Revista Ingeniería e Innovación
ISSN: 2346-0474 (En línea)

Política De Seguridad Hsts O Seguridad De Transporte Http Estricta Y Su Implementacion En Entornos Web.

Hsts Security Policy Or Strict Http Transport Security And Its Implementation In Web Environments

Anderson S Flórez[1] *
UNIPAMPLONA

José G Chacón[2]
UNIPAMPLONA

Renzo A Chía[3]
UNIPAMPLONA

Albenis E Flórez[4]
UNIPAMPLONA

Johel E Rodriguez[5]
UNIPAMPLONA

\

[1] *Ing. Sistemas, Profesor OC UNIPAMPLONA, Villa del Rosario, Colombia.*
[2] *Msc Ciencias de la computación, Profesor TCO, Villa del Rosario, Colombia, Correo: jose.chacon@unipamplona.edu.co*
[3] *Ing. Sistemas, Egresado UNIPAMPLONA, Villa del Rosario, Colombia*
[4] *Ing. Sistemas, Egresado UNIPAMPLONA, Villa del Rosario, Colombia*
[5] *5Msc TIC, Docente UNISIMON, Cúcuta, Colombia*

RESUMEN

Este articulo consiste en determinar y mejorar la seguridad en aplicaciones web, utilizando la política de seguridad estricta HSTS en un servidor web Apache, la cual permitirá a los usuarios mediante un navegador web solo visualizar páginas web con https, bloqueando las páginas que no posean la capa de seguridad SSL/TLS. Secure Sockets Layer (SSL capa de puertos seguro) y Transport Layer Security (TLS seguridad de la capa de transporte) en su protocolo web y a su vez proteger la información que el usuario dispone, de tal forma que los atacantes no podrán convertir la página web con el protocolo de seguridad https en una página sin protocolo de seguridad http, manteniendo así este bien preciado protegido de manera segura, brindándole al usuario la confidencialidad, integridad y disponibilidad de la información.

PALABRAS CLAVE: Apache, Seguridad, Confidencialidad, Disponibilidad, HSTS, Http, Https, Integridad.

ABSTRACT

This article consists of determining and improving security in web applications, using HSTS security policy on an Apache web server, which will allow users through a web browser to only view web pages with https, blocking pages that do not have the layer SSL / TLS security. Secure Sockets Layer (SSL) and Transport Layer Security (TLS) in its web protocol and in turn protect the information that the user has, so that the attackers cannot convert the web page With the https security protocol on a page without http security protocol, thus keeping this precious asset securely protected, providing the user with the confidentiality, integrity and availability of the information.

KEYWORDS: Apache, Security, Confidentiality, Availability, HSTS, Http, Https, Integrity.

INTRODUCCIÓN

La evolución que tienen las aplicaciones web en el mundo es asombrosa, debido a esto la internet es muy utilizada, todo el mundo quiere tener su empresa o sus productos y mucha información valiosa en la internet, gracias a la facilidad con que las personas pueden ingresar a internet, ya sea mediante un teléfono móvil, un computador o una tablet, por medio de conexiones en red wifi, red cableada o hasta en un paquete de datos.

Debido a las vulnerabilidades que han tenido los diferentes aplicativos web y sistemas de información más comunes en el mundo, los usuarios ponen en duda la validez de la información que se encuentra en las páginas web y colocan en duda, si la información que ellos disponen está siendo de verdad protegida, por este motivo y debido a que una página con https puede ser también vulnerada surge la necesidad de utilizar la una política seguridad más completa, para que las páginas que son visualizadas por los usuarios sean páginas web con el protocolo https no vulnerable a ataques.

Pero esta facilidad de navegar en internet no nos asegura que todas las páginas que estamos visitando sean seguras, de hecho, la

mayoría de páginas que visitamos en internet no son seguras, debido a todo esto surgen grandes vulnerabilidades acerca de la información que ingresamos y obtenemos en el internet.

Por este motivo las grandes empresas en desarrollo de software quieren hacer sitios web más seguros y una de los mecanismos que permiten que la web sea más segura es el HSTS o seguridad de transporte http estricta. Permitiendo que cualquier servidor web solo cargue páginas web que tenga el protocolo HTTPS de esta manera, se estaría eliminando una serie de vulnerabilidades y la gente solo navegaría por sitios web seguros.

El Protocolo HTTP (Hypertext Transfer Protocol o Protocolo de transferencia de hipertexto). Es un protocolo típico de petición-respuesta, que controla la transferencia de datos entre el servidor y el cliente (tal como un navegador web). El tráfico HTTP es una comunicación punto a punto, pero hay muchos casos de uso en los que esta comunicación puede beneficiarse Inclusión de elemento activo adicional - servidor proxy. El servidor proxy es el elemento central en la comunicación entre el servidor y el cliente. Este elemento sólo puede re direccionar la comunicación o verificar el protocolo de aplicación. El servidor proxy HTTP puede acelerar el acceso a los recursos y realizar operaciones de inspección o supervisión. Proteger la privacidad de los usuarios se puede proporcionar también. (Sysel & Doležal, 2014). (Rosado & Verjel ,2020).

HTTP también se utiliza como un protocolo genérico para la comunicación entre los agentes de usuario y proxies / gateways a otros sistemas de Internet, incluidos los soportados por el SMTP, NNTP, FTP, Gopher y WAIS protocolos. De esta forma, HTTP permite el acceso básico de hipermedia a los recursos disponibles de diversas aplicaciones.(Leach et al., 1999).

HTTP es un protocolo que permite la obtención de recursos, como documentos HTML. Es la base de cualquier intercambio de datos en la Web y un protocolo cliente-servidor, lo que significa que las solicitudes son iniciadas por el destinatario, generalmente el navegador Web. Un documento completo se reconstruye a partir de los diferentes sub-documentos obtenidos, por ejemplo, texto, descripción del diseño, imágenes, videos, secuencias de comandos y más.(“An overview of HTTP - HTTP | MDN,” 2017).

Los Componentes de sistemas basados en HTTP. HTTP es un protocolo cliente-servidor: las solicitudes son enviadas por una entidad, el user-agent (o un proxy en su nombre). La mayoría de las veces el user-agent es un navegador Web, pero puede ser cualquier cosa, por ejemplo, un robot que rastrea la Web para rellenar y mantener un índice de motor de búsqueda esto se muestra en figura 1.

Cada solicitud individual se envía a un servidor, que lo manejará y proporcionará una respuesta, llamada respuesta. Entre esta petición y la respuesta hay numerosas entidades, designadas colectivamente como proxies, que realizan operaciones diferentes y actúan como pasarelas o cachés.

Ahora bien, Qué puede ser controlado por HTTP Esta naturaleza extensible de HTTP ha permitido, con el tiempo, más control y funcionalidad de la Web. Métodos de caché o de autenticación fueron funciones manejadas temprano en el historial HTTP. La capacidad de relajar la restricción de origen, por el contrario, sólo se ha añadido en los años 2010.

Aquí hay una lista de características comunes controlables con HTTP.

- Cache
- Autenticación
- Proxy y tunelización
- Sesiones
- Flujo HTTP

- Mensajes HTTP

Siguiendo en el mismo contexto, El protocolo de transferencia de hipertexto seguro (HTTPS) es la versión segura de HTTP, el protocolo sobre el que se envían los datos entre su navegador y el sitio web al que está conectado. El 'S' al final de HTTPS significa 'Seguro'. Significa que todas las comunicaciones entre su navegador y el sitio web están encriptados. HTTPS se utiliza a menudo para proteger las transacciones en línea altamente confidenciales, como la banca en línea y los formularios de pedidos de compras en línea (Castro, Velásquez & Castro 2020).

Las páginas HTTPS suelen utilizar uno de los dos protocolos seguros para cifrar las comunicaciones: SSL (Secure Sockets Layer) o TLS (Transport Layer Security). Ambos protocolos TLS y SSL utilizan lo que se conoce como un sistema de infraestructura de clave pública (PKI) "asimétrico". Un sistema asimétrico utiliza dos «claves» para cifrar las comunicaciones, una clave «pública» y una clave «privada». Cualquier cosa cifrada con la clave pública sólo puede ser descifrada por la clave privada y viceversa.

Como sugieren los nombres\, la clave 'privada' debe mantenerse estrictamente protegida y sólo debe ser accesible al propietario de la clave privada. En el caso de un sitio web, la clave privada permanece seguramente instalada en el servidor web. A la inversa, la clave pública está destinada a ser distribuida a cualquier persona ya todos los que necesiten poder descifrar información cifrada con la clave privada.

Cuando solicita una conexión HTTPS a una página web, el sitio web enviará inicialmente su certificado SSL a su navegador. Este certificado contiene la clave pública necesaria para iniciar la sesión segura. Sobre la base de este intercambio inicial, su navegador y el sitio web a continuación, iniciar el "apretón de manos SSL". El apretón de manos SSL implica la generación de secretos compartidos para establecer una conexión segura entre usted y el sitio web.

Cuando se utiliza un certificado digital SSL de confianza durante una conexión HTTPS, los usuarios verán un icono de candado en la barra de direcciones del navegador. Cuando un certificado de validación extendida está instalado en un sitio web, la barra de direcciones se volverá verde.

Todas las comunicaciones enviadas a través de conexiones HTTP normales están en "texto sin formato" y pueden ser leídas por cualquier hacker que logre entrar en la conexión entre su navegador y el sitio web. Esto presenta un claro peligro si la "comunicación" está en un formulario de pedido e incluye los detalles de su tarjeta de crédito o número de seguro social. Con una conexión HTTPS, todas las comunicaciones se cifran de forma segura. Esto significa que incluso si alguien logró romper en la conexión, no sería capaz de descifrar cualquiera de los datos que pasa entre usted y el sitio web.

Beneficios del protocolo de transferencia de hipertexto seguro

Los principales beneficios de un certificado HTTPS son:

La información del cliente, como números de tarjetas de crédito, está cifrada y no puede ser interceptada.

Los visitantes pueden verificar que es un negocio registrado y que es propietario del dominio.

Los clientes tienen más probabilidades de confiar y completar compras de sitios que utilizan HTTPS.

(“HTTP to HTTPS | What is a HTTPS Certificate,” 2017).

Los navegadores Web como Internet Explorer, Firefox y Chrome también muestran un icono de candado en la barra de direcciones para indicar visualmente que una conexión HTTPS está en efecto.

1. MARCO TEÓRICO

1.1. Robot

El objetivo principal del protocolo SSL es proporcionar privacidad y fiabilidad entre dos aplicaciones de comunicación. El protocolo se compone de dos capas. En el nivel más bajo, superpuesto a un protocolo de transporte fiable (por ejemplo, TCP [TCP]), se encuentra el protocolo de registro SSL. El SSL Record Protocol se utiliza para la encapsulación de varios protocolos de nivel superior. Uno de estos protocolos encapsulados, el SSL Handshake Protocol, permite que el servidor y el cliente se autentiquen entre sí y negocien un algoritmo de cifrado y claves criptográficas antes de que el protocolo de aplicación transmita o reciba su primer byte de datos.

Una ventaja de SSL es que es independiente del protocolo de aplicación. Un protocolo de nivel superior puede superponerse al Protocolo SSL de forma transparente. El protocolo SSL proporciona seguridad de conexión que tiene tres propiedades básicas:

La conexión es privada. El cifrado se utiliza después de un primer apretón de manos para definir una clave secreta. La criptografía simétrica se utiliza para el cifrado de datos (por ejemplo, DES, RC4, etc.).

La identidad de los compañeros puede ser autenticada mediante criptografía asimétrica o de clave pública (por ejemplo, RSA, DSS, etc.).

La conexión es fiable. El transporte de mensajes incluye una comprobación de integridad del mensaje utilizando un MAC con clave. Las funciones de hash seguras (por ejemplo, SHA, MD5, etc.) se utilizan para cálculos MAC.

Los objetivos del Protocolo SSL v3.0, por orden de prioridad, son: (Freier, Kocher, & Karlton, 1996).

- Seguridad criptográfica
- Interoperabilidad
- Extensibilidad
- Eficiencia relativa

Una de las nuevas características existentes en varios navegadores es la adición de HTTP Strict Transport Security. HSTS permite que un sitio pueda solicitar siempre contactarse a través de HTTPS. HSTS es compatible con Google Chrome, Firefox, Safari, Opera y el IE. El tema es que HSTS re direcciona a los usuarios que tienden a escribir http: // en el navegador, y omiten por completo el esquema la mayor parte del tiempo. Sin embargo, HTTP es inseguro. Un atacante puede tomar esa conexión, manipularlo y sólo los usuarios de ojos de águila, podrían darse cuenta de que los han redirigido a https://www.bank0famerica.com o algo así. A partir de entonces, el usuario está bajo el control del atacante, que puede interceptar contraseñas, direcciones, etc. a voluntad.

Un servidor con HSTS habilitado puede incluir la siguiente cabecera en una respuesta HTTPS: Strict-Transport-Security: max-age=16070400; includeSubDomains.

Cuando el navegador ve esto, recordará por el número dado de segundos, que el dominio actual sólo debe contactarse a través de HTTPS. En el futuro, si el usuario omite el esquema, HTTPS, (http: //) es el valor predeterminado. De hecho, todas las solicitudes de direcciones URL en el dominio actual serán redirigidos a HTTPS.

"Lista HSTS precarga", si usted posee un sitio que le gustaría ver incluido en la lista precargada de HSTS se puede presentar en https://hstspreload.appspot.com. Un subconjunto seleccionado de los miembros de la lista HSTS precargada: Google, Paypal, Twitter, Simple, Linode, Stripe, Lastpass.(the chromium projects, 2014) .

Durante muchos años, se ha trabajado para aumentar el uso del cifrado entre

nuestros usuarios y Google. Hoy en día, la gran mayoría de estas conexiones son encriptados, y nuestro trabajo continúa en este esfuerzo.

Para proteger aún más a los usuarios, hemos dado un paso más para fortalecer la forma en que utilizamos el cifrado de datos en tránsito mediante la implementación de HTTP estricta Transporte Seguridad- HSTS de corto en la www.google.com dominio. HSTS impide que la gente accidentalmente navegar a direcciones URL HTTP convirtiendo automáticamente las direcciones URL HTTP inseguros en HTTPS URL protegidas. Los usuarios pueden navegar a estas direcciones URL HTTP tecleando manualmente una dirección URL de protocolo HTTP-menos o en la barra de direcciones, o siguiendo los enlaces desde otros sitios web HTTP. Preparación para la puesta en marcha lo general, la implementación de HSTS es un proceso relativamente básico. Sin embargo, debido a las complejidades particulares de Google, que teníamos que hacer algún trabajo de preparación adicional que no habrían necesitado la mayoría de los otros dominios que hacer. Por ejemplo, hemos tenido que hacer frente a contenido mixto, malas HREF, vuelve a dirigir a HTTP, y otros temas como la actualización de los servicios tradicionales que podrían causar problemas a los usuarios que tratan de acceder a nuestro dominio de núcleo. Este proceso no estuvo exento de dificultades. Despliegue y próximos pasos Hemos activado HSTS para www.google.com, pero algo de trabajo permanece en nuestra lista de despliegue. En lo inmediato, estamos enfocados en el aumento de la duración que la cabecera, está activo ('max-age'). Hemos establecido inicialmente max-age de la cabecera de un día; la corta duración ayuda a mitigar el riesgo de cualquier problema potencial con este despliegue. Al aumentar el máximo de edad, sin embargo, se reduce la probabilidad de que una solicitud inicial a www.google.com pasa a través de HTTP. En los próximos meses, vamos a la rampa hasta el max-age de la cabecera para al menos un año. El cifrado de datos en tránsito ayuda a mantener nuestros usuarios y sus datos seguros. Estamos muy contentos de ser HSTS ejecución y seguiremos extenderlo a más dominios y productos de Google en los próximos meses.(“Blog de seguridad de Google en línea: Llevar HSTS a www.google.com,” 2016).

Http strict transport security (hsts) (o seguridad de transporte http estricta). Es un mecanismo que permita a los sitios Web a declararse accesible sólo a través de conexiones seguras, y / o para los usuarios para poder dirigir su agente (s) de usuario para interactuar con los sitios dados solamente a través de conexiones seguras. Esta política general se conoce como HTTP Strict Transport Security (HSTS). La política es declarada por los sitios Web a través del campo de encabezado de respuesta HTTP estricta-Transporte-Seguridad, y / o por otros medios, como la configuración de agente de usuario. (Jackson & Barth, 2008)

HTTP Strict Transport Security o Seguridad de transporte HTTP estricta HSTS, es una política de seguridad web establecida para evitar ataques que puedan interceptar comunicaciones, cookies, etc. Según este mecanismo un servidor web declara a los agentes de usuario compatibles, por ejemplo, un sitio web, que puede interactuar con ellos solamente mediante conexiones HTTP seguras (es decir, en capas HTTP sobre TLS/SSL). HSTS es un protocolo de normas de IETF Internet Engineering Task Force (IETF) (en español, Grupo de Trabajo de Ingeniería de Internet) y se especifica en el RFC 6797. La política HSTS es comunicada por el servidor al agente de usuario a través de un campo de la cabecera HTTP de respuesta denominado "libre Transport-Security". La política HSTS especifica un período de tiempo durante el cual el agente de usuario deberá acceder al servidor sólo en forma segura.(Jackson & Barth, 2008)

Cabe mencionar los efectos de la política

de HSTS, tal como se aplica por un UA (user agent) en confirman interacciones con una gran cantidad de recursos web que tienen ese tipo de política (conocido como un anfitrión HSTS), se resumen así:

UA (user agent) transforman referencias inseguras URI (Uniforme Resource Identifier, es decir, identificador de recursos uniforme.) a un anfitrión en HSTS seguro de referencias URI antes de la eliminación de referencias.

El UA (user agent) termina cualquier intento de conexión de transporte seguro sobre cualquier no seguro y todos los errores o advertencias de transporte seguro.

HSTS se ocupa de tres clases de amenazas: atacantes de red pasivos, atacantes de redes activas y desarrolladores webs imperfectos. Sin embargo, explícitamente no es un remedio para otras dos clases de amenazas: phishing y malware. Las amenazas que se abordan, así como las amenazas que se abordan brevemente a continuación. Amenazas abordadas.Los atacantes de red pasivos

Cuando un usuario navega por la web en una red local inalámbrica (por ejemplo, una Red de área local inalámbrica 802.11 a base de) un atacante cerca puede posiblemente espiar en el protocolo no cifrado de Internet del usuario conexiones, tales como HTTP, independientemente de si es o no lo local red inalámbrica en sí está asegurado. Herramientas de detección inalámbrica gratuitas (por ejemplo, [Aircrack-ng]) permiten tales ataques pasivos de escucha, incluso si la red inalámbrica local está funcionando de una manera segura. Un atacante red pasiva hace uso de tales herramientas y puede secuestrar las sesiones web del usuario mediante la obtención de cookies que contengan credenciales de autenticación (Jackson & Barth, 2008). Por ejemplo, existen herramientas ampliamente disponibles, tales como Firesheep (una extensión del navegador web) [Firesheep], que permiten a su portador para obtener las cookies de sesión de otros usuarios locales de diversas aplicaciones web. DEC Márquez, TV Pérez (2018).

Un atacante determinado puede montar un ataque activo, ya sea mediante la suplantación de identidad del servidor DNS de un usuario o, en una red inalámbrica, mediante la falsificación de tramas de red u ofreciendo un punto de acceso gemelo maligno denominado. Si el usuario está detrás de un enrutador doméstico inalámbrico, un atacante puede intentar reconfigurar el enrutador usando contraseñas predeterminadas y otras vulnerabilidades. Algunos sitios, como los bancos, se basan en el transporte seguro de extremo a extremo para protegerse a sí mismos y a sus usuarios contra esos atacantes activos. Desafortunadamente, los navegadores permiten a sus usuarios a optar fácilmente por estas protecciones para ser utilizables para sitios que implementan incorrectamente el transporte seguro, por ejemplo, generando y firmando sus propios certificados (sin distribuir su certificado de autoridad de certificación Navegadores de los usuarios).

La seguridad de un sitio de otro modo seguro (es decir, todo su contenido se materializa a través de "https" URIs) puede ser comprometida completamente por un atacante activo que explora un simple error, como la carga de una hoja de estilo en cascada o un SWF (Shockwave Flash) sobre una conexión insegura (tanto las hojas de estilo en cascada como las películas SWF pueden codificar la página de incrustación, para sorpresa de muchos desarrolladores web, además de algunos navegadores no emiten las llamadas "advertencias de contenido mixto" Conexiones). Incluso si los desarrolladores del sitio escudriñan cuidadosamente su página de inicio de sesión para "contenido mixto", una sola incrustación insegura en cualquier lugar del sitio global

compromete la seguridad de su página de inicio de sesión porque un atacante puede grabar (es decir, controlar) la página de inicio inyectando código, Un guion) en otra, inseguramente cargada, página del sitio. Amenazas no abordadas. Suplantación de identidad (Phishing)

Los ataques de phishing se producen cuando un atacante solicita las credenciales de autenticación del usuario al alojar un sitio falso ubicado en un dominio distinto del sitio real, tal vez dirigiendo tráfico al sitio falso mediante el envío de un enlace en un mensaje de correo electrónico. Los ataques de phishing pueden ser muy efectivos porque los usuarios encuentran difícil distinguir el sitio real de un sitio falso. HSTS no es una defensa contra el phishing; Sino que complementa muchas defensas de phishing existentes instruyendo al navegador para proteger la integridad de la sesión y los tokens de autenticación de larga duración (Jackson & Barth, 2008).

La Vulnerabilidades de Malware y Navegador. Debido a que HSTS se implementa como un mecanismo de seguridad del navegador, se basa en la confiabilidad del sistema del usuario para proteger la sesión. El código malicioso que se ejecuta en el sistema del usuario puede comprometer una sesión del navegador, independientemente de si se utiliza HSTS.

Declaración de Host HSTS. Un host HTTP se declara un Host HSTS emitiendo a user agents (UAs) una Política HSTS, la cual es representada y transportada a través del campo de encabezado de respuesta HTTP Strict-Transport-Security sobre transporte seguro (por ejemplo, TLS). Tras el recibo y el procesamiento sin errores de esta cabecera por una UA conformada, la UA considera al host como un host HSTS conocido.

Política HSTS. Una política de HSTS dirige a UAs a comunicarse con un host HSTS conocido sólo a través de un transporte seguro y especifica la duración del tiempo de retención de políticas. La política HSTS anula explícitamente el procesamiento UA de referencias URI, entrada de usuario (por ejemplo, a través de la "barra de ubicación") u otra información que, en ausencia de la política HSTS, podría ocasionar que UAs se comuniquen inseguros con el Host HSTS Conocido. Una política HSTS puede contener una directiva opcional - includeSubDomains - especificando que esta política HSTS también se aplica a todos los hosts cuyos nombres de dominio son subdominios del nombre de dominio del Host HSTS Conocido.

Política de HSTS Almacenamiento y mantenimiento por agentes de usuario Los UAs almacenan e indexan las políticas de HSTS basadas estrictamente en los nombres de dominio de los Hosts HSTS emisores. Esto significa que los UA mantendrán la Política HSTS de cualquier Host HSTS dado por separado de cualquier Política HSTS emitida por cualquier otro HSTS Hosts cuyos nombres de dominio sean superdomains o subdominios del nombre de dominio del Host HSTS dado. Sólo el host HSTS puede actualizar o puede provocar la eliminación de su política de HSTS emitida. Esto lo logra enviando campos de encabezado de respuesta HTTP de Strict-Transport-Security a UAs con nuevos valores para la duración de la política y la aplicabilidad del subdominio.

Por lo tanto, los UAs almacenan en caché la información de política HSTS "más reciente" en nombre de un Host HSTS. La especificación de una duración de tiempo cero indica a la UA que elimine la Política HSTS (incluida cualquier directiva includeSubDomains) para ese HSTS Host.

Agente de usuario HSTS Policy Enforcement. Cuando se establece una conexión HTTP a un host determinado, no obstante, lo instigado, la UA examina su caché de Hosts HSTS Conocidos para ver si hay alguno con nombres de dominio que son superdomains del nombre de dominio del host dado. Si se encuentra cualquiera, y de aquellos sí. Cualquiera tiene la directiva

includeSubDomains afirmada, entonces la política de HSTS se aplica al host dado.. De lo contrario, la política HSTS se aplica al host dado sólo si el host dado es conocido por la UA como un host HSTS. (Sadasivan et al., 2012).

2. RESULTADOS

La seguridad de la información, según ISO 27001, consiste en la preservación de su confidencialidad, integridad y disponibilidad, así como de los sistemas implicados en su tratamiento, dentro de una organización. Así pues, estos tres términos constituyen la base sobre la que se cimienta todo el edificio de la seguridad de la información:

Confidencialidad: La información no se pone a disposición ni se revela a individuos, entidades o procesos no autorizados.

Integridad: Mantenimiento de la exactitud y completitud de la información y sus métodos de proceso.

Disponibilidad: Acceso y utilización de la información y los sistemas de tratamiento de la misma por parte de los individuos, entidades o procesos autorizados cuando lo requieran.

El proyecto OWASP (Open Web Application Security Project) tiene como objetivo ofrecer una metodología, de libre acceso y utilización, que pueda ser utilizada como material de referencia por parte de los arquitectos de software, desarrolladores, fabricantes y profesionales de la seguridad involucrados en el diseño, desarrollo, despliegue y verificación de la seguridad de las aplicaciones y servicios web.

La guía empieza estableciendo el principio básico de seguridad que cualquier aplicación o servicio web debe cumplir:

Validación de la entrada y salida de información

La entrada y salida de información es el principal mecanismo que dispone un atacante para enviar o recibir código malicioso contra el sistema. Por tanto, siempre debe verificarse que cualquier dato entrante o saliente es apropiado y en el formato que se espera. Las características de estos datos deben estar predefinidas y debe verificarse en todas las ocasiones.

2.1. Diseños simples

Los mecanismos de seguridad deben diseñarse para que sean los más sencillos posibles, huyendo de sofisticaciones que compliquen excesivamente la vida a los usuarios. Si los pasos necesarios para proteger de forma adecuada una función o modulo son muy complejos, la probabilidad de que estos pasos no se ejecuten de forma adecuada es muy elevada.

2.2. Utilización y reutilización de componentes de confianza

Debe evitarse reinventar la rueda constantemente. Por tanto, cuando exista un componente que resuelva un problema de forma correcta, lo más inteligente es utilizarlo.

2.3. Defensa en profundidad

Nunca confiar en que un componente realizará su función de forma permanente y ante cualquier situación. Hemos de disponer de los mecanismos de seguridad suficientes para que cuando unos componentes del sistema fallen ante un determinado evento, otros sean capaces de detectarlo.

Tan seguros como en eslabón más débil La frase "garantizamos la seguridad, ya que se utiliza SSL" es realmente muy popular, pero también es muy inexacta. La utilización de SSL garantiza que el tráfico en tránsito entre el servidor y el cliente se encuentra cifrado, pero no garantiza nada acerca de los mecanismos de seguridad existentes.

Ofrecer la mínima información Ante una

situación de error o una validación negativa, los mecanismos de seguridad deben diseñarse para que faciliten la mínima información posible. De la misma forma, estos mecanismos deben estar diseñados para que una vez denegada una operación, cualquier operación posterior sea igualmente denegada.

Por tanto, no debemos fiarnos únicamente de los mecanismos de seguridad "exteriores", sino que es preciso identificar cuáles son los puntos precisos en los que deben establecerse las medidas de seguridad. Si nosotros no hacemos este trabajo, seguro que los atacantes si lo harán.

Otros aspectos tratados en la guía son: consideraciones de arquitectura, mecanismos de autenticación, gestión de sesiones de usuario, control de acceso, registro de actividad, prevención de problemas comunes, consideraciones de privacidad y criptografía. (Van der Stock, 2005). La guía está compuesta como se muestra en la figura 2.

Vulnerabilidades En La Web Son todos aquellos problemas de seguridad que afectan las páginas web, por lo general estos problemas permiten modificación y extracción de la información, lo cual es muy grave para las organizaciones; la mayoría de estos son registrados por medio de un identificador de CVE (CommonVulnerabilty Exposure) el cual es un diccionario de los problemas encontrados en la internet. Estos son algunos de esos problemas de seguridad:

Inyección de código SQL.(“PHP: Inyección de SQL - Manual,” 2008)., Ejecución de comandos (Command Execution), Desbordamiento de buffer (Buffer Overflow), Denegación de servicio (DoS), IMPLEMENTACIÓN DE HSTS EN DESARROLLOS WEB LOCALES.

En esta sesión se realizara la implementación de la política de seguridad HTTP Strict Transport Security (HSTS) en un servidor web Apache bajo sistema operativo Ubuntu. SERVIDOR APACHE2. En este servidor se hará la configuración necesaria de los archivos para que la política de seguridad estricta HSTS pueda funcionar correctamente y se puede llevar acabo la implementación del proyecto.

Modelo entidad relación. En la figura 3 se muestra modelo relacional de la base de datos para la aplicación web reporsoft.

Subida de la aplicación al servidor web apache2 Se procede a subir la aplicación al servidor web apache2. A continuación, pasos a seguir:

Paso 1: Nos ubicamos en la siguiente ruta: equipo/var/www/html una vez acá creamos una carpeta con el nombre de la aplicación, para este proyecto la aplicación se llama reporsoft, una vez creada la carpeta pegamos nuestros archivos en esta. Si todo salió bien debería tener algo como esto:

Paso 2: Darle permiso a la carpeta, Si no tenemos permiso para manipular su contenido, se lo damos con lo siguiente. Cambiamos el propietario del directorio y el grupo que debe usarlo. Reemplazar USUARIO con el nombre de usuario que esté utilizando la terminal los siguientes comandos: sudo chown -R USUARIO:www-data /var/www

Se le dan permisos de lectura y ejecución para todos y de escritura sólo al propietario: sudo chmod -R 755 /var/www.

Paso 3: Comprobar que se allá subido correctamente la aplicación web al servidor web apache2, para ello abrimos una ventana en el navegador y colocar localhost seguido del nombre de la carpeta donde se encuentra la aplicación web: localhost/reportsoft

Configuración de los módulos del servidor apache2. Para que la política de seguridad estricta HSTS funcione en el servidor apache2, deben Estar habilitados los siguientes modulo: el modulo del ssl y el módulo header. Se procede a habilitar los módulos ssl y header respectivamente: A continuación, pasos a seguir:

Paso 1: Abrir una terminal y copiar el siguiente comando: sudo a2enmod ssl luego reiniciamos

El servidor apache: sudo service apache2 restart, luego copiar el siguiente comando:

sudo a2enmod headers y finalizamos reiniciando apache2 de nuevo.

El comando a2enmod nos permite habilitar cualquier modulo en apache2.

Si todo salió bien copiamos el siguiente comando: sudo apche2ctl –t –D DUMP_MODULES lo ejecutamos y debería mostrarnos esto:

El comando sudo apche2ctl –t –D DUMP_MODULES nos permite visualizar que modelos tiene activo en servidor apache2.

Creación del certificado SSL auto firmado con Openssl.

Paso 1: Instalamos Opensssl con el siguiente comando: sudo apt-get install openssl

Paso 2: Crear un certificado SSL auto firmado, vamos a empezar por la creación de un subdirectorio dentro de las carpetas de configuración de Apache para colocar los archivos de certificado que vamos a estar creando copiar el siguiente comando: sudo mkdir /etc/apache2/ssl

Paso 3: Ahora que tenemos un lugar para colocar la llave y el certificado, podemos crear los dos en un solo paso escribiendo lo siguiente en la consola: sudo openssl req -x509 -nodes -days 365 -newkey rsa: 2048 -keyout /etc/apache2/ssl/apache.key -out etc/apache2/ssl/apache.crt

Paso 4: Luego en la terminal copiar y ejecutar el siguiente comando: cd /etc/apache2/sites-available/, luego abrimos es archivo default-ssl.conf con el siguiente comando: sudo gedit default-ssl.conf y agregamos estas dos líneas:

SSLCertificateFile /etc/apache2/ssl/apache.crt

SSLCertificateKeyFile /etc/apache2/ssl/apache.key

.

Pasó 5: Luego habilitamos el ssl con el siguiente comando: sudo a2ensite default-ssl.conf

Pasó 6: Comprobamos el https, abrir una ventana en el navegador y colocar: https://localhost/report.

Configuración del virtual host para cambiar la ip por un dominio local.

Paso 1: En una terminal copiar el siguiente comando: sudo nano /etc/hosts ejecutar y agregar el nombre del dominio.

Re direccionar el tráfico https a https:

Paso 1: En una terminal copiar el siguiente comando: sudo a2emod rewrite ejecutarlo

Paso 2: modificar el archivo 00-default.conf agregándole las siguientes líneas:

RewriteCond %{Https} off

RewriteRule (.*) https://{HTTP_HOST}%{REQUEST_URI}

Probar el dominio

Paso 1: Abrir una ventana en el navegador y escribir el nombre del dominio que se configuro en el paso anterior, para este proyecto se utilizó este dominio https://www.renzoreport.com

Implementar la política de seguridad estricta (HSTS) en el servidor web apache.

Paso 1: Abrir una terminal copiar y ejecutar el siguiente comando: cd /etc/apache2/sites-available/, luego abrimos es archivo default-ssl.conf con el siguiente comando: sudo gedit default-ssl.conf y agregamos esta línea: Header always set Strict-Transport-Security “max-age=63072000; includeSubdomains”

Paso 2: Reiniciar el servidor apache: sudo service apache2 restart.

Paso 3: Si todo salió bien debería mostrar algo como esto

Verificar que la política de seguridad estricta HSTS si funciona.

Paso 1: En una terminal copiar el siguiente comando: curl -s -D- -k https://www.renzoreport.com |more

3. DISCUSIÓN

La implementación de protocolos de seguridad desde la apertura de la página web le brinda al usuario final la certeza, de realizar cualquier procedimiento, sea de consulta, diligenciamiento de formularios o incluso transacciones o pagos, procesos transparentes muchas veces para estos usuario pero de mucha utilidad a la hora de realizar todas las transacciones.

4. CONCLUSIONES

La implementación de protocolos de seguridad desde la apertura de la página web le brinda al usuario final la certeza, de realizar cualquier procedimiento, sea de consulta, diligenciamiento de formularios o incluso transacciones o pagos, procesos transparentes muchas veces para estos usuario pero de mucha utilidad a la hora de realizar todas las transacciones.

El uso e implementación de esta política de seguridad (HSTS) en el servidor web apache ayuda a que este, solo muestre páginas web con que tengan el protocolo HTTPS, así las páginas web están más seguras si se ven envueltas en un ataque.

El protocolo HTTPS el atacante puede capturar los datos trasmitidos, pero lo que no puede es descifrar la información ya que esta encriptada.

El protocolo HTTP es más fácil de vulnerar un atacante puede capturar todos los datos que se están transmitiendo este protocolo y asa conseguir la información que es el bien más preciado de toda empresa o entidad.

REFERENCIAS

[1]. Ajax | MDN. (2016). Retrieved May 17, 2017, from https://developer.mozilla.org/en-US/docs/AJAX

[2]. An overview of HTTP - HTTP | MDN. (2017). Retrieved May 17, 2017, from https://developer.mozilla.org/en-US/docs/Web/HTTP/Overview

[3]. Blog de seguridad de Google en línea: Llevar HSTS a www.google.com. (2016). Retrieved May 16, 2017, from https://security.googleblog.com/2016/07/bringing-hsts-to-wwwgooglecom.html

[4]. Can I use... Support tables for HTML5, CSS3, etc. (2015). Retrieved May 16, 2017, from http://caniuse.com/#feat=stricttransportsecurity

[5]. Castro Márquez, D. E. ., Velásquez Pérez, T., & Castro Silva, H. F. (2020). Integración de seguridad y gestión de servicios en el gobierno de las tecnologías de la información. Revista Colombiana de Tecnologías de Avanzada, 2(32), 62-67. https://doi.org/10.24054/16927257.v32.n32.2018.108

[6]. CSS | MDN. (2016). Retrieved May 17, 2017, from https://developer.mozilla.org/es/docs/Web/CSS

[7]. CSS 3 - CSS | MDN. (2016). Retrieved May 17, 2017, from https://developer.mozilla.org/en-US/docs/Web/CSS/CSS3

[8]. DEC Márquez, TV Pérez (2018). Integración de seguridad y gestión de servicios en el gobierno de las tecnologías de la información 2018 ISSN: 1692-7257 - Vol2 – N 32 - 2018. DEC Márquez, TV Pérez (2018). Integración de seguridad y gestión de servicios en el gobierno de las tecnologías de la

información 2018 ISSN: 1692-7257 - Vol2 – N 32 - 2018.

[9]. Freier, A., Kocher, P., & Karlton, P. (1996). The SSL Protocol Version 3.0. Retrieved from https://tools.ietf.org/html/draft-ietf-tls-ssl-version3-00

[10]. HTML5 - HTML | MDN. (2017). Retrieved May 17, 2017, from https://developer.mozilla.org/es/docs/HTML/HTML5

[11]. HTML | MDN. (2017). Retrieved May 17, 2017, from https://developer.mozilla.org/es/docs/Web/HTML

[12]. HTTP to HTTPS | What is a HTTPS Certificate. (2017). Retrieved May 17, 2017, from https://www.instantssl.com/ssl-certificate-products/https.html

[13]. Jackson, C., & Barth, A. (2008). ForceHTTPS : Protecting High-Security Web Sites from Network Attacks. Computers and Society, 525–533. https://doi.org/10.1145/1367497.1367569

[14]. JavaScript - Glosario | MDN. (2017). Retrieved May 17, 2017, from https://developer.mozilla.org/es/docs/Glossary/JavaScript

[15]. jQuery. (2017). Retrieved May 17, 2017, from https://jquery.com/

[16]. Leach, P. J., Berners-Lee, T., Mogul, J. C., Masinter, L., Fielding, R. T., & Gettys, J. (1999). Hypertext Transfer Protocol -- HTTP/1.1. Retrieved from https://tools.ietf.org/html/rfc2616

[17]. MySQL | La base de datos de código abierto más popular | Oracle América Latina. (2014). Retrieved May 17, 2017, from https://www.oracle.com/lad/mysql/index.html

[18]. M. Solarte, G. A. Ramírez and D. A. Jaramillo, “Hábitos de ingreso y resultados en las

[19]. evaluaciones en cursos en línea masivos con reconocimiento académico,” Ing. E Innov., vol.5, 1, 2017.

[20]. PHP: ¿Qué es PHP? - Manual. (2008). Retrieved May 17, 2017, from http://php.net/manual/es/intro-whatis.php

[21]. PHP: Inyección de SQL - Manual. (2008). Retrieved May 17, 2017, from http://php.net/manual/es/security.database.sql-injection.php

[22]. Rosado Gómez, A. A. ., & Verjel Ibáñez, A. . (2020). Aplicación de la minería de datos en la educación en línea. Revista colombiana de tecnologías de avanzada, 1(29), 92-98. https://doi.org/10.24054/16927257.v29.n29.2017.194

[23]. Sadasivan, G., Brownlee, J., Claise, B., & Quittek, J. (2012). Architecture for IP flow information export. RFC Editor. Retrieved from https://tools.ietf.org/html/rfc6797

[24]. Sysel, M., & Doležal, O. (2014). An Educational HTTP Proxy Server. Procedia Engineering, 69, 128–132. https://doi.org/10.1016/j.proeng.2014.02.212

[25]. the chromium projects. (2014). HTTP Strict Transport Security - The Chromium Projects. Retrieved from https://www.chromium.org/hsts

[26]. van der Stock, A. (2005). Una Guía para Construir Aplicaciones y Servicios Web Seguros Edición 2.0 Black Hat. Retrieved from https://www.owasp.org/images/b/b2/OWASP_Development_Guide_2.0.1_Spanish.pdf

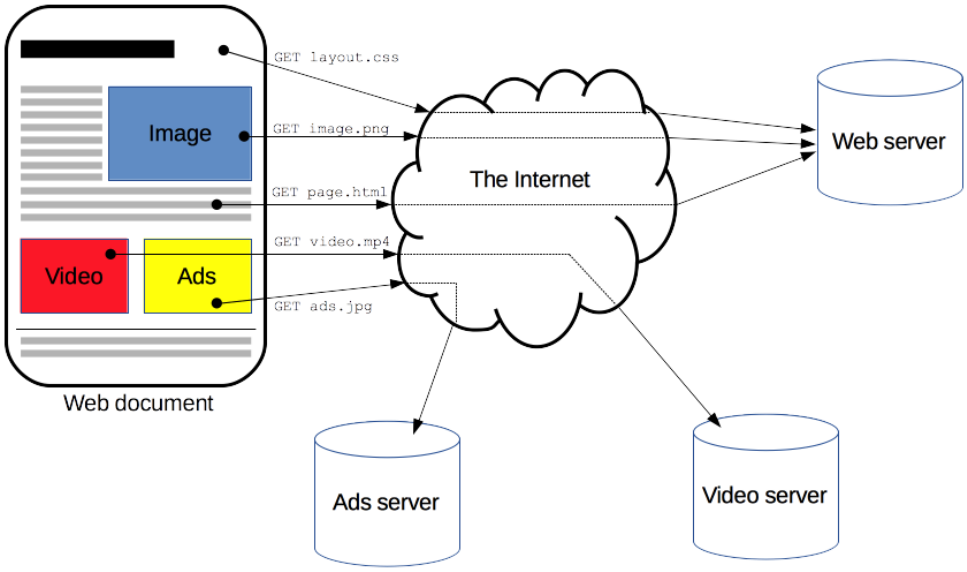

Figura 1 Los clientes y los servidores se comunican mediante el intercambio de mensajes individuales (en contraposición a una corriente de datos.

Fuente: *https://developer.mozilla.org/en-US/docs/Web/HTTP/Overview*

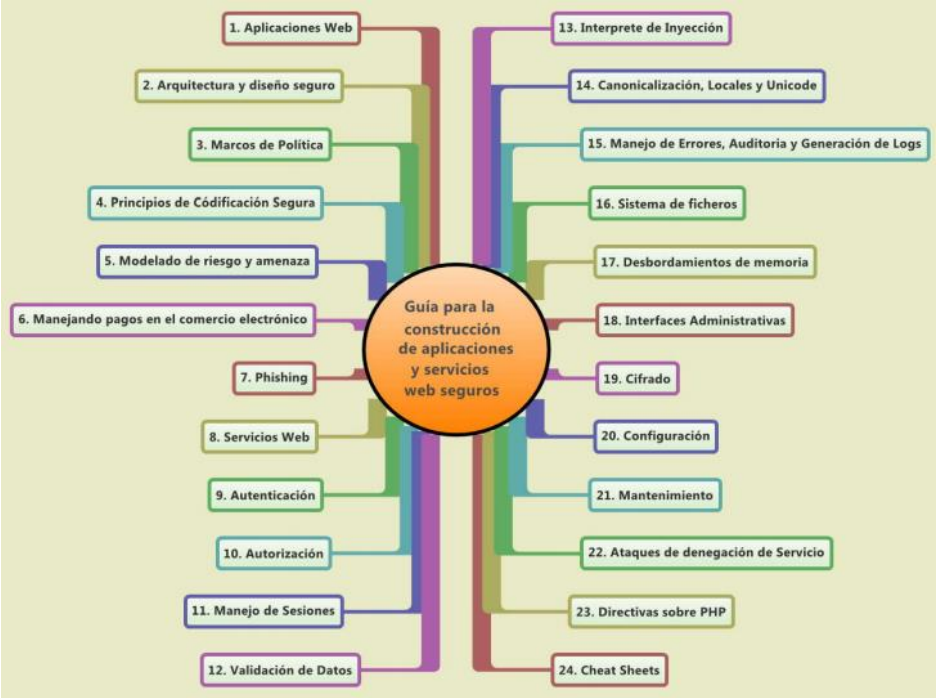

Figura 2 Guía para el desarrollo de aplicaciones y servicios seguros

Fuente: https://www.owasp.org/images/b/b2/OWASP_Development_Guide_2.0.1_Spanish.pdf

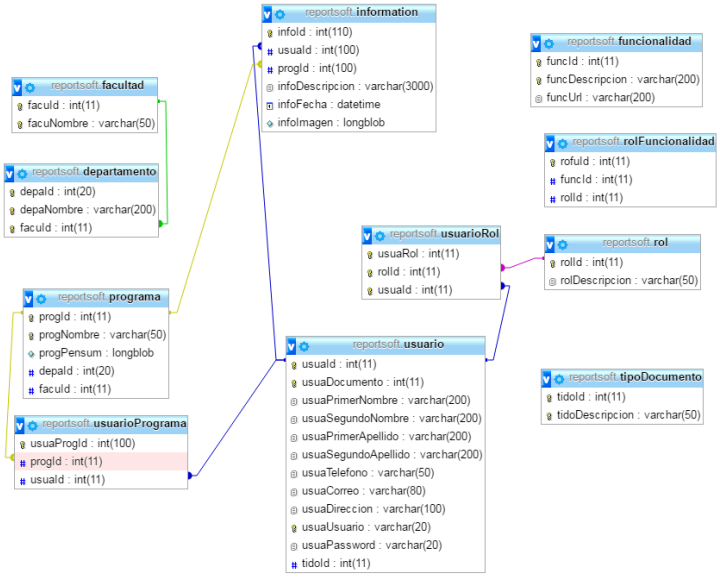

Figura 3 Modelo relacional de la base de datos reportsoft

Fuente: Elaboración propia