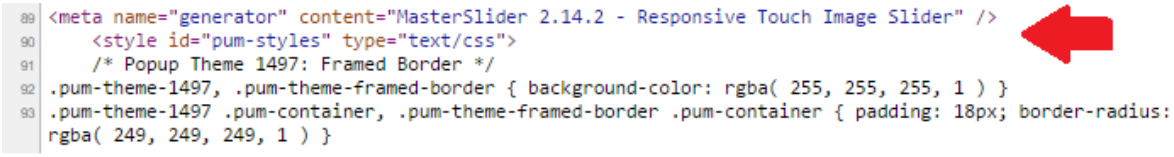
Figura 44. Comentario versión plugin Yoast SEO v2.3.5

```
<!-- This site is optimized with the Yoast SEO plugin v2.3.5 -
https://yoast.com/wordpress/plugins/seo/ -->
<link rel="canonical" href="https://siembraviva.com/home/" />
<meta property="og:locale" content="es_ES" />
<meta property="og:type" content="website" />
<meta property="og:title" content="Inicio | SiembraViva" />
<meta property="og:url" content="https://siembraviva.com/home/" />
<meta property="og:site_name" content="SiembraViva" />
<meta name="twitter:card" content="summary"/>
<meta name="twitter:title" content="Inicio | SiembraViva"/>
<meta name="twitter:site" content="@SiembraViva"/>
<meta name="twitter:domain" content="SiembraViva"/>
<meta name="twitter:creator" content="@SiembraViva"/>
<script type='application/ld+json'>
{"@context":"http:\/\/schema.org","@type":"WebSite","url":"https:\/\/siembravi
va.com\/home\/","name":"Siembraviva | Mercado de comida
saludable","alternateName":"Mercado Online de comida saludable en
Medell\u00edn","potentialAction":
{"@type":"SearchAction","target":"https:\/\/siembraviva.com\/home\/?s=
{search_term_string}","query-input":"required name=search_term_string"}}
</script>
<script type='application/ld+json'>
{"@context":"http:\/\/schema.org","@type":"Organization","url":"https:\/\/siem
braviva.com\/home\/","sameAs":["https:\/\/www.facebook.com\/siembraviva?
fref=ts","https:\/\/instagram.com\/siembraviva\/","https:\/\/www.youtube.com\/
channel\/UCzgXNgadU4S0pzyUrGWPs3g","https:\/\/twitter.com\/SiembraViva"],"name
":"SiembraViva","logo":"http:\/\/newhtml.siembraviva.com\/svhome\/wp-
content\/uploads\/logo.png"}</script>
<meta name="alexaVerifyID" content="EuOn3PUofT6b2hSdDZbAyT49ekg" />
<meta name="google-site-verification"
content="_kL0K1fNghb0X8Fm6x3UjN6Pn_071d35EVxZX85AmUw" />
<!-- / Yoast SEO plugin. -->
```

Fuente: Autor, SiembraViva.com/home/

En la línea 89 de la figura 45 se encuentra metainformación del plugin MasterSlider - Responsive Touch Image Slider, en donde se evidencia la versión utilizada del plugin 2.14.2. Esta información es generada automáticamente por el plugin y puede ser sensible para la obtención de información de los diversos complementos que funcionan en el sitio web.

Figura 45. Comentario versión MasterSlider 2.14.2 - Responsive Image Slider



Fuente: Autor, SiembraViva.com/home/

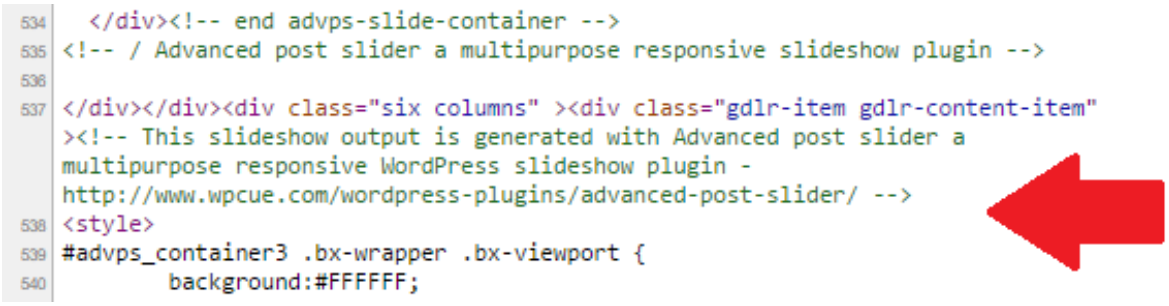
En la línea 135 de la figura 46 se encuentra un comentario HTML con información del tema Lawyer Base Wordpress versión: 1.01, en donde se evidencia la versión utilizada de esta plantilla.

Figura 46. Comentario versión tema Lawyer Base Wordpress 1.01

Fuente: Autor, SiembraViva.com/home/

En la línea 537 de la figura 47 se encuentra un comentario HTML con información del plugin Advanced post slider, en donde se evidencia la utilización de la versión 2.5.0.

Figura 47. Comentario versión plugin Advanced post slider 2.5.0



Fuente: Autor, SiembraViva.com/home/

En la línea 1488 de la figura 48 se encuentra un comentario de JavaScript con información que nos permite identificar que utilizan el plugin Contact Form 7, en donde se evidencia la utilización de la versión 4.3.

Figura 48. Comentario versión plugin Contact Form 7 versión 4.3

Fuente: Autor, SiembraViva.com/home/

En la línea 1498 de la figura 49 se encuentra metainformación del plugin llamado Display Posts Grid, List Without Coding - Content Views.

Figura 49. Comentario versión plugin Display Posts Grid

Fuente: Autor, SiembraViva.com/home/

En la línea 1526 de la figura 50 se encuentra metainformación del plugin llamado Popup Maker, en donde se evidencia la utilización de la versión 1.4.11.

Figura 50. Comentario versión plugin Popup Maker versión 1.4.11

Fuente: Autor, SiembraViva.com/home/

7.3.5. Gestión de extensiones de archivo (OWASP-CM-005).

Las extensiones de archivo son utilizadas comúnmente en los servidores web para determinar fácilmente que tecnologías lenguajes/plugins deben ser utilizados para responder a las peticiones web. A pesar de que este comportamiento es consistente con los RFCs y estándares web, el uso de extensiones de archivo estándar proporciona a quien realiza las pruebas de intrusión información de utilidad acerca de las tecnologías de base utilizadas en una aplicación web, y simplifica mucho la tarea de determinar el escenario de ataque a utilizar sobre tecnologías específicas. Adicionalmente, una configuración inadecuada en los servidores web podría revelar fácilmente información confidencial sobre credenciales de acceso.

Buscamos la existencia de archivos conocidos que puedan contener información sensible, o archivos que no tienen ninguna razón para estar en el servidor web:

Tabla 9. Búsqueda de archivos conocidos

Extensión de archivo	Estado
.inc	Sin evidencia
.sql	Sin evidencia
.asa	Sin evidencia
.zip	Sin evidencia
.tar	Sin evidencia
.gz	Sin evidencia
.tgz	Sin evidencia

Extensión de archivo	Estado
.rar	Sin evidencia
.java	Sin evidencia
.py	Sin evidencia
.txt	Sin evidencia
.doc	Sin evidencia
.docx	Sin evidencia
.rtf	Sin evidencia
.pdf	Sin evidencia
.xls	Sin evidencia
.ppt	Sin evidencia
.bak	Sin evidencia
.old	Sin evidencia
Fuente: El autor	

7.3.6. Archivos antiguos, copias de seguridad y sin referencias (OWASP-CM-006).

Es común que los archivos que se encuentren alojados en el servidor sea gestionados por el servidor, algunos archivos no cuentan con la debida documentación o referencias; este error puede posibilitar que se conozca información sensible sobre características de la infraestructura. Dentro del código fuente de las páginas web se tienen versiones desactualizadas de algunos complementos que son documentados para llevar un control de cambios y facilitar la tarea de los desarrolladores de depurar el código o conocer algunas características del producto, el funcionamiento del sistema o aprovechar vulnerabilidades con el uso de puertas traseras.

Las tares automatizadas en los servidores puede generar archivos que pueden generar vulnerabilidades; tareas como copias de seguridad, dejar archivos obsoletos en el árbol de directorios o archivos creados por actualizaciones dentro del servidor. Las copias que se realizan pueden generar extensiones no deseadas, no relacionadas con los archivos originales.

Un ejemplo de este tipo son los archivos tales como login.php, al hacer una modificación en el código fuente de este archivo se va a generar otro de tipo login.php.old después de ser actualizado; el archivo login.php solo puede ser leído por el servido, por el contrario login.php.old de haber una copia en el servidor y ser accedido, puede ser abierto y visualizarse como un documento plano con el navegador. De conocerse lo contenido de un archivo importante es posible conocer la lógica del negocio o información relevante que puede exponer la aplicación a ataques.

Existen otro tipo de archivos que solo deben ser visualizados o accedidos por la aplicación, no deben ser accedidos o manipulados por el usuario; por ejemplo, archivos de configuración, archivos de datos, registros, etc.

Contramedidas:

- No se deben editar los archivos dentro del mismo servidor, los editores generan copias de seguridad que pueden ser alojadas dentro del servidor y que sean visibles para los usuarios.
- Al realizar tareas de backup se puede caer en el error de comprimirlos en extensiones tales con .zip o .tar; este tipo de extensiones son potencialmente peligrosas y no deben estar alojadas en el servidor.
- La buena gestión de archivos garantiza que existen referencias o ficheros obsoletos dentro de la aplicación.
- Se debe configurar la aplicación para que no cree archivos y que estos sean almacenados en el servidor, al ser visibles para los usuarios van a ser manipulados o explotados como vulnerabilidad.

La auditoría aplicada sobre la página siembraviva.com no permite realizar pruebas al servidor por motivo de la disponibilidad del sitio, pero el desarrollador dice que la administración de los archivos se realiza con el administrador de versiones tortoiseSVN para la administración de versiones y el control de archivos dentro del servidor.

Dentro de la aplicación el usuario no interactúa el servidor con ningún tipo de archivo, lo cual contribuye a que la pagina no se vea amenazada a ataques que se puedan filtrar al sistema internamente. El sistema se encuentra bien protegido para ataques de este tipo.

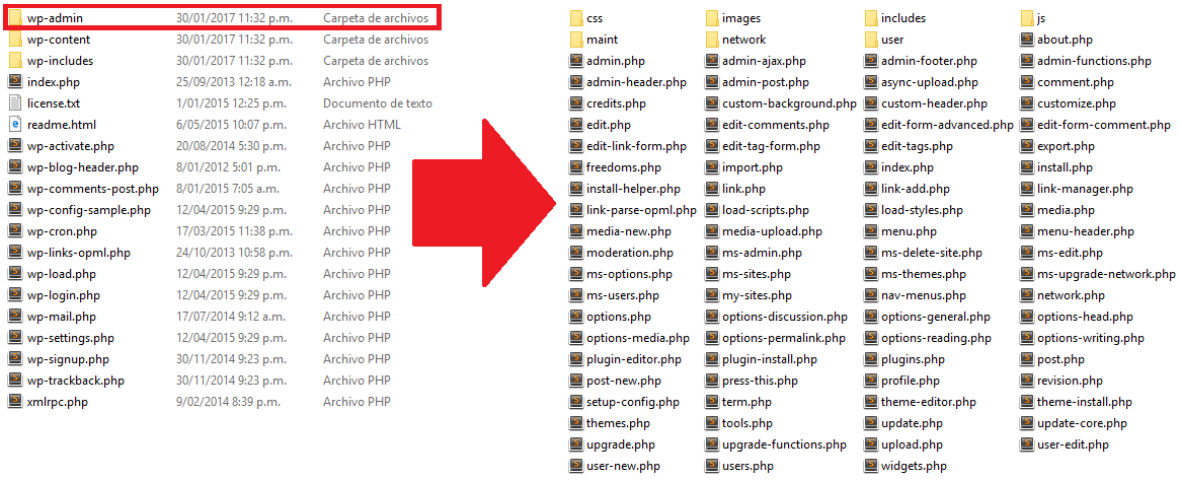
7.3.7. Interfaces de administración de la infraestructura y de la aplicación (OWASP-CM-007).

Las interfaces de administración del sitio web están diseñadas para escalonar los privilegios de los usuarios y permitir acceder al panel de administración con las funcionalidades acordes al tipo de usuario, roles y capacidades.

- Super Admin - Quien cuenta con acceso a las características completas de administración.
- Administrador - Quien tiene acceso a todas las características de administración de un sitio en particular.
- Editor - Quien puede publicar y editar entradas, propias y de otros usuarios.
- Autor - Quien puede publicar y editar sus propias entradas.
- Colaborador - Quien puede escribir y editar sus propias entradas, pero no publicarlas.
- Subscriptor - Quien solamente puede editar su perfil.

Las pruebas de interfaces de administración e infraestructura deben ser llevadas a cabo para revelar si las funcionalidades privilegiadas pueden ser accedidas por un usuario estándar o no autorizado. En la figura 51 se detallan los directorios y ficheros utilizados en la instalación del sitio web SiembraViva.com

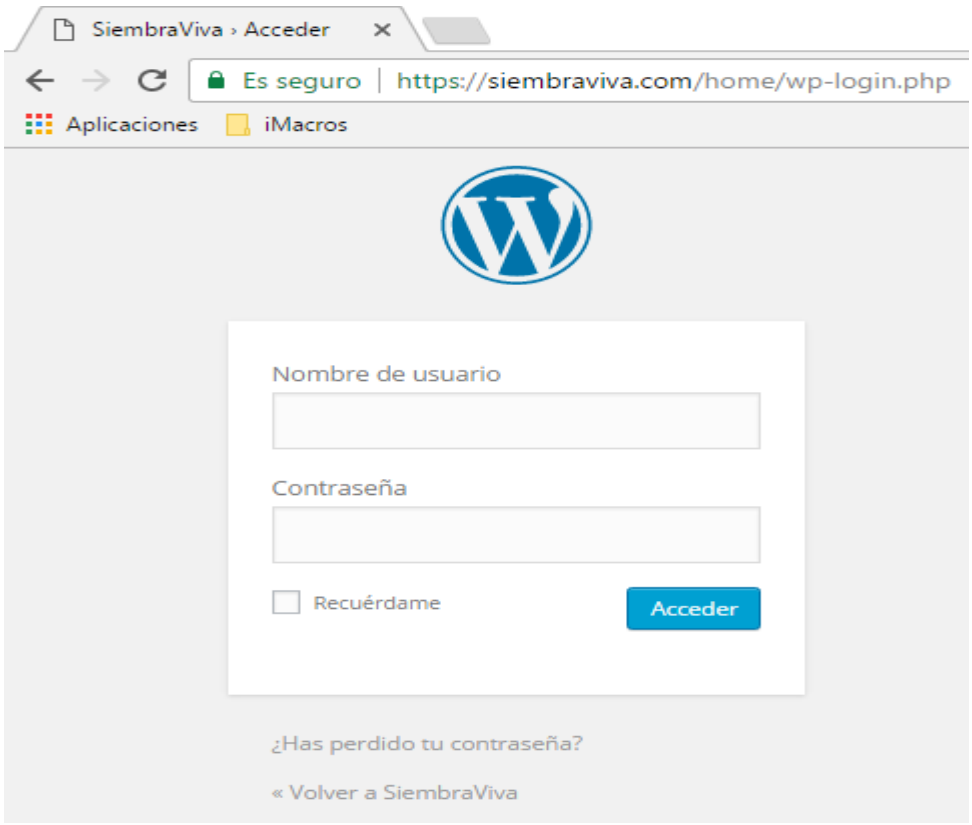
Figura 51. Directorios y ficheros de administración



Fuente: Contenido directorio /wp-admin/

Al ingresar al directorio /wp-admin/, es redireccionado al panel de administración de usuarios, para el inicio de sesión según el rol de dicho usuario (Ver figura 52).

Figura 52. Inicio de sesión panel de administración



Fuente: Autor, SiembraViva.com/home/wp-admin/

7.3.8. Métodos HTTP y XST (OWASP-CM-008).

Los métodos GET y POST son los métodos HTTP más comunes para acceder a la información proporcionada en un servidor web, los métodos HTTP pueden ser utilizados con propósitos dañinos si el servidor web se encuentra más

configurado. Adicionalmente, Cross Site Tracing (XST), es una forma de cross site scripting utilizando el método HTTP TRACE.

Aunque los métodos GET y POST son los métodos más comunes para acceder a la información proporcionada por el servidor web, el protocolo de transferencia de hipertexto (en inglés HTTP) permite otros diversos métodos según RFC 2616[12] donde describe la versión 1.1 HTTP (estándar actual). Se definen los siguientes ocho métodos:

- HEAD
- GET
- POST
- PUT
- DELETE
- TRACE
- OPTIONS
- CONNECT

Los siguientes métodos HTTP pueden plantear un riesgo para la seguridad de una aplicación web, ya que un atacante puede modificar los archivos almacenados en el servidor web como por ejemplo robas las credenciales de usuarios legítimos. En concreto, los métodos que deberían ser desactivados son los siguientes:

PUT: Este método permite a un cliente subir nuevos archivos al servidor web. Un atacante puede explotarlo subiendo archivos maliciosos. Por ejemplo: Un archivo ASP que ejecuta ordenes invocando a cmd.exe, o simplemente utilizando el servidor víctima como un repositorio de archivos.

---

[12] W3. 2004. RFC2616 - Hypertext Transfer Protocol HTTP/1.1. Network Working Group. [En línea] 09 de octubre de 2004. [Citado el: 04 de febrero de 2017.] https://www.w3.org/Protocols/rfc2616/rfc2616.html.

DELETE: Este método permite a un cliente borrar un archivo en el servidor web. Un atacante puede explotarlo como una forma simple y directa de cambiar el contenido que muestra el site para montar un ataque DoS (Denegación de servicio).

CONNECT: Este método podría permitir a un cliente utilizar el servidor web a modo de proxy.

TRACE: Simplemente devuelve al cliente cualquier cadena que se le haya enviado al servidor, y se utiliza principalmente para depuración. Este método, aparentemente inofensivo, se puede utilizar para montar un ataque conocido como Cross Site Tracing, descubierto por Jeremiah Grossman.

Una página web necesita uno o varios de estos métodos para acceder a cierta información del servidor, es importante comprobar que está siendo utilizada adecuadamente limitado a usuarios confiables y bajo condiciones seguras.

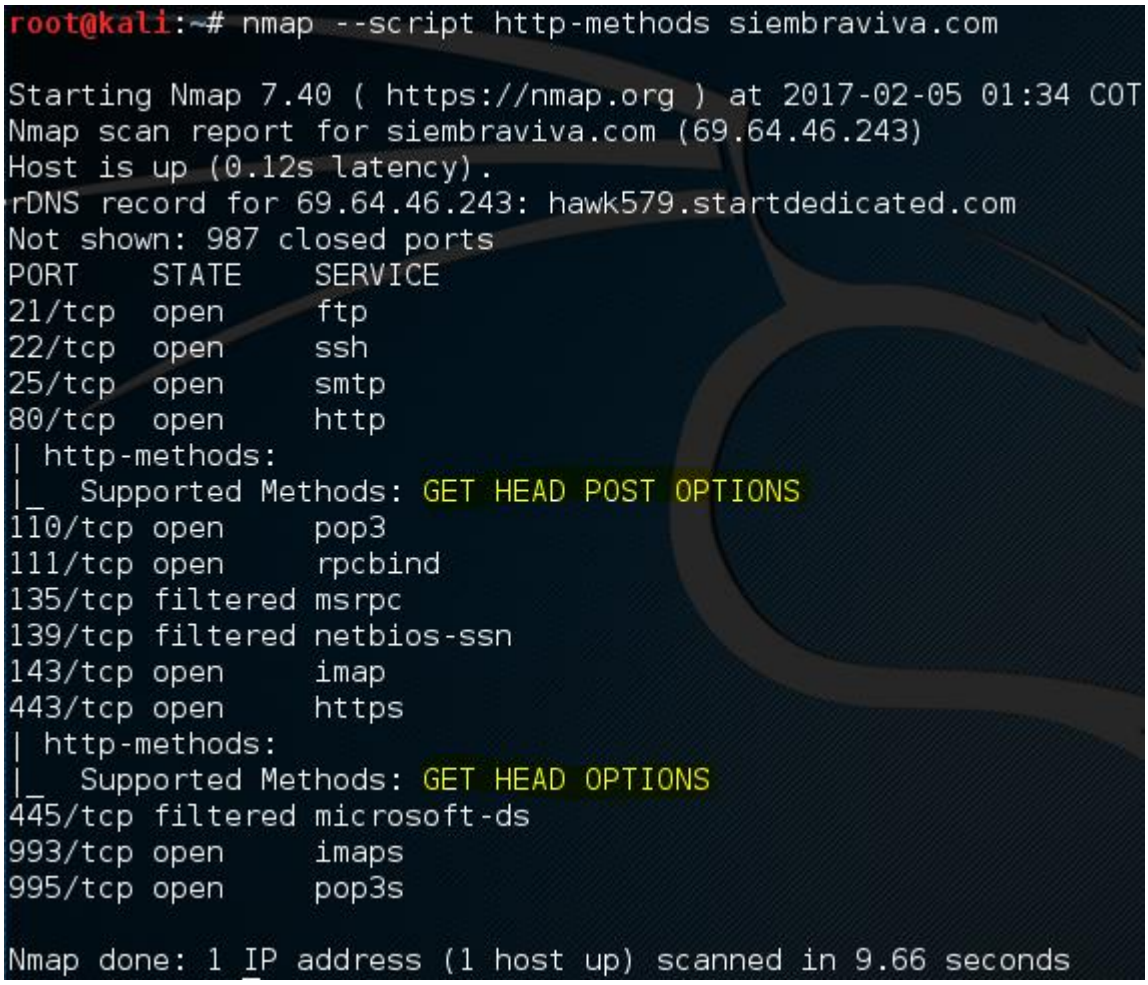
7.3.8.1 Métodos HTTP soportados por SiembraViva.com.

Para realizar esta prueba, utilizamos el método HTTP OPTIONS que proporciona la forma más directa y efectiva de descubrir los métodos utilizados en el sitio web. El RFC 2616 afirma que "*el método OPTIONS representa una petición de información acerca de las opciones de comunicación disponibles en la cadena petición/respuesta identificada por la petición-URI*".

Esta petición se realiza con la ayuda de nmap ejecutando el siguiente script, *nmap --script http-methods siembraviva.com.*

En la figura 53 ejecutamos el Script que permite enumerar los métodos "potencialmente riesgosos", aunque no todos representan un riesgo de seguridad, es importante conocer que métodos son soportados por el servidor web.

Figura 53. Métodos HTTP soportados por el servidor



Fuente: Autor, nmap 7.40

Los métodos soportados por el servidor en el puerto 80 http son GET, HEAD, POST, OPTIONS. En el puerto 443 https son GET, HEAD, OPTIONS.

7.4. COMPROBACIÓN DEL SISTEMA DE AUTENTICACIÓN

La autenticación es el proceso de verificar la identidad digital de la persona que intenta acceder al sistema o a los servicios web durante la comunicación. Por

ejemplo, durante el proceso de registro al sistema e-commerce SiembraViva, o el proceso de inicio de sesión.

7.4.1. Transmisión de credenciales a través de un canal cifrado (OWASP-AT-001).

Realizamos la comprobación de que los datos de autenticación como credenciales del usuario son trasmitidos a través de un canal cifrado para evitar interceptaciones por usuario no autorizados o maliciosos.

Este análisis se centra en determinar que el sitio web SiembraViva.com utiliza los mecanismos de seguridad apropiados entre la comunicación desde el servidor web y el cliente, mecanismos de seguridad como utilización del protocolo HTTPS, validación de formularios, etc.

Se realiza prueba ingresando con un usuario convencional a la página siembraviva.com, inmediatamente al ingresar el navegador Google Chrome arroja una alerta en la barra de dirección y notifica de una posible manipulación de los datos al ser sensible a ataques la información del usuario. Evidencia

Figura 54. Transmisión por canal cifrado HTTP

Fuente: Autor, SiembraViva.com

Según la notificación que aparece en la barra de direcciones de la figura 54, el sitio web no cuenta con una conexión segura para el usuario. La información de los usuarios se encuentra en riesgo.

7.4.2. Enumeración de usuarios (OWASP-AT-002).

Con esta prueba se pretende conocer la existencia de un usuario dentro de la aplicación Web acorde a una serie de mecanismos sencillos que se tienen incorporados en la implementación de la aplicación. Errores en la configuración o en el diseño pueden contribuir al conocimiento de un usuario, por ejemplo, al intentar ingresar a la aplicación se envían los datos y se muestra un mensaje notificando que las credenciales son incorrectas o que el usuario ya está usando la aplicación en el momento. De hacerse múltiples pruebas en el sistema se pueden conocer una lista de usuarios y con esta información proceder a hacer un ataque de fuerza bruta.

Esta información es de dominio del administrador de la página y no nos puede ser suministrada para evitar ser filtrada; esta prueba no poder ser efectuada.

7.4.3. Pruebas de diccionario sobre cuentas de Usuario o cuentas predeterminadas (OWASP-AT-003).

Los servidores donde son ejecutadas las aplicaciones web cuentan con una base de software que facilita la administración o provee las funcionalidades necesarias para que el desarrollo se comporte como fue creado. Las configuraciones por defecto cuentan con contraseñas que son de dominio público o fáciles de descifrar, la correcta administración de estas contraseñas puede disminuir el riesgo inherente que existe en este tipo de aplicaciones.

Algunos responsables son:

- Personal TI que no cuenta con la experiencia suficiente para saber que contraseñas son o no seguras; además la frecuencia que deben ser cambiadas.
- Algunas aplicaciones cuentan con usuarios predeterminados que no es posible que sean eliminados.
- Una práctica peligrosa es dejar espacios en blanco como contraseña en las aplicaciones.
- Es necesario incentivar en los usuarios la creación de contraseñas con un nivel de complejidad aceptable. No usar contraseñas comunes, combinar caracteres con números, mayúsculas, etc.
- Los desarrolladores crean puertas traseras para acceder de manera fácil a la aplicación, es importante auditar el software para eliminarlas y evitar infiltraciones.

Las pruebas de diccionario aumentan la latencia de la red y afectan el desempeño de la página siembraviva.com, por ese motivo y solicitud del administrador del sitio no es posible efectuar las pruebas.

7.4.4. Pruebas de Fuerza Bruta (OWASP-AT-004).

Fuerza bruta es un conocido ataque que trata de penetrar un sistema para sacar provecho de este o extraer información importante. El ataque consiste en tener un problema y probar todas las soluciones posibles hasta obtener lo deseado, en el caso de una aplicación web es importante conocer una cuenta que exista dentro del sistema y luego proceder a probar cada una de las contraseñas posibles hasta obtener el acceso.

Existen diferentes tipos de autenticación para acceder en un sistema, identificación biométrica, certificados, contraseñas que se usan una sola vez, tokens, entre otras. Las aplicaciones web en su mayoría cuentan con la autenticación de un usuario y su correspondiente contraseña, estos datos son almacenados en las bases de datos de la aplicación, como buena práctica son los datos de la contraseña se les aplica un algoritmo de cifrado y así poner un grado de dificultad para el atacante.

De darse un ataque exitoso de tipo fuerza bruta se puede tener acceso a sectores de la aplicación donde se pueden ocultar partes de código malicioso y explotar algunas vulnerabilidades, se puede tener acceso a los paneles de administración de la aplicación, sonsacar documentos privados de la organización o de los usuarios, existen muchas posibilidades de un ataque de tener éxito un ataque de este tipo.

Una parte importante al hacer un ataque de fuerza bruta es importante realizar un análisis del tipo de autenticación que se tiene implementado en la aplicación, existen dos categorías; autenticación HTTP y autenticación basada en formulario HTML.

Autenticación HTTP

Básica

Este tipo de autenticación es fácilmente descifrable por el atacante, por esta razón no es recomendable su uso. Comúnmente es usado un formulario de tipo nickname y contraseña para realizar el ingreso de un usuario a una aplicación WEB; el servidor al recibir la orden procesa los datos y envía un mensaje de tipo 401, se arma una cadena que contiene los siguientes elementos: WWW-Authenticate: Basic realm="wwwsiembraviva", los primeros elementos de la cadena (Authenticate y Basic) son obligatorios y el valor final sirve para identificar desde que dominio se realizó la petición. El cliente recibe una cadena (Authorization: Basic amFjb2Jvb2RpYWFwYW9sYQ==), la cadena contiene los dos primeros elementos de manera obligatoria y al final se agrega el valor de la contraseña codificado en base-64.

Autenticación de Acceso de Resumen

En este tipo de autenticación la base es la autenticación básica expuesta anteriormente, la diferencia radica que no se usa un algoritmo base 64 si no que se usa un tipo de codificación hash (MD5) tanto para cifrar la contraseña del login y se añade un código que es usado por única vez por parte del navegador cliente para procesar la petición. La respuesta que el servidor arroja es un texto plano y este es procesado por la aplicación.

Autenticación basada en formulario HTML

Los métodos de autenticación que se han descrito anteriormente funcionan de manera correcta si la conexión que se establece está cifrada, una sesión SSL. Actualmente son usados métodos más complejos, personalizados y basados en HTML. El formulario básico HTML cuenta con la siguiente estructura:

```
<form method="POST" action="login">
<input type="text" name"username">
<input type="password" name="password">
</form>
```

En el caso de la página siembraviva.com se evidencia que el tipo de autenticación que se encuentra implementado es de tipo HTML, a continuación, en la figura 55 se muestra la estructura de la página:

Figura 55. Autenticación método POST

```
                <a id="login-fb" href="/login-fb" class="connectfb">
                    <i class="mdi mdi-facebook"></i>
                        Ingresa con Facebook
                </a>
                <center>o Regístrate con tu Correo</center>
                <p id="login-error" class="login-error"> </p>
                <div class="input-with-icon">
                    <i class="mdi mdi-account"></i>
                    <input name="name" type="text" placeholder="Nombre" required>
                </div>
                <div class="input-with-icon">
                    <i class="mdi mdi-at"></i>
                    <input name="email" type="mail" placeholder="Correo Electronico" required>
                </div>
                <br>
                <div class="input-with-icon">
                    <i class="mdi mdi-key-variant"></i>
                    <input name="password" type="password" placeholder="Contraseña" required>
                </div>
                <div class="row column text-center">
                    <small>
                        <label class="privacy-login">
                        <input id="terms-register-check" name="terms" type="checkbox" ng-model="model.accept_terms"
required" checked required>
                        <span class="welcome-text">Acepto</span>
                        <a target="_blank" class="welcome-link" href="/home/terminos-y-condiciones/">Términos</a>
                        <span class="welcome-text"> y </span><a target="_blank" class="welcome-link" href="/home/po
                        </label>
                    </small>
                </div>
                <div class="row column text-center">
                    <button type="submit" class="button secondary large expanded">Registrar</button>
                </div>
            </div>
        <input type='hidden' id='registerStep' value=''>
        </form>
```

Fuente: Autor.

Ataques de diccionario

El ataque de este tipo trata de encontrar los usuarios o contraseñas de una página web apoyándose en herramientas automatizadas o scripts, los archivos que contienen las palabras pueden ser creados y configurados acorde al tipo de usuario; es importante realizar un estudio para conocer al usuario, comúnmente se usa ingeniería social o reconocimiento activo/pasivo.

Ataques de búsqueda

El ataque busca encontrar en una combinación de letras mayúsculas o minúsculas, números, caracteres especiales y las cuales se les configura una

longitud según al análisis que se haga de la cadena a encontrar. Es un proceso que tarda mucho tiempo debido a la complejidad de las cadenas a analizar, por ejemplo, en una cadena de 8 caracteres puede contar con 200.000 millones de posibilidades.

Ataques de búsqueda basados en patrones

Existen aplicaciones para realizar ataques de fuerza bruta, tal es el caso de “Jhon the Ripper”. Este ataque incorpora elementos de búsqueda, pero les adiciona modificaciones de la contraseña basándose en el nombre de usuario en tiempo real o se configuran modificaciones a la cadena basándose en una subcadena preestablecida. Un ejemplo, min primer intento, m1n segundo intento, minm1n tercer intento

El administrador no permitió realizar esta prueba, la página se vería afectada en su rendimiento y afectar adicionalmente el servicio a los usuarios.

7.4.5. Saltarse el sistema de autenticación (OWASP-AT-005).

La mayoría de las aplicaciones utilizan sistemas de autenticación para lograr obtener acceso a la información privada o confidencial, establecer ciertos permisos pueden proveer una buena seguridad, pero no todos los métodos de autenticación son eficaces.

Muchos de los sistemas de autenticación pueden saltarse mediante sentencias, registros o peticiones cuando estos métodos son mal configurados o implementados. Esto puede conseguirse modificando la URL dada como parámetro, o manipulando un formulario, o falsificando sesiones.

El administrador no permitió realizar esta prueba por motivos de funcionamiento de la página.

7.4.5.1. Peticiones directas de paginas

Consiste en verificar el control de acceso de las páginas de registro e inicio de sesión, validando si es posible saltarse el sistema de autenticación.

Se realizan directamente las peticiones en las páginas donde el acceso o inicio se sesión son requeridas, la pagina realiza la verificación de credenciales antes de permitir el acceso a dicho contenido.

Navegando directamente en una página protegida con control de acceso, este método realiza la correcta verificación antes de mostrar su contenido tal como se puede evidenciar en la figura 56.

Figura 56. Petición directa ingreso a la pagina

Fuente: Autor, SiembraViva.com

Si navegamos sobre la URL https://siembraviva.com/dashboard, esta realiza la validación y solicita iniciar sesión con sus credenciales para poder mostrar su contenido.

Todas las peticiones directas a las páginas de SiembraViva son validadas y cumplen con el método correcto para mostrar su contenido.

7.4.6. Comprobar Sistemas de recordatorio/restauración de contraseñas vulnerables (OWASP-AT-006).

Frecuentemente los usuarios de las aplicaciones Web olvidan las contraseñas que almacenan para poder realizar la autenticación en el sistema, los desarrolladores implementan funciones especiales para realizar este proceso, por ejemplo, enviar un email con un enlace de recuperación a la dirección que se tiene configurada en el sistema. El usuario se le envía una contraseña temporal o una antigua a la dirección registrada, el usuario accede y el usuario solicita cambio de la que tiene asignada temporalmente.

Los usuarios al realizar el registro en una aplicación Web en algunos casos, dependiendo del nivel de seguridad, se les realiza una serie de preguntas para complementar el proceso de autenticación. Al momento de realizar la solicitud de restablecimiento de contraseña se le realizan estas preguntas al usuario y así comprobar su identidad, las preguntas son de temas muy personales que en la práctica no son fácil de acceder por parte de gente del común. Un ejemplo de pregunta puede ser la materia favorita en secundaria o el nombre de la mascota de la infancia.

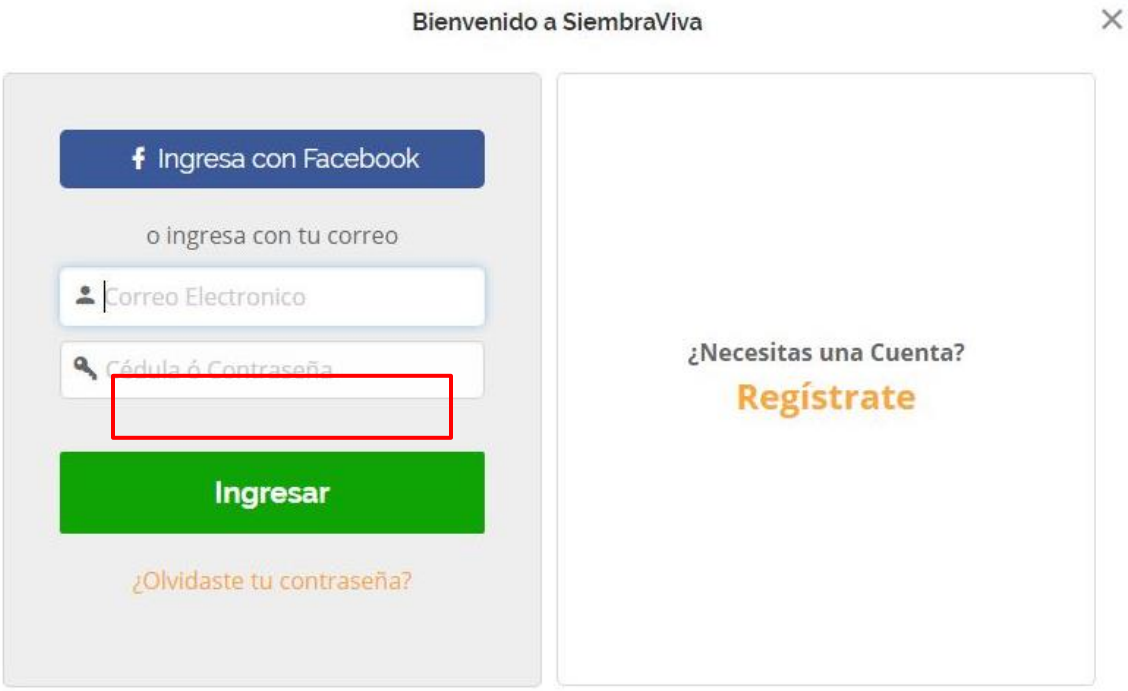
Se cuenta con un mecanismo adicional, guardar las contraseñas en el navegador o como se conoce cachear los password; el navegador detecta el nombre de usuario y busca la contraseña almacenada previamente por el usuario. Es una medida poco recomendable debido a que quien tenga acceso al sistema del usuario va a tener acceso fácil, es una medida que deja expuestos los datos.

En el caso de siembraviva.com se tiene una manera de reestablecer las contraseñas que se ajusta a los que se han explicado anteriormente. El usuario le solicita al sistema que se haga un restablecimiento, el sistema solicita que se ingresa la dirección de correo electrónico registrada, el sistema envía un correo al usuario, dentro del correo existe un enlace donde se le dirige a un formulario que le solicita la nueva contraseña.

Paso 1:

En la pantalla de registro se debe buscar la etiqueta que está relacionada con el olvido de la contraseña (ver figura 57), por parte de usuario. Generalmente todas las aplicaciones Web cuentan con este mecanismo para restablecer la contraseña.

Figura 57. Solicitud para recordar la contraseña



Fuente: Autor, SiembraViva.com

Paso 2:

Al momento que la aplicación web recibe la petición del restablecimiento se muestra un formulario que solicita el correo electrónico registrado, también es posible retroceder el proceso si se recordó la contraseña tal como se puede observar en la figura 58.

Figura 58. Recordar la contraseña

Fuente: Autor, SiembraViva.com

Paso 3:

En el buzón de entrada del correo llega la notificación de la solicitud que se le hizo a la página, el asunto muestra el propósito de la solicitud.

Figura 59. Notificación restablecimiento de contraseña

Fuente: Autor, SiembraViva.com

Paso 4:

En el cuerpo del mensaje se pueden evidenciar los logos de la empresa, lo cual le da credibilidad al mensaje y le permite tener mayor confianza al usuario para realizar el proceso (ver figura 60). El proceso no se realiza en el correo

directamente si no en una página nueva, el correo cuenta con un tiempo de caducidad.

Figura 60. Restablecimiento de contraseña

Fuente: Autor, SiembraViva.com

Paso 5:

En la página emergente se muestra un formulario en el cual se digita y confirma la contraseña, es importante resaltar que en este punto la página confía en el correo electrónico digitado y que no se le ha realizado ninguna pregunta de seguridad.

El sistema solicita al usuario una contraseña con muy pocos requerimientos de seguridad, las contraseñas deben tener unos requerimientos fuertes de seguridad dada la naturaleza de la página. El usuario debería complementar la solicitud respondiendo algunas preguntas de seguridad o comprobar la identidad de quien hace la petición con algún mecanismo complementario.

Figura 61. Restablecer contraseña

Fuente: Autor, SiembraViva.com

Al analizar el proceso de reset de contraseña es importante realizarse preguntas relacionadas a como se desarrolló el proceso, a continuación, se van a plantear una serie de preguntas basadas en el manual de pruebas OWASP:

- ¿El proceso de restauración tiene intentos ilimitados o la cuenta es bloqueada al realizar un número determinado de intentos?
  Al realizar el proceso de restauración se realizó en cinco ocasiones y el sistema no presento ningún mensaje que advirtiera sobre los intentos que faltaban hasta realizar un bloqueo de la cuenta, el sistema implementado permite realizar el proceso las veces que sea necesario para el usuario.
- ¿Se realizan preguntas de seguridad al efectuar el registro? ¿Estas preguntas son usadas comprobar la identidad del usuario al realizar la restauración?
  El sistema de registro implementado en siembraviva.com no cuenta con preguntas de seguridad, el sistema no cuenta con esta funcionalidad. Es importante anotar que por este motivo el proceso de reset no cuenta con esta medida de seguridad, la cual contribuye a comprobar la identidad del usuario.

El proceso de reset de la contraseña en la página siembraviva.com debería tener medidas de seguridad complementarias que brinden mayor confiabilidad para el usuario y fortalezcan los datos que se encuentran almacenados en el sistema.

7.4.7. Pruebas de gestión del Caché de Navegación y de salida de sesión (OWASP-AT-007).

Al iniciar un usuario la sesión es importante que el navegador cuente con algunas funcionalidades que permitan cerrar la sesión cuando esta quede inactiva, además que es importante que no sean almacenados datos sensibles en los datos del navegador.

Existen dos escenarios para que una sesión sea cerrada:

- Cierre de sesión por parte del usuario
- Cierre de sesión por parte del sistema debido a que no hay actividad del usuario

Al momento de crear una aplicación web es importante tener muchos factores que le pueden brindar al usuario un mayor grado de confiabilidad y adicionalmente darle credibilidad al manejo que se le dan a los datos dentro del sistema. La implementación de una funcionalidad o funcionalidades que administren de manera correcta y confiable las sesiones dentro de la aplicación es fundamental para prevenir que no sean explotadas vulnerabilidades o que los datos del usuario estén en peligro de ser robados por algún atacante.

Dependiendo del lenguaje de programación es importante implementar funciones que invaliden la sesión de lado del servidor, con este paso eliminar las cookies no se hace necesario; al invalidar la sesión esta desaparece y se cierra en el servidor, el usuario deja de existir en el sistema. A continuación, se muestra cómo se pueden invocar las funciones dependiendo del lenguaje de programación:

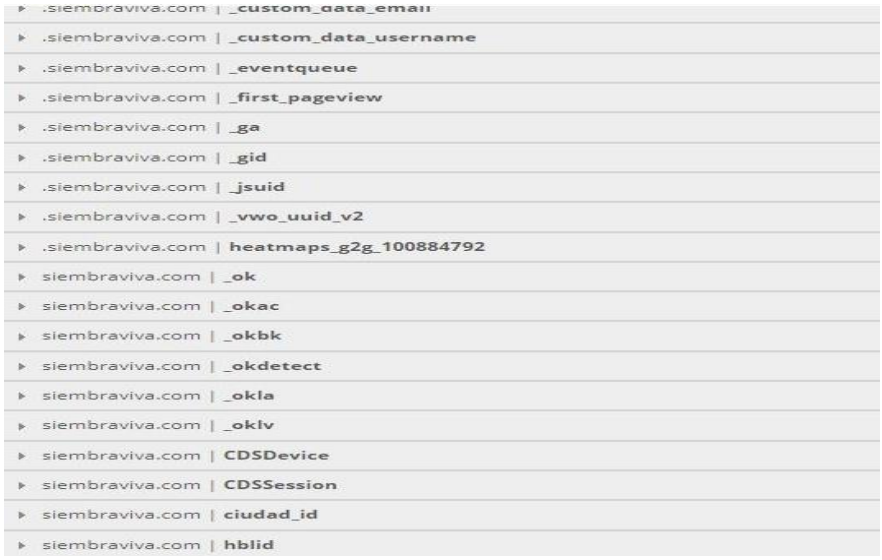
- session_destroy() en PHP
- httpSession.invalidate() en Java
- session.abandon() en .Net

Las sesiones creadas por parte de los usuarios son sensibles a sufrir ataques de tipo “cookie replay”, este ataque lo que busca principalmente es imitar los datos

de la sesión de un usuario y así simular dentro del sistema que es el dueño de la sesión; este ataque busca “resucitar” una sesión. Los computadores públicos ponen en riesgo a los usuarios que inicien sesiones dentro del navegador puesto que existen algunas aplicaciones que no eliminan cookies ni invalidan las sesiones creadas y con simplemente ir “atrás” con el navegador se pueden reutilizar sesiones y poner en riesgo los datos sensibles de los usuarios. Una medida adicional es que el navegador asigne un tiempo determinado para que estén con “vida” los datos de la sesión del usuario, esto puede disminuir que al ser secuestrada una sesión los datos puedan ser reutilizados y así el atacante pueda infiltrarse en el sistema.

En la página SiembraViva.com se verifico cómo se comportan las cookies y como es el manejo de estas, existen complementos para los navegadores que permiten hacer un análisis de las cookies. Para realizar pruebas en la página que se está analizando se usó un plugin llamado EditThisCookie en el navegador Google Chrome, el plugin muestra todas las cookies que se tienen para una sesión en particular y el valor de cada una de ellas. A continuación, se ilustran las cookies que se tienen para una sesión según la figura 62:

Fig. 62. Cookies almacenadas por SiembraViva.com

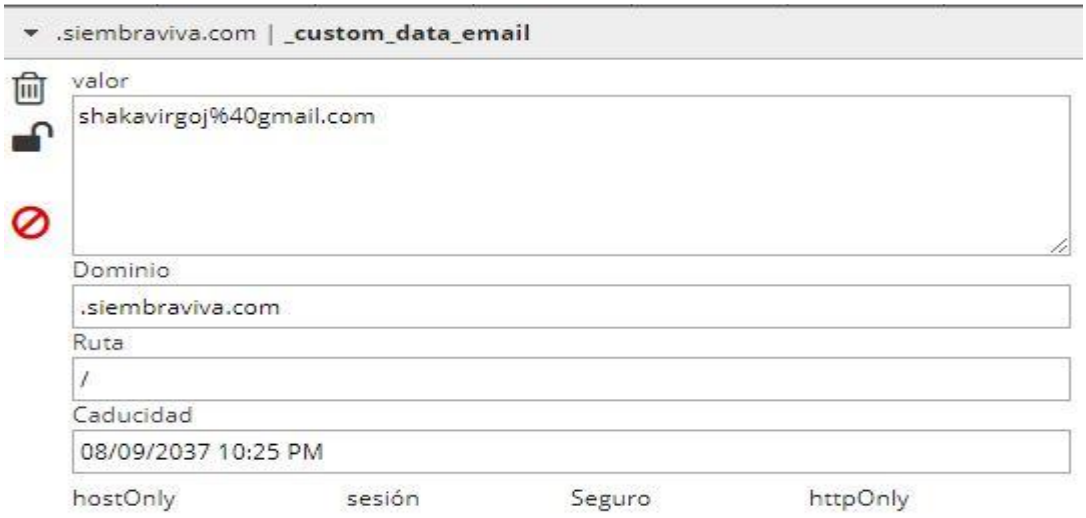


Fuente: Autor, SiembraViva.com

En la imagen anterior se aprecian las cookies custom_data_email y custom_data_username, estas cookies son los datos que permiten identificar la sesión del usuario y realizar la validación en el servidor de la identidad de quien va a iniciar sesión. En el momento se en las dos cookies esta la siguiente información almacenada:

En el caso de la cookie custom_data_email se tiene el siguiente resultado evidenciado en la figura 63 y 64.

Figura 63. Cookies custon_data_email



Fuente: Autor, SiembraViva.com

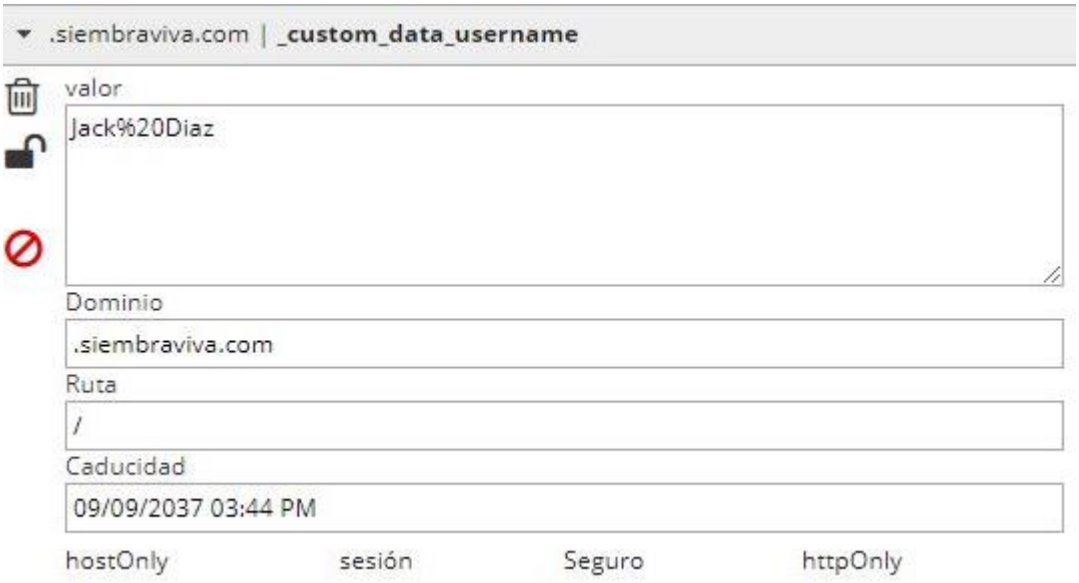
Al analizar la cookie se encuentra que nunca expira, siendo una cookie que mantiene activa o no la sesión de un usuario, debería tener un tiempo de vida corto y el cual depende de la inactividad de un usuario.

La vida de la cookie es muy importante definirla debido a que en esta variable se pueden evitar muchas complicaciones, el desarrollador debe tener en cuenta que el poner un tiempo muy corto puede traer molestias al usuario y si el tiempo de vida de una cookie es muy largo va a generar una sesión que puede permanecer abierta y ser aprovechada por personas con intereses maliciosos. De no darse un correcto cierre o destrucción de la sesión con un simple atrás en el navegador se

puede abrir una sesión, sacar datos importantes y hasta hacer vulnerable un sistema.

En el caso de la cookie custom_data_username se tiene lo siguiente

Figura 64. Cookies custon_data_username



Fuente: Autor, SiembraViva.com

La fecha de caducidad de esta cookie es muy elevada y lo que hace es que la sesión no se destruya, la sesión va a estar abierta a pesar de que el usuario no esté haciendo nada en la página; esto genera que los datos del usuario estén vulnerables y que cualquiera que use el equipo y encuentre la sesión abierta puede usarla, suplantando al propietario. Las sesiones deben rastrear permanentemente si el usuario está haciendo uso de la aplicación.

Las cookies también pueden ser manipulables o no, en su contenido, dependiendo de lo que el desarrollador quiera proteger. Existen cookies dentro del sistema que pueden afectar el correcto funcionamiento y por otra parte hay otras cookies que tienen menor relevancia, con la manipulación de una cookie se puede cambiar el usuario registrado en una sesión o hasta se puede tener acceso

con un perfil diferente dentro del sistema, por ejemplo, acceder como administrador.

A través de la extensión de Google Chrome “Edit this Cookie” se intentó editar cookies existentes y no se tuvo éxito con la prueba, el desarrollador protegió las cookies para que estas no fueran editables y que si se procediera un cambio este no replique en el servidor.

Un manejo adecuando de las sesiones debe tener en cuenta la duración de la misma, dependiendo del tipo de página Web se pueden tener tiempos de duración de 60 minutos o en otros casos que no se cierre la sesión de no ser por orden del usuario. Por la naturaleza de siembraviva.com se debe tener un tiempo de cierre de sesión corto, la razón radica en que esta página maneja información de pago con tarjetas de crédito u otros tipos de formas de pago; al realizar el análisis de la sesión la página no cerro nunca aun sin haberse detectado actividad dentro de la aplicación, esto no es lo óptimo, siendo de carácter e-commerce se debería tener un tiempo corto de cierre automático.

7.4.8. Pruebas de CAPTCHA (OWASP-AT-008).

El CAPTCHA de las siglas (Completely Automated Public Turing test to tell Computers and Humans Apart) que en español significa prueba de Turing completamente automática y pública para diferenciar ordenadores de humanos. Consiste en una prueba desafío-respuesta utilizada para determinar cuando el visitante al sitio web es un humano o no lo es.

El Captcha no es considerado un control de autenticación, pero si es un sistema eficiente para reducir ataques de enumeración, ingreso, registros, recuperación de contraseña y envió automático de peticiones GET/POST. etc.

El sitio web SiembreViva.com no tiene implementado ningún sistema de Captcha para realizar el registro a nuevos usuarios, inicio de sesión, recuperación de contraseñas, etc. Facilitando de esta forma el registro o uso indiscriminado de nuevos usuarios en el sitio web.

Sin Captcha el atacante puede obtener nombres de usuario válidos, números telefónicos o cualquier otra información sensitiva en poco tiempo, envío automático de muchas peticiones GET/POST en poco tiempo cuando no es deseable (ejemplo, inundación de SMS/MMS/correos), Captcha proporciona una

función limitante de velocidad. En la figura 65 se muestra uno de los formularios sensibles ya que no tiene implementado el Captcha.

Figura 65. Sin implementación de Captcha

Fuente: El autor

7.4.9. Pruebas para autenticación de factores múltiples (OWASP-AT-009).
Evaluar la fortaleza de un sistema de autenticación es una tarea critica ya que suele considerarse una etapa delicada al tener relación con información sensible de clientes.

El objetivo de estas pruebas es verificar factores adicionales que aseguren que el usuario tenga dispositivo de acceso físico además de la contraseña. El dispositivo proporcionado al usuario puede ser capaz de comunicarse directa e independientemente con la infraestructura de autenticación usando un canal de

comunicación adicional; esta característica en particular es a veces conocida como “separación de canales”.

Bruce Schneier en 2005 observó que hace algunos años “*todas las amenazas eran pasivas: escuchas y adivinación de contraseñas. Hoy, las amenazas son más activas: phishing y caballos de Troya*”[13]. En realidad, las amenazas comunes que un MFAS debería abordar correctamente en un entorno Web incluye:

1. Robo de credenciales (Phishing, escuchas, MITM)

2. Credenciales débiles (Ataques de diccionarios de contraseñas y fuerza bruta)

3. Ataques basados en sesión (Session Riding, Session Fixation)

4. Ataques de troyanos y Malware

5. Reuso de contraseñas (Usar la misma contraseña para diferentes fines u operaciones, ejemplo transacciones diferentes).

7.4.9.1. Debilidades conocidas.

- Permite el registro de nuevos usuarios sin ningún nivel de seguridad en las contraseñas ingresadas.
- Permite contraseñas inseguras de bajo nivel como 1234, 0000, 0101, etc.
- No solicita contraseñas alfanuméricas ni exige un número determinado de caracteres.
- No verifica el email registrado por el usuario, es propenso a registros inválidos.
- No utiliza sistemas eficientes de Captcha.

[13] Schneier, Bruce y ComputerWeekly. 2005. Insider Threat Statistics. [En línea] From Europe, junio de 2005. [Citado el: 10 de septiembre de 2017.] https://www.schneier.com/blog/archives/2005/12/insider_threat.html.

7.4.9.2. Fortalezas conocidas.

- Integra el registro e inicio de sesión con redes sociales como Facebook.
- No acepta registros con email duplicados.

7.4.10. Probar por situaciones adversas (OWASP-AT-010).

Al realizarse una o más acciones sobre un mismo elemento en una aplicación Web es importante que dicho elemento replique los cambios efectuados sobre el mismo de una manera inmediata, si los cambios no se realizan instantáneamente se puede afectar el funcionamiento de la aplicación y los resultados arrojados pueden ser inesperados y traer problemas en el desempeño de la aplicación.

La naturaleza de la página siembraviva.com es el de ofertar productos agrícolas y que el usuario pueda comprar el producto deseado sin inconvenientes; pero que puede sucedería si varios usuarios compran al mismo tiempo el mismo producto y el stock no da abasto, que sucedería si varios usuarios acceden a un bono de oferta y este no se agota jamás (habiendo sido diseñado para tener un límite).

Una “race condition” es una situación que no ha sido planeada para que se dé bajo ningún caso, este tipo de situaciones hacen que el sistema falle y que los resultados que se desean no sean obtenidos. Los procesos dentro de una aplicación dependen, en gran parte de los casos, uno del otro y las variables de no cambiar de proceso en proceso pueden afectar en gran medida las respuestas que arroja el servidor a los usuarios.

OWASP en esta prueba sugiere ser cuidadosos, la razón radica en que encontrar y probar condiciones de carrera es peligroso para la aplicación; aumenta la latencia de la red, cargar excesivamente el servidor, afectar el stock de un producto, entre otras.

El desarrollador de la página SiembraViva.com no autoriza la realización de esta prueba debido a que puede afectar el servicio a los usuarios y generar perdidas económicas al sacar de operación la aplicación.

7.5.1. Pruebas para el esquema de gestión de sesiones (owasp-sm-001)

Las cookies son el núcleo del manejo de sesiones y son las encargadas de mantener las sesiones activas entre cada una de las peticiones que se hacen por parte del usuario. Las cookies información que identifica a los usuarios, contienen identificadores que distingue una sesión de otra y que almacenan las acciones

que vaya realizando el usuario; las cookies proporcionan el estado de un usuario dentro de la aplicación.

En el caso de siembraviva.com las cookies se usan para almacenar la información del carro de compras mientras el usuario realiza la búsqueda de otros productos, sin la implementación de cookies para esta funcionalidad, cada que el usuario realiza una nueva operación en el sistema la información se perdería.

La seguridad de las cookies es un asunto de cuidado, una cookie que no es protegida correctamente puede dar pie a que un usuario sea clonado o incrementar los privilegios que tenía configurados inicialmente.

7.6. PRUEBAS DE VALIDACIÓN DE DATOS

Las aplicaciones Web se ven sometidas continuamente a nuevas amenazas desarrolladas por personas sin escrúpulos que desean sacar provecho de la información que ha sido confiada por parte de los clientes a las empresas, pero existe una amenaza que es mucho más nociva para los sistemas y es el caso de la información que es ingresa por los usuarios y la información que es creada por el sistema por los procesos que han sido implementados por parte de los desarrolladores, la cual es usada por otros procesos o es mostrada a los usuarios cuando hacen algún tipo de petición o interacción con el sistema.

Dentro de los problemas que pueden darse en una aplicación por datos de entrada en la aplicación se destacan: desbordamiento de Buffer, inyección SQL, ataque UNICODE, entre otros.

Para el desarrollo del informe se han elegido las pruebas XSS e inyección SQL para determinar el nivel de seguridad con el que se cuenta a nivel de datos.

7.6.1. Pruebas de XSS. Cross Site Scripting

Los datos que son ingresados en las aplicaciones Web son una fuente de vulnerabilidades constante, la validación incorrecta de cada uno de los datos importantes en el sistema puede afectar el sistema y los datos que fueron confiados por parte del usuario.

Los ataques XSS tiene como objetivo usar las entradas de los usuarios y modificar la salida esperada, la salida que arroje es generalmente algo malicioso como realizar un secuestro del browser que se esté usando por parte del usuario o sustraer información del usuario sin permiso.

En la mayoría de los casos este tipo de ataque se realiza usando HTML y en algunos casos, más complejos, se usa JavaScript. El sitio Web en este punto solo se usa como el canal para desarrollar el ataque, XSS aprovecha las vulnerabilidades que existen para secuestrar información de los usuarios, guardar la información ingresada a través de los campos que son entradas de datos, robar historiales, entre otros fines.

Como primera medida se debe identificar los elementos dentro de la aplicación Web que sirven para ingresar datos, en este caso se va a usar la entrada de búsqueda dentro de la aplicación Web SiembraViva y se va a probar el siguiente script para identificar la vulnerabilidad.

La sintaxis que se debe seguir para crear el script es la misma con la cual se realizan estas acciones en HTML, es importante definir qué tipo de script se va a usar; por ejemplo, existe en ocasiones el código se puede ingresar sin variaciones y en otras es importante usar codificación ASCII para saltarse las validaciones que se realizan en las entradas de datos para evitar este tipo de ataques.

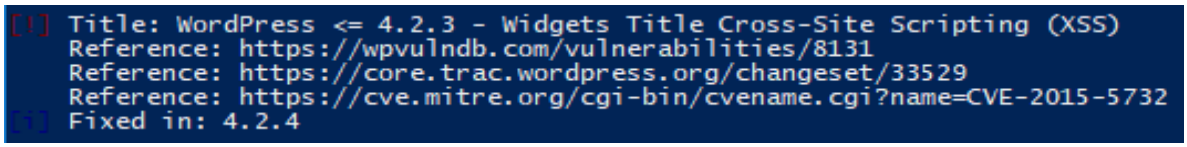
7.6.1.1. Análisis XSS. Cross Site Scripting.

Desde las figuras 66 a la 80 se realizó el análisis de vulnerabilidades con la aplicación wpscan logrando detectar cerca de 41 vulnerabilidades de las cuales 28 de ellas están relacionadas a vulnerabilidades XSS debido a extensiones o plugins desactualizados.

Figura 66. XSS Authenticated Stored

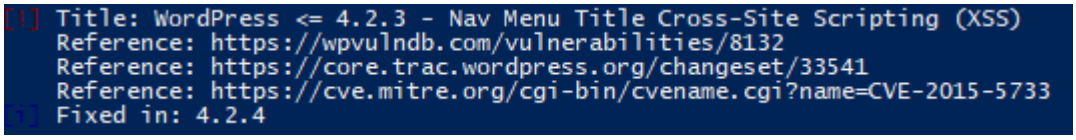
Fuente: Autor WPscan

Figura 67. XSS Widget title



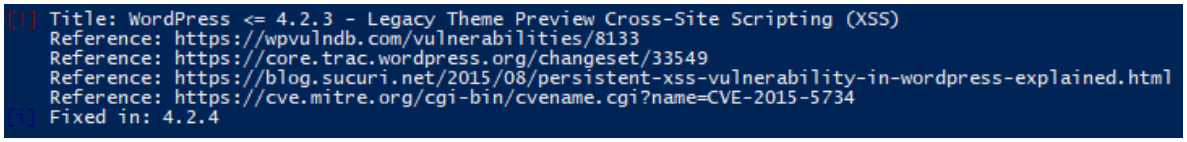
Fuente: Autor WPscan

Figura 68. XSS Nav Menu Title



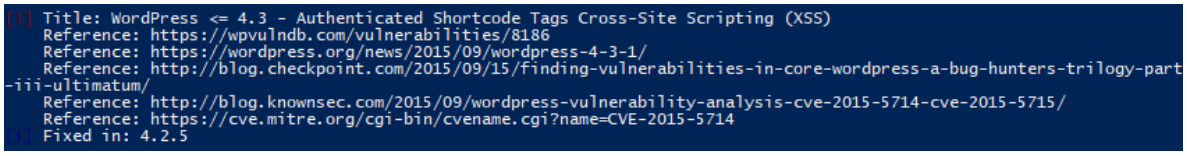
Fuente: Autor WPscan

Figura 69. XSS Legacy Theme Preview



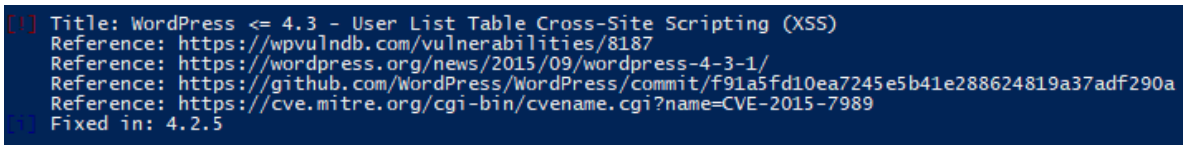
Fuente: Autor WPscan

Figura 70. XSS Authenticated Shortcode Tags



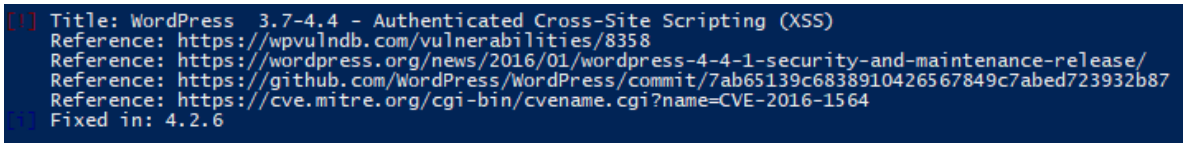
Fuente: Autor WPscan

Figura 71. XSS User List Table



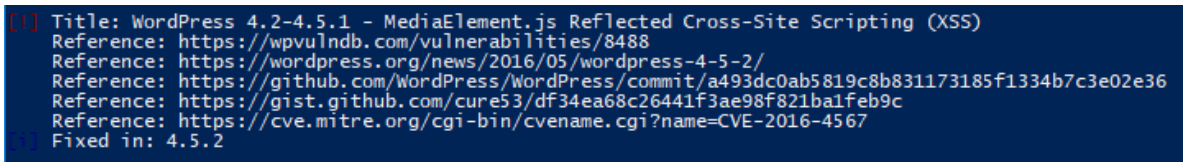
Fuente: Autor WPscan

Figura 72. XSS Authenticated Cross-Site



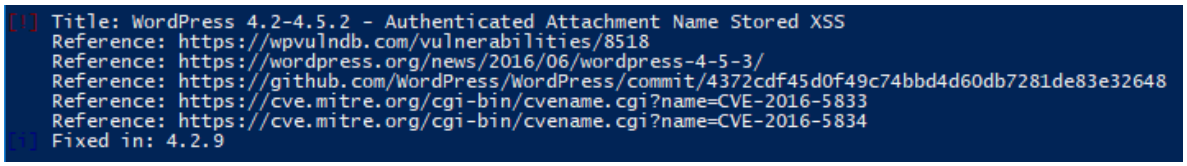
Fuente: Autor WPscan

Figura 73. XSS MediaElement.js Reflected



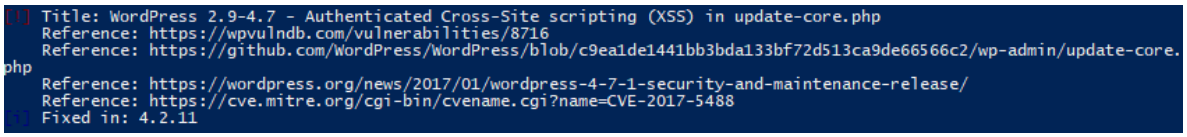
Fuente: Autor WPscan

Figura 74. XSS Authenticated Attachment Names Stored



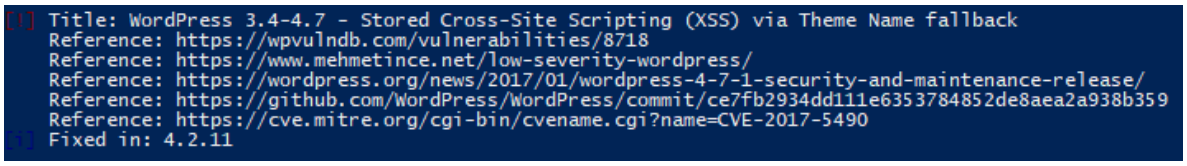
Fuente: Autor WPscan

Figura 75. XSS Authenticated



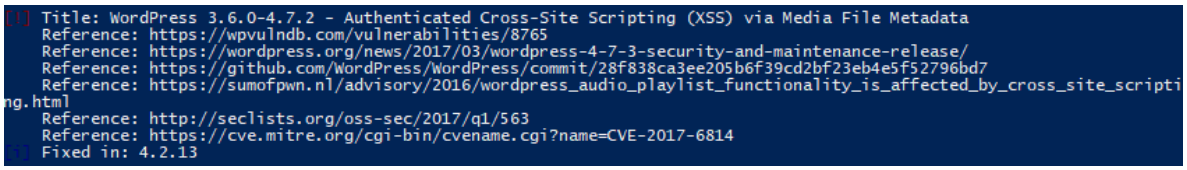
Fuente: Autor WPscan

Figura 76. XSS Stored via theme name fallback



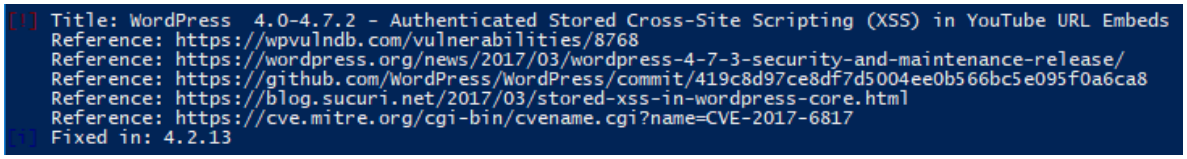
Fuente: Autor WPscan

Figura 77. XSS vía media File MetaData



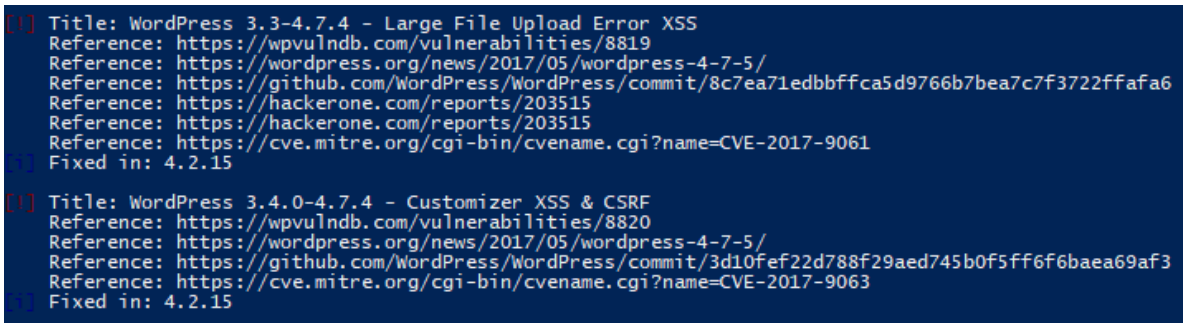
Fuente: Autor WPscan

Figura 78. XSS Authenticated Stored in Youtube URL Embeds



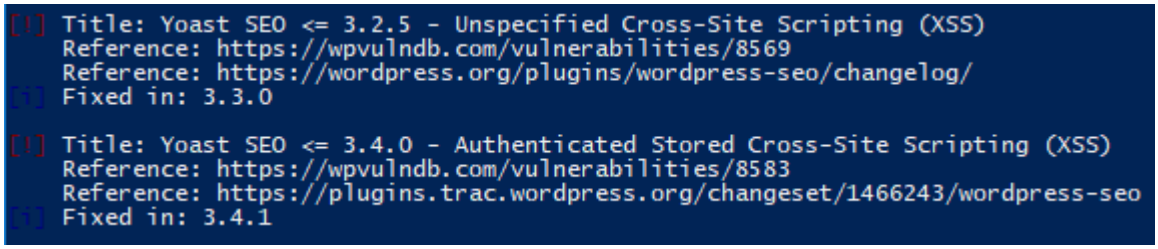
Fuente: Autor WPscan

Figura 79. XSS Large File Upload



Fuente: Autor WPscan

Figura 80. XSS Authenticated Stored Yoast SEO 3.4.0

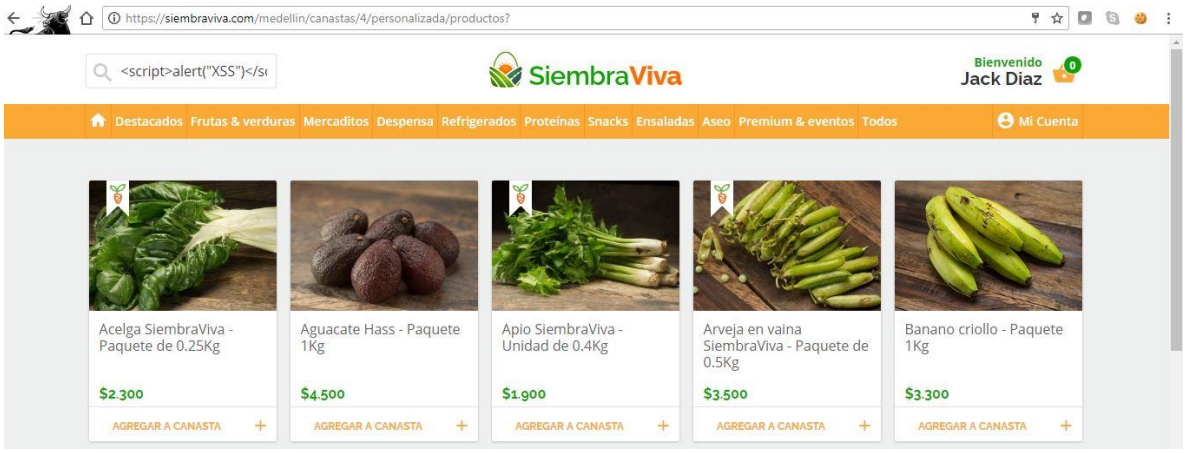


Fuente: Autor WPscan

Prueba 1

<script>alert("XSS")</script>

Figura 81. Prueba 1 Alert XSS



Fuente: Autor, SiembraViva.com

La respuesta que se esperaba por parte del navegador era que se mostrara una ventana emergente con el mensaje “XSS”, pero la respuesta del navegador no fue esta. Ver evidencia en la figura 81 y 82.

Evidencia 1

Figura 82. Evidencia Alert XSS

Fuente: Autor, SiembraViva.com

Como se puede apreciar en la imagen anterior el mensaje no fue mostrado, los sitios Web en la actualidad cuentan con mecanismos que identifican las cadenas

de script en tiempo real y los ataques de este tipo son bloqueados por parte del navegador. Para hacer ataques exitosos se debe agregar codificación ASCII en la cadena que deseamos ingresar en la aplicación.

Prueba 2

Las páginas Web cuentan con seguridad básica para los ataques de este tipo, por esta razón existen complementos para facilitar el ataque y saltar las medidas de seguridad que se han implementado. Una forma es volver código ASCII la cadena que se desea probar en la aplicación Web, un ejemplo es el siguiente:

<script>alert(String.fromCharCode(88,83,83))</script>

La función String.fromCharCode permite convertir la cadena a una cadena que no sea tan fácil de analizar para el navegador, el resultado de la cadena se evidencia en la figura 83.

Figura 83. Prueba 2 Alert XSS

Fuente: Autor, SiembraViva.com

La página Web Siembraviva cuenta con controles de seguridad que ayudan a proteger de estos ataques XSS, luego de realizarse pruebas se puede deducir que las medidas tomadas para evitar o mitigar este tipo de ataques son eficaces y se deben revisar periódicamente cuales técnicas en este tipo de ataques van actualizándose o que otras son desarrolladas y evitar daños a la aplicación.

7.6.2. Inyección SQL.

Un ataque de inyección SQL consiste en inyectar datos a una consulta SQL desde el apartado de cliente, modificar datos como (insertar/actualizar/borrar), lograr operaciones administrativas en la base de datos y hasta poder reiniciar el DBMS acompañado de una infinidad de posibilidades que hacen de este ataque una vulnerabilidad critica para el sistema.

Los ataques de Inyección SQL están divididos en tres clases:

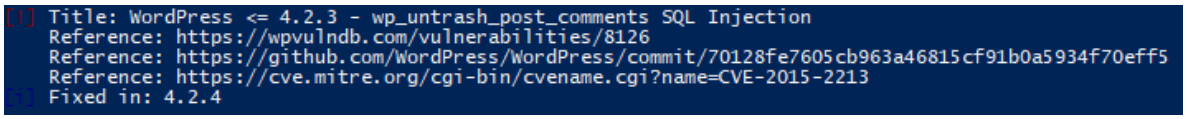
- **Inband:** los datos son extraídos usando el mismo canal que es usado para inyectar el código SQL. Este es el tipo de ataque más simple, en el que los datos recibidos se muestran en la propia aplicación web.
- **Out-of-band:** los datos son extraídos usando un canal diferente (por ejemplo, un correo con el resultado de la consulta es generado y enviado al auditor).
- **Inferetial:** no hay transferencia de datos, pero el auditor puede reconstruir la información enviando peticiones y observando el comportamiento que mantiene el servidor de bases de datos.

En la prueba se busca reconstruir de una forma lógica una consulta SQL correcta, si la aplicación devuelve un mensaje de error, se reconstruye fácilmente de forma lógica a la consulta original.

7.6.2.1. Análisis inyección SQL.

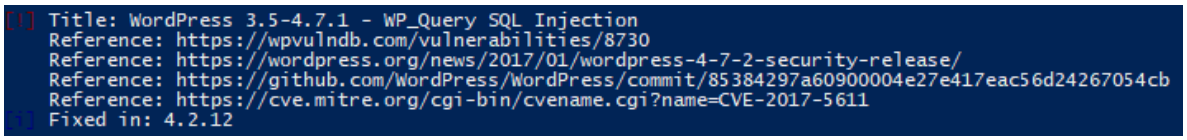
En las figuras 84 a la 88, se realizó el análisis de vulnerabilidades con la aplicación wpscan logrando detectar cerca de 41 vulnerabilidades de las cuales 4 de ellas relacionadas a vulnerabilidades SQL inyección debido a extensiones o plugins desactualizados. Una vulnerabilidad es de tipo *Potential SQL Injection*.

Figura 84. SQL Injection wp_untrash_post_comments



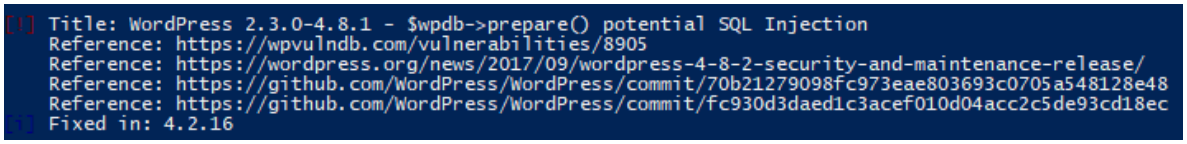
Fuente: Autor WPscan

Figura 85. SQL Injection wp_Query



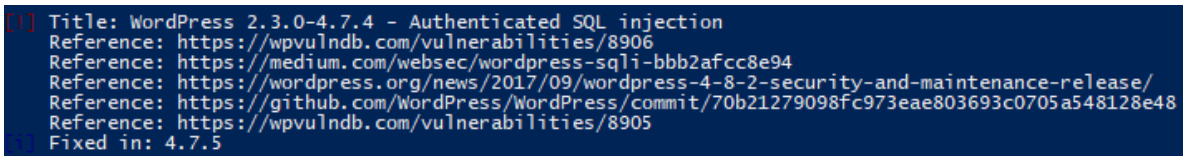
Fuente: Autor WPscan

Figura 86. Potential SQL Injection $wpdb->prepare()



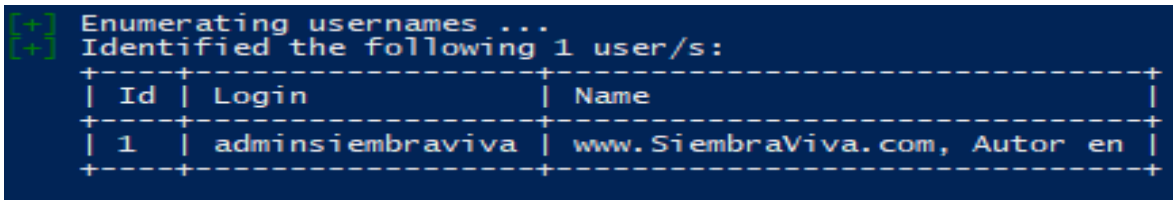
Fuente: Autor WPscan

Figura 87. SQL Injection Authenticated



Fuente: Autor WPscan

Figura 88. Enumerating usernames



Fuente: Autor WPscan

7.8. PRUEBAS DE DENEGACIÓN DE SERVICIO

Las pruebas de denegación de servicio (en inglés, Denial of Service, Dos), son un ataque utilizado para interrumpir la comunicación del servidor con los visitantes válidos del sitio web. El objetivo fundamental de un ataque DoS es usar usuarios maliciosos para aumentar el tráfico de conexión al servidor y de esta forma conseguir hacer la conexión incapaz de responder debido al gran volumen de peticiones que recibe.

Sin embargo, esta prueba no fue posible realizarla debido a que el administrador del sistema no autorizo la ejecución de dicha prueba ya que el riesgo de afectar el servicio que prestan a sus clientes es muy alto.

8. INFORMES

Como resultado del proyecto se anexa el informe final de las pruebas realizadas, el nivel de seguridad y los controles definidos de acuerdo con los resultados de las pruebas para el mejoramiento de la seguridad del sistema e-Commerce SiembraViva.com.

8.1. CONTROLES DE SEGURIDAD

Propósito Control	Descripción
Definición política de seguridad	
Definir la política de seguridad en la que se fundamenta la compañía	Cada desarrollador, propietarios y usuarios de la página Siembraviva debe conocer la política de seguridad, en el caso de los usuarios se debe comunicar al realizarse el registro.
Evaluar y analizar la política de seguridad	Periódicamente se debe actualizar y depurar la política de la organización, es importante documentarse sobre las amenazas que se van dando en el entorno informático.
Estructurar la seguridad de la información	
Acuerdos de privacidad	La protección de los datos de los usuarios es un factor importante, definir qué datos se protegen y cuáles no. Los usuarios deben confiar en la seriedad de la página al manejar los datos.
Definir roles en la política de seguridad de información	Se debe tener claridad en las tareas y quien procede a realizar estas tareas, es vital la organización de la política y estructurarla para no dejar factores sin control. El tipo de

	página que es siembraviva necesita una política clara y definida.
Manejo de Activos	
Uso responsable de los activos	El procesamiento de la información debe complementarse con políticas de uso responsables de los activos que la procesan. Definir, documentar y gestionar de manera correcta estos activos.
Definición de los activos	Los activos que intervienen directa o indirectamente en la organización se deben vigilar constantemente, además que deben estar identificados.
Gestión recursos humanos	
Actores y sus deberes	Las personas encargadas de desarrollar y mantener la aplicación web deben tener definidas sus responsabilidades para que se garantice la seguridad de la información.
Seguridad informática a nivel de hardware	
Consideraciones de los computadores	Los desarrolladores que intervienen en el proceso de implementación y construcción de la página siembraviva deben contar con las medidas de seguridad para proteger el código fuente y no filtrar información importante del core de la aplicación. Se debe implementar seguridad perimetral como CCTV, sensores o herramientas de protección.

Seguridad informática a nivel de hardware	
Seguridad perimetral	Implementar soluciones para la detección de intrusos en tiempo real y complementar esto con políticas de seguridad que ayuden a mitigar las acciones que puedan afectar la aplicación.
Política de respaldo	Las aplicaciones web no están exentas de fallas, por esta razón es importante que se hagan copias de seguridad de la información y del desarrollo. Se debe definir la cantidad de tiempo entre los respaldos.
Control de acceso	
Roles de los usuarios y sus privilegios	Los usuarios y los desarrolladores deben tener definido el alcance de cada uno en la aplicación, no se debe improvisar en los permisos de cada persona que interactúe con la aplicación.
Política de contraseñas	Es importante tener una definición clara de los componentes para que una contraseña sea segura o no; número de caracteres minino, mayúsculas o minúsculas, caracteres especiales, entre otros. Además, cuáles son los mecanismos de recuperación de las contraseñas.
Control de acceso	Los accesos que se den dentro de la aplicación deben ser guardados y documentados para posibles procesos de auditoria interna o externa.

8.2. VALORACIÓN DEL RIESGO

Para realizar la valoración del riesgo se utilizó la siguiente tabla cualitativa teniendo en cuenta la frecuencia del impacto de las amenazas.

Valor			**Criterio**
100	Muy frecuente	MF	A diario
10	Frecuente	F	Mensualmente
1	Normal	FN	Una vez al año
1/10	Poco frecuente	PF	Cada varios años

MA: muy alto
A: alto
M: medio
B: bajo
MB: muy bajo

Dimensiones de impacto

D = Disponibilidad
I = Integridad
C = Confidencialidad
A = Autenticidad
T = Trazabilidad

ACTIVO	AMENAZA	IMPACTO					F	RIESGO
		D	I	C	A	T		
APLICACIONES INFORMATICAS	[E.1] Errores de los usuarios						F	
	[E.2] errores del administrador						FN	
	[E.4] Errores de configuración						FN	
	[E.14] Escapes de información						PF	
	[E.18] Destrucción de información						PF	
	[A.11] Acceso no autorizado						FN	
	[A.15] Modificación de la información						PF	
SERVICIOS	[E.20] Vulnerabilidades de los programas						PF	
	[A.5] Suplantación de la identidad del usuario						FN	
	[A.8]Difusión de Software dañino						FN	
	[A.24]Denegación de Servicios						PF	
REDES DE COMUNICACIONES	[N.*] Desastres Naturales						PF	
	[I.5]Avería de origen físico o lógico						FN	
	[I.8]Fallo de Servicio de comunicaciones						PF	
	[E.2] Errores del administrador						PF	
	[A.4]Manipulación de Configuración.						PF	
EQUIPAMIENTO INFORMATICO	[N.1] Fuego.						PF	
	[I.2] Daños por Agua.						PF	
	[I.5] Avería de origen físico o lógico						PF	
	[E.23] Errores de mantenimiento/ actualización de Equipos						FN	

	(hardware).							
	[A.11] Acceso no Autorizado.						PF	
	[A.23] Manipulación de los equipos.						FN	
EQUIPAMIENTO AUXILIAR	[I.5] Avería de origen físico o lógico						PF	
INSTALACIONES	[A.26] Ataque destructiva						PF	
PERSONAL	[E.7] Deficiencia en la organización.						FN	
	[E.15]Alteración accidental de la información						FN	
	[A.30]Ingeniería Social						PF	

8.3. INFORME DE PRUEBAS

8.3.1. Resumen ejecutivo.

El foco principal del análisis en el sitio web SiembraViva.com se centró en identificar vulnerabilidades de seguridad mediante el análisis de resultados de las pruebas de pentesting aplicando la metodología OWASP en el sistema e-commerce SiembraViva.

La metodología usada para la investigación y las pruebas de pentesting incluyen entre otras herramientas software de rastreo entrada y salida de resultados, validaciones de datos, Sniffers, mapeo de versiones de software, software de mapeo de debilidades de parches, actualizaciones y control de versiones en plugin, themes, complementos, software no seguro e ingeniería social.

Todas las pruebas de caja negra fueron realizadas verificando la funcionalidad del sitio sin entrar en la manipulación de la estructura interna del código fuente o acceso a servidores, detalles de implementación o escenarios internos de producción.

En las pruebas de caja negra nos enfocamos en las entradas y salidas que nos arrojaban las herramientas de pentesting para obtener el detalle de cuales de estas entradas y salidas presentan algún impacto en vulnerabilidades del sitio web.

El principal objetivo con los resultados de las pruebas OWASP es identificar y con ello minimizar los problemas de seguridad, mejorando la administración mediante el análisis de los resultados de las pruebas de pentesting, estas pruebas se empezaron a ejecutar en enero de 2017 con la prueba de intrusión en el sitio web SiembraViva.com, recopilando la mayor cantidad de información de la manera como opera. Esta prueba de reconocimiento pasivo de información se realizó sin tocar los servidores ni alterar la operatividad del sitio web.

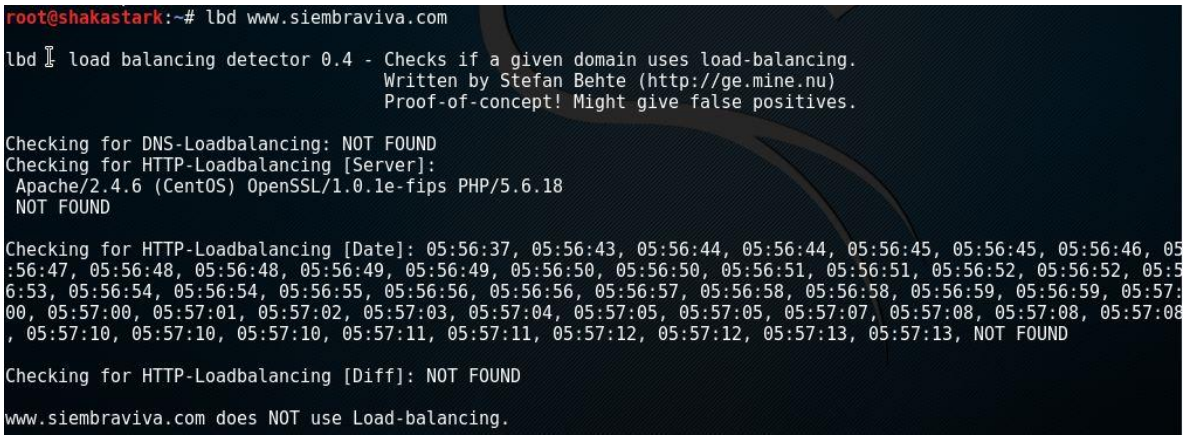
Se utilizaron herramientas de acceso público, motores de búsqueda, scanners, enviando peticiones HTTP simples, o peticiones especialmente diseñadas, para forzar a la aplicación filtrar información al exterior con mensajes de error devueltos, o revelar las versiones y tecnología en uso por la aplicación web.

Luego se realizaron los análisis de infraestructura o topología web del servidor, revelando importante información de datos sensibles del código fuente a nivel de cliente sin ser intrusivos, métodos y funciones HTTP permitidos, métodos de autenticación y respuesta, validaciones de datos y hasta las configuraciones de infraestructura tecnológicas usadas en el sistema web e-commerce.

Las auditorías manuales de los niveles de cifrado SSL son débiles mediante OpenSSL. Conexión mediante consola a SiembraViva.com con SSLv2.

Se encontraron varias vulnerabilidades con el certificado emitido por Comodo CA el 26 de febrero de 2016 con clave publica de tipo RSA a 2048 bits, su algoritmo de firma digital sha256 con encriptación RSA. Este servidor es vulnerable al ataque de POODLE. Grado cubierto C. (CVE-2014-3566). Ver en la figura 89 el protocolo SSL3.0, tal como se utiliza en OpenSSL a través de 1.0.1i y otros productos, utiliza un relleno de CBC no determinista, lo que facilita a los atacantes del tipo “man-in-the-middle” obtener datos de texto claro a través de un ataque “padding-oracle”.

Figura 89. Vulnerabilidad con Certificado Comodo CA

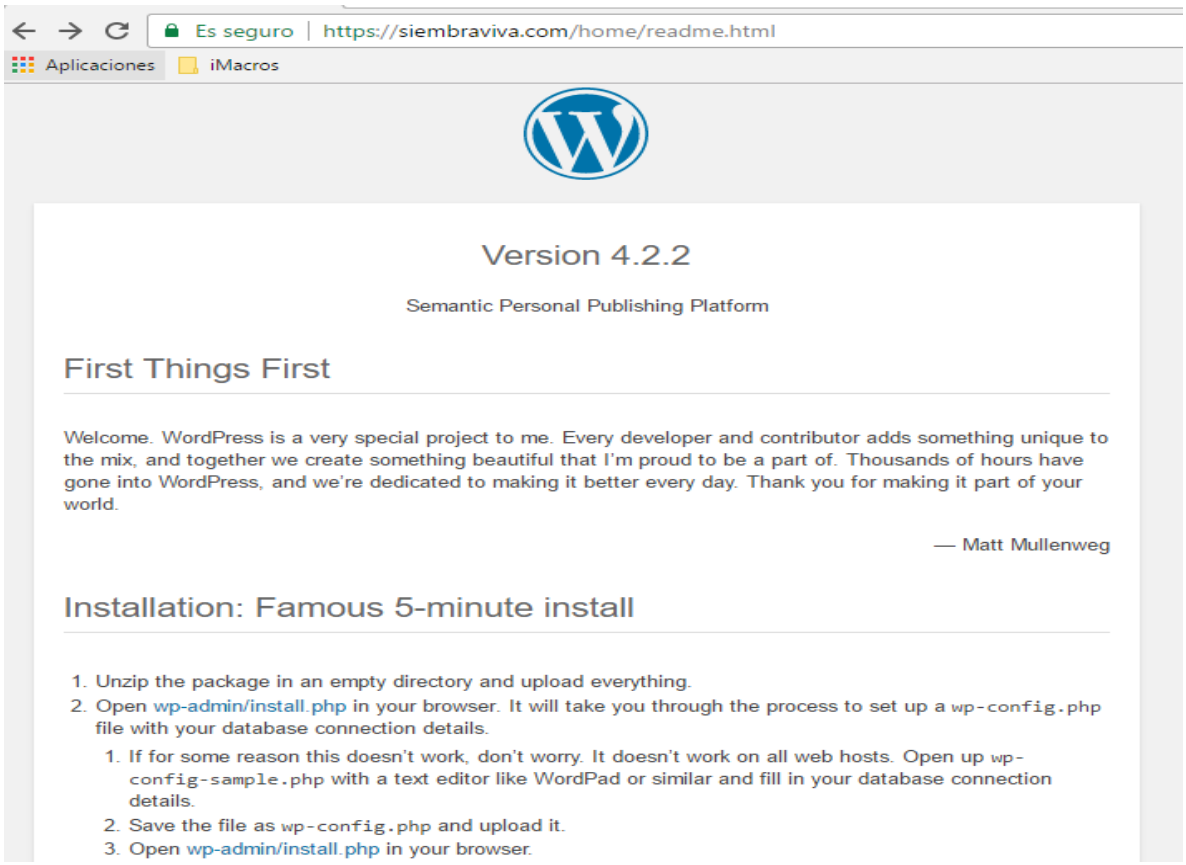


Fuente: Autor, SiembraViva.com

Se logro evidenciar que el servidor del sitio web SiembraViva.com a la fecha de las pruebas de octubre 2017, tal como lo muestra la figura 90, no contaba con un balanceador de cargas. Se identifico que el sitio web corre sobre una distribución CentOS de Linux, en la revisión de las interfaces de administración empleadas

para gestionar las diferentes partes de la arquitectura del sitio web, se encuentra que el sitio web SiembraViva.com trabaja bajo el sistema de gestión de contenidos Wordpress en su versión tentativa 4.2.2, según los datos arrojados por la herramienta Wappalyzer. Esta información suele proporcionar detalles que pueden ser utilizados en la recolección de información e identificación del sistema de gestión de contenidos.

Figura 90. Archivo Readme, marzo 2017



Fuente: Autor, SiembraViva.com

En la figura 91 a fecha de noviembre de 2017, se evidencia la correcta actualización de WordPress 4.2.2 a 4.8 con la cual se solucionan varias de las vulnerabilidades encontradas en la versión anterior.

Figura 91. Archivo Readme, febrero 2018

Fuente: Autor, SiembraViva.com

8.3.2. Consideraciones técnicas generales

8.3.2.1. Alcance de la evaluación.

La revisión de la página Siembraviva se realizó para detectar falencias en la seguridad y para verificar si los componentes implementados en el desarrollo están actualizados.

El manual de pruebas OWASP cuenta con un gran número de pruebas que contribuyen a conocer las deficiencias del sistema a analizar; las pruebas se pueden categorizar dependiendo de qué tanta afectación tiene en el sistema

realizar la prueba. Existen pruebas que no afectan la operación normal de la página y otras que afectan potencialmente la operación del sistema.

En consenso con el propietario y desarrollador de la página se han realizado las pruebas que no afecten la operación del sistema, la aplicación debe estar en operación 24/7 dado que los acuerdos con los usuarios son atender sus solicitudes en tiempo real. Al realizar la reunión para explicar las pruebas que debían ser aplicadas sobre el sistema los propietarios solicitaron que las pruebas de acceso a código o que alteren la disponibilidad de la página debían ser descartadas.

Las pruebas que fueron aplicadas dentro de la página fueron las que se relacionan con la identificación de la seguridad perimetral del sistema, además de identificar que tan actualizados están los componentes que integran el sistema.

La definición del alcance de la evaluación aplicada sobre la aplicación Web Siembraviva se define en analizar y proponer medidas que contribuyan a proteger el contenido de los usuarios y a asegurar que los pilares de la seguridad de la información se cumplan; siendo la disponibilidad de la aplicación algo fundamental de ser garantizado.

8.3.2.2. Objetivos de la evaluación.

1. Realizar una revisión de las vulnerabilidades y amenazas existentes en la página Web de e-commerce siembraviva.com basándose en la metodología OWASP.
2. Recomendar medidas para mitigar los riesgos a los cuales se encuentre propensos la página.
3. Presentar un informe que presente los resultados obtenidos de la aplicación de la metodología OWASP.

8.3.2.3. Hallazgos encontrados.

Lenguaje de Programación PHP 5.6.18

Para esta versión de PHP se tienen alrededor de 38 vulnerabilidades reportadas en el CVE, dentro de las cuales 7 vulnerabilidades están muy marcadas en el sistema analizado y pueden afectar la disponibilidad del sitio, dentro del listado resaltamos las siguientes evidenciadas en las figuras 92, 93 y 94.

- La función multipart_buffer_headers en main/rfc1867.c presenta una falla en el algoritmo, esto hace vulnerable las aplicaciones Web a ataques de denegación de servicio. CVE-2015-4024
- A través de un argumento que se puede inyectar en las funciones set_include_path, tempnam, rmdir, o readlink, se pueden realizar accesos sin autorización sobre los archivos o directorios. CVE-2015-4025
- La función ftp_genlist() sufre un colapso cuando se le envían números muy extensos, se desborda la función. Se considera sencilla de explotar. CVE-2015-4022
- Esta función phar_parse_tarfile al recibir un nombre nulo falla y sufre un desbordamiento de números enteros. CVE-2015-4021

Tabla 10. Vulnerabilidades CVE PHP 5.6.18

CVE ID	Tipo de vulnerabilidad	Fecha de publicación	Puntuación CVSS	Acceso	Complejidad
CVE-2017-16642	+Info	2017-11-07	5.0	Remoto	Bajo
CVE-2016-7478	DoS	2017-01-11	5.0	Remoto	Bajo
CVE-2016-6297	DoS Overflow	2016-07-25	6.8	Remoto	Medio
CVE-2016-6296	DoS Overflow	2016-07-25	7.5	Remoto	Bajo
CVE-2016-6295	DoS	2016-07-25	7.5	Remoto	Bajo
CVE-2016-6294	DoS	2016-07-25	7.5	Remoto	Bajo
CVE-2016-	DoS	2016-07-25	4.3	Remoto	Medio

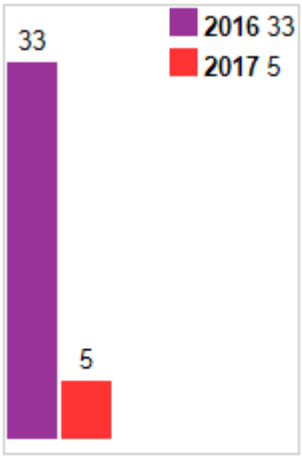
CVE ID	Tipo de vulnerabilidad	Fecha de publicación	Puntuación CVSS	Acceso	Complejidad
6292					
CVE-2016-6291	DoS Overflow Mem. Corr. +Info	2016-07-25	7.5	Remoto	Bajo
CVE-2016-6290	DoS	2016-07-25	7.5	Remoto	Bajo
CVE-2016-6289	DoS Overflow	2016-07-25	6.8	Remoto	Medio
CVE-2016-5773	DoS Exec Code	2016-08-07	7.5	Remoto	Medio
CVE-2016-5772	DoS Exec Code	2016-08-07	7.5	Remoto	Bajo
CVE-2016-4473	Exec Code	2017-06-08	7.5	Remoto	Bajo
CVE-2015-8994	+Priv	2017-03-02	6.8	Remoto	Medio
CVE-2016-3141	DoS Overflow Mem. Corr.	2016-03-31	7.5	Remoto	Bajo
CVE-2016-4070	DoS Overflow	2016-05-20	5.0	Remoto	Bajo
CVE-2016-4537	DoS	2016-05-21	7.5	Remoto	Medio
CVE-2016-4544	DoS Overflow	2016-05-21	7.5	Remoto	Bajo
Fuente: CVE Details, Security Vulnerabilities PHP 5.6.18. https://www.cvedetails.com/version/192558/PHP-PHP-5.6.18.html					

Figura 92. Tendencias de la vulnerabilidad a lo largo del tiempo PHP 5.6.18

Year	# of Vulnerabilities	DoS	Code Execution	Overflow	Memory Corruption	Sql Injection	XSS	Directory Traversal	Http Response Splitting	Bypass something	Gain Information	Gain Privileges	CSRF	File Inclusion	# of exploits
2016	33	31	8	18	2						2				
2017	5	2	2								1	1			
Total	38	33	10	18	2						3	1			
% Of All		86.8	26.3	47.4	5.3	0.0	0.0	0.0	0.0	0.0	7.9	2.6	0.0	0.0	

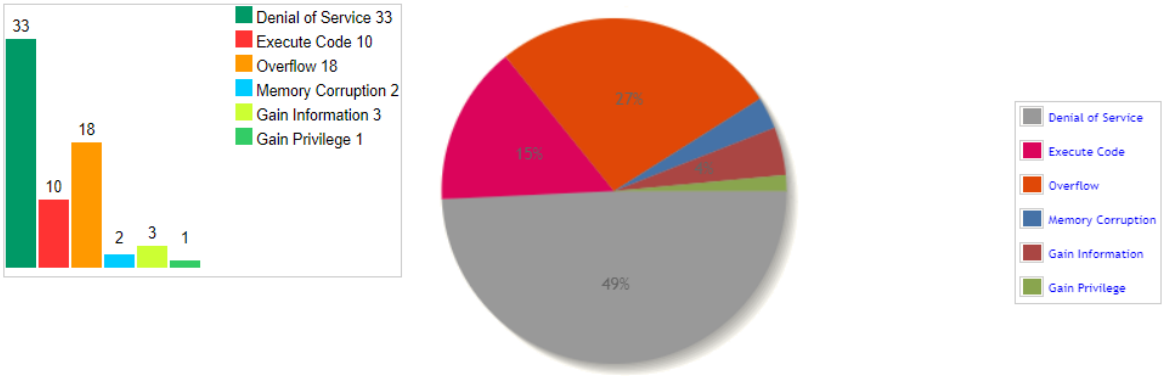
Fuente: Vulnerability Statistics CVE

Figura 93. Vulnerabilidades por año PHP 5.6.18



Fuente: Vulnerability Statistics CVE

Figura 94. Vulnerabilidades por tipo PHP 5.6.18



Fuente: Vulnerability Statistics CVE

Apache 2.4.6

Dentro del servidor en la aplicación se encuentra un componente llamado *mod_cache* el cual se le encontró una nueva función, al enviarse una serie de parámetros puede llevar a una falla del servicio, afecta la disponibilidad de la aplicación. CVE-2013-4352.

Por otro lado, Apache Server 2.4.6 presenta alrededor de 17 vulnerabilidades reportadas en el CVE.

Tabla 11. Vulnerabilidades Apache Server 2.4.6

CVE ID	Tipo de vulnerabilidad	Fecha de publicación	Puntuación CVSS	Acceso	Complejidad
CVE-2017-9798	DoS +Info	2017-09-18	5.0	Remoto	Bajo
CVE-2017-9788	DoS +Info	2017-07-13	6.4	Remoto	Bajo
CVE-2017-7679	Overflow	2017-06-19	7.5	Remoto	Bajo
CVE-2016-8743	Http R.Spl.	2017-07-27	5.0	Remoto	Bajo
CVE-2016-8612		2018-03-09	3.3	Local Network	Bajo
CVE-2016-2161		2017-07-27	5.0	Remoto	Bajo
CVE-2016-0736		2017-07-27	5.0	Remoto	Bajo
CVE-2015-3185	Bypass	2015-07-20	4.3	Remoto	Medio
CVE-2014-8109	Bypass	2014-12-29	4.3	Remoto	Medio
CVE-2014-0231	DoS	2014-07-20	5.0	Remoto	Bajo
CVE-2014-0226	DoS Exec Code Overflow +Info	2014-07-20	6.8	Remoto	Medio
CVE-2014-0118	DoS	2014-07-20	4.3	Remoto	Medio
CVE-2014-0117	DoS	2014-07-20	4.3	Remoto	Medio
CVE-2014-0098	DoS	2014-03-18	5.0	Remoto	Bajo
CVE-2013-6438	DoS	2014-03-18	5.0	Remoto	Bajo
CVE-2013-4352	DoS	2014-07-20	4.3	Remoto	Medio

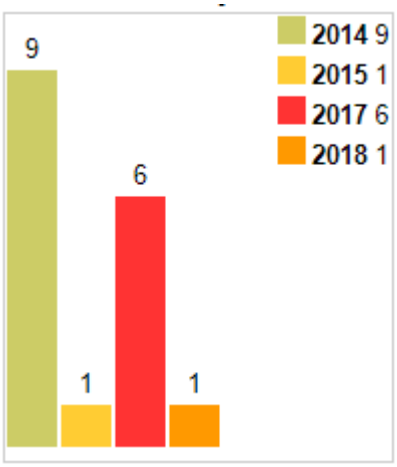
CVE ID	Tipo de vulnerabilidad	Fecha de publicación	Puntuación CVSS	Acceso	Complejidad
Fuente: CVE Details, Security Vulnerabilities Apache HTTP Server 2.4.6. https://www.cvedetails.com/version/161846/Apache-Http-Server-2.4.6.html					

Figura 95. Tendencias vulnerabilidad a lo largo del tiempo Apache Server 2.4.6

Year	# of Vulnerabilities	DoS	Code Execution	Overflow	Memory Corruption	Sql Injection	XSS	Directory Traversal	Http Response Splitting	Bypass something	Gain Information	Gain Privileges	CSRF	File Inclusion	# of exploits
2014	9	8	1	1						1	1				1
2015	1									1					
2017	6	1		1					1		1				
2018	1														
Total	17	9	1	2					1	2	2				1
% Of All		52.9	5.9	11.8	0.0	0.0	0.0	0.0	5.9	11.8	11.8	0.0	0.0	0.0	

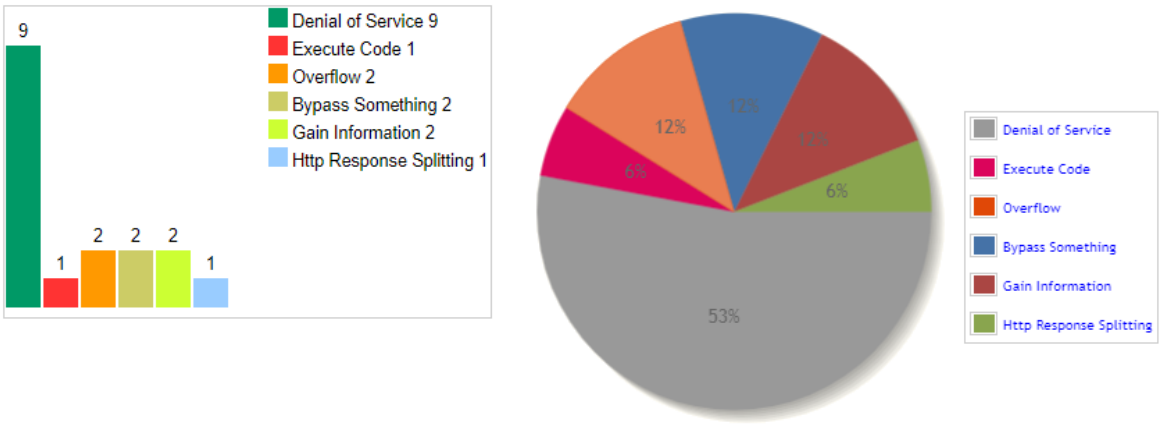
Fuente: Vulnerability Statistics CVE

Figura 96. Vulnerabilidades por año Apache Server 2.4.6



Fuente: Vulnerability Statistics CVE

Figura 97. Vulnerabilidades por tipo Apache Server 2.4.6



Fuente: Vulnerability Statistics CVE

OpenSSL 1.0.1 e-fips

Esta versión del paquete de soluciones criptográficas es vulnerable al ataque conocido como HeartBleed y sus derivados, es un ataque que pone en alto riesgo la información privada de la aplicación y con esta información poder manipular subsistemas o servicios dentro de la aplicación. CVE-2014-0160. Los resultados se resaltan gráficamente en las figuras 98, 99 y 100.

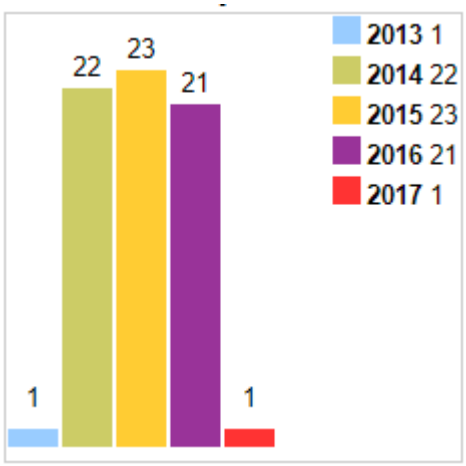
Las (1) implementaciones TLS y (2) DTLS en OpenSSL 1.0.1 antes 1.0.1g no manejan adecuadamente los paquetes Heartbeat Extension, lo que permite a los atacantes remotos obtener información sensible de la memoria del proceso a través de paquetes diseñados que activan sobre el búfer sobre-lectura. como se demostró al leer claves privadas, relacionadas con d1_both.c y t1_lib.c, también conocido como el error Heartbleed.

Figura 98. Tendencias vulnerabilidad a lo largo del tiempo en Openssl 1.0.1e

Year	# of Vulnerabilities	DoS	Code Execution	Overflow	Memory Corruption	Sql Injection	XSS	Directory Traversal	Http Response Splitting	Bypass something	Gain Information	Gain Privileges	CSRF	File Inclusion	# of exploits
2013	1	1													
2014	22	16	1	3						1	3				2
2015	23	17		4	4						1				
2016	21	14		5	2						5				
2017	1			1											
Total	68	48	1	13	6					1	9				2
% Of All		70.6	1.5	19.1	8.8	0.0	0.0	0.0	0.0	1.5	13.2	0.0	0.0	0.0	

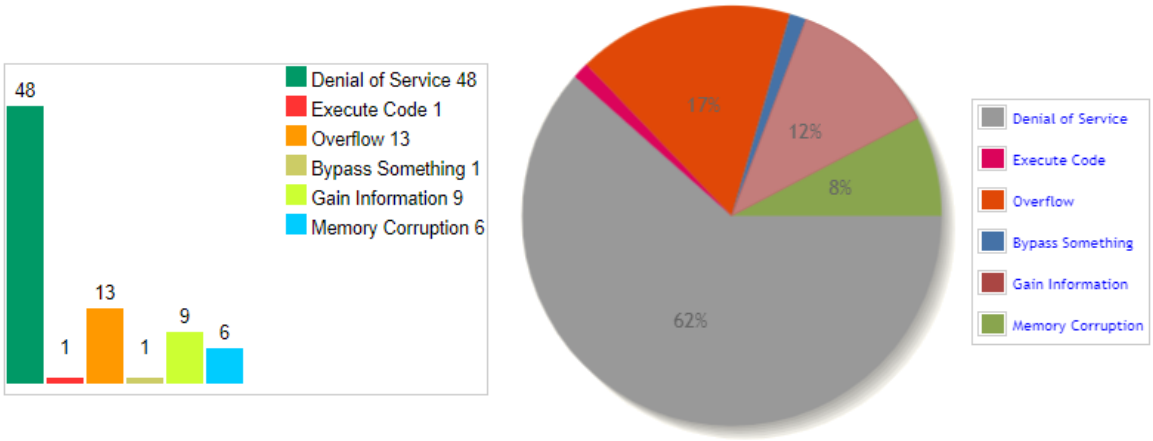
Fuente: Vulnerability Statistics CVE

Figura 99. Vulnerabilidades por año en Openssl 1.0.1e



Fuente: Vulnerability Statistics CVE

Figura 100. Vulnerabilidades por tipo Openssl 1.0.1e



Fuente: Vulnerability Statistics CVE

Gestor de Contenido WordPress 4.2.2

Vulnerabilidad en el componente quick draft, a través de un campo de entrada dentro de la aplicación se puede realizar un ataque de Cross Site Scripting. De realizarse este ataque se puede afectar la integridad de los datos dentro de la aplicación. CVE-2015-5623. Los resultados se resaltan gráficamente en las figuras 101, 102 y 103.

Por otro lado, WordPress 4.2.2 presentaba alrededor de 4 vulnerabilidades reportadas en el CVE.

Tabla 12. Vulnerabilidades Wordpress 4.2.

CVE ID	Tipo de vulnerabilidad	Fecha de publicación	Puntuación CVSS	Acceso	Complejidad
CVE-2018-5776	XSS	2018-01-18	4.3	Remoto	Medio
CVE-2017-14719	Dir. Trav.	2017-09-23	5.0	Remoto	Bajo

CVE ID	Tipo de vulnerabilidad	Fecha de publicación	Puntuación CVSS	Acceso	Complejidad
CVE-2015-5623	Bypass	2015-08-03	4.0	Remoto	Bajo
CVE-2015-5622	XSS	2015-08-03	3.5	Remoto	Medio
Fuente: CVE Details, Security Vulnerabilities WordPress 4.2.2. https://www.cvedetails.com/version/185073/Wordpress-Wordpress-4.2.2.html					

CVE-2018-5776: Todas las versiones de WordPress antes de 4.9.2, tienen vulnerabilidad a ataques de tipo XSS en los archivos de recuperación de Flash en MediaElement (wp-includes/js/mediaelement), (CVE Details, 2018).

CVE-2017-14719: Antes de la versión 4.8.2, WordPress era vulnerable a un ataque transversal de directorio durante las operaciones de descompresión en los componentes ZipArchive y PclZip.

CVE-2015-5623: WordPress antes de 4.2.3 no verificaba correctamente la capacidad edit_posts, que permite a los usuarios autenticados remotos eludir las restricciones de acceso previstas y crear borradores aprovechando el rol de Suscriptor, como lo demuestra una acción de guardado posterior a wp-admin/post.php.

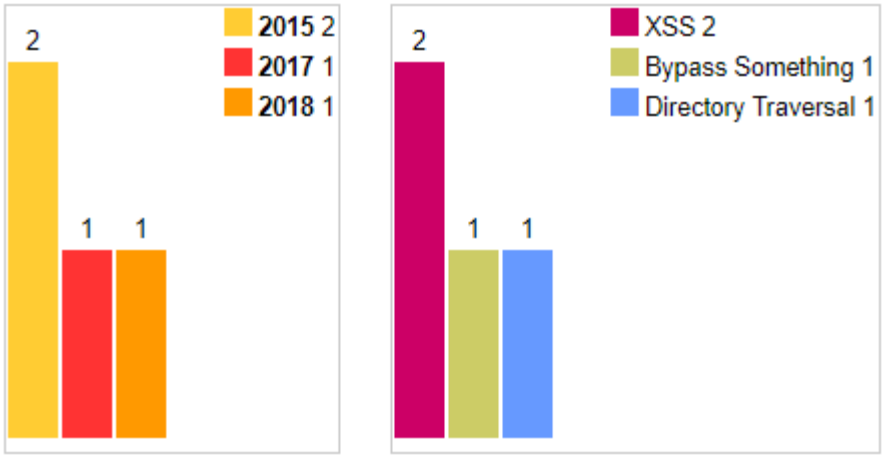
CVE-2015-5622: La vulnerabilidad de scripts cruzados (XSS) en WordPress antes de la versión 4.2.3 permite a los usuarios autenticados remotos inyectar script web o HTML arbitrario al aprovechar el rol Autor o Colaborador para colocar un código abreviado en un elemento HTML relacionado con wp-includes/kses.php y wp-includes/shortcodes.php. (CVE Details, 2015).

Figura 101. Tendencias vulnerabilidad a lo largo del tiempo en WordPress 4.2.2

Year	# of Vulnerabilities	DoS	Code Execution	Overflow	Memory Corruption	Sql Injection	XSS	Directory Traversal	Http Response Splitting	Bypass something	Gain Information	Gain Privileges	CSRF	File Inclusion	# of exploits
2015	2						1			1					
2017	1							1							
2018	1						1								
Total	4						2	1		1					
% Of All		0.0	0.0	0.0	0.0	0.0	50.0	25.0	0.0	25.0	0.0	0.0	0.0	0.0	

Fuente: Vulnerability Statistics CVE

Figura 102. Vulnerabilidades por año y tipo en WordPress 4.2.2



Fuente: Vulnerability Statistics CVE

Plugins vulnerables o que requieren actualización

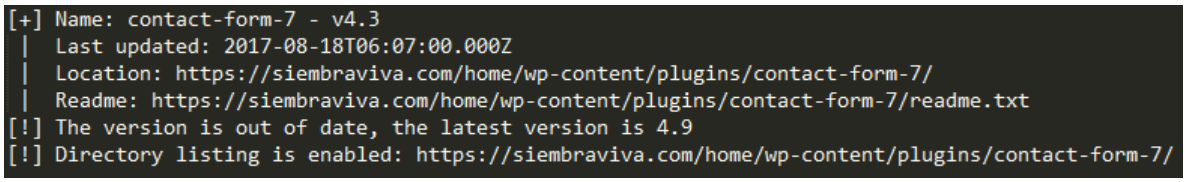
En octubre de 2017 se realizó un análisis completo en la identificación de los complementos o Plugins que utiliza SiembraViva para la gestión de su contenido y administración del sitio web, para lo cual se utilizó la herramienta Wp-Scan 2.9.2 Dev. En las figuras 103 a la 111 se pueden ver los detalles de los plugin vulnerables que requieren atención.

Figura 103. Advanced-post-slider - v2.5.0

```
[+] Name: advanced-post-slider - v2.5.0
 |  Latest version: 2.5.0 (up to date)
 |  Last updated: 2015-12-26T14:46:00.000Z
 |  Location: https://siembraviva.com/home/wp-content/plugins/advanced-post-slider/
 |  Readme: https://siembraviva.com/home/wp-content/plugins/advanced-post-slider/readme.txt
[!] Directory listing is enabled: https://siembraviva.com/home/wp-content/plugins/advanced-post-slider/
```

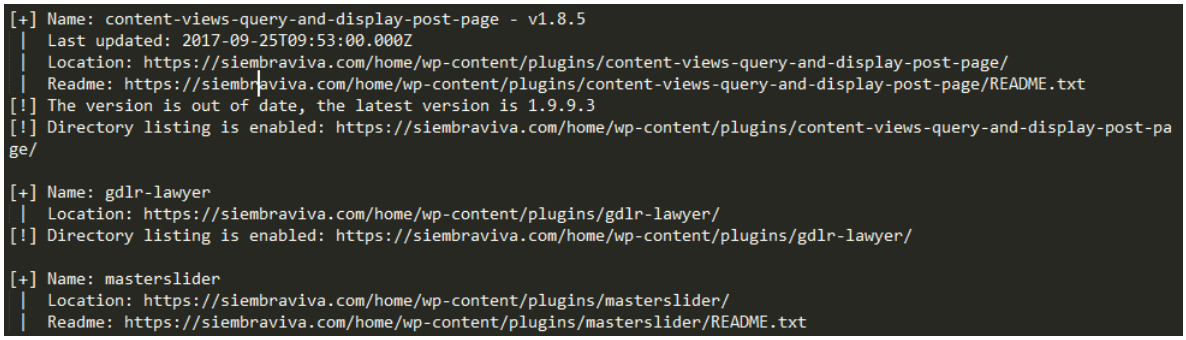
Fuente: Wp-Scan 2.9.2. Análisis octubre 2017.

Figura 104. Contact-form-7 - v4.3



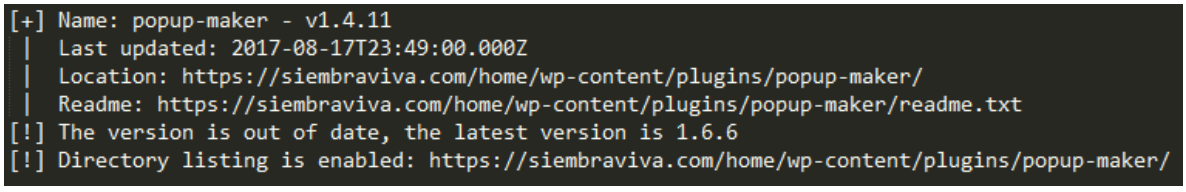
Fuente: Wp-Scan 2.9.2. Análisis octubre 2017.

Figura 105. Content-views-query-and-display-post-page - v1.8.5



Fuente: Wp-Scan 2.9.2. Análisis octubre 2017.

Figura 106. Popup-maker - v1.4.11



Fuente: Wp-Scan 2.9.2. Análisis octubre 2017.

Figura 107. Popup Maker <= 1.6.4 - Authenticated Cross-Site Scripting (XSS)

```
[!] Title: Popup Maker <= 1.6.4 - Authenticated Cross-Site Scripting (XSS)
    Reference: https://wpvulndb.com/vulnerabilities/8878
    Reference: https://plugins.trac.wordpress.org/changeset/1697216/#file3
    Reference: https://jvn.jp/en/jp/JVN92921024/index.html
    Reference: https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2017-2284
[i] Fixed in: 1.6.5
```

Fuente: Wp-Scan 2.9.2. Análisis octubre 2017.

Figura 108. Recipes-writer - v1.0.4

```
[+] Name: recipes-writer - v1.0.4
 |  Latest version: 1.0.4 (up to date)
 |  Last updated: 2014-09-19T03:13:00.000Z
 |  Location: https://siembraviva.com/home/wp-content/plugins/recipes-writer/
 |  Readme: https://siembraviva.com/home/wp-content/plugins/recipes-writer/readme.txt
[!] Directory listing is enabled: https://siembraviva.com/home/wp-content/plugins/recipes-writer/
```

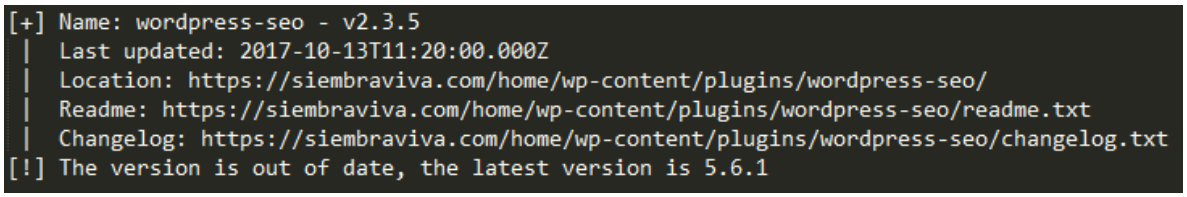
Fuente: Wp-Scan 2.9.2. Análisis octubre 2017.

Figura 109. WP Google Map Plugin <= 2.3.9 - Authenticated (XSS)

```
[!] Title: WP Google Map Plugin <= 2.3.9 - Authenticated Cross-Site Scripting (XSS)
    Reference: https://wpvulndb.com/vulnerabilities/8271
    Reference: http://cinu.pl/research/wp-plugins/mail_7f44ee1674f69bba54409d00ca562e8d.html
    Reference: http://blog.cinu.pl/2015/11/php-static-code-analysis-vs-top-1000-wordpress-plugins.html
[i] Fixed in: 3.0.0
```

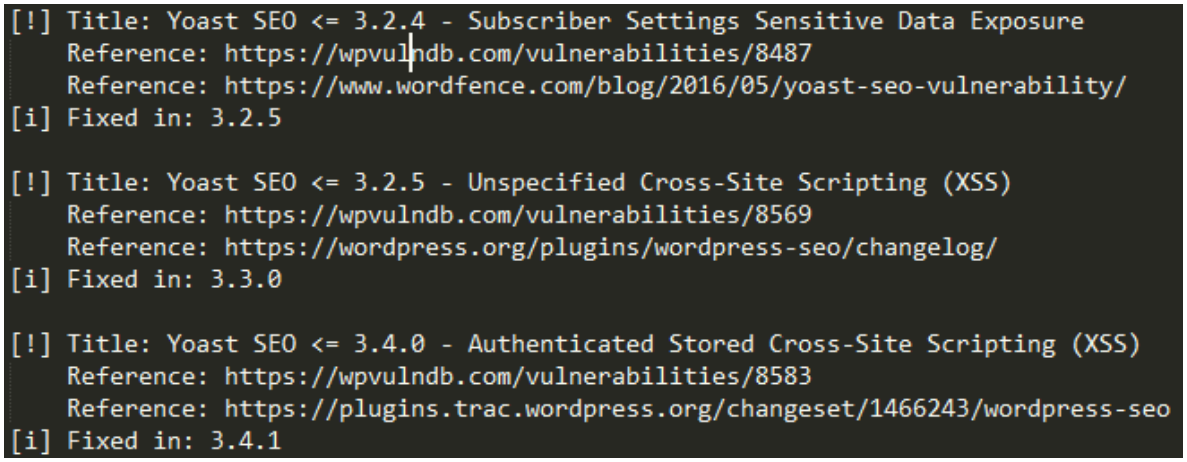
Fuente: Wp-Scan 2.9.2. Análisis octubre 2017.

Figura 110. Wordpress-seo - v2.3.5



Fuente: Wp-Scan 2.9.2. Análisis octubre 2017.

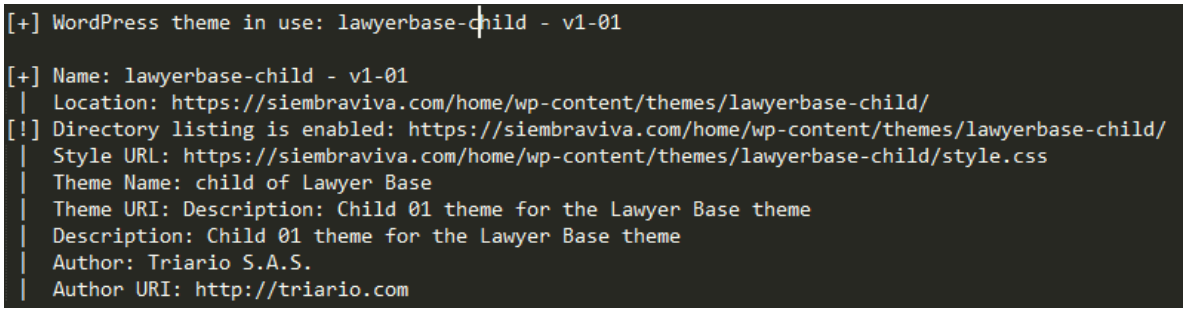
Figura 111. Yoast SEO <= 3.2.5 - Unspecified Cross-Site Scripting (XSS)



Fuente: Wp-Scan 2.9.2. Análisis octubre 2017.

Themes vulnerables o requieren actualización

Figura 112. Lawyerbase-child - v1-01



Fuente: Wp-Scan 2.9.2. Análisis octubre 2017.

Figura 113. Lawyerbase - v1.01

Fuente: Wp-Scan 2.9.2. Análisis octubre 2017.

En las figuras 112 y 113 se pueden observar los detalles de los themes vulnerables que requieren atención.

Resumen de Pruebas Realizadas

Numero de Referencia	Nombre de la Prueba	Efectuada	No Efectuada
OWASP-IG-001	Spiders, Robots y Crawlers	X	
OWASP-IG-002	Descubrimiento/Reconocimiento mediante motores de búsqueda	X	
OWASP-IG-003	Identificación de puntos de entrada de la aplicación	X	
OWASP-IG-004	Pruebas de firma digital de aplicaciones web	X	
OWASP-IG-005	Descubrimiento de Aplicaciones	X	
OWASP-IG-006	Análisis de Códigos de Errores	X	
OWASP-CM-001	Pruebas SSL/TLS (SSL Versión, Algoritmos, longitud de Claves, Validez de Certificado Digital)	X	

Numero de Referencia	Nombre de la Prueba	Efectuada	No Efectuada
OWASP-CM-002	Prueba de DB Listener	X	
OWASP-CM-003	Prueba de Gestión de Configuración de Infraestructura	X	
OWASP-CM-004	Prueba de Gestión de Configuración de Aplicación	X	
OWASP-CM-005	Prueba del Gestor de Extensión de Ficheros	X	
OWASP-CM-006	Antiguo, backup y ficheros no referenciados	X	
OWASP-CM-007	Interfase de Administración de Aplicación e Infraestructura	X	

Numero de Referencia	Nombre de la Prueba	Efectuada	No Efectuada
OWASP-CM-008	Prueba de métodos HTTP y XST	X	
OWASP-AT-001	Transporte de Credenciales sobre canal cifrado	X	
OWASP-AT-002	Prueba para Enumeración de usuarios		X
OWASP-AT-003	Prueba de detección de Cuentas de Usuario Adivinables (Diccionario)		X
OWASP-AT-004	Prueba de Fuerza Bruta		X
OWASP-AT-005	Prueba para evitar los esquemas de autenticación		X
OWASP-AT-006	Prueba de recordatorio de contraseña y restablecimiento	X	
OWASP-AT-007	Prueba de Cierre de Sesión y Gestión de Cache de Navegación	X	

Numero de Referencia	Nombre de la Prueba	Efectuada	No Efectuada
OWASP-AT-008	Prueba de CAPTCHA	X	
OWASP-AT-009	Prueba de Autenticación de Múltiple Factores	X	
OWASP-AT-010	Prueba de Condiciones de Carrera		X
OWASP-SM-001	Prueba del Esquema de Gestión de Sesión	X	
OWASP-SM-002	Prueba de atributos de Cookies		X
OWASP-SM-003	Prueba de Fijación de Sesión		X

Numero de Referencia	Nombre de la Prueba	Efectuada	No Efectuada
OWASP-SM-004	Prueba de Variables de Sesión Expuestas		X
OWASP-SM-005	Prueba de CSRF		X
OWASP-AZ-001	Prueba de Ruta Transversal		X
OWASP-AZ-002	Prueba para Evitar Esquema de Autorización		X
OWASP-AZ-003	Prueba de escalada de privilegios		X

Numero de Referencia	Nombre de la Prueba	Efectuada	No Efectuada
OWASP-BL-001	Prueba de Lógica de Negocio		X
OWASP-DV-001	Prueba de XSS Reflejado		X
OWASP-DV-002	Prueba de XSS Almacenado	X	
OWASP-DV-003	Prueba de XSS basado en DOM	X	
OWASP-DV-004	Prueba de XSS basado en Flash		X
OWASP-DV-005	Inyección SQL	X	
OWASP-DV-006	Inyección LDAP		X
OWASP-DV-007	Inyección ORM		X

Numero de Referencia	Nombre de la Prueba	Efectuada	No Efectuada
OWASP-DV-008	Inyección XML		X
OWASP-DV-009	Inyección SSI		X
OWASP-DV-010	Inyección XPath		X
OWASP-DV-011	Inyección IMAP/SMTP		X
OWASP-DV-012	Inyección de Código	X	
OWASP-DV-013	Inyección de Ordenes del Sistema Operativo		X
OWASP-DV-014	Desbordamiento de *buffer*	X	
OWASP-DV-015	Prueba de Vulnerabilidad incubada		X

Numero de Referencia	Nombre de la Prueba	Efectuada	No Efectuada
OWASP-DV-016	Prueba de HTTP Splitting/Smuggling		X
OWASP-DS-001	Prueba de Ataques a través de Comodines SQL		X
OWASP-DS-002	Bloqueo de Cuentas de Usuarios		X
OWASP-DS-003	Pruebas de DoS mediante Desbordamiento de Buffer		X
OWASP-DS-004	Asignación de Objeto de Usuario Especificado		X

Numero de Referencia	Nombre de la Prueba	Efectuada	No Efectuada
OWASP-DS-005	Entrada de usuario como un contador de bucle		X
OWASP-DS-006	Prueba de Escritura en Disco de data provista por Usuario		X
OWASP-DS-007	Fallo en Liberar Recursos		X
OWASP-DS-008	Almacenamiento de demasiados datos en Sesión		X
OWASP-WS-001	Recopilación de Información de WS		X
OWASP-WS-002	Prueba de WSDL		X
OWASP-WS-003	Prueba en la Estructura del XML		X
OWASP-WS-004	Prueba del XML a nivel de contenido		X

Numero de Referencia	Nombre de la Prueba	Efectuada	No Efectuada
OWASP-WS-005	Prueba de REST/parámetros HTTP GET		X
OWASP-WS-006	Adjuntos SOAP maliciosos		X
OWASP-WS-007	Prueba de Repetición		X
OWASP-AJ-001	Vulnerabilidades Ajax		X
OWASP-AJ-002	Pruebas Ajax		X
Fuente: El autor, Metodologías pruebas OWASP			

9. CONCLUSIONES

Al aplicar las pruebas de la metodología OWASP se obtuvieron resultados de manera global del estado actual de la seguridad en el sistema e-commerce SiembraViva logrando identificar los riesgos a los cuales se veía enfrentado si no se tomaban las acciones correctivas para disminuir el riesgo a su nivel mínimo.

Con el resultado del análisis de riesgos de seguridad de la información del sistema e-commerce SiembraViva.com, se logró establecer los niveles de seguridad implementados en la actualidad que les permitiera con mayor fundamento tomar acciones correctivas evitando posibles fallos y perdida de información debido a vulnerabilidades encontradas en el sistema utilizado.

Las pruebas aplicadas sobre el sistema e-commerce SiembraViva.com fueron enfocadas hacia las pruebas de caja negra las cuales permitieron verificar la funcionalidad sin tomar en cuenta la estructura del software y su código fuente, detalles de su implementación o escenarios de aplicación. En las pruebas de caja negra nos enfocamos en las entradas y salidas que nos arrojaban las herramientas de pentesting para obtener el detalle de cuales de estas entradas y salidas presentan algún impacto en las vulnerabilidades del sitio web.

La aplicación del manual de pruebas OWASP se completó con un informe final para dar a conocer los hallazgos a las partes interesadas; en el caso de SiembraViva.com se presentan informes que dan a conocer las partes que se ven expuestas a ataques, pruebas aplicadas en el sistema y criticidad de las fallas encontradas. El informe es la mejor forma de presentar los problemas de una manera formal para que el administrador o personas involucradas puedan tomar acciones correctivas en la menor brevedad posible.

Fue elaborado el informe ejecutivo y técnico como resultado del análisis del sitio web SiembraViva.com con el cual se logró especificar las vulnerabilidades de seguridad mediante el análisis de resultados de las pruebas de pentesting aplicadas en base a la metodología OWASP.

10. RECOMENDACIONES

Generar una política empresarial y de usuarios para utilizar contraseñas con un grado de complejidad óptimo, hacer que los empleados y los usuarios las cambien periódicamente y dentro de la compañía no sean transmitidas entres los empleados.

Mantener el entorno de producción y core de Wordpress con las más recientes actualizaciones que cumplen con los lineamientos y compatibilidad con cada uno de los complementos, plugin y temas utilizados en el sitio web, verificando cada actualización en el entorno de pruebas para cerciorarse que todo funciona sin ningún tipo de problema.

Realizar una copia de seguridad periódica del sitio web y de las bases de datos, es vital que haya prevención de darse un ataque y no perder los datos de los usuarios y de la organización.

Revisar el manejo de cookies del sitio web, tienen un ciclo de vida muy amplio y esto puede generar ataques de suplantación sobre el sitio Web. Las cookies de sesión deberían tener un ciclo de vida entre dos horas y tres horas.

Añadir en los formularios de sesión y compras el captcha para validar si se es humano, es importante evitar el inicio de sesión de bots que pueden generar gran cantidad de peticiones sobre la página y generar un ataque de denegación de servicios.

11. BIBLIOGRAFÍA

Common Vulnerabilities and Exposures. 2014. Ataque de POODLE CVE-2014-3566. *CVE.* [En línea] 14 de mayo de 2014. [Citado el: 10 de diciembre de 2016.] https://cve.mitre.org/cgi-bin/cvename.cgi?name=cve-2014-3566.

—. 2016. Vulnerabilidad Oracle OpenSSL CBC Ciphersuites CVE-2014-3566. *CVE.* [En línea] 29 de enero de 2016. [Citado el: 10 de diciembre de 2016.] https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2016-2107.

CVE Details. 2015. Vulnerability Details : CVE-2015-5622. *Vulnerability Feeds & Widgets.* [En línea] 03 de agosto de 2015. [Citado el: 15 de abril de 2018.] https://www.cvedetails.com/cve/CVE-2015-5622/.

—. 2018. Vulnerability Details : CVE-2018-5776. *Vulnerability Feeds & WidgetsNew .* [En línea] 18 de enero de 2018. [Citado el: 15 de abril de 2018.] https://www.cvedetails.com/cve/CVE-2018-5776/.

DETAILS, CVE. 2018. Vulnerability Statistics PHP 5.6.18. *The ultimate security vulnerability datasource.* [En línea] 2018. [Citado el: 28 de marzo de 2018.] https://www.cvedetails.com/version/192558/PHP-PHP-5.6.18.html.

DragonJAR, Restrepo, Jaime Andrés y Tibaquirá, Jacobo. 2016. DragonJAR Soluciones y Seguridad Informática S.A.S. *Comunidad DragonJAR.* [En línea] 20 de marzo de 2016. http://www.dragonjar.org/.

FERNÁNDEZ, Gabriel Maciá y TEODORO García, Pedro. 2007. *Ataques de denegación de servicio a baja tasa contra servidores.* Granada : Editorial de la Universidad de Granada, 2007.

fjavierm, Mundo Informático. 2011. OWASP Top Ten: A3 – Pérdida de autenticación y gestión de sesiones. [En línea] 1 de enero de 2011. [Citado el: 21 de mayo de 2016.] https://infow.wordpress.com/2011/01/01/owasp-top-ten-a3-perdida-de-autenticacion-y-gestion-de-sesiones/.

Grossman, Jeremiah. Chief of Security Strategy (SentinelOne). *Founder of WhiteHat Security.* [En línea] [Citado el: 04 de febrero de 2017.] https://www.jeremiahgrossman.com/.

HISPASEC y Ropero, Antonio . 2013. OWASP: Los diez riesgos más críticos en aplicaciones web (2013). [En línea] 21 de noviembre de 2013. [Citado el: 26 de mayo de 2016.] http://unaaldia.hispasec.com/2013/11/owasp-los-diez-riesgos-mas-criticos-en.html.

IETF. 2005. IDWG - Intrusion Detection Working Group. [En línea] 2 de agosto de 2005. [Citado el: 21 de abril de 2016.] https://datatracker.ietf.org/wg/idwg-charter.html/.

Internet Ya - Soluciones Web. 2013. [En línea] Que es OWASP TOP 10, abril de 2013. [Citado el: 20 de mayo de 2016.] http://www.internetya.co/que-es-owasp-top-10/.

Jiménez Guanoluisa, Cristian Geovanny y Facultad, Universidad Nacional de Chimborazo. 2015. Propuesta de técnicas de seguridad en sistemas web dinámicos para disminuir las vulnerabilidades de acceso a los sistemas web. caso aplicativo: sistema académico para el bachillerato internacional de la unidad educativa riobamba. [En línea] 2015. [Citado el: 20 de mayo de 2016.] http://dspace.unach.edu.ec/bitstream/51000/726/1/UNACH-EC-ISC-2015-0010.pdf.

José María, Alonso Cebrián, y otros. 2009. Ataques a BB. DD. SQL Injection. [En línea] 2009. [Citado el: 21 de mayo de 2016.] http://psoluciones.net/recursos/Ataques-a-bases-de-datos.pdf. PID_00191663.

Klus, Javier Fernando. 2011. *¿Cómo auditar controles de aplicación?* [En línea] 28 de agosto de 2011. [Citado el: 10 de junio de 2016.] http://www.auditool.org/blog/auditoria-de-ti/265-na-breve-introduccion-de-como-auditar-controles-de-aplicacion.

Larrieu, Cyrille. 2003. *Sistema de detección de intrusiones (IDS).* [En línea] 29 de enero de 2003. [Citado el: 11 de mayo de 2016.] http://es.ccm.net/contents/162-sistema-de-deteccion-de-intrusiones-ids.

OWASP. 2013. A1-Injection. *OWASP.* [En línea] 23 de junio de 2013. [Citado el: 21 de marzo de 2016.] https://www.owasp.org/index.php/Top_10_2013-A1-Injection.

—. 2016. Open Web Application Security Project. *Acerca OWASP.* [En línea] 01 de febrero de 2016. [Citado el: 18 de marzo de 2016.] https://www.owasp.org/index.php/About_OWASP#The_OWASP_Foundation.

—. 2010. Top 10–2010–the ten most critical web application security risks. *The Open Web Application Security (2010).* 2010.

OWASP.ORG. 2013. Los diez riesgos mas criticos en aplicaciones web - OWASP Top 10 - 2013. [En línea] 2013. [Citado el: 1 de 23 de 2016.] https://www.owasp.org/images/5/5f/OWASP_Top_10_-_2013_Final_-_Espa%C3%B1ol.pdf.

owasp.org. 2008. Top 10 2007-Referencia Insegura y Directa a Objetos. [En línea] 19 de septiembre de 2008. [Citado el: 20 de mayo de 2016.] https://www.owasp.org/index.php/Top_10_2007-Referencia_Insegura_y_Directa_a_Objetos.

—. 2008. Top 10 2007-Secuencia de Comandos en Sitios Cruzados (XSS). [En línea] 19 de septiembre de 2008. [Citado el: 21 de mayo de 2016.] https://www.owasp.org/index.php/Top_10_2007-Secuencia_de_Comandos_en_Sitios_Cruzados_(XSS).

—. 2010. Top 10 2010-A4-Insecure Direct Object References. [En línea] 01 de mayo de 2010. [Citado el: 22 de mayo de 2016.] https://www.owasp.org/index.php/Top_10_2010-A4-Insecure_Direct_Object_References.

—. 2013. Top 10 2013-A7-Missing Function Level Access Control. [En línea] Foundation OWASP, 23 de junio de 2013. [Citado el: 24 de mayo de 2016.] https://www.owasp.org/index.php/Top_10_2013-A7-Missing_Function_Level_Access_Control.

Pajuelo Grandez, Laura Janett. 2015. Implementación de un metodología de pruebas de penetración en el area del aseguramientos de la calidad EAP ENERGY. [En línea] 2015. [Citado el: 25 de mayo de 2016.] http://www.repositorioacademico.usmp.edu.pe/bitstream/usmp/1176/1/pajuelo_glj.pdf.

PHP:NET. 2012. [En línea] Ejemplo Inyección de SQL, 5 de marzo de 2012. [Citado el: 21 de mayo de 2016.] http://php.net/manual/es/security.database.sql-injection.php.

Schneier, Bruce y ComputerWeekly. 2005. Insider Threat Statistics. [En línea] From Europe, junio de 2005. [Citado el: 10 de septiembre de 2017.] https://www.schneier.com/blog/archives/2005/12/insider_threat.html.

Security artWork y Monteagudo, David . 2010. OWASP TOP 10 (III): Pérdida de autenticación y Gestión de Sesiones. [En línea] 24 de marzo de 2010. [Citado el: 20 de mayo de 2016.] http://www.securityartwork.es/2010/03/24/owasp-top-10-iii-perdida-de-autenticacion-y-gestion-de-sesiones/.

SiembraViva. 2014. Acerca de SiembraViva.com. [En línea] 2014. [Citado el: 07 de febrero de 2017.] https://siembraviva.com/home/quienes-somos/.

SSL Labs, Ivan Ristic. 2013. RC4 in TLS is Broken: Now What? *Qualys Blog.* [En línea] 19 de marzo de 2013. [Citado el: 11 de diciembre de 2016.] https://blog.qualys.com/ssllabs/2013/03/19/rc4-in-tls-is-broken-now-what.

SSL Labs, Ivan Ristic. 2013. SSL Labs: Deploying Forward Secrecy. [En línea] 25 de junio de 2013. [Citado el: 11 de diciembre de 2016.] https://blog.qualys.com/ssllabs/2013/06/25/ssl-labs-deploying-forward-secrecy.

Universidad de Vigo. 2014. Seguridad en el desarrollo de aplicaciones. [En línea] noviembre de 2014. [Citado el: 25 de mayo de 2016.] http://ccia.ei.uvigo.es/docencia/SSI-grado/1314/apuntes/tema3.pdf.

Valle Padilla, Jessica Janneth y Gavidia Villacrés, Marco Vinicio. 2015. ANÁLISIS DE VULNERABILIDADES DE SOFTWARE PARA MEJORAR LA SEGURIDAD EN LOS SISTEMAS INFORMÁTICOS. [En línea] diciembre de

2015. [Citado el: 20 de mayo de 2016.] http://dspace.unach.edu.ec/bitstream/51000/731/1/UNACH-EC-ISC-2015-0012.pdf.

Vigo Scheiner, Gonzalo. 2015. Seguridad Ágil. [En línea] 2015. [Citado el: 27 de mayo de 2016.] http://diposit.ub.edu/dspace/bitstream/2445/68455/2/memoria.pdf.

W3. 2004. RFC2616 - Hypertext Transfer Protocol HTTP/1.1. *Network Working Group.* [En línea] 09 de octubre de 2004. [Citado el: 04 de febrero de 2017.] https://www.w3.org/Protocols/rfc2616/rfc2616.html.

Williams, Jeff y Dave, Wichers. 2010. OWASP top 10–2010. *OWASP Foundation.* 15 de abril de 2010, pág. 8.