UNIVERSIDAD NACIONAL SAN CRISTÓBAL DE HUAMANGA

FACULTAD DE INGENIERÍA DE MINAS, GEOLOGÍA Y CIVIL

ESCUELA DE FORMACIÓN PROFESIONAL DE INGENIERÍA DE SISTEMAS

TESIS PARA OPTAR EL TÍTULO DE INGENIERO DE SISTEMAS

"RIESGOS DE SEGURIDAD EN EL DESARROLLO DE APLICACIONES WEB, AYACUCHO, 2014"

PRESENTADO POR:
BACH. ROMÁN CANCHARI GUTIÉRREZ

ASESOR:
ING. MANUEL AVELINO LAGOS BARZOLA

11 DE AGOSTO DE 2014

"RIESGOS DE SEGURIDAD EN EL DESARROLLO DE APLICACIONES WEB, AYACUCHO, 2014".

RECOMENDADO : **11 DE JULIO DEL 2014**

APROBADO : **07 DE AGOSTO DEL 2014**

MSc. Ing. CARLOS A. PRADO PRADO
PRESIDENTE

Ing. EDITH F. GUEVARA MOROTE
MIEMBRO

Ing. MANUEL A. LAGOS BARZOLA
MIEMBRO

Ing. FLORO N. YANGALI GUERRA
SECRETARIO DOCENTE

Según el acuerdo constatado en el Acta, levantada el 07 de agosto del 2014, en la Sustentación de Tesis presentado por el Bachiller en Ingeniería de Sistemas Sr. Román CANCHARI GUTIÉRREZ, con la Tesis Titulado "RIESGOS DE SEGURIDAD EN EL DESARROLLO DE APLICACIONES WEB, AYACUCHO, 2014", fue calificado con la nota de QUINCE (15) por lo que se da la respectiva APROBACIÓN.

RECOMENDADO : 11 DE JULIO DEL 2014

APROBADO : 07 DE AGOSTO DEL 2014

MSc. Ing. CARLOS A. PRADO PRADO
PRESIDENTE

Ing.EDITH F. GUEV ARAMOROTE
MIEMBRO

Ing. MANUEL A. LAGOS BARZOLA
MIEMBRO

Ing. FLORO A. VANGALI GUERRA
SECRETARIO DOCENTE

Dedicatoria

Dedico esta tesis a mis padres Juan y Emilia y a mis hermanos.

Agradecimientos

- *A mis hermanos Edmundo, Sérgio, Dorisa y Ortencia, por apoyarme siempre.*
- *Al Ing. Manuel Lagos Barzola., por su tiempo en las correcciones de esta tesis.*
- *Al Ing. José Elías Yauri Vidalón, por animarme siempre.*
- *A mis amigos de TelWifi.SAC, por compartir sus conocimientos conmigo.*
- *Al Perú, por mi educación escolar y universitario.*

Índice general

Índice de figuras

Índice de cuadros

Resumen

El objetivo de la investigación es identificar los riesgos de seguridad contemplados en el desarrollo de aplicaciones web, ciudad de Ayacucho, 2014. Mediante el análisis documental, encuestas, entrevistas y tecnologías de internet. Con el propósito de conocer si los riesgos de seguridad más importantes de acuerdo al OWASP top 10 son contemplados en el desarrollo de aplicaciones web en las organizaciones más representativas de nuestro medio y la finalidad de plantear una serie de recomendaciones respecto a cómo minimizar los riesgos de seguridad a los desarrolladores y administradores de aplicaciones web

El diseño de la investigación es no experimental pues no se manipulan las variables sino se estudian tal como se presentan. La recolección de datos se realizó un un período de tiempo establecido, por lo tanto se considera que el diseño de la investigación es transeccional o transversal.

Para la investigación la población está representada por las organizaciones públicas y privadas más representativas de nuestro medio donde se desarrollan aplicaciones web, de octubre del 2013 hasta abril del 2014.

Para identificar los riesgos de seguridad en aplicaciones web se elaboró y se utilizó un cuestionario y una guía de entrevista, estos instrumentos de recolección de datos cumplen con las características de fiabilidad y validez de instrumentos , los mismos que dan garantía para su uso en la presente investigación.

Como resultado, se ha identificado que los riesgos de seguridad más importantes en aplicaciones web como la inyección SQL, XSS y CSRF, en la mayoría de los casos son desconocidos en las organizaciones más representativas de la ciudad de Ayacucho.

Palabras Claves: seguridad en aplicaciones web, riesgos en aplicaciones web, SQLi, XSS, OWASP top 10, SANS/CWE, WASC.

Introducción

Las aplicaciones web son apenas un poco más que un conjunto de archivos de hipertexto ligados que presenta información mediante texto y algunas gráficas. Sin embargo, a medida que el comercio electrónico adquieren mayor importancia, las WebApps evolucionan hacia ambientes computacionales sofisticados que no sólo proporcionan características, funciones de cómputo y contenidos independientes al usuario final, sino que están integradas con bases de datos corporativas y aplicaciones del negocio.

Por otra parte, la seguridad de las aplicaciones web son un campo reciente en la seguridad de la información, soluciones tradicionales como firewalls e IDs, no detectan muchos de los ataques a aplicaciones web, además la cantidad de ataques se ha incrementado en los últimos años y se estima que aumentarán en el futuro.

En la presente investigación se identifican los riesgos de seguridad contemplados en el desarrollo de aplicaciones web, ciudad de Ayacucho, 2014. Mediante el análisis documental, encuestas, entrevistas y tecnologías de internet. Con el propósito de conocer si los riesgos de seguridad más importantes de acuerdo al OWASP top 10 son contemplados en el desarrollo de aplicaciones web en las organizaciones más representativas de nuestro medio y la finalidad de plantear una serie de recomendaciones respecto a cómo minimizar los riesgos de seguridad a los desarrolladores y administradores de aplicaciones web.

La presente investigación se organiza de la siguiente manera:

En el Primer Capítulo se trata sobre el Planteamiento de la Investigación, justifica el porqué de la investigación y se plantea el objetivo y cómo este será alcanzado.

En el Segundo Capítulo se desarrolla el Marco Teórico, se identifican los antecedentes de la investigación y se definen los conceptos necesarios dentro del contexto de la investigación.

En el Tercer Capítulo de describe la Metodología de la Investigación, se determina la poblaciín y muestra de la investigación y se define las técnicas e instrumentos como

el análisis documental, encuestas y entrevistas.

El el Cuarto Capítulo Análisis y Resultados de la Investigación, se realiza el análisis y tratamiento de los datos sobre la muestra seleccionada en el capítulo III.

En el Quinto Capítulo se describe las Conclusiones y Recomendaciones de la investigación.

Capítulo 1

Planteamiento de la Investigación

1.1. Diagnóstico y Enunciado del Problema

Las auditorías de seguridad de las aplicaciones web también llamadas a veces tests de intrusión o hacking ético, tienen el inconveniente de costar muchísimo dinero y tardar mucho tiempo en su realizacíin. Por otro lado, la validez de sus resultados pierde vigencia con rapidez: una aplicación diagnosticada como segura hoy puede dejar de serlo dentro de diez días o de diez semanas, ya que van apareciendo nuevas vulnerabilidades, se va modificando la plataforma, se añaden funcionalidades a las ya existentes, etc. En definitiva, las auditorías de seguridad son costosas, lentas y sus resultados rápidamente quedan obsoletas [Marañón, 2009, p.84].

Configuraciones mal hechas, falta de atención y software con errores pueden crear desastres en la internet. Una de las áreas primarias de vulnerabilidad es a través de conexiones WWW. Por diseño, los servicios WWW pretenden ser abiertos y receptivos y usualmente actúan como una interfaz de recursos de información de valor. De este modo, es crítico que estos servicios sean seguros para salvaguardar la seguridad de la información [OWASP, 2004].

WhiteHat Security (2010). Afirma que la probabilidad de que un sitio web tenga una clase de vulnerabilidad es como se muestra en la figura 1.1.

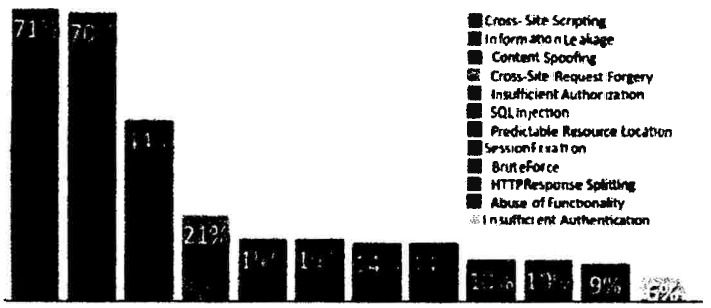


Figura 1.1: Probabilidad de que un sitio web tenga una vulnerabilidad (Fuente: WhiteHat security, 2010)

De acuerdo con [BBC, 2013] "La industria del hackeo, la industria del hackeo criminal, es en realidad la actividad criminal más grande en el mundo",(...) "Genera más dinero para los criminales que cualquier otro tipo de tráfico de drogas o armas (...) Es una industria enorme". Este mismo medio también publica en sus páginas:(...) Lo llamaron "el crimen en internet más grande de la historia". Se robaron 160 tarjetas de crédito de 12 compañías en Estados Unidos. A una le robaron US$ 200 millones y a otra, US$ 93 millones. Las autoridades arrestaron a dos de los cinco hackers que, según los reportes, también tuvieron acceso durante dos años al sistema del Nasqad, la bolsa de valores electrónica más grande de Estados Unidos.

Por otra parte, para ser un cracker o pirata informático no es necesario ser ningún genio de la informática, en internet hay un verdadero arsenal que permite invadir las redes de ordenadores con muy poco esfuerzo intelectual, muchas de estas herramientas de intrusión son muy sofisticados, que muestran incluso una interfaz muy amigable para el usuario, por lo tanto, cualquier persona con conocimientos básicos en redes e informática puede convertirse en un cracker. La verdad es que el nivel medio de la sofisticación del atacante es ahora mucho más pequeño, pero las herramientas utilizadas en los ataques son cada vez más potentes y fáciles de usar.

Resumiendo, las auditorías de las aplicaciones web son costosas, lentas y sus resultados rápidamente quedan obsoletas. Además, por diseño los servicios WWW pretenden ser abiertos y receptivos y usualmente actúan como una interfaz de recursos de información de valor, de igual manera el número de ataques a las aplicaciones web y las pérdidas económicas aumentan cada año y por último en internet hay un verdadero arsenal que permite invadir las redes de ordenadores con muy poco esfuerzo intelectual, muchas de estas herramientas de intrusión son muy sofisticados que muestran incluso una interfaz muy amigable para el usuario.

1.2. Formulación del Problema de Investigación

1.2.1. Problema principal

¿Cuáles son los riesgos de seguridad contemplados en el desarrollo de aplicaciones web, ciudad de Ayacucho, 2014?

1.2.2. Problemas específicos

A. ¿El riesgo de inyección SQL, está contemplado en el desarrollo de aplicaciones web?

B. ¿El riesgo de pérdida de autenticación y gestión de sesiones, está contemplado en el desarrollo de aplicaciones web?

C. ¿El riesgo de secuencia de comandos en sitios cruzados (XSS), está considerado en el desarrollo de aplicaciones web?

D. ¿El riesgo de configuración de seguridad incorrecta, está considerado en el desarrollo de aplicaciones web?

E. ¿El riesgo de exposición de datos sensibles, está contemplado en el desarrollo de aplicaciones web?

F. ¿El riesgo de falsificación de peticiones en sitios cruzados (CRSRF), está contemplado en el desarrollo de aplicaciones web?

G. ¿El riesgo de uso de componentes con vulnerabilidades conocidas, está considerado en el desarrollo de aplicaciones web?

1.3. Objetivos de la Investigación

1.3.1. Objetivo general

Identificar los riesgos de seguridad contemplados en el desarrollo de aplicaciones web, ciudad de Ayacucho, 2014. Mediante el análisis documental, encuestas, entrevistas y tecnologías de internet. Con el propósito de conocer si los riesgos de seguridad más importantes de acuerdo al OWASP top 10 son contemplados en el desarrollo de aplicaciones web en las organizaciones más representativas de nuestro medio y la finalidad de plantear una serie de recomendaciones respecto a cómo minimizar los riesgos de seguridad a los desarrolladores y administradores de aplicaciones web y así colaborar con la gestión de la seguridad informática en estas organizaciones.

1.3.2. Objetivos específicos

A. Identificar si el riesgo de inyección SQL está contemplado en el desarrollo de aplicaciones web.

B. Identificar si el riesgo de pérdida de autenticación y gestión de sesiones está contemplado en el desarrollo de aplicaciones web.

C. Identificar si el riesgo de secuencia de comandos en sitios cruzados (XSS) está considerado en el desarrollo de aplicaciones web.

D. Identificar si el riesgo de configuración de seguridad incorrecta está considerado en el desarrollo de aplicaciones web.

E. Identificar si el riesgo de exposición de datos sensibles está contemplado en el desarrollo de aplicaciones web.

F. Identificar si el riesgo de falsificación de peticiones en sitios cruzados (CRSRF) está contemplado en el desarrollo de aplicaciones web.

G. Identificar si el riesgo de uso de componentes con vulnerabilidades conocidas está considerado en el desarrollo de aplicaciones web.

1.4. Hipótesis de la Investigación

1.4.1. Hipótesis general

Los riesgos de seguridad contemplados en el desarrollo de aplicaciones web, ciudad de Ayacucho, 2014, son: inyección SQL, pérdida de autenticación y gestión de sesiones, secuencia de comandos en sitios cruzados (XSS), configuración de seguridad incorrecta, exposición de datos sensibles, falsificación de peticiones en sitios cruzados(CSRF) y uso de componentes con vulnerabilidades conocidas.

1.5. Justificación de la Investigación

En nuestro medio no existen investigaciones relacionados a los riesgos de seguridad contemplados en el desarrollo de aplicaciones web que puedan servir de línea base para investigaciones a un nivel más profundo como de tipo descriptivo o explicativo.

La seguridad de las aplicaciones web son un campo reciente en la seguridad de la información, soluciones tradicionales como firewalls e IDs, no detectan muchos de los

ataques a aplicaciones web, además la cantidad de ataques se ha incrementado en los últimos años y se estima que aumentarán en el futuro (Innovae, 2011).

1.6. Importancia de la Investigación

1.6.1. Importancia técnica

Con esta investigación se da a conocer los riesgos de seguridad más importantes que se contemplan en el desarrollo de las aplicaciones web en las organizaciones más representativas de la ciudad de Ayacucho. Como parte final de la investigación se plantearán una serie de sugerencias para disminuir los riesgos de seguridad críticos no contemplados en el desarrollo de aplicaciones web.

1.6.2. Importancia económica

Según un estudio realizado por el CSI / FBI Computer Crime and Security , los problemas de seguridad principalmente provocan un gran impacto económico. En 1998 sólo en EE.UU. $5.000 millones de dólares en pérdidas económicas causadas por delitos informáticos. Estudios más recientes hablan de cantidades desorbitantes. En el año 2000, 273 organizaciones reportaron pérdidas por un valor de $265, 589,940 dólares (CSI / FBI, 2008, Citado en Informatica64, 2013).

De igual manera, el número de ataques y las pérdidas económicas que estos problemas producen aumentan cada año. Existe una necesidad real de protegerse ante estos ataques. Es necesario estudiar la seguridad de los sistemas y adecuarlas a las necesidades actuales (Alonso, 2011). Si bien es cierto que en nuestro medio no existen estadísticas acerca de las pérdidas económicas, en definitiva tampoco estamos aislados de los problemas de seguridad a nivel mundial que conllevarían a cuantiosas pérdidas en las organizaciones.

1.6.3. Importancia social

Los riesgos de seguridad en aplicaciones web tienen un impacto directo en la confidencialidad, integridad, disponibilidad de la información o en la continuidad de los servicios prestados mediante las aplicaciones web.

Existen organizaciones públicas y privadas que administran información confidencial de clientes , socios u otro tipo de información; y que por medio de sus aplicaciones web y/o servidores pueden ser vulnerados afectando a los propietarios de dicha información.

1.7. Delimitación de la Investigación

1.7.1. Delimitación espacial

La investigación se realizó en la ciudad de Ayacucho - Perú, considerando las organizaciones públicas y privadas más representativas donde se desarrollan aplicaciones web.

1.7.2. Delimitación temporal

La investigación se realizó durante el periodo de octubre del 2013 hasta abril del 2014, el tiempo estimado de la investigación fue de siete meses, el financiamiento estuvo a cargo del investigador.

1.7.3. Delimitación conceptual

Se estudió los riesgos de seguridad en aplicaciones web del OWASP top 10 (Open Web Application Security Project, en español Proyecto Abierto de Seguridad de Aplicaciones Web). Así mismo, para el estudio de las variables se consideró el CWE SANS 25 (TOP 25 Most Dangerous Software Errors) y WASC (Web Application Security Consortium).

De los diez riesgos de seguridad del OWASP top 10, se estudió siete riesgos por ser este un tema muy amplio y se descartaron tres, estos son: referencia directa insegura a objetos, ausencia de control de acceso a las funciones y redirecciones y reenvíos no validados.

Capítulo 2

Marco Teórico

2.1. Antecedentes de la Investigación

2.1.1. En el ámbito local

En el año 2012, Cruz Ayala, Javier, en su trabajo de tesis titulado: Técnicas de protección para mejorar la seguridad de una aplicación web. Mediante el lenguaje java, implementó algunas técnicas de protección para mejorar la seguridad de una aplicación web, entre ellos: cifrado de datos, creación de certificados digitales, configuración de apache tomcat para el uso del protocolo https, implementación de un teclado virtual, implementación de captcha y finalmente para la protección de la URL usó el from controller.

2.1.2. En el ámbito nacional

En el año 2012, Zavaleta De la Cruz, Yury Daniel, en su trabajo de tesis titulado: Impacto de ataques SQL injection en los portales web interactivos de las empresas del sector TI de la ciudad de Trujillo, llegó a la conclusión de que sin importar la tecnología que se use, mientras un sistema Web interaccione con la base de datos, en las distintas etapas del ciclo de vida de dicho sistema; la aplicación de los tópicos de la metodología por cada proceso del framework permite llevar un mejor control y, en el mejor de los casos, la mitigación de la Vulnerabilidad SQL Injection.

2.1.3. En el ámbito internacional

En el año 2012, Martin Kompan, en la universidad Masaryk University, facultad de informática, en su tesis de maestría titulado: Enterprise Web Application Security, llegó a la conclusión de que el problema de secuestro de sesión es la vulnerabilidad más grave en las aplicaciones web. Así mismo recomienda implementar políticas de uso de contraseñas más estrictas que no permitan que los usuarios utilicen contraseñas débiles y susceptibles a ataques de diccionario.

En el año 2012, Delgado Caballero, Gerson Geovanny, en su trabajo de investigación titulado: Metodología de pruebas de inyección SQL para entornos web, llegó a la conclusión de que el eslabón más débil en una aplicación web para que permita ataques de inyección SQL es la poca o ninguna validación de las variables de entrada. Así mismo indica que los desarrolladores deben reconocer que la seguridad es un componente fundamental de cualquier producto de software y deben incluir buenas prácticas de programación en el software que están desarrollando.

2.2. Marco Teórico

2.2.1. Software

"El software es instrucciones (programas de cómputo) que cuando se ejecutan proporcionan las características, función y desempeño buscados"[Demarco. 1995, citado en Pressman,2010, p.3].

2.2.1.1. Dominios de aplicación del software

Actualmente, hay siete grandes categorías de software de computadora que plantean retos continuos a los ingenieros de software[Pressman, 2010. p.6-7].

a. Software del sistema.
b. Software de aplicación.
c. Software de ingeniería y ciencias.
d. Software incrustado.
e. Software de línea de productos.
f. Aplicaciones web.
g. Software de inteligencia artificial.

2.2.2. Fundamentos de la web

El éxito espectacular de la web se basa en dos puntos fundamentales: el protocolo HTTP y el lenguaje HTML. Uno permite una implementación simple y sencilla de un sistema de comunicaciones y el otro nos proporciona un mecanismo de composición de páginas enlazadas simple y fácil.

2.2.2.1. El protocolo HTTP

El protocolo HTTP (*hypertext transfer protocol*) es el protocolo base de la WWW. se trata de un protocolo simple, orientado a conexión y sin estado. La razón de que esté orientado a conexión es que emplea para su funcionamiento un protocolo de comunicaciones (TCP, *Transport control protocol*). El protocolo no mantiene estado, es decir cada transferencia de datos es una conexión independiente a la anterior, sin relación alguna entre ellas[Mateu, 2004, p.13-14].

Existe una variante de HTTP llamada HTTPS (S por *secure*) que utiliza el protocolo de seguridad SSL (*secure socket layer*) para cifrar y autenticar el tráfico entre cliente y servidor, siendo esta muy usada por los servidores web de comercio electrónico, así como por aquellos que contienen información personal o confidencial[Mateu, 2004, p.13-14].

Protocolo utilizado por la World Wide Web para el intercambio de documentos de hipertexto entre un servidor web y un cliente web[securityfocus, 2013].

2.2.2.2. El lenguaje HTML

Se trata de un lenguaje de marcas (se utiliza marcas en el interior del texto) que nos permite representar de forma rica el contenido y también referenciar otros recursos (imágenes, etc.), enlaces a otros documentos (la característica más destacada del WWW), mostrar formularios para posteriormente procesarlos, etc[Mateu, 2004, p.19-20].

Siglas de HyperText Markup Language, lenguaje de formateo utilizado para crear páginas Web[securityfocus, 2013].

2.2.3. Vulneravilidad

"Dado que una aplicación posee un conjunto de activos (recursos de valor como los datos en una base de datos o en el sistema de archivos), una vulnerabilidad es una

debilidad en un activo que hace posible a una amenaza. Así que una amenaza es un caso potencial que puede dañar un activo mediante la explotación de una vulnerabilidad. Un test es una acción que tiende a mostrar una vulnerabilidad en la aplicación" |OWASP, 2008|.

"Estimación de la exposición efectiva de un activo a una amenaza. Se determina por dos medidas: frecuencia de ocurrencia y degradación causada"|Magerit, 2006|.

"La vulnerabilidad puede definirse como la falta de una garantía que causa una debilidad y que podría ser explotada. Las vulnerabilidades pueden surgir desde el diseño, implementación y configuración de hardware, software o procesos"|Scambray et al., 2010, p. 26|.

2.2.4. Aplicación web

En su forma más simple, las WebApps son apenas un poco más que un conjunto de archivos de hipertexto ligados que presenta información mediante texto y algunas gráficas. Sin embargo, a medida que el comercio electrónico y las aplicaciones B2B adquieren mayor importancia, las WebApps evolucionan hacia ambientes computacionales sofisticados que no sólo proporcionan características, funciones de cómputo y contenidos independientes al usuario final, sino que están integradas con bases de datos corporativas y aplicaciones de negocio|Pressman, 2010, p.7|.

2.2.4.1. La naturaleza única de las WebApps

La gran mayoría de WebApps presentan los siguientes atributos|Pressman, 2010, p.9|.

a. Uso intensivo de redes.

b. Concurrencia.

c. Carga impredecible.

d. Rendimiento.

e. Disponibilidad.

f. Orientada a los datos.

g. Contenido sensible.

h. Evolución continua.

i. Inmediatez.

j. Seguridad.

k. Estética.

2.2.5. Vulnerabilidad de las aplicaciones web

Son los fallos que se pueden presentar en el análisis, diseño e implantación de las aplicaciones web, así como también en la correcta configuración una vez puesta en marcha, estos fallos no pueden ser resueltos íntegramente y sólo se puede minimizar su efecto tomando las políticas de seguridad adecuadas y estando atento a los problemas de seguridad o a las nuevas vulnerabilidades que puede aparecer en cualquier momento|OWASP, 2008|.

2.2.6. Riesgo

Efecto de la incertidumbre en los objetivos |ISO/IEC27000, 2014|.

Estimación del grado de exposición a que una amenaza se materialice sobre uno o más activos causando daños o perjuicios a la organización |Magerit, 2006|.

Combinación de la probabilidad de un evento y sus consecuencias |NTP-ISO/IEC17799, 2007|.

2.2.6.1. Análisis de riesgo

El análisis de riesgo es un proceso sistemático para estimar la magnitud de los riesgos al que esta expuesto una organización. El análisis de riesgo permite determinar como es, cuanto vale y como de protegidos se encuentran los activos |Magerit, 2006|.

Proceso de comprender la naturaleza del riesgo para determinar el nivel del riesgo |ISO/IEC27000, 2014|.

2.2.6.2. Sistema de gestión de seguridad de la información

Es un sistema de gestión que comprende la política, la estructura organizativa, los procedimientos, los procesos y los recursos necesarios para implantar la gestión de la seguridad de la información. Este sistema es la herramienta de que dispone la dirección de las organizaciones para llevar a cabo las políticas y los objetivos de seguridad (integridad, confidencialidad, disponibilidad, asignación de responsabilidad, autenticación, etc.) |AENOR, 2004|.

"Debido a la complejidad de llevar a cabo un plan de seguridad, es necesario una metodología. Por este motivo aparecieron los sistemas de gestión de la seguridad de la

información (SGSI). En general, cualquier sistema de gestión de la seguridad, tendrá que comprender la política, la estructura organizativa, los procedimientos, los procesos y los recursos necesarios para implantar la gestión de la seguridad de la información dentro de una organización" [Colobran, 2008].

2.2.7. OWASP

OWASP (en inglés acrónimo de Open Web Application Security Project, en español "Proyecto abierto de seguridad en aplicaciones web") es un proyecto de código abierto dedicado a determinar y combatir las causas que hacen que las aplicaciones web sean inseguros. La Fundación OWASP es un organismo sin ánimo de lucro que apoya y gestiona los proyectos e infraestructura de OWASP. La comunidad OWASP está formada por empresas, organizaciones educativas y particulares de todo mundo. Juntos constituyen una comunidad de seguridad informática que trabaja para crear artículos, metodologías, documentación, herramientas y tecnologías que se liberan y pueden ser usadas gratuitamente por cualquiera.

Los documentos con más éxito de OWASP incluyen la Guía OWASP y el ampliamente adoptado documento de autoevaluación OWASP Top 10. Las herramientas OWASP más usadas incluyen el entorno de formación WebGoat, la herramienta de pruebas de penetración WebScarab y las utilidades de seguridad para entornos .NET OWASP DotNet. OWASP cuenta con unos 50 capítulos locales por todo el mundo y miles de participantes en las listas de correo del proyecto. OWASP ha organizado la serie de conferencias AppSec para mejorar la construcción de la comunidad de seguridad de aplicaciones web.

2.2.7.1. OWASP Top 10

Es un documento de alto nivel que se centra sobre las vulnerabilidades más críticas de las aplicaciones web. El objetivo principal del Top 10 es educar desarrolladores, diseñadores, arquitectos, gerentes, y organizaciones sobre las consecuencias de las vulnerabilidades de seguridad más importantes en aplicaciones web [OWASP, 2013].

Esta importante actualización representa una lista concisa y enfocada sobre los diez riesgos más críticos sobre seguridad en aplicaciones. El OWASP Top 10 ha sido siempre sobre riesgos, pero esta actualización lo evidencia de mayor manera respecto a ediciones anteriores. También provee información adicional sobre como evaluar estos riesgos en

información (SGSI). En general, cualquier sistema de gestión de la seguridad, tendrá que comprender la política, la estructura organizativa, los procedimientos, los procesos y los recursos necesarios para implantar la gestión de la seguridad de la información dentro de una organización" |Colobran, 2008|.

2.2.7. OWASP

OWASP (en inglés acrónimo de Open Web Application Security Project, en español "Proyecto abierto de seguridad en aplicaciones web") es un proyecto de código abierto dedicado a determinar y combatir las causas que hacen que las aplicaciones web sean inseguros. La Fundación OWASP es un organismo sin ánimo de lucro que apoya y gestiona los proyectos e infraestructura de OWASP. La comunidad OWASP está formada por empresas, organizaciones educativas y particulares de todo mundo. Juntos constituyen una comunidad de seguridad informática que trabaja para crear artículos, metodologías, documentación, herramientas y tecnologías que se liberan y pueden ser usadas gratuitamente por cualquiera.

Los documentos con más éxito de OWASP incluyen la Guía OWASP y el ampliamente adoptado documento de autoevaluación OWASP Top 10. Las herramientas OWASP más usadas incluyen el entorno de formación WebGoat, la herramienta de pruebas de penetración WebScarab y las utilidades de seguridad para entornos .NET OWASP DotNet. OWASP cuenta con unos 50 capítulos locales por todo el mundo y miles de participantes en las listas de correo del proyecto. OWASP ha organizado la serie de conferencias AppSec para mejorar la construcción de la comunidad de seguridad de aplicaciones web.

2.2.7.1. OWASP Top 10

Es un documento de alto nivel que se centra sobre las vulnerabilidades más críticas de las aplicaciones web. El objetivo principal del Top 10 es educar desarrolladores, diseñadores, arquitectos, gerentes, y organizaciones sobre las consecuencias de las vulnerabilidades de seguridad más importantes en aplicaciones web |OWASP, 2013|.

Esta importante actualización representa una lista concisa y enfocada sobre los diez riesgos más críticos sobre seguridad en aplicaciones. El OWASP Top 10 ha sido siempre sobre riesgos, pero esta actualización lo evidencia de mayor manera respecto a ediciones anteriores. También provee información adicional sobre como evaluar estos riesgos en

sus aplicaciones [OWASP, 2013].

Cuadro 2.1: Evolución de los Top 10 de OWASP(Fuente: elaboración propia)

Top 10 2003	Top 10 2004	Top 10 2007	Top 10 2010	Top 10 2013
A1-Parámetros invalidados	A1-Entrada no validada	A1-Secuencia de comandos en sitios cruzados(XSS)	A1-Inyección	A1-Inyección
A2-Control de acceso interrumpido	A2-Control de acceso interrumpido	A2-Fallas de inyección	A2-Secuencia de comandos en sitios cruzados(XSS)	A2-Pérdida de autenticación y gestión de sesiones
A3-Administración de cuentas y sesión interrumpida	A3-Administración de autenticación y sesión interrumpida	A3-Ejecución de ficheros malintencionados	A3-Pérdida de autenticación y gestión de sesiones	A3-Secuencia de comandos en sitios cruzados(XSS)
A4-Fallas de cross site scripting(XSS)	A4-Fallas de cross site scripting(XSS)	A4-Referencia insegura y directa a objetos	A4-Referencia directa insegura a objetos	A4-Referencia directa insegura a objetos
A5-Desbordamiento de bufer	A5-Desbordamiento de bufer	A5-Falsificación de peticiones en sitios cruzados(CSRF)	A5-Falsificación de peticiones en sitios cruzados(CSRF)	A5-Configuración de seguridad incorrecta
A6-Fallas de inyección de comandos	A6-Fallas de inyección	A6-Revelación de información y gestión incorrecta de errores	A6-Defectuosa configuración de seguridad	A6-Exposición de datos sensibles
A7-Problemas de manejo de errores	A7-Manejo inadecuado de errores	A7-Pérdida de autenticación y gestión de sesiones	A7-Almacenamiento criptográfico inseguro	A7-Ausencia de control de acceso a las funciones
A8-Uso inseguro de criptografía	A8-Almacenamiento inseguro	A8-Almacenamiento criptográfico inseguro	A8-Falla de restricción de acceso a URL	A8-Falsificación de peticiones en sitios cruzados(CSRF)

Continua en la siguiente página

Cuadro 2.1 – *Continuación de la página anterior*

Top 10 2003	Top 10 2004	Top 10 2007	Top 10 2010	Top 10 2013
A9-Fallas de administración remota(no aplicable)	A9-Negación de servicio	A9-Comunicaciones inseguras	A9-Protección insuficiente en la capa de transporte	A9-Uso de componentes con vulnerabilidades conocidas
A10-Configuración indebida de servidor web y de aplicación	A10-Administración de configuración insegura	A10-Falla de restricción de acceso a URL	A10-Redirecciones y reenvíos no validados	A10-Redirecciones y reenvíos no validados

2.2.8. Riesgos de seguridad en aplicaciones web

"Los atacantes pueden potencialmente usar muchas rutas diferentes a través de su aplicación para causar daño en su negocio u organización, cada una de estas rutas representa un riesgo que puede, o no, ser lo suficientemente serio para merecer atención. A veces estas rutas son triviales de encontrar y explotar y a veces son extremadamente difíciles. De manera similar, el daño causado puede ir de ninguno hasta incluso sacarlo del negocio"[OWASP, 2008].

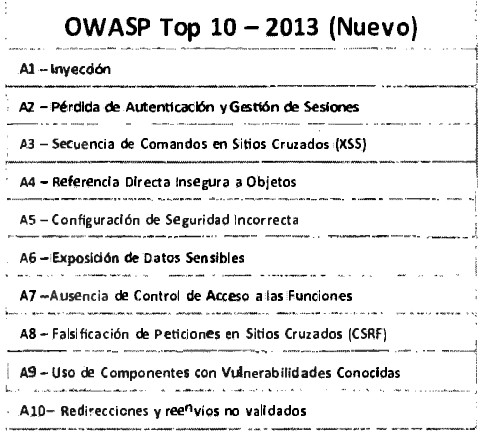


Figura 2.1: Los 10 riesgos de seguridad en aplicaciones web(Fuente: Owasp top 10, 2013)

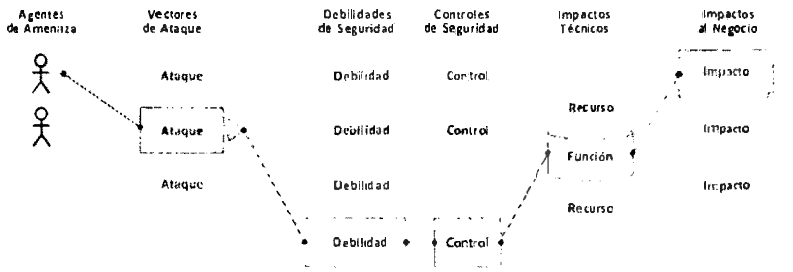


Figura 2.2: Riesgos de seguridad en aplicaciones (Fuente: Owasp top 10, 2013)

2.2.8.1. Inyección SQL

Agentes de Amenaza	Vectores de Ataque	Debilidades de Seguridad		Impactos Técnicos	Impactos al negocio
Específico de la Aplicación	[illegible]	Prevalencia COMÚN	Detección PROMEDIO	[illegible]	Específico de la aplicación/negocio
Considere a cualquier persona que pueda enviar información no confiable al sistema, incluyendo usuarios externos, usuarios internos y administradores.	El atacante envía ataques con cadenas simples de texto, los cuales explotan la sintaxis del intérprete a vulnerar. Casi cualquier fuente de datos puede ser un vector de inyección, incluyendo las fuentes internas.	Las fallas de inyección ocurren cuando una aplicación envía información no confiable a un intérprete. Estas fallas son muy comunes, particularmente en el código antiguo. Se encuentran, frecuentemente, en las consultas SQL, LDAP, Xpath o NoSQL; los comandos de SO, intérpretes de XML, encabezados de SMTP, argumentos de programas, etc. Estas fallas son fáciles de descubrir al examinar el código, pero difíciles de descubrir por medio de pruebas. Los analizadores y «fuzzers» pueden ayudar a los atacantes a encontrar fallas de inyección.		Una inyección puede causar pérdida o corrupción de datos, pérdida de responsabilidad, o negación de acceso. Algunas veces, una inyección puede llevar a el compromiso total de el servidor.	Considere el valor de negocio de los datos afectados y la plataforma sobre la que corre el intérprete. Todos los datos pueden ser robados, modificados o eliminados. ¿Podría ser dañada su reputación?

Figura 2.3: Inyección (Fuente: Owasp top 10, 2013)

Un ataque de inyección SQL consiste en la inserción o "inyección" de datos en una consulta SQL desde un cliente de la aplicación. El éxito de una inyección SQL puede leer datos sensibles de la base de datos, modificar los datos (insertar/actualizar, borrar), realizar operaciones de administración sobre la base de datos (como reiniciar el DBMS), recuperar el contenido de un archivo del sistema de archivos del DBMS y en algunos casos, ejecutar ordenes en el sistema operativo. Los ataques de inyección SQL son un tipo de ataques de inyección, en los que órdenes SQL son inyectados en texto para afectar la correcta realización de una consulta SQL predefinida[OWASP, 2008, p.221-222].

Es un tipo de vulnerabilidad en las WebApps en la que un atacante puede manipular un comando SQL para recuperar la información de la base de datos. Este tipo de ataque se produce sobre todo cuando una aplicación web se ejecuta mediante el uso de los datos proporcionados por el usuario sin la validación adecuada. Se puede permitir el acceso a

información confidencial, como los números de tarjetas de crédito o datos financieros y permite a un atacante crear, leer, actualizar, modificar o borrar los datos almacenados en la base de datos. Se trata de una falla en las WebApps y no una cuestión de base de datos o servidor web. La mayoría de los programadores no son conscientes de esta amenaza [CEH, 2013].

Independientemente del tipo de ataque, una inyección SQL correcta requiere que el atacante pueda construir una consulta SQL correcta. Si la aplicación devuelve un mensaje de error a causa de una consulta incorrecta, entonces en fácil reconstruir de forma lógica la consulta original y, por lo tanto, entender cómo realizar una inyección correctamente. Sin embargo, si la aplicación oculta los mensajes de error, un auditor puede conseguir por medio de ingeniería inversa la lógica de la consulta original. Este último caso se conoce como "inyección SQL ciega" (blind SQL injection).

Tipos de ataques de inyección SQL

Cuadro 2.2: Tipos de inyección SQL(Fuente: [Trejos, 2010])

Tipo de SQLi	Método de trabajo	Ejemplo
Tautologías	Se inyecta código en la cadena SQL, de la consulta, en una o en todas las declaraciones condicionales, de forma que siempre sean evaluadas como verdaderas. En este caso se aprovecha un parámetro de entrada vulnerable, usado para construir la condición. El ataque resulta efectivo, cuando de la tabla objetivo, se logra recuperar cuando menos un registro.	SELECT * from TblUser where Username '' or 1=1–and pass= ''
Errores Lógicos / Consultas Ilegales	Se aplica reingeniería sobre los mensajes de errores, enviados desde la base de datos como respuesta, para obtener información acerca del esquema de la base de datos. Un ejemplo es la adición de la comilla dentro de la cadena de texto suministrada desde el parámetro de entrada.	SELECT * FROM TblSupplier WHERE NameSupplier – 'O'Reill'
Basados en el Operador Union	Es posible modificar el conjunto de resultados de la consulta original, cambiando su lógica. A través de un parámetro de entrada vulnerable, se adiciona una segunda consulta mediante el operador union. Es posible recuperar resultados de diferentes tablas producto de la unión de las tablas de la consulta original y de la segunda consulta adicionada.	SELECT * FROM TblSupplier WHERE NameSupplier = '' UNION ALL SELECT * From TblConsumer WHERE 1 – 1

Continua en la siguiente página

Cuadro 2.2 – *Continuación de la página anterior*

Tipo de SQLi	Método de trabajo	Ejemplo
Piggy-backed Queries	Este tipo de ataque no modifica la lógica de la consulta original. A su vez, adiciona una nueva consulta totalmente diferente a la primera, con lo que la base de datos recibe más de una consulta SQL. Este ataque depende de la configuración de la base de datos, cuando permite múltiples consultas en una única cadena de texto SQL.	SELECT * FROM TblUser WHERE UserName = ''; DROP TABLE TblUser-'AND pass = ''
Basada en Inferencia	La inyección a ciegas permite inferir tomando en cuenta el comportamiento de la página cuando se envía una pregunta (cierto/falso) al servidor. Si la respuesta es verdadera, la aplicación continúa con su funcionamiento normal. En caso contrario, el funcionamiento de la página difiere del habitual. El timing attacks es similar a la inyección a ciegas, pero utiliza un método de inferencia diferente. La estructura de la consulta a ejecutar es un enunciado de la forma SELECT IF(expression, true, false) donde en unas de las ramas se incluye una función de tiempo. Por ejemplo BENCHMARK()o WAITFOR DELAY. Midiendo el incremento o decremento del tiempo de repuesta, es posible determinar la rama seguida por la consulta e inferior en la respuesta.	Declare @s varchar(8000); select @s db-name(); if (ascii(substring(@s,1,1)) and (power(2, 0))) >0 waitfor delay '0:0:5'
Basada en Stores Procedures	Lo primero es conocer cuál producto de base de datos está dando soporte. El problema se deriva de la extensión en la funcionalidad que la mayoría de los productos de bases de datos ofrecen mediante la incorporación de procedimientos almacenados. Estos procedimiento almacenados pueden incluso interactuar con el sistema operativo. Es posible elaborar consultas maliciosas que aprovechen la funcionalidad de un procedimiento almacenado para tomar el control del servidor o bloquearlo.	Simplequoted.asp?city seattle';EXEC master.dbo.xp-cmdshell 'cmd.exe dir c:

Continua en la siguiente página.

Cuadro 2.2 – *Continuación de la página anterior*

Tipo de SQLi	Método de trabajo	Ejemplo
Codificación Variable	Más que un tipo de ataque, esto permite evadir los mecanismos de detección y prevención de intrusiones. Mediante la codificación, es posible ocultar a los mecanismos de seguridad patrones conocidos de ataque. Para codificar las cadenas de texto, se utilizan varios métodos de codificación tales como hexadecimal, ASCII o UNICODE). Esta técnica de codificación hace difícil plantear algún mecanismo de seguridad basado en la revisión de código, sencillamente porque resulta difícil manejar todas las posibles variaciones.	declare @q varchar(8000); select @q 0x73656c6563742040 4076657273696f6e; exec(@q) El resultado de la codificación es 'select @@version'
Combinación de Ataques	Las técnicas descritas arribas se pueden combinar para alcanzar más de un objetivo a la vez. Por ejemplo, es posible recuperar información del esquema de la base de datos mediante consultas ilegales, al mismo tiempo que se apunta a identificar el producto de la base de datos para aprovechar el uso de procedimientos almacenados que den acceso al sistema operativo. Todo ello, oculto bajo un método de codificación que no sea detectado por los mecanismos de seguridad.	

El alcance de este riesgo de seguridad según WASC (2010), CWE/SANS top 25 (2011) y OWASP top 10 (2013):

Problemas de seguridad:

- Lectura, modificación o borrado de datos en una base de datos.
- Ejecución de comandos en la base de datos.
- Evasión de los mecanismos de autenticación
- Modificación de los datos sin autorización
- Descarga de archivos desde el servidor de base de datos comprometido
- Carga de archivos con código malicioso
- Interrupción de los servicios y del sistema.
- Escalamiento de privilegios.
- Robo de información confidencial.

Plataformas aplicables:

- **Lenguajes de programación:** Independiente del lenguaje de programación.
- **Clases de tecnología:** Servidor de base de datos.

Modos de introducción: Esta debilidad aparece típicamente en las partes de las aplicaciones donde el usuario interactúa con la base de datos.

Consecuencias comunes:

- **Confidencialidad:** Lectura de datos de la base de datos.
- **Integridad:** Modificación de datos de la base de datos.
- **Control de acceso:** Pasan los mecanismos de protección, usados para identificar el nombre de usuario y contraseña. Obtención de privilegios de una cuenta.

Probabilidad de ocurrencia: Muy alto.

Ejemplos demostrativos: La aplicación utiliza datos no confiables en la construcción de la siguiente consulta vulnerable SQL:

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

El atacante modifica el parámetro "id" en su navegador para enviar: ' or '1' '1. Esto cambia el significado de la consulta devolviendo todos los registros de la tabla accounts en lugar de solo el cliente solicitado.

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

En el peor caso, el atacante utiliza esta vulnerabilidad para invocar procedimientos almacenados especiales en la base de datos que permiten la toma de posesión de la base de datos y posiblemente también al servidor que aloja la misma.

Métodos de detección:

- Análisis estático automatizado
- Análisis dinámico automatizado
- Análisis manual

Cómo prevenirlos:

- Para prevenir la inyección SQL se requiere mantener los datos no confiables separados de comandos y consultas.
- Utilizar librerías que administre la validación de entradas.

- Hacer filtros que dejen pasar sólo lo que está bien, aquello con error debe ser rechazado.
- Configurar correctamente el servidor para evitar lo que el atacante pueda hacer si logra ejecutar (mínimos privilegios).
- Validar contra una especificación rigurosa de lo que debe ser permitido en todas las: *Cabeceras, Cookies, Cadenas de petición, Campos de formularios, Campos escondidos, etc. (todos los parámetros).*

2.2.8.2. Pérdida de autenticación y gestión de sesiones

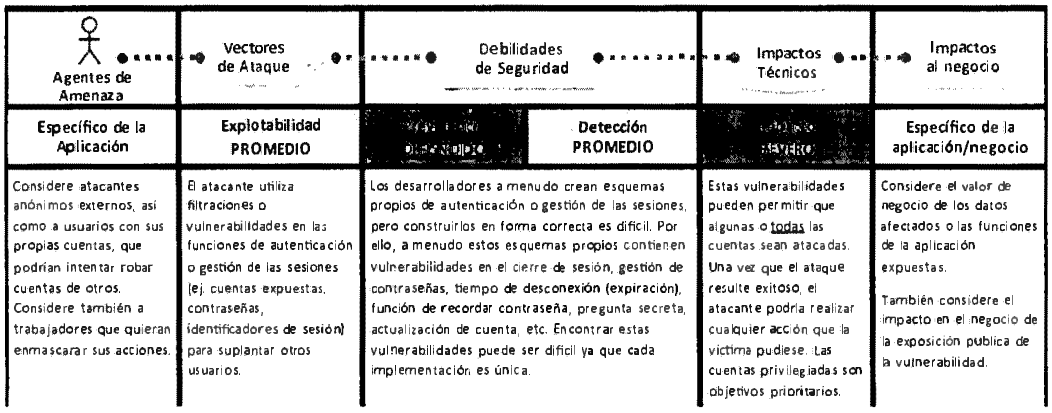


Figura 2.4: Pérdida de autenticación y gestión de sessiones (Fuente: Owasp top 10, 2013)

Las funciones de la aplicación relacionadas a la autenticación y gestión de sesiones son frecuentemente implementadas incorrectamente, permitiendo a los atacantes comprometer contraseñas, llaves, token de sesiones, o explotar otras fallas de implementación para asumir la identidad de otros usuarios|OWASP, 2013|.

El software no implementa medidas suficientes para evitar múltiples intentos fallidos de autenticación dentro de un corto plazo, haciéndola más susceptible a ataques de fuerza bruta|CWE/SANS, 2011|.

Ocurren cuando las credenciales de la cuenta y los tokens de sesión no están bien protegidos|Herrera, 2012, p.40|.

El alcance de este riesgo de seguridad según WASC (2010), CWE/SANS top 25 (2011) y OWASP top 10 (2013):

Problemas de seguridad:

- Comprometer las contraseñas y cookies de sesión.

- Asumir la identidad de otros usuarios.
- Robo de sesiones abiertas.
- Reutilización de sesiones.
- Cambio de usuario o contraseña.

Plataformas aplicables:

- **Lenguajes de programación:** Independiente del lenguaje de programación.

Consecuencias comunes:

- **Confidencialidad:** Lectura de datos de la aplicación.
- **Integridad:** Ejecución de códigos o comandos no autorizados.
- **Disponibilidad:** Denegación de servicio: *exit / restart.*
- **Control de acceso:** Asumir identidad, obtener privilegios de la cuenta asociada.

Probabilidad de ocurrencia: Alto a muy alto.

Ejemplos demostrativos:

Ejemplo 01: Una aplicación de reserva de vuelos que soporta re-escritura de direcciones URL poniendo los identificadores de sesión en la propia dirección:

```
http://example.com/sale/saleitems;jsessionid=2↩
   P00C2JDPXM00QSNDLPSKHCJUN2JV?dest=Hawaii
```

Un usuario autenticado en el sitio quiere mostrar la venta a sus amigos. Envía por correo electrónico el enlace anterior, sin ser consciente de que está proporcionando su identificador de sesión. Cuando sus amigos utilicen el anterior enlace utilizarán su sesión y su tarjeta de crédito.

Ejemplo 02: No se establecen correctamente los tiempos de desconexión en la aplicación. Un usuario utiliza un ordenador público para acceder al sitio. En lugar de utilizar la función de "Cerrar sesión", cierra la pestaña del navegador y se marcha. Un atacante utiliza el mismo navegador al cabo de una hora, y ese navegador todavía se encuentra autenticado.

Ejemplo 03: Un atacante dentro de la organización, o externo, consigue acceder a la base de datos de contraseñas del sistema. Las contraseñas de los usuarios no se encuentran cifradas, mostrando todas las contraseñas en texto claro al atacante.

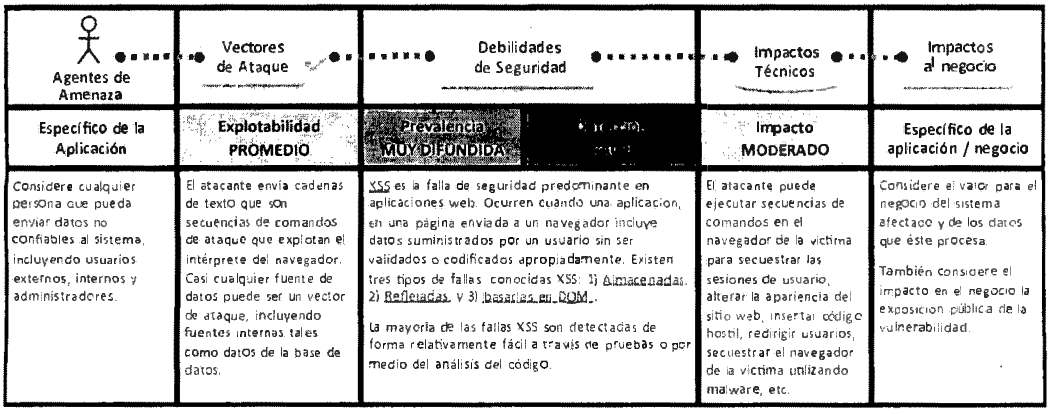
Métodos de detección:

- Análisis manual.
- Análisis estático automatizado.

Cómo prevenirlos:

- Usar mecanismos de autenticación y de manejo de sesión comerciales *(no reinventar la rueda)*.
- Administración de contraseñas: *contraseñas complejas (letras, números y símbolos), con rotación.*
- Bloqueo de usuarios: *al equivocarse en el intento de ingresar se debe de bloquear la cuenta del usuario.*
- Control de cambios de contraseñas: *requerir la contraseña vieja y no enviar por correo electrónico (salvo una nueva temporal).*
- Contraseñas cifradas: *deben guardarse y transmitirse siempre cifradas las contraseñas de los usuarios con hoshes no reversibles.*
- Proteger credenciales en tránsito: *usar mecanismos de challenge-response para evitar que viaje el hash y cifrar la conexión siempre con SSL.*
- Proteger el ID de la sesión: *el ID debe ser una cadena larga, compleja y totalmente aleatoria. Usar algoritmos comerciales para evitar malos diseños. Idealmente usar siempre SSL para proteger el número. Nunca poner el ID en el url.*
- Lista de Cuentas: *evitar que el atacante pueda obtener listas o nombres de usuarios válidos.*
- Cache del Navegador: *usar POST en lugar de GETs y marcar todas las etiquetas para evitar que se guarden las credenciales del usuario en el cache.*
- Relación de Confianza: *evitar relaciones de confianza entre componentes de la aplicación. Cada componente, módulo o sistema deberá de autenticarse para evitar que sea mal utilizada esa conexión.*

2.2.8.3. Secuencia de comandos en sitios cruzados (XSS)



Agentes de Amenaza	Vectores de Ataque	Debilidades de Seguridad	Impactos Técnicos	Impactos al negocio
Específico de la Aplicación	Explotabilidad PROMEDIO	Prevalencia MUY DIFUNDIDA	Impacto MODERADO	Específico de la aplicación / negocio
Considere cualquier persona que pueda enviar datos no confiables al sistema, incluyendo usuarios externos, internos y administradores.	El atacante envía cadenas de texto que son secuencias de comandos de ataque que explotan el intérprete del navegador. Casi cualquier fuente de datos puede ser un vector de ataque, incluyendo fuentes internas tales como datos de la base de datos.	XSS es la falla de seguridad predominante en aplicaciones web. Ocurren cuando una aplicación, en una página enviada a un navegador incluye datos suministrados por un usuario sin ser validados o codificados apropiadamente. Existen tres tipos de fallas conocidas XSS: 1) Almacenadas, 2) Reflejadas, y 3) basadas en DOM. La mayoría de las fallas XSS son detectadas de forma relativamente fácil a través de pruebas o por medio del análisis del código.	El atacante puede ejecutar secuencias de comandos en el navegador de la víctima para secuestrar las sesiones de usuario, alterar la apariencia del sitio web, insertar código hostil, redirigir usuarios, secuestrar el navegador de la víctima utilizando malware, etc.	Considere el valor para el negocio del sistema afectado y de los datos que éste procesa. También considere el impacto en el negocio la exposición pública de la vulnerabilidad.

Figura 2.5: XSS (Fuente: Owasp top 10, 2013)

Las fallas XSS ocurren cada vez que una aplicación toma datos no confiables y los envía al navegador web sin una validación y codificación apropiada. XSS permite a los atacantes ejecutar secuencia de comandos en el navegador de la víctima los cuales pueden secuestrar las sesiones de usuario, destruir sitios web, o dirigir al usuario hacia un sitio malicioso[OWASP, 2013].

Por lo general, un usuario malintencionado va a confeccionar un script del lado del cliente, que cuando analizada por un navegador web realiza alguna actividad (como el envío de todas las cookies de sitios a una dirección de correo electrónico dada). Este script se carga y se ejecuta por cada usuario que visita el sitio web. Dado que el sitio que solicita para ejecutar el script tiene acceso a las cookies que se trate, el script malicioso aprovecha esta funcionalidad[CWE/SANS, 2011].

Cross-site scripting (XSS) es un agujero de seguridad típico de las aplicaciones web, donde el atacante puede inyectar código en una aplicación web que se ha de ejecutar en el sitio del visitante. Esto es posible siempre que la entrada del usuario se muestra en el sitio Web , por ejemplo, en los libros de visitas, esta forma de ataque se puede utilizar para explotar los errores del navegador, como desbordamientos de búfer y defectos del activeX o para robar las cookies [Neff, 2002].

[...] mecanismo para transportar un ataque al navegador del usuario final [...], estos pueden ser de dos tipos: almacenados o reflejados[Herrera, 2012, p.25].

El ataque más comunes que se realizan con el cross-site scripting consiste en la divulgación de la información almacenada en las cookies de los usuarios. La consecuen-

cia de un ataque XSS es la misma independientemente de si se almacena o refleja. La diferencia está en cómo llega la carga en el servidor.

El alcance de este riesgo de seguridad según WASC (2010), CWE/SANS top 25 (2011) y OWASP top 10 (2013):

Problemas de seguridad:

- Compromiso de la sesión del usuario.
- Envío del usuario y contraseña y otros datos al sitio del atacante.
- Ataque a la máquina local: *Divulgación de archivos, Instalación de troyanos.*
- Enmascarar contenido para engañar al usuario.
- Navegación forzada.
- Inserción de comandos.
- Buffer overflow.
- Ataque de formato de cadena de caracteres.
- Manipulación de cookies y campos escondidos.

Plataformas aplicables:

- **Lenguajes de programación:** Independiente del lenguaje de programación.
- **Paradigmas de arquitectura:** Basada en la web.
- **Clase de tecnología:** Servidor web.

Consecuencias comunes:

- **Confidencialidad:** Lectura de datos de la aplicación.
- **Integridad:** Ejecución de comandos o códigos no autorizados.
- **Disponibilidad:** Denegación de servicio: *exit / restart.* En algunas circunstancias, puede ser posible ejecutar código arbitrario en el ordenador de la víctima cuando cross-site scripting se combina con otros defectos.
- **Control de acceso:** Lectura de datos de la aplicación.

Probabilidad de ocurrencia: Alto a muy alto.

Factores que hacen posible la explotación: XSS almacenado tiene su inicio con los sitios web que ofrecen un "libro de visitas" a los visitantes. Los atacantes podrían incluir JavaScript en su entrada del libro y en todos los visitantes posteriores a la página de libro de visitas se ejecute el código malicioso. Las vulnerabilidades

XSS son causadas por el código que incluye datos no validados en una respuesta HTTP.

Ejemplos demostrativos:

Ejemplo 01: La aplicación utiliza datos no confiables en la construcción del siguiente código HTML sin validar o escapar los datos:

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

El atacante modifica el parámetro 'CC' en el navegador:

```
'><script>document.location= 'http://www.attacker.com/cgi-bin/↩
    cookie.cgi? foo='+document.cookie</script>'.
```

Esto causa que el identificador de sesión de la víctima sea enviado al sitio web del atacante, permitiendo secuestrar la sesión actual del usuario. Notar que los atacantes pueden también utilizar XSS para anular cualquier defensa CSRF que la aplicación pueda utilizar.

Métodos de detección:

- Análisis estático automatizado.
- Caja negra.

Cómo prevenirlos:

- **Arquitectura y diseño:** Utilizar una biblioteca o un marco que no permite que esta debilidad se produzca o proporciona construcciones que hacen de esta debilidad más fácil de evitar. Son ejemplos de bibliotecas de Microsoft Anti-XSS, el módulo de codificación OWASP ESAPI y Apache Wicket.
- Validar contra una especificación rigurosa de lo que debe ser permitido en todas las: *Cabeceras, Cookies, Cadenas de petición, Campos de formularios, Campos escondidos, etc. (todos los parámetros).*
- No usar filtros negativos, pueden ser burlados fácilmente.
- Adicionalmente se pueden cambiar caracteres peligrosos a la salida para evitar que sean ejecutados por javascript a nivel del navegador.

```
< = &lt;
> = &gt;
( = &#40;
) = &#41;
# = &#35;
```

```
6 & = &#38;
```

- Desactivar todo tipo de server side includes para que no puedan ser abusados por un atacante.

```
1 <!--#include virtual="nombre archivo"-->
2 <!--#exec cmd="comando"-->
```

- Filtrado de caracteres de salida HTML

2.2.8.4. Configuración de seguridad incorrecta

Agentes de Amenaza	Vectores de Ataque	Debilidades de Seguridad		Impactos Técnicos	Impactos al negocio
Específico de la Aplicación	[illegible]	Prevalencia COMÚN	[illegible]	Impacto MODERADO	Específico de la aplicación / negocio
Considere atacantes anónimos externos así como usuarios con sus propias cuentas que pueden intentar comprometer el sistema También considere personal interno buscando enmascarar sus acciones	Un atacante accedea cuentas por defecto, páginas sin uso, fallas sin parchear, archivos y directorios sin protección, etc para obtener acceso no autorizado o conocimiento del sistema	Las configuraciones de seguridad incorrectas pueden ocurrir a cualquier nivel de la aplicación, incluyendo la plataforma, servidor web, servidor de aplicación, base de datos, framework, y código personalizado. Los desarrolladores y administradores de sistema necesitan trabajar juntos para asegurar que las distintas capas están configuradas apropiadamente. Las herramientas de detección automatizadas son útiles para detectar parches omitidos, fallos de configuración uso de cuentas por defecto, servicios innecesarios etc		Estas vulnerabilidades frecuentemente dan a los atacantes acceso no autorizado a algunas funcionalidades o datos del sistema Ocasionalmente provocan que el sistema se comprometa totalmente	El sistema podría ser completamente comprometido sin su conocimiento Todos sus datos podrían ser robados o modificados, irrevocablemente en el tiempo los costes de recuperación podrían ser altos

Figura 2.6: Configuración de seguridad incorrecta (Fuente: Owasp top 10, 2013)

Según la OWASP es un riesgo de seguridad en WebApps a causa de no tener definida e implementada una configuración segura para la aplicación, marcos de trabajo, servidor de aplicación, servidor web, base de datos y plataforma. Todas estas configuraciones deben ser definidas, implementadas y mantenidas ya que por lo general no son seguras por defecto. Esto incluye mantener todo el software actualizado, incluidas las librerías de código utilizadas por la aplicación[OWASP, 2013].

Los servidores web y los de aplicación no son seguros por defecto al ser instalados, es complicado mantener un bardening eficiente para estos servicios, además muchas aplicaciones no están desarrolladas para funcionar sobre servidores asegurados[Herrera, 2012, p.47].

El alcance de este riesgo de seguridad según WASC (2010), CWE/SANS top 25 (2011) y OWASP top 10 (2013):

Problemas de seguridad:

- Explotación de vulnerabilidades (Exploits).
- Acceso no autorizado a recursos sensitivos.
- Explotación de vulnerabilidades en servicios relacionados.
- Obtención de información de mensajes de error.
- Ataques de hombre en el medio.

Plataformas aplicables:

- **Lenguajes de programación:** Independiente del lenguaje de programación.

Consecuencias comunes:

- **Confidencialidad:** Lectura de datos de la aplicación, lectura de archivos o directorios.
- **Integridad:** Modificación de datos de la aplicación, modificación de archivos o directorios.
- **Disponibilidad:** Denegación de servicio: exit / restart.
- **Control de acceso:** Asumir identidad de otros usuarios.

Probabilidad de ocurrencia: Alto.

Ejemplos demostrativos:

- **Ejemplo 01:** El siguiente código establece el umask del proceso a 0 antes de crear un archivo y escribir "Hola mundo" en el archivo.

Listing 2.1: Lenguaje C

```
#define OUTFILE "hello.out"

umask(0);
FILE *out;
/* Ignore CWE-59 (link following) for brevity */
out = fopen(OUTFILE, "w");
if (out) {
fprintf(out, "hello world!\n");
fclose(out);
}
```

Después de ejecutar este programa en un sistema UNIX, ejecutando el comando "ls -l" podría devolver el siguiente resultado:

```
-rw-rw-rw- 1 username 13 Nov 24 17:58 hello.out
```

La cadena rw-rw - rw- indica que el propietario, grupo y el mundo (todos los usuarios) pueden leer el archivo y escribir en él.

- **Ejemplo 02:** La consola de administrador del servidor de aplicaciones se instaló automáticamente y no se ha eliminado las cuentas por defecto. Un atacante descubre las páginas por defecto de administración que están en su servidor, se conecta con las contraseñas por defecto y lo toma.

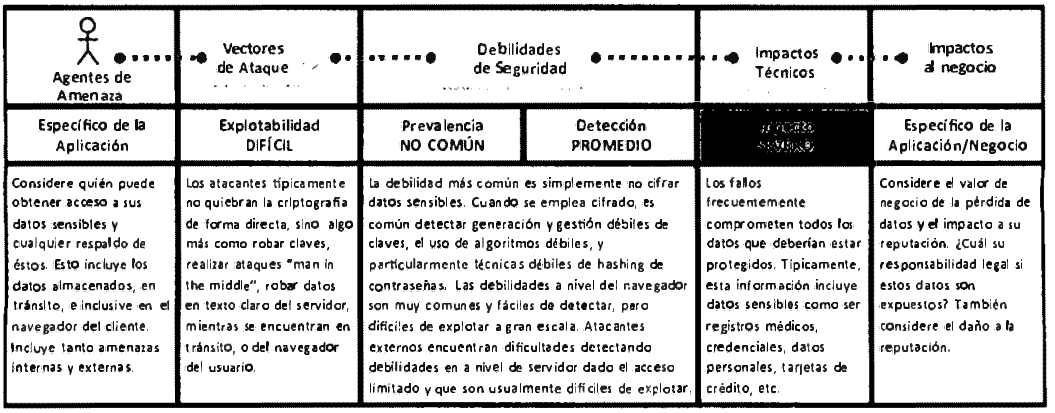
Métodos de detección:

- Análisis manual.
- Análisis estático automatizado.
- Análisis dinámico automatizado.
- Caja negra.

Cómo prevenirlos:

- **Arquitectura y diseño:** dividir el software en áreas anónimas, normales, privilegiadas y administrativas. Reducir la superficie de ataque cuidadosamente, definir roles, privilegios o grupos de usuarios distintos.
- **Configuración del sistema:** para todos los archivos de configuración, los ejecutables y bibliotecas, asegúrese de que sólo pueden leerse y escribirse por el administrador del programa.
- **Implementación e instalación:** establecer explícitamente los permisos predeterminados en la configuración más restrictiva posible. también establece los permisos adecuados durante la instalación del programa.
- Contar con una guía de hardening para su configuración particular de servidor web y servidor de aplicación: *hay muchas guías que se pueden usar como punto de partida: OWASP, CERT, SANS, NSA, Microsoft, etc.*
- Usar sistemas asegurados tanto en desarrollo como en producción.
- El hardening debe incluir: *configuración de todos los mecanismos de seguridad, dar de baja todos los servicios no usados, establecer roles, permisos y cuentas, incluyendo deshabilitar todas las cuentas por defecto o cambiar sus contraseñas, registro de eventos y alertas.*
- El mantenimiento es igual de importante, revisar, actualizar y parchar los sistemas constantemente y actualizar las guías.

2.2.8.5. Exposición de datos sensibles



Agentes de Amenaza	Vectores de Ataque	Debilidades de Seguridad		Impactos Técnicos	Impactos al negocio
Específico de la Aplicación	Explotabilidad DIFÍCIL	Prevalencia NO COMÚN	Detección PROMEDIO	[redacted]	Específico de la Aplicación/Negocio
Considere quién puede obtener acceso a sus datos sensibles y cualquier respaldo de éstos. Esto incluye los datos almacenados, en tránsito, e inclusive en el navegador del cliente. Incluye tanto amenazas internas y externas.	Los atacantes típicamente no quiebran la criptografía de forma directa, sino algo más como robar claves, realizar ataques "man in the middle", robar datos en texto claro del servidor, mientras se encuentran en tránsito, o del navegador del usuario.	La debilidad más común es simplemente no cifrar datos sensibles. Cuando se emplea cifrado, es común detectar generación y gestión débiles de claves, el uso de algoritmos débiles, y particularmente técnicas débiles de hashing de contraseñas. Las debilidades a nivel del navegador son muy comunes y fáciles de detectar, pero difíciles de explotar a gran escala. Atacantes externos encuentran dificultades detectando debilidades en a nivel de servidor dado el acceso limitado y que son usualmente difíciles de explotar.		Los fallos frecuentemente comprometen todos los datos que deberían estar protegidos. Típicamente, esta información incluye datos sensibles como ser registros médicos, credenciales, datos personales, tarjetas de crédito, etc.	Considere el valor de negocio de la pérdida de datos y el impacto a su reputación. ¿Cuál su responsabilidad legal si estos datos son expuestos? También considere el daño a la reputación.

Figura 2.7: Exposición de datos sensibles (Fuente: Owasp top 10, 2013)

Muchas aplicaciones web no protegen adecuadamente los datos sensibles, tales como tarjetas de crédito, NSSs, y credenciales de autenticación con mecanismos de cifrado o hashing. Los atacantes pueden modificar o robar tales datos protegidos inadecuadamente para conducir robos de identidad, fraudes de tarjeta de crédito u otros crímenes[OWASP, 2013].

El uso de un algoritmo no estándar es peligroso porque un atacante determinado puede ser capaz de romper el algoritmo y comprometer cualquier dato que ha sido protegido, técnicas bien conocidas pueden existir para romper el algoritmo[CWE/SANS, 2011].

El alcance de este riesgo de seguridad según CWE/SANS top 25 (2011) y OWASP top 10 (2013):

Problemas de seguridad:

- No cifrar datos sensibles.
- Elección pobre de algoritmo de cifrado.
- Intentar inventar el nuevo algoritmo de cifrado.

Plataformas aplicables:

- **Lenguajes de programación:** Independiente del lenguaje de programación.

Consecuencias comunes:

- **Confidencialidad:** Lectura de datos de la aplicación.

- **Integridad:** Modificación de datos de la aplicación.
- **Control de acceso:** Obtener privilegios, asumir identidad de otros usuarios.

Probabilidad de ocurrencia: Muy alto.

Ejemplos demostrativos:

- **Ejemplo 01:** Estos ejemplos de código utilizan el estándar de cifrado de datos (DES). Una vez considerado un algoritmo fuerte, es ahora considerado como insuficiente para muchas aplicaciones. Ha sido reemplazado por Advanced Encryption Standard (AES).

Listing 2.2: Ejemplo lenguaje C y C++

```
1 EVP_des_ecb();
```

Listing 2.3: Ejemplo lenguaje Java

```
1 Cipher des=Cipher.getInstance("DES...");
2 des.initEncrypt(key2);
```

Listing 2.4: Ejemplo lenguaje PHP

```
1 function encryptPassword($password){
2 $iv_size = mcrypt_get_iv_size(MCRYPT_DES, MCRYPT_MODE_ECB);
3 $iv = mcrypt_create_iv($iv_size, MCRYPT_RAND);
4 $key = "This is a password encryption key";
5 $encryptedPassword = mcrypt_encrypt(MCRYPT_DES, $key, ↩
     $password, MCRYPT_MODE_ECB, $iv);
6 return $encryptedPassword;
7 }
```

- **Ejemplo 02:** Una aplicación cifra los números de tarjetas de crédito en una base de datos utilizando cifrado automático de la base de datos. Esto significa que también se descifra estos datos automáticamente cuando se recuperan, permitiendo por medio de una debilidad de inyección de SQL recuperar números de tarjetas en texto claro. El sistema debería cifrar dichos número usando una clave pública, y permitir solamente a las aplicaciones de back-end descifrarlo con la clave privada.
- **Ejemplo 03:** Un sitio simplemente no utiliza SSL para todas sus páginas que requieren autenticación. El atacante monitorea el tráfico en la red (como puede ser una red inalámbrica abierta) y obtiene la cookie de sesión del usuario. El atacante reenvía la cookie y secuestra la sesión, accediendo los

datos privados del usuario.

Métodos de detección:

- Análisis automático.
- Análisis manual.

Cómo prevenirlos:

- **Requerimientos:** Especificar claramente qué datos son valiosos que deberían estar protegidas por encriptación. Cualquier transmisión o almacenamiento de esta información deberían utilizar los algoritmos de cifrado considerado robusto y seguro.
- **Arquitectura y diseño:** Utilizar librerías o frameworks. Tener en cuenta la característica de cifrado ESAPI.
- **Implementación:** Cuando se utilizan técnicas aprobadas usarlos correctamente, estos pasos son a menudo esenciales para la prevención de ataques comunes.
- Para contraseñas use algoritmos irreversibles (hashes usando sha1 o md5).
- Asegúrese de que los secretos (llaves, certificados, contraseñas) sean almacenados de forma segura.

2.2.8.6. Falsificación de peticiones en sitios cruzados(CSRF)

Agentes de Amenaza	Vectores de Ataque	Debilidades de Seguridad	Impactos Técnicos	Impactos al negocio
Específico de la Aplicación	**Explotabilidad PROMEDIO**	**Prevalencia COMÚN**	**Impacto MODERADO**	**Específico de la aplicación/negocio**
Considere cualquier persona que pueda cargar contenido en los navegadores de los usuarios, y así obligarlos a presentar una solicitud para su sitio web. Cualquier sitio web o canal HTML que el usuario acceda puede realizar este tipo de ataque.	El atacante crea peticiones HTTP falsificadas y engaña a la víctima mediante el envío de etiquetas de imágenes, XSS u otras técnicas. Si el usuario está autenticado, el ataque tiene éxito.	CSRF aprovecha el hecho que la mayoría de las aplicaciones web permiten a los atacantes predecir todos los detalles de una acción en particular. Dado que los navegadores envían credenciales como cookies de sesión de forma automática, los atacantes pueden crear páginas web maliciosas que generan peticiones falsificadas que son indistinguibles de las legítimas. La detección de fallos de tipo CSRF es bastante fácil a través de pruebas de penetración o de análisis de código.	Los atacantes pueden cambiar cualquier dato que la víctima esté autorizada a cambiar, o a acceder a cualquier funcionalidad donde esté autorizada, incluyendo registro, cambios de estado o cierre de sesión.	Considerar el valor de negocio asociado a los datos o funciones afectados. Tener en cuenta lo que representa no estar seguro si los usuarios en realidad desean realizar dichas acciones. Considerar el impacto que tiene en la reputación de su negocio.

Figura 2.8: Falsificación de peticiones en sitios cruzados (Fuente: Owasp top 10, 2013)

Un ataque CSRF obliga al navegador de una víctima autenticada a enviar una petición HTTP falsificado, incluyendo la sesión del usuario y cualquier otra información de autenticación incluida automáticamente a una aplicación web vulnerable. Esto permite

al atacante forzar al navegador de la víctima para generar pedidos que la aplicación vulnerable piensa son peticiones legítimas provenientes de la víctima[OWASP, 2013].

La aplicación web no tiene, o puede no, verificar suficientemente si una petición bien formada, válida y coherente intencionalmente fue proporcionada por el usuario que ha presentado la solicitud[CWE/SANS, 2011].

Todos los marcos de aplicaciones web son susceptibles a estos ataques. La sesión legítima del usuario puede ser aprovechada por un atacante para llevar a cabo acciones en nombre de la víctima. Estos pueden ser de dos tipos: almacenados o reflejados[Herrera, 2012, p.45].

El alcance de este riesgo de seguridad según WASC (2010), CWE/SANS top 25 (2011) y OWASP top 10 (2013):

Problemas de seguridad:

- Compromiso de la sesión de un usuario.
- Envío de usuario, contraseña y otros datos al sitio del atacante.
- Ejecución de transacciones automatizadas.
- Robo de información privada.

Plataformas aplicables:

- **Lenguajes de programación**: Independiente del lenguaje de programación.
- **Clase de tecnología**: Servidor web

Modos de introducción: Arquitectura y diseño.

Consecuencias comunes:

- **Confidencialidad**: Lectura de datos de la aplicación.
- **Integridad**: Modificación de datos de la aplicación.
- **Disponibilidad**: Denegación de servicios (DoS): exit / restart.
- **Control de acceso:** Obtener privilegios, asumir identidad de otros usuarios.

Probabilidad de ocurrencia: De medio a alto.

Ejemplos demostrativos:

- **Ejemplo 01**: Este ejemplo de código PHP intenta garantizar el proceso de envío de formulario al validar el usuario que ha enviado que el formulario tiene una sesión válida. Un ataque CSRF no podría haberlo evitado por esta respuesta porque el atacante forja una solicitud a través del navegador del

usuario en el cual existe ya una sesión válida. El siguiente código HTML está diseñado para permitir a un usuario actualizar un perfil.

Listing 2.5: Ejemplo lenguaje HTML

```
<form action="/url/profile.php" method="post">
<input type="text" name="firstname"/>
<input type="text" name="lastname"/>
<br/>
<input type="text" name="email"/>
<input type="submit" name="submit" value="Update"/>
</form>
```

Métodos de detección:

- Análisis manual.
- Análisis automático.

Cómo prevenirlos:

- **Arquitectura y diseño:** Utilizar librerías o frameworks, por ejemplo, utilizar paquetes de anti-CSRF como la OWASP CSRFGuard. Otro ejemplo es el control ESAPI Session Management, que incluye un componente de CSRF. No utilice el método GET para cualquier solicitud que desencadena un cambio de estado.
- **Implementación:** compruebe el encabezado HTTP Referer para ver si la solicitud se originó desde una página esperada, esto podría romper funcionalidad legítima, puesto que los usuarios o proxies pueden haver desactivado enviando el Referer por motivos de privacidad.
- Re-autentique al usuario para realizar transacciones críticas.
- Usar captcha.
- Lo ideal vuelve a ser el componente centralizado o librería que administre la validación de entradas.
- No utilice el método GET para transporte de datos sensitivos.
- Un POST sencillo no es suficiente protección.
- Validar contra una especificación rigurosa, de lo que debe ser permitido en todas las: *cabeceras, cookies, cadenas de petición, campos de formularios, campos escondidos, etc. (todos los parámetros).*
- No usar filtros negativos, pueden ser burlados fácilmente.

2.2.8.7. Utilización de componentes con vulnerabilidades conocidas

Agentes de Amenaza	Vectores de Ataque	Debilidades de Seguridad		Impactos Técnicos	Impactos al negocio
Específico de la Aplicación	Explotabilidad PROMEDIO	[illegible]	Detectabilidad DIFÍCIL	Impacto MODERADO	Específico de la aplicación / negocio
Algunos componentes vulnerables (por ejemplo frameworks) pueden ser identificados y explotados con herramientas automatizadas, aumentando las opciones de la amenaza más allá del objetivo atacado.	El atacante identifica un componente debil a través de escaneos automáticos o anilisis manuales. Ajusta el exploit como lo necesita y ejecuta el ataque. Se hace mas dificil siel componente es ampliamente utilizado en la aplicacion.	Virtualmente cualquier aplicación tiene este tipo de problema debido a que la mayoría de los equipos de desarrollo no se enfocan en asegurar que sus componentes / bibliotecas sa encuentren actualizadas. En muchos casos, los desarrolladores no conocen todos los componentes que utilizan, y menos sus versiones. Dependencias entre componentes dificultan incluso mas el problema.		El rango completo de debilidades incluye inyección, control de acceso roto. XSS etc. El impacto puede ser desde mínimo hasta apoderamiento completo del equipo y compromiso de los datos.	Considere qué puede significar cada vulnerabilidad para el negocio controlado por la aplicación afectada Puede ser trivial o puede significar compromiso completo

Figura 2.9: Componentes con vulnerabilidades conocidas (Fuente: Owasp top 10, 2013)

Algunos componentes tales como las librerías, los frameworks y otros módulos de software casi siempre funcionan con todos los privilegios. Si se ataca un componente vulnerable esto podría facilitar la intrusión en el servidor o una perdida de datos. Las aplicaciones que utilicen componentes con vulnerabilidades conocidas debilitan las defensas de la aplicación y permiten ampliar el rango de posibles ataques e impactos|OWASP, 2013|.

El software especifica los permisos de un recurso crítico para la seguridad de una manera que permita ese recurso ser leído o modificado por actores involuntarios|CWE/SANS, 2011|.

El alcance de este riesgo de seguridad según WASC (2010), CWE/SANS top 25 (2011) y OWASP top 10 (2013):

Problemas de seguridad:

- Pérdida de datos.
- Intrusión al servidor.

Plataformas aplicables:

- **Lenguajes de programación:** Independiente del lenguaje de programación.

Consecuencias comunes:

- **Confidencialidad:** lectura de datos de la aplicación.
- **Integridad:** Modificación de datos de la aplicación.

- **Disponibilidad:** denegación de servicios (DoS): exit / restart.
- **Control de acceso:** obtener privilegios, asumir identidad de otros usuarios.

Probabilidad de ocurrencia: de medio a alto.
Ejemplos demostrativos:

- **Ejemplo 01:** los siguientes componentes vulnerables fueron descargados 22 mil veces en el 2011.

  Apache CXF Authentication Bypass: debido a que no otorgaba un token de identidad, los atacantes podían invocar cualquier servicio web con todos los permisos. (Apache CXF es un framework de servicios, no confundir con el servidor de aplicaciones de Apache.)

  Spring Remote Code Execution: el abuso de la implementación en Spring del componente *Expression Languaje* permitió a los atacantes ejecutar código arbitrario, tomando el control del servidor. Cualquier aplicación que utilice cualquiera de esas bibliotecas vulnerables es susceptible de ataques.

  Ambos componentes son directamente accesibles por el usuario de la aplicación, otras bibliotecas vulnerables, usadas ampliamente en una aplicación, puede ser mas difíciles de explotar.

Métodos de detección:

- Análisis manual.
- Análisis estático automático.
- Análisis dinámico automático.
- Caja negra.

Cómo prevenirlos:

- **Implementación:** cuando se utiliza un recurso crítico como un archivo de configuración, compruebe para ver si el recurso tiene permisos inseguros (tales como ser modificables por cualquier usuario regular).
- **Arquitectura y diseño:** dividir el software en áreas anónimas, normales, privilegiadas y administrativas. Reducir la superficie de ataque, definir roles, privilegios o grupos de usuarios distintos.
- **Arquitectura, diseño y operación:** establecer explícitamente los permisos predeterminados o la configuración más restrictiva posible, también

establece los permisos adecuados durante la instalación del programa, esto le impedirá heredar los permisos inseguros de cualquier usuario que instala o ejecuta el programa.

- **Configuración del sistema:** para todos los archivos de configuración, ejecutables y bibliotecas asegúrese de que sólo pueden leerse y escribirse por el administrador del programa.
- **Operación, configuración del sistema:** asegurar que el software funciona correctamente bajo el Federal Desktop Core Configuration (FDCC) o un equivalente de hardening de la guía de configuración, que muchas organizaciones utilizan para limitar la superficie de ataque y riesgo potencial en la implementación del software.
- Identificar todos los componentes y la versión que están ocupando, incluyendo dependencias (ejm: la versión del plugin).
- Revisar la seguridad del componente en bases de datos públicas, lista de correos del proyecto y lista de correo de seguridad, y mantenerlos actualizados.
- Establecer políticas de seguridad que regulen el uso de componentes, como requerir ciertas prácticas en el desarrollo de software, pasar test de seguridad, y licencias aceptables.
- Sería apropiado considerar agregar capas de seguridad alrededor del componente para deshabilitar funcionalidades no utilizadas y/o asegurar aspectos débiles o vulnerables del componente.

2.2.9. Escáneres de sitios web

Lo más importante a remarcar es que estas herramientas son genéricas, es decir, que no están diseñadas para tu código específico, sino para aplicaciones en general. Lo que significa que aunque pueden encontrar algunos problemas genéricos, no tienen el conocimiento suficiente sobre tu aplicación como para permitirles detectar la mayoría de los fallos. Por experiencia, las incidencias de seguridad más serias son aquellas que no son genéricas, sino profundamente intrincadas en tu lógica de negocio y diseño a medida de la aplicación|OWASP, 2008, p.9|.

Según WhiteHatsecurity (2013), alguna de las características de los escáneres de aplicaciones web:

2.2.9.1. Lo bueno de los escáneres de sitios web

- Es repetible y medible (al menos en teoría).
- Reduce la probabilidad de error humano y la fatiga en las pruebas
- Optimiza el tiempo intensivo y tareas "aburrido".
- Ayuda a las personas que de otra manera no podrían ser calificados con la habilidad suficientes para hacer las pruebas.
- Es barato.

2.2.9.2. Lo malo de los escáneres de sitios web

- Aún requiere de una persona experta para operar.
- Los informes son confusos.
- Los resultados son propensos a errores.
- Los resultados son a menudo no recurribles.

2.2.9.3. La realidad de la mayoría de los escáneres

- Los escáneres tienden a tener un conocimiento limitado.
- Los escáneres no van a encontrar todo.
- Los escáneres fallan en "hacer las cosas".
- Los escáneres no pueden leer: *"Check your profile for your updated request ", " Check your email for your confirmation", "Hello, Admin!', "The password for this username does not match"," <!- u/p uadmin:uadmin->", "Click here only if you are sure you want to delete your account."*

2.2.10. Herramientas

2.2.10.1. SQLmap

SQLmap es una herramienta desarrollada en python para realizar inyección de código sql automáticamente. Su objetivo es detectar y aprovechar las vulnerabilidades de inyección SQL en aplicaciones web. Una vez que se detecta una o más inyecciones SQL en el host de destino, el usuario puede elegir entre una variedad de opciones entre ellas, enumerar los usuarios, los hashes de contraseñas, los privilegios, las bases de datos , o todo el volcado de tablas / columnas específicas del DBMS , ejecutar su propio SQL SELECT, leer archivos específicos en el sistema de archivos y mucho más.

2.2.10.2. BeEf

La BeEF es la abreviatura de The Browser Exploitation Framework. Es una herramienta de pruebas de penetración que se centra en el navegador web.

BeEF permite la prueba de intrusión profesional para evaluar la situación de seguridad actual de un entorno de destino mediante el uso de vectores de ataque del lado del cliente. A diferencia de otros marcos de seguridad, BeEF mira más allá del perímetro de la red endurecido y sistema cliente, y examina explotabilidad en el marco de la puerta abierta: el navegador web. BeEF enganchará uno o más de los navegadores web y los utilizan como cabezas de playa para el lanzamiento de los módulos de comando dirigidas y nuevos ataques contra el sistema desde dentro del contexto del explorador.

2.2.10.3. Kali linux v1.04

Es una distribución LiveCD de Linux, lo que significa que no hace falta instalar el sistema operativo ya que podemos correrlo directamente desde su CD o incluso desde un pendrive. Es una distribución creada en especial para quienes están relacionados con la seguridad informática[offensive security, 2013].

Capítulo 3

Metodología de la Investigación

3.1. Tipo de Investigación

Cuadro 3.1: Tipo de investigación

De acuerdo al	Clasificación	Descripción
Alcance	Transversal	Porque el análisis de los datos se realiza en un lapso de tiempo específico, de octubre del 2013 hasta abril del 2014.
Análisis de datos	Mixta	Porque se analiza y vincula datos cuantitativos y cualitativos en un mismo estudio para responder a los planteamientos del problema.
Profundidad	Exploratorio	Porque se examina un problema de investigación poco estudiado en nuestro medio y no se han abordado con anterioridad investigaciones detalladas y suficientes.
Lugar	De campo	Porque se analizan las variables de investigación tal como existen en la realidad.

3.2. Diseño de Investigación

El diseño de la investigación es no experimental pues no se manipulan las variables sino se estudian tal como se presentan. La recolección de datos se realizó en un periodo de tiempo establecido, por lo tanto se considera que el diseño de la investigación es transeccional o transversal.

3.3. Población y Muestra

3.3.1. Población

La población está constituida por las organizaciones más representativas donde se desarrollan aplicaciones web, dentro de la ciudad de Ayacucho - Perú, 2013 - 2014.

3.3.1.1. Criterios de inclusión y exclusión

- Ubicación: ciudad de Ayacucho.
- Periodo: de octubre del 2013 hasta abril del 2014.
- Organizaciones: públicas y privadas más representativas que tengan personal o área dedicado al desarrollo de aplicaciones web.

La población es finita, porque existe un número determinado de organizaciones donde se desarrollan aplicaciones web.

3.3.2. Muestra

El muestreo que se realizó fue el no probabilístico con juicio de experto y criterio de saturación.

El marco muestral fue para aquellas organizaciones donde desarrollan aplicaciones web, y que se encuentran ubicadas dentro de la ciudad de Ayacucho, 2013 y parte del 2014.

La cantidad de la muestra fue de 19 organizaciones, entre ellas 9 privadas y 10 píblicas.

Para determinar el tamaño de la muestra se hizo una selección intencional, para lo cual se consideró algunos puntos como:

- Aquellas empresas que se dedican al desarrollo de aplicaciones web.
- Aquellas organizaciones públicas y privadas más representativas donde desarrollan aplicaciones web en la ciudad de Ayacucho.
- Aquellas organizaciones existentes en el periodo de octubre del 2013 hasta abril del 2014.

Cuadro 3.2: Relación de la muestra

Nro	Institución	Dirección	Púb/Priv
1	C.A.C. San Cristobal de Huamanga	Jr. 28 de julio N° 113 - 117	Privado
2	Makipura Microfinanzas	Jr. 2 de mayo N° 210	Privado
3	C.A.C. Santa María Magdalena	Jr. San Martín N° 558	Privado
4	C.A.C. Federación de Mercados Ayacucho	Calle san juan de Dios N° 101	Privado
5	C.A.C. Fortaleza de Ayacucho	Av. mariscal cáceres N° 1228	Privado
6	Clínica el Nazareno	Jr. Quinua N° 428	Privado
7	NECSSEIN SRL	Jr. 9 de diciembre N° 253	Privado
8	TEL&WIFI SAC- Telecomunicaciones	Asoc. Aprovisa Mz. Z, Lt. 12	Privado
9	ELEKTRA SA	Jr. 2 de mayo N° 171	Privado
10	Gobierno Regional Ayacucho Sede Central	Jr. Callao Nro. 122	Público
11	Municipalidad Provincial de Huamanga	Portal Municipal N° 44	Público
12	Servicio de Administración tributaria Huamanga	Jr. 9 de diciembre N° 491	Público
13	Municipalidad distrital de San Juan Bautista	Jr. españa N° 119	Público
14	Corte Superior de Justicia Ayacucho	Portal Constitución N° 20	Público
15	Dirección Regional Agraria - Ayacucho	Av. independencia N° 604	Público
16	RENIEC - Ayacucho	Jr. San Martín N° 471	Público
17	EPSASA - Ayacucho	Jr. Manco Cápac N° 342	Público
18	COFOPRI - Ayacucho	Urb. Mariscal Cáceres Mz. H, Lt. 11	Público
19	SUNARP - Ayacucho	Av. Ramón Castilla N° 489	Público

3.4. Variables e Indicadores

3.4.1. Definición conceptual de las variables

3.4.1.1. Variable de estudio

Cuadro 3.3: Variable de estudio

Variable	Descripción
Riesgos de seguridad en aplicaciones web	Son las diferentes rutas empleadas a través de una aplicación web para causar daño en un negocio u organización, cada una de estas rutas representa un riesgo que puede, o no, ser lo suficientemente serio para merecer atención.

3.4.1.2. Indicadores de la variable de estudio

Cuadro 3.4: Indicadores de la variable de estudio

Indicador	Descripción
A1: Inyección SQL	Es un método de infiltración de código intruso que se vale de una vulnerabilidad informática presente en una aplicación en el nivel de validación de las entradas para realizar consultas a una base de datos.
A2: Pérdida de autenticación y gestión de sesiones	Es un agujero de seguridad típico de las aplicaciones web relacionadas a la autenticación y gestión de sesiones, que son frecuentemente implementadas incorrectamente, permitiendo a los atacantes comprometer contraseñas, llaves, token de sesiones, o explotar otras fallas de implementación para asumir la identidad de otros usuarios.
A3: Secuencia de comandos en sitios cruzados (XSS)	Es un tipo de inseguridad informática o agujero de seguridad típico de las aplicaciones web que permite a una tercera parte inyectar en páginas web vistas por el usuario código JavaScript u otro lenguaje script similar, permitiendo a los atacantes ejecutar secuencia de comandos en el navegador de la víctima los cuales pueden secuestrar las sesiones de usuario, destruir sitios web, o dirigir al usuario hacia un sitio malicioso.

Continua en la siguiente página

Cuadro 3.4 – *Continuación de la página anterior*

Indicadores	Descripción
A4: Configuración de seguridad incorrecta	Es un problema de seguridad en aplicaciones web que sucede a causa de no tener definida e implementada una configuración segura para la aplicación, marcos de trabajo, servidor de aplicación, servidor web y base de datos.
A5: Exposición de datos sensibles	Es un problema de seguridad en aplicaciones web y sucede cuando las aplicaciones no protegen adecuadamente los datos sensibles, tales como los números de tarjetas de crédito y credenciales de autenticación con mecanismos de cifrado o hashing.
A6: Falsificación de peticiones en sitios cruzados(CSRF)	Es un problema de seguridad en aplicaciones web, sucede cuando se obliga al navegador de una víctima autenticada a enviar una petición HTTP falsificado, incluyendo la sesión del usuario y cualquier otra información de autenticación incluida automáticamente a una aplicación web vulnerable.
A7: Uso de componentes con vulnerabilidades conocidas	Es un problema de seguridad en aplicaciones web. Se presenta cuando las librerías, los frameworks y otros módulos de la aplicación poseen vulnerabilidades. Si se ataca un componente vulnerable esto podría facilitar la intrusión en el servidor o una perdida de datos.

3.4.2. Definición operacional de las variables de estudio

Cuadro 3.5: Variables e indicadores

Variables	Indicadores
VI: Riesgos de seguridad en aplicaciones web	X1: Inyección SQL
	X2: Pérdida de autenticación y gestión de sesiones
	X3: Secuencia de comandos en sitios cruzados (XSS)
	X4:Configuración de seguridad incorrecta
	X5: Exposición de datos sensibles
	X6: Falsificación de peticiones en sitios cruzados(CSRF)
	X7: Uso de componentes con vulnerabilidades conocidas

3.5. Técnicas e Instrumentos de Recolección de Datos

3.5.1. Técnicas de recolección de datos

3.5.1.1. El análisis documental

Es una forma de investigación técnica, un conjunto de operaciones intelectuales, que buscan describir y representar los documentos de forma unificada sistemática para facilitar su recuperación. Comprende el procesamiento analítico-sintético que, a su vez, incluye la descripción bibliográfica y general de la fuente, la clasificación, anotación, extracción, traducción y la confección de reseñas.

3.5.1.2. La encuesta

Es un estudio observacional en el que el investigador busca recaudar datos por medio de un cuestionario previamente diseñado, sin modificar el entorno ni controlar el proceso que está en observación (como sí lo hace en un experimento). Los datos se obtienen realizando un conjunto de preguntas normalizadas dirigidas a una muestra representativa o al conjunto total de la población estadística en estudio, integrada a menudo por personas, empresas o entes institucionales, con el fin de conocer estados de opinión, características o hechos específicos.

3.5.1.3. La entrevista

Es la herramienta básica y fundamental de la investigación. Una entrevista es una conversación dirigida que nos permite recopilar información importante con un propósito específico. La entrevista es el método más directo y económico que se utiliza para obtener información. En la entrevista podemos utilizar dos tipos de preguntas: abiertas y cerradas. Además existen dos formas: La entrevista estructurada y la entrevista no estructurada.

3.5.2. Instrumentos de recolección de datos

Para la recolección de los datos se hizo la solicitud respectiva a las organizaciones públicas y privadas que conforman la muetra seleccionada. Se adjunta el Anexo A con las respectivas solicitudes a dichas organizaciones.

3.5.2.1. Guía de revisión documental

Para la técnica del análisis documental se utilizó la guía de revisión documental, para ello, se ha elaborado una guía de revisión documental considerando las siguientes variables: autor, año, título, muestra, diseño de investigación, instrumento o método de recolección, análisis de datos, conclusiones de las relaciones, lugar de ubicación de la muestra de estudio, tipo de organizaciones investigadas, aspectos investigados.

Cuadro 3.6: Técnica de recolección de datos (Fuente: [Horna, 2012])

Variables	Definición
Autor	Apellidos e iniciales del autor.
Año	Año de publicación del artículo
Título	Título del artículo de investigación
Muestra	Número y descripción de la muestra usada.
Diseño	Si la investigación es exploratoria, descriptiva, correlacional o explicativa (experimental o cuasiexperimental), de análisis de casos o meta - analítico.
Instrumentos/Métodos de recolección de datos	Nombre del instrumento empleado. Modalidad: encuesta, escala, guía de entrevista, focus group, etc
Análisis de datos	Métodos estadísticas o softwares especializados empleados para el análisis de los datos.
Conclusiones	Listado de las principales conclusiones del estudio.
Lugar de ubicación de la muestra	País o ciudad en dónde su ubica físicamente la muestra.

3.5.2.2. El cuestionario

Para la técnica de la encuesta se utilizó el cuestionario. Las preguntas del cuestionario se han elaborado principalmente mediante el análisis documental, para ello se ha considerado los recursos usados en el marco teórico del capítulo II, entre ellos: [OWASP, 2004], [OWASP, 2007], [OWASP, 2008], [OWASP, 2010], [OWASP, 2013], [ASVS, 2013], [CWE/SANS, 2011], [WASC, 2010], Conferencias Defcon y finalmente las observaciones y sugerencias del asesor de tesis.

Las encuestas fueron aplicadas a los desarrolladores y administradores de aplicacio-

nes web, los tipos de preguntas fueron cerradas, además corresponden a un cuestionario de forma estructurada.

3.5.2.3. La guía de la entrevista

Para la técnica de la entrevista se utilizó la guía de entrevista. Las preguntas de la entrevista se han elaborado principalmente mediante el análisis documental, para ello se ha considerado los recursos usados en el marco teórico del capítulo II, entre ellos: [OWASP, 2004], [OWASP, 2007], [OWASP, 2008], [OWASP, 2010], [OWASP, 2013], [ASVS, 2013] ,[CWE/SANS, 2011], [WASC, 2010], Conferencias Defcon y finalmente las observaciones y sugerencias del asesor de tesis.

Los entrevistados fueron los desarrolladores y administradores de aplicaciones web, además las preguntas fueron abiertas con el objetivo de conocer toda información sobre los riesgos de seguridad.

Cuadro 3.7: Técnicas, Instrumentos y Fuentes y sus principales ventajas y desventajas (Fuente: [Romero, 2011])

Técnica	Instrumento	Informantes o fuentes	Principales ventajas	Principales Desventajas
Encuestas	Cuestionario	Desarrolladores de aplicaciones web. Administradores de BD. Administradores de servidores.	Aplicable a gran número de informantes. Sobre gran número de datos.	Poca profundidad.
Entrevistas	Guía de entrevistas	Desarrolladores de aplicaciones web. Administradores de BD. Administradores de servidores.	Permite profundizar los aspectos interesantes.	Difícil y costosa Sólo aplicable a un pequeño número de informantes importantes.
Análisis documental	Fichas: Textuales, de resumen, etc.	Fuente: Libros especializados, internet.	Muy objetiva. Puede constituir evidencia.	Aplicación Limitada a fuentes documentales.

CUESTIONARIO SOBRE SEGURIDAD EN APLICACIONES WEB

El siguiente cuestionario está enfocado a los desarrolladores de aplicaciones web, tiene como objetivo obtener o recavar información sobre los riesgos de seguridad en aplicaciones web. Para asegurar la fiabilidad de la investigación se pide al entrevistado dar respuestas verídicas. Se le agradece de antemano por su tiempo.

Organización: _____

Dirección: _____ **Fecha:** _____

PREGUNTAS GENERALES

Instrucciones: Marque con un **X** el que crea conveniente.

1a. ¿Qué conocimiento tiene Ud. de los riesgos de seguridad en aplicaciones web del OWASP top 10?

- ☐ Domino el tema
- ☐ Conozco el tema
- ☐ Conozco poco
- ☐ Tengo referencias
- ☐ Desconozco el tema

1b. Si domina el tema o conoce el tema ¿Aplica Ud. los principios de OWASP a la seguridad de sus aplicaciones web?

- ☐ Sí
- ☐ No

2. ¿Qué industria o estándar a adoptado Ud. para implementar las cuestiones de seguridad en sus aplicaciones web?

- ☐ OWASP
- ☐ NIST
- ☐ SANS 25
- ☐ CWE
- ☐ Ninguna de las anteriores
- ☐ Otros (especifique)

_____

_____

3. ¿Qué porcentaje de sus aplicaciones web han sido testeados en busca de vulnerabilidades de seguridad?

- ☐ 0%, sin pruebas realizadas
- ☐ Menos del 25 %
- ☐ Menos del 50 %
- ☐ De 51 a 75%
- ☐ De 76 a 100 %

4. Para el testeo de las vulnerabilidades de seguridad de sus aplicaciones web ¿Qué técnicas utiliza usted?

- ☐ Escáneres de vulnerabilidad
- ☐ External pen testing
- ☐ Pruebas estáticas automatizadas
- ☐ Pruebas dinámicas automatizadas
- ☐ Análisis manual
- ☐ Ninguna de las anteriores
- ☐ Otros (especifique)

_____

_____

5. ¿Con qué frecuencia hace las pruebas de seguridad a sus aplicaciones web?

- ☐ Uno al año
- ☐ Dos veces al año
- ☐ Cada trimestre
- ☐ Cada vez que cambia el código
- ☐ No se realizan pruebas

6. ¿En qué etapa del desarrollo del software comprueba Ud. la seguridad de sus aplicaciones web?

- ☐ Producción
- ☐ Desarrollo
- ☐ Testing y control de calidad
- ☐ Ninguna de las anteriores
- ☐ Otros (especifique)

_____

_____

7. En los últimos dos años ¿Cuántas veces han sido hackeados sus aplicaciones web?

- ☐ 0, nunca
- ☐ 1 a 5 veces
- ☐ 6 a 10 veces
- ☐ Más de 10 veces
- ☐ No lo sé

8. Para usted ¿Cuáles son las dos razones más importantes para asegurar sus aplicaciones web?

- ☐ Cumplimiento (como PCI, HIPAA, GLBA o directivas de privacidad)

- □ Protección de datos
- □ Incidencias en el negocio
- □ Pérdida de ingresos
- □ Pérdida de clientes
- □ Protección del trabajo
- □ Otro (especifique)

_____

_____

PREGUNTAS ESPECÍFICAS

9. **Cuando desarrolla una aplicación web ¿Valida rigurosamente todas los campos de los formularios (incluido los comandos SQL y código javascript), campos escondidos, cabeceras, cookies y cadenas de petición?** □ Sí □ No
10. **¿Comprueba que sus aplicaciones web hacen uso del control de sesiones del marco de trabajo (framework) utilizado para el desarrollo de sus aplicaciones web?** □ Sí □ No
11. **¿Verifica que todos los errores producidos como resultado del rechazo de acceso por la aplicación web, están correctamente manejados por la aplicación?** □ Sí □ No
12. **¿Hace uso de procedimientos almacenados en lugar de consultas dinámicas?** □ Sí □ No
13. **¿Comprueba si los campos de ingreso de usuarios y contraseñas en sus aplicaciones web están protegidos contra un ataque de fuerza bruta?** □ Sí □ No
14. **¿Sus aplicaciones web cuentan con funciones para proteger contra el uso de contraseñas comúnmente elegidos?** □ Sí □ No
15. **¿Verifica que todos las credenciales de autenticación se guardan y se transmiten siempre cifradas con hashes no reversibles y no así en el código fuente?** □ Sí □ No
16. **¿Comprueba que sólo los identificadores de sesión generados por la aplicación web son reconocidos como válidos?** □ Sí □ No
17. **¿Verifica que las sesiones en las aplicaciones web caducan después de un período de tiempo de inactividad?** □ Sí □ No
18. **¿Comprueba que la aplicación web no permite que usuarios concurrentes dupliquen sesiones?** □ Sí □ No
19. **¿Comprueba que sus aplicaciones web no son susceptibles a ataques de Cross Site Scripting (XSS) o que los controles de seguridad impiden XSS?** □ Sí □ No
20. **¿Utiliza algún tipo de biblioteca que no permite que el riesgo de Cross Site Scripting (XSS) se produzca ?** □ Sí □ No
21. **¿Comprueba que todos los datos ingresados por un usuario que son la salida a HTML (incluyendo elementos HTML, atributos HTML, Java script, valores de datos, bloques de CSS y atributos URI) están debidamente validados en la aplicación web?** □ Sí □ No
22. **¿Cuenta con una guía de hardening (como los de OWASP, CERT, SANS, NSA, Microsoft, etc. que ayudan a configurar correctamente los sistemas) para personalizar los componentes de sus aplicaciones web?** □ Sí □ No
23. **¿Conoce usted, todos los número de puertos, servicios, páginas, cuentas y privilegios que están configurados actualmente en su servidor web)?** □ Sí □ No
24. **¿Indique los algoritmos criptográficos que utiliza?**

Criptografía simétrica:

- □ DES
- □ 3DES
- □ RC5
- □ AES
- □ Otros (especifique)

_____

_____

Criptografía asimétrica:

- □ RSA
- □ DSA
- □ ElGamal
- □ Criptografía de curva elíptica
- □ Otros (especifique)

_____

_____

25. ¿Comprueba que los datos confidenciales procesados por la aplicación web son cifrados tanto en el almancenamiento como en la transmición de datos? □ Sí □ No
26. ¿Utiliza TLS (Seguridad de la Capa de Transporte) para todas las conexiones (incluyendo tanto internas como externas) en las que impliquen datos sensibles o funciones especiales? □ Sí □ No
27. ¿ Utiliza algún tipo de librería que proporcione construcciones que hacen más fácil de evitar el riesgo de CSRF (Falsificación de Peticiones en Sitios Cruzados) ? □ Sí □ No
28. ¿Implementa usted funciones de reautenticación de usuarios para realizar operaciones críticas? □ Sí □ No
29. ¿Implementa Ud. mecanismos como los captcha para comprobar que las solicitudes provenientes son de un usuario real? □ Sí □ No
30. ¿Indique los nombres y las versiones de los componentes de sus aplicaciones web?

Componente	Nombre del componente	Versión
Lenguajes de programación		
Frameworks de desarrollo		
SGDB		
Servidor Web		
Servidor de aplicaciones		
Otros		

31. ¿Revisa, actualiza y parcha constantemente los componentes de la aplicación (servidor web, servidor de aplicación, etc) y actualiza las guías? □ Sí □ No
32. ¿Revisa la seguridad de los componentes de sus aplicaciones web en listas de correo de seguridad tales como: securityfocus, hugtraq, CERT, etc. y los mantiene actualizados? □ Sí □ No

GUÍA DE ENTREVISTA SOBRE SEGURIDAD EN APLICACIONES WEB

La siguiente entrevista está enfocado a los desarrolladores de aplicaciones web, tiene como objetivo obtener o recavar información sobre los riesgos de seguridad en aplicaciones web. Además servirá para corroborar la información obtenida en el cuestionario usado en la presente investigación.

Organización: _____

Dirección: _____ **Fecha:** _____

PREGUNTAS DE LA ENTREVISTA

1. **¿Qué riesgos de seguridad o vulnerabilidades en aplicaciones web conoce Ud.?**
   - ☐ Inyección SQL
   - ☐ Perdida de Autenticación y Gestión de Sesiones
   - ☐ Secuencia de Comandos en Sitios Cruzados (XSS)
   - ☐ Configuración de Seguridad Incorrecta
   - ☐ Exposición de Datos Sensibles
   - ☐ Falsificación de Petición en Sitios Cruzados (CSRF)
   - ☐ Uso de Componentes con Vulnerabilidades Conocidas
   - ☐ Ninguna de las anteriores
   - ☐ Otros (notas adicionales)

   _____

   _____

2. **¿Qué información tiene Ud. del riesgo o vulnerabilidad de Inyección SQL y qué soluciones conoce para prevenirlo?**

   Conocimientos:

   - ☐ Domina el tema
   - ☐ Conoce el tema
   - ☐ Conoce poco
   - ☐ Tiene referencias
   - ☐ Desconoce el tema
   - ☐ Otros (Notas adicionales)

   _____

   _____

   Soluciones que conoce:

   - ☐ Validación de datos ingresados.
   - ☐ Verificación de errores.
   - ☐ Uso de librerías.
   - ☐ Uso de procedimientos almacenados.
   - ☐ Otros (Notas adicionales)

   _____

   _____

3. **¿Qué información tiene Ud. del riesgo o vulnerabilidad de Perdida de Autenticación y Gestión de Sesiones y qué soluciones conoce para prevenirlo?**

   Conocimientos:

   - ☐ Domina el tema
   - ☐ Conoce el tema
   - ☐ Conoce poco
   - ☐ Tiene referencias
   - ☐ Desconoce el tema
   - ☐ Otros (Notas adicionales)

   _____

   _____

   Soluciones que conoce:

   - ☐ Protección de ataques de fuerza bruta.
   - ☐ Validación de contraseñas comúnmente elegidos.
   - ☐ Cifrado de las credenciales.
   - ☐ Caducidad de las sesiones.
   - ☐ No se permiten usuarios concurrentes.
   - ☐ Otros (Notas adicionales)

   _____

   _____

4. **¿Qué información tiene Ud. del riesgo o vulnerabilidad de Secuencia de Comandos en Sitios Cruzados (XSS) y qué soluciones conoce para prevenirlo?**

   Conocimientos:

   - ☐ Domina el tema
   - ☐ Conoce el tema

- ☐ Conoce poco
- ☐ Tiene referencias
- ☐ Desconoce el tema
- ☐ Otros (Notas adicionales)

Soluciones que conoce:

- ☐ Comprobación y uso de controles de seguridad que impiden XSS.
- ☐ Uso de librerías que impiden XSS.
- ☐ Validación datos ingresados.
- ☐ Otros (Notas adicionales)

---

5. **¿Qué información tiene Ud. del riesgo o vulnerabilidad de Configuración de Seguridad Incorrecta y qué soluciones conoce para prevenirlo?**

Conocimientos:

- ☐ Domina el tema
- ☐ Conoce el tema
- ☐ Conoce poco
- ☐ Tiene referencias
- ☐ Desconoce el tema
- ☐ Otros (Notas adicionales)

Soluciones que conoce:

- ☐ Uso de guías de hardening.
- ☐ Administración de puertos, servicios, etc.
- ☐ Eliminación de cuentas por defecto.
- ☐ Conf. de seguridad en el framework.
- ☐ Otros (Notas adicionales)

---

6. **¿Qué información tiene Ud. del riesgo o vulnerabilidad de Exposición de Datos Sensibles qué soluciones conoce para prevenirlo?**

Conocimientos:

- ☐ Domina el tema
- ☐ Conoce el tema
- ☐ Conoce poco
- ☐ Tiene referencias
- ☐ Desconoce el tema
- ☐ Otros (Notas adicionales)

Soluciones que conoce:

- ☐ Módulos criptográficos certificados.
- ☐ Uso correcto de módulos criptográficos.
- ☐ Cifrado de datos confidenciales.
- ☐ CA de confianza.
- ☐ Uso de TLS.
- ☐ Otros (Notas adicionales)

---

7. **¿Qué información tiene Ud. del riesgo o vulnerabilidad de Falsificación de Petición en Sitios Cruzados (CSRF) y qué soluciones conoce para prevenirlo?**

Conocimientos:

- ☐ Domina el tema
- ☐ Conoce el tema
- ☐ Conoce poco
- ☐ Tiene referencias
- ☐ Desconoce el tema
- ☐ Otros (Notas adicionales)

Soluciones que conoce:

- ☐ Uso de librerías o framework
- ☐ Reautenticación de usuarios
- ☐ Uso de CAPTCHA
- ☐ Otros (Notas adicionales)

---

8. **¿Qué información tiene Ud. del riesgo o vulnerabilidad de Uso de Componentes con Vulnerabilidades Conocidas y qué soluciones conoce para prevenirlo?**

Conocimientos:

- ☐ Domina el tema
- ☐ Conoce el tema
- ☐ Conoce poco
- ☐ Tiene referencias
- ☐ Desconoce el tema
- ☐ Otros (Notas adicionales)

Soluciones que conoce:

- ☐ Software actualizado
- ☐ Revisa, actualiza los componentes de la WepApps
- ☐ Uso de librerías
- ☐ Se revisa la seguridad en las listas de correos (securityfocus, bugtraq, etc.)
- ☐ Otros (Notas adicionales)

3.5.2.4. Técnicas, instrumentos y fuentes utilizados por cada problema y objetivo de la investigación

Cuadro 3.8: Técnicas, instrumentos y fuentes para estudiar los problemas de investigación

Problemas	Objetivos	Técnicas, instrumentos y fuentes
(a) ¿El riesgo de Inyección, está contemplado en el desarrollo de aplicaciones web?	(a) Identificar si el riesgo de inyección está contemplado en el desarrollo de aplicaciones web.	Cuestionario (preguntas: 9, 11 y 12). Guía de entrevista (preguntas: 1,2). y la guía de revisión documental.
(b) ¿El riesgo de Pérdida de Autenticación y Gestión de Sesiones, está contemplado en el desarrollo de aplicaciones web?	(b) Identificar si el riesgo de pérdida de autenticación y gestión de sesiones está contemplado en el desarrollo de aplicaciones web.	Cuestionario (preguntas: 13, 14, 15, 16, 17 y 18). Guía de entrevista (preguntas: 1,3). y la guía de revisión documental.
(c) ¿El riesgo de Secuencia de Comandos en Sitios Cruzados (XSS), está considerado en el desarrollo de aplicaciones web?	(c) Identificar si el riesgo de secuencia de comandos en sitios cruzados (XSS) está considerado en el desarrollo de aplicaciones web.	Cuestionario (preguntas: 10, 19, 20 y 21). Guía de entrevista (preguntas: 1,4). y la guía de revisión documental.
(d) ¿El riesgo de Configuración de Seguridad Incorrecta, está considerado en el desarrollo de aplicaciones web?	(d) Identificar si el riesgo de Configuración de Seguridad Incorrecta está considerado en el desarrollo de aplicaciones web.	Cuestionario (preguntas: 22 y 23). Guía de entrevista (preguntas: 1,5). y la guía de revisión documental.
(e) ¿El riesgo de Exposición de Datos Sensibles, está contemplado en el desarrollo de aplicaciones web?	(e) Identificar si el riesgo de Exposición de Datos Sensibles, está contemplado en el desarrollo de aplicaciones web.	Cuestionario (preguntas: 24, 25 y 26). Guía de entrevista (preguntas: 1,6). y la guía de revisión documental.
(f) ¿El riesgo de Falsificación de Peticiones en Sitios Cruzados (CSRF), está contemplado en el desarrollo de aplicaciones web?	(f) Identificar si el riesgo de Falsificación de Peticiones en Sitios Cruzados (CSRF) está contemplado en el desarrollo de aplicaciones web.	Cuestionario (preguntas: 27, 28 y 29). Guía de entrevista (preguntas: 1,7). y la guía de revisión documental.

Continua en la siguiente página

Cuadro 3.8 – *Continuación de la página anterior*

Problemas	Objetivos	Técnicas, instrumentos y fuentes
(g) ¿El riesgo de Uso de Componentes con Vulnerabilidades Conocidas, está considerado en el desarrollo de aplicaciones web?	(g) Identificar si el riesgo de Uso de Componentes con Vulnerabilidades Conocidas está considerado en el desarrollo de aplicaciones web.	Cuestionario (preguntas: 30, 31 y 32). Guía de entrevista (preguntas: 1,8). y la guía de revisión documental.

3.6. Fiabilidad y Validez de los Instrumentos

Los instrumentos de medición se han elaborado siguiendo las siguientes etapas:

1. Definición del constructo o concepto que se medirá.
2. Definir el propósito y alcance del instrumento.
3. Elaborar la composición de los ítems.
4. Definir y ordenar cada ítem.
5. Codificar las respuestas.
6. Establecer una puntuación para los ítems.
7. Iniciar la evaluación de calidad del instrumento.
8. Hacer una prueba piloto.
9. Mejorar la prueba sobre la base del estudio piloto.
10. Aplicación final.

Todo instrumento se construye para medir o registrar una variable o conjunto de variables a través de un número de preguntas, afirmaciones o indicadores (llamados "ítems"). En la práctica es casi imposible que una medición sea perfecta, generalmente se tiene un grado de error. Desde luego, se trata que este error sea el mínimo posible y para ello hay formas de calcular la fiabilidad y la validez [Horna, 2012].

3.6.1. Fiabilidad

La fiabilidad se relaciona con la precisión y congruencia, es el grado en que la aplicación repetida de un instrumento al mismo sujeto, objeto o situación, produce iguales resultados. Además, es la capacidad del instrumento de producir resultados

congruentes (iguales), cuando se aplica por segunda o tercera vez, en condiciones tan parecidas como sea posible |Horna, 2012|.

En la investigación el tipo de fiabilidad del instrumento fue de fidelidad de las fuentes y publicidad en el registro.

3.6.2. Validez

La validez, por su parte, es el grado en que un instrumento realmente mide la variable que pretende medir. La validez se refiere al grado de evidencia acumulada sobre qué mide el instrumento, justifica la particular interpretación que se va a hacer del instrumento |Horna, 2012|.

Hay tres tipos de validez, que son enfoques complementarios: de contenido, de constructo y de criterio.

En la investigación el tipo de validez del instrumento fue de contenido (criterio de jueces o de expertos).

3.7. Formas de Tratamiento de los Datos

Los datos obtenidos mediante la aplicación de los instrumentos antes mencionados, fueron incorporados a un sistema de composición de textos, orientado especialmente a la creación de libros, documentos científicos y técnicos LaTeX 2_ε(kile) y SPSS, y con precisiones porcentuales y relaciones u ordenamientos, los promedios son presentados como informaciones, en forma de gráficos, cuadros o resúmenes.

Capítulo 4

Análisis y Resultados de la Investigación

Se adjunta el Anexo B con los cuestionarios de las instituciones públicas, Anexo C con los cuestionarios de las instituciones privadas y el Anexo D con las guías de la entrevista.

4.1. Análisis y Tratamiento de Datos

4.1.1. Elaboración de tablas de frecuencia

Para elaborar las tablas de frecuencia se requiere hacer dos operaciones: la clasificación y la tabulación. La clasificación (consiste en determinar las categorías, los distintos valores que toman las variables o los intervalos de clase) y la tabulación (consiste en contabilizar cuantas veces se repite cada uno de los distintos valores o categorías de las variables).

Para elaborar las tablas de frecuencia se usó el programa estadístico SPSS v22.0, este software una vez codificado las preguntas y se hayan ingresado todas las respuestas nos ayuda a generar las tablas de frecuencia y los estadísticos como tendencia central (media, mediana, moda, suma), dispersion (desviación estándar, varianza, rango, mínimo, máximo, error estándar media), valores percentiles (cuartiles, percentiles) y de distribución (asimetría y curtosis).

4.1.1.1. Codificación de preguntas en SPSS

Mediante el software SPSS se hizo la codificación de las preguntas del cuestionario y de la guía de entrevista.

Codificación de las preguntas del cuestionario

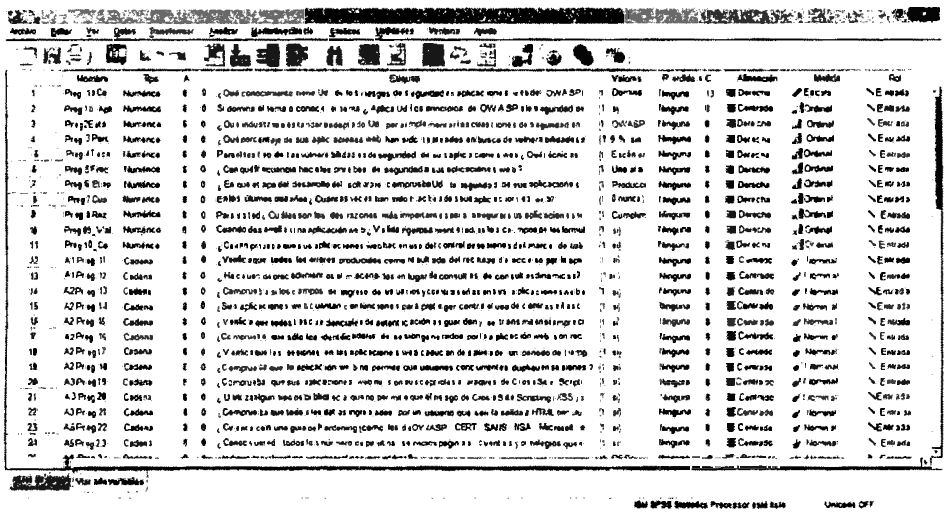


Figura 4.1: Codificación de preguntas del cuestionario

Codificación de las preguntas de la guía de entrevista

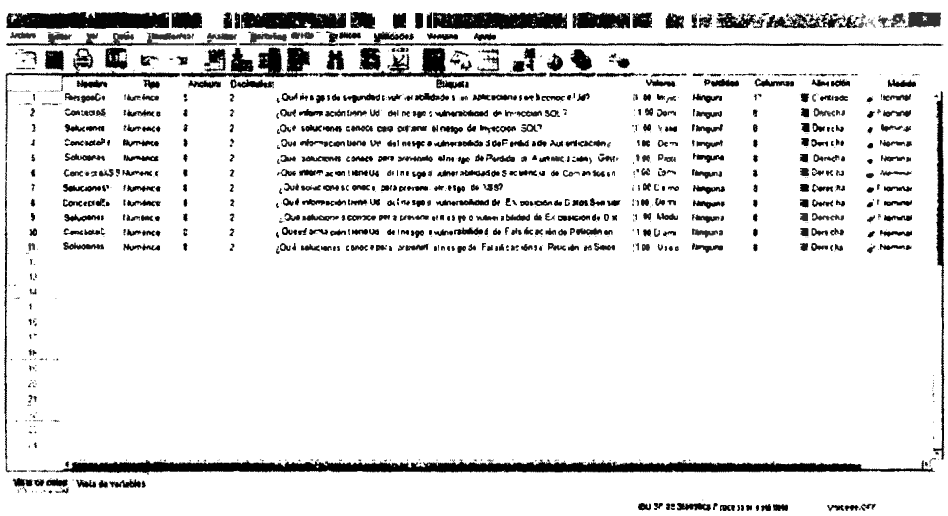


Figura 4.2: Codificación de preguntas de la guía de entrevista

4.1.1.2. Vaciado de respuestas en SPSS

Ingreso de las respuestas obtenidas del cuestionario y la guía de entrevista en el programa estadístico SPSS.

Vaciado de respuestas del cuestionario

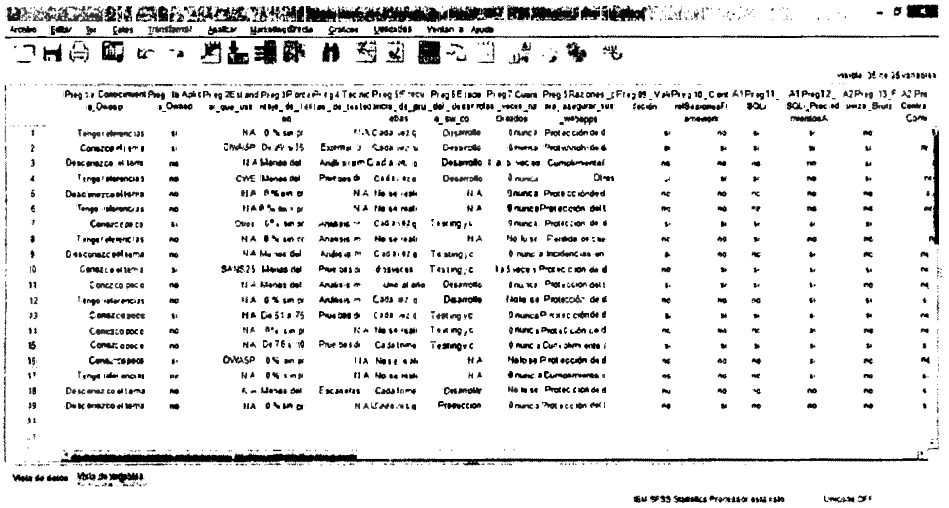


Figura 4.3: Vaciado de respuestas de los cuestionarios

Vaciado de respuestas de las guías de entrevista

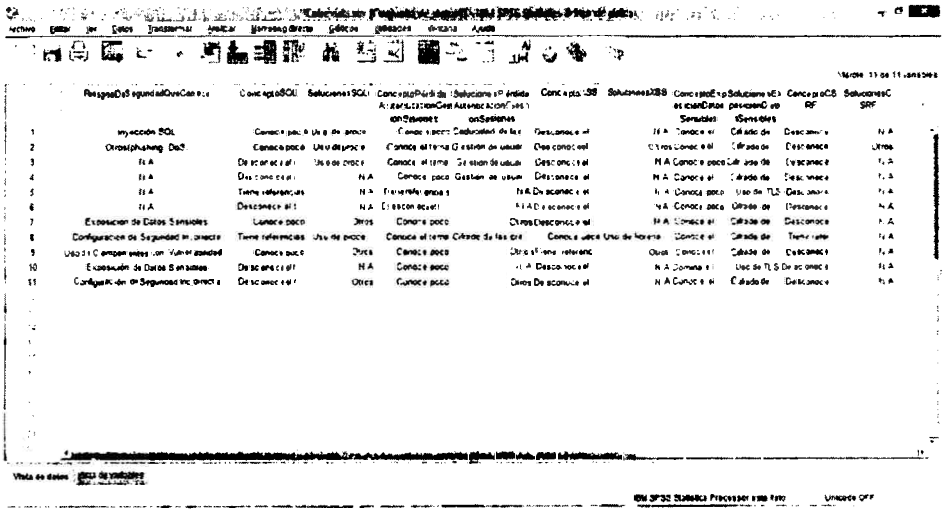


Figura 4.4: Vaciado de respuestas de las guías de entrevista

4.1.2. Clasificación de las preguntas del cuestionario y de la guía de entrevista

4.1.2.1. Clasificación de las preguntas del cuestionario

Preguntas generales

Cuadro 4.1: Clasificación de las preguntas generales del cuestionario

Objetivo	Preguntas generales
Obtener información relacionada sobre los riesgos de seguridad en aplicaciones web de las organizaciones más representativas de la ciudad de Ayacucho.	**Preg1a.** ¿Qué conocimiento tiene Ud. de los riesgos de seguridad en aplicaciones web del OWASP top 10? **Preg1b.** Si domina el tema o conoce el tema ¿Aplica Ud. los principios de OWASP a la seguridad de sus aplicaciones web? **Preg2.** ¿Qué industria o estándar a adoptado Ud. para implementar las cuestiones de seguridad en sus aplicaciones web? **Preg3.** ¿Qué porcentaje de sus aplicaciones web han sido testeados en busca de vulnerabilidades de seguridad? **Preg4.** Para el testeo de las vulnerabilidades de seguridad de sus aplicaciones web ¿Qué técnicas utiliza usted? **Preg5.** ¿Con qué frecuencia hace las pruebas de seguridad a sus aplicaciones web? **Preg6.** ¿En qué etapa del desarrollo del software comprueba Ud. la seguridad de sus aplicaciones web? **Preg7.** En los últimos dos años ¿Cuántas veces han sido hackeados sus aplicaciones web? **Preg8.** Para usted ¿Cuáles son las dos razones más importantes para asegurar sus aplicaciones web?

Preguntas específicas

Cuadro 4.2: Clasificación de las preguntas específicas del cuestionario

Objetivos	Preguntas específicas

Continua en la siguiente página

Cuadro 4.2 – *Continuación de la página anterior*

Objetivos	Preguntas específicas
(a) Identificar si el riesgo de inyección SQL está contemplado en el desarrollo de aplicaciones web.	**Preg9.** Cuando desarrolla una aplicación web ¿Valida rigurosamente todos los campos de los formularios (incluido los comandos SQL), campos escondidos, cabeceras, cookies y cadenas de petición? **Preg11.** ¿Verifica que todos los errores producidos como resultado del rechazo de acceso por la aplicación web están correctamente manejados por la aplicación? **Preg12.** ¿Hace uso de procedimientos almacenados en lugar de consultas dinámicas?
(b) Identificar si el riesgo de pérdida de autenticación y gestión de sesiones está contemplado en el desarrollo de aplicaciones web.	**Identificación del riesgo de pérdida de autenticación** **Preg13.** ¿Comprueba si los campos de ingreso de usuarios y contraseñas en sus aplicaciones web están protegidos contra un ataque de fuerza bruta? **Preg14.** ¿Sus aplicaciones web cuentan con funciones para proteger contra el uso de contraseñas comúnmente elegidos? **Preg15.** ¿Verifica que todos las credenciales de autenticación se guardan y se transmiten siempre cifradas con hashes no reversibles y no así en el código fuente? **Identificación del riesgo de gestión de sesiones** **Preg16.** ¿Comprueba que sólo los identificadores de sesión generados por la aplicación web son reconocidos como válidos? **Preg17.** ¿Verifica que las sesiones en las aplicaciones web caducan después de un período de tiempo de inactividad? **Preg18.** ¿Comprueba que la aplicación web no permite que usuarios concurrentes dupliquen sesiones?

Continua en la siguiente página

Cuadro 4.2 – *Continuación de la página anterior*

Objetivos	Preguntas específicas
(f) Identificar si el riesgo de Falsificación de Peticiones en Sitios Cruzados (CSR_F) está contemplado en el desarrollo de aplicaciones web.	**Preg27.** ¿Utiliza algún tipo de librería que proporcione construcciones que hacen más fácil de evitar el riesgo de CSRF (Falsificación de Peticiones en Sitios Cruzados) ? **Preg28.** ¿Implementa usted funciones de reautenticación de usuarios para realizar operaciones críticas? **Preg29.** ¿Implementa Ud. mecanismos como los CAPTCHA para comprobar que las solicitudes provenientes son de un usuario real?
(g) Identificar si el riesgo de Uso de Componentes con Vulnerabilidades Conocidas está considerado en el desarrollo de aplicaciones web.	**Preg30.** ¿Indique los nombres y las versiones de los componentes de sus aplicaciones web? **Preg31.** ¿Revisa, actualiza y parcha constantemente los componentes de la aplicación (servidor web, servidor de aplicación, etc) y actualiza las guías? **Preg32.** ¿Revisa la seguridad de los componentes de sus aplicaciones web en listas de correo de seguridad tales como: securityfocus, bugtraq, CERT, etc. y los mantiene actualizados?

4.1.2.2. Clasificación de las preguntas de la guía de entrevista

Cuadro 4.3: Clasificación de las preguntas de la guía de entrevista

Objetivo	Preguntas de la guía de entrevista

Continua en la siguiente página

Cuadro 4.3 – *Continuación de la página anterior*

Objetivo	Preguntas de la guía de entrevista
Obtener o recabar información sobre los riesgos de seguridad en aplicaciones web, así mismo para corroborar las respuestas obtenidas en el cuestionario.	**Preg1.** ¿Qué riesgos de seguridad o vulnerabilidades en aplicaciones web conoce Ud.? **Preg2.** ¿Qué información tiene Ud. del riesgo o vulnerabilidad de Inyección SQL y qué soluciones conoce para prevenirlo? **Preg3.** ¿Qué información tiene Ud. del riesgo o vulnerabilidad de Perdida de Autenticación y Gestión de Sesiones y qué soluciones conoce para prevenirlo? **Preg4.** ¿Qué información tiene Ud. del riesgo o vulnerabilidad de Secuencia de Comandos en Sitios Cruzados (XSS) y qué soluciones conoce para prevenirlo? **Preg5.** ¿Qué información tiene Ud. del riesgo o vulnerabilidad de Configuración de Seguridad Incorrecta y qué soluciones conoce para prevenirlo? **Preg6.** ¿Qué información tiene Ud. del riesgo o vulnerabilidad de Exposición de Datos Sensibles qué soluciones conoce para prevenirlo? **Preg7.** ¿Qué información tiene Ud. del riesgo o vulnerabilidad de Falsificación de Petición en Sitios Cruzados (CSRF) y qué soluciones conoce para prevenirlo? **Preg8.** ¿Qué información tiene Ud. del riesgo o vulnerabilidad de Uso de Componentes con Vulnerabilidades Conocidas y qué soluciones conoce para prevenirlo?

4.2. Presentación de Resultados

4.2.1. Resultado del cuestionario

4.2.1.1. Resultado de las preguntas generales

Preg.1a: ¿Qué conocimiento tiene Ud. de los riesgos de seguridad en aplicaciones web del OWASP top 10?

Interpretación: Del cuadro 4.4 se observa que el 10.5 % de los encuestados conocen el tema, de igual forma el 31.6 % conocen poco, el 31.6 % tiene referencias y el 26.3 % desconoce el tema.

Cuadro 4.4: Conocimiento del OWASP top 10 (Fuente: elaboración propia basado en encuestas, 2014)

	Frecuencia	Porcentaje	Porcentaje válido	Porcentaje acumulado
Conozco el tema	2	10.5	10.5	10.5
Conozco poco	6	31.6	31.6	42.1
Tengo referencias	6	31.6	31.6	73.7
Desconozco el tema	5	26.3	26.3	100.0
Total	19	100.0	100.0	

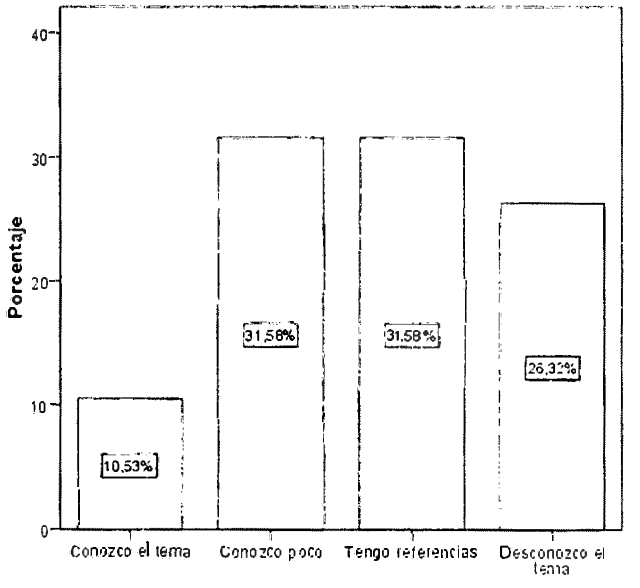


Figura 4.5: Conocimiento del OWASP top 10 (Fuente: elaboración propia basado en encuestas, 2014)

Preg.1b: Si domina el tema o conoce el tema ¿Aplica Ud. los principios de OWASP a la seguridad de sus aplicaciones web?

Interpretación: Del cuadro 4.5 se observa que el 31.6 % de los encuestados aplican los principios de OWASP a la seguridad de sus aplicaciones web, mientras que el 68.4 % no lo hace.

Cuadro 4.5: Aplicación de los principios de OWASP (Fuente: elaboración propia basado en encuestas, 2014)

	Frecuencia	Porcentaje	Porcentaje válido	Porcentaje acumulado
si	6	31.6	31.6	31.6
no	13	68.4	68.4	100.0
Total	19	100.0	100.0	

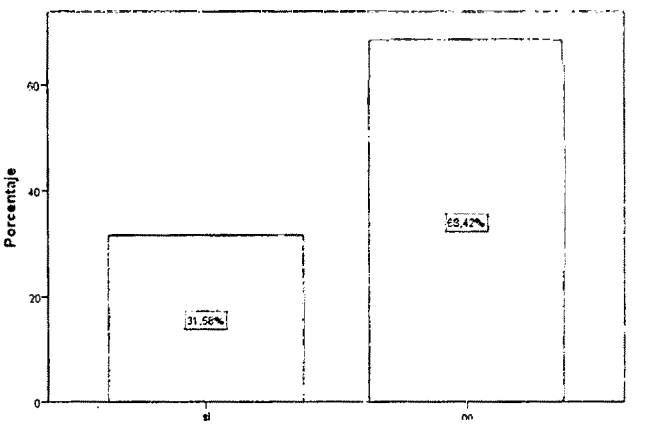


Figura 4.6: Aplicación de los principios de OWASP (Fuente: elaboración propia basado en encuestas, 2014)

Preg.2: ¿Qué industria o estándar a adoptado Ud. para implementar las cuestiones de seguridad en sus aplicaciones web?

Interpretación: Del cuadro 4.6 se observa que el 10.5 % de los encuestados usa OWASP para implementar las cuestiones de seguridad en aplicaciones web. Así mismo, el 5.3 % utiliza SANS, el 5.3 % utiliza CWE, el 73.7 % N.A y el 5.3 % otros.

Cuadro 4.6: Estándar adoptado para la seguridad de las aplicaciones web (Fuente: elaboración propia basado en encuestas, 2014)

	Frecuencia	Porcentaje	Porcentaje válido	Porcentaje acumulado
OWASP	2	10.5	10.5	10.5
SANS 25	1	5.3	5.3	15.8
CWE	1	5.3	5.3	21.1
N.A	14	73.7	73.7	94.7
Otros	1	5.3	5.3	100.0
Total	19	100.0	100.0	

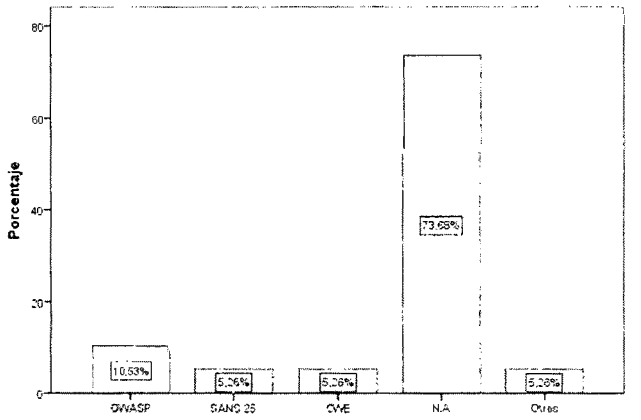


Figura 4.7: Estándar adoptado para la seguridad de las aplicaciones web (Fuente: elaboración propia basado en encuestas, 2014)

Preg.3: ¿Qué porcentaje de sus aplicaciones web han sido testeados en busca de vulnerabilidades de seguridad?

Interpretación: Del cuadro 4.7 se observa que el 52.6 % de los encuestados no han realizado pruebas (0 %). Así mismo, el 15.8 % lo hicieron menos del 25 % de sus aplicaciones web, el 15.8 % menos del 50 % de sus aplicaciones web, el 10.5 % del 51 a 75 % de sus aplicaciones web y el 5.3 % de 76 a 100 % de sus aplicaciones web.

Cuadro 4.7: Porcentaje de WebApps testeados en busca de vulnerabilidades (Fuente: elaboración propia basado en encuestas, 2014)

	Frecuencia	Porcentaje	Porcentaje válido	Porcentaje acumulado
0% sin pruebas	10	52.6	52.6	52.6
Menos del 25 %	3	15.8	15.8	68.4
Menos del 50 %	3	15.8	15.8	84.2
De 51 a 75%	2	10.5	10.5	94.7
De 76 a 100%	1	5.3	5.3	100.0
Total	19	100.0	100.0	

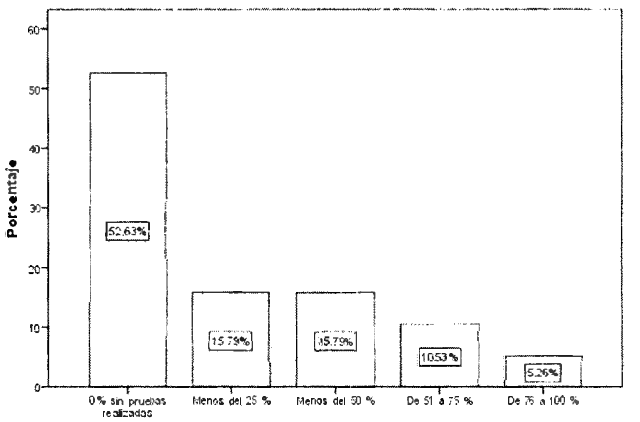


Figura 4.8: Porcentaje de WebApps testeados en busca de vulnerabilidades (Fuente: elaboración propia basado en encuestas, 2014)

Preg.4: Para el testeo de las vulnerabilidades de seguridad de sus aplicaciones web ¿Qué técnicas utiliza usted?

Interpretación: Del cuadro 4.8 se observa que el 5.3 % de los encuestados utilizan escáneres de vulnerabilidad. Así mismo, el 5.3 % utiliza external pen testing, el 21.1 % utiliza pruebas dinámicas automatizadas , el 31.6 % análisis manual y el 36.8 % N.A.

Cuadro 4.8: Técnicas utilizados para el testeo de WebApps (Fuente: elaboración propia basado en encuestas, 2014)

	Frecuencia	Porcentaje	Porcentaje válido	Porcentaje acumulado
Escáneres de vulnerabilidad	1	5.3	5.3	5.3
External pen testing	1	5.3	5.3	10.5
Pruebas dinámicas automatizadas	4	21.1	21.1	31.6
Análisis manual	6	31.6	31.6	63.2
N.A	7	36.8	36.8	100.0
Total	19	100.0	100.0	

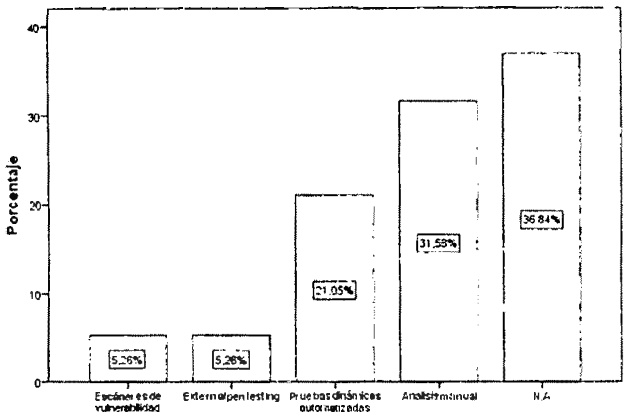


Figura 4.9: Técnicas utilizados para el testeo de WebApps (Fuente: elaboración propia basado en encuestas, 2014)

Preg.5: ¿Con qué frecuencia hace las pruebas de seguridad a sus aplicaciones web?

Interpretación: Del cuadro 4.9 se observa que el 5.3 % de los encuestados hace las pruebas una ves al año. Así mismo el 5.3 % lo hacen cada semestre, el 10.5 % cada trimestre, el 47.4 % cada vez que cambia el código y el 31.6 % no realizan pruebas.

Cuadro 4.9: Frecuencia de las pruebas de seguridad en WebApps (Fuente: elaboración propia basado en encuestas, 2014)

	Frecuencia	Porcentaje	Porcentaje válido	Porcentaje acumulado
Uno al año	1	5.3	5.3	5.3
Cada semestre	1	5.3	5.3	10.5
Cada trimestre	2	10.5	10.5	21.1
Cada vez que cambia el código	9	47.4	47.4	68.4
No se realizan pruebas	6	31.6	31.6	100.0
Total	19	100.0	100.0	

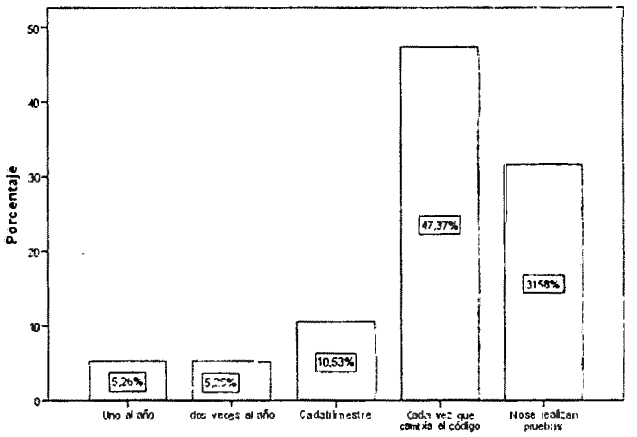


Figura 4.10: Frecuencia de las pruebas de seguridad en WebApps (Fuente: elaboración propia basado en encuestas, 2014)

Preg.8: Para usted ¿Cuáles son las dos razones más importantes para asegurar sus aplicaciones web?

Interpretación: Del cuadro 4.10 se observa que el 15.8 % de los encuestados lo hacen por cumplimiento de PCI, HIPAA, o políticas de seguridadí mismo el 52.6 % lo hacen para la protección de datos, el 5.3 % incidencias en el negocio, el 5.3 % pérdida de clientes, el 15.8 % protección del trabajo y el 5.3 % otros.

Cuadro 4.10: Razones importantes para asegurar las aplicaciones web (Fuente: elaboración propia basado en encuestas, 2014)

	Frecuencia	Porcentaje	Porcentaje válido	Porcentaje acumulado
Cumplimiento de PCI, HIPAA, etc.	3	15.8	15.8	15.8
Protección de datos	10	52.6	52.6	68.4
Incidencias en el negocio	1	5.3	5.3	73.7
Pérdida de clientes	1	5.3	5.3	78.9
Protección del trabajo	3	15.8	15.8	94.7
Otros	1	5.3	5.3	100.0
Total	19	100.0	100.0	

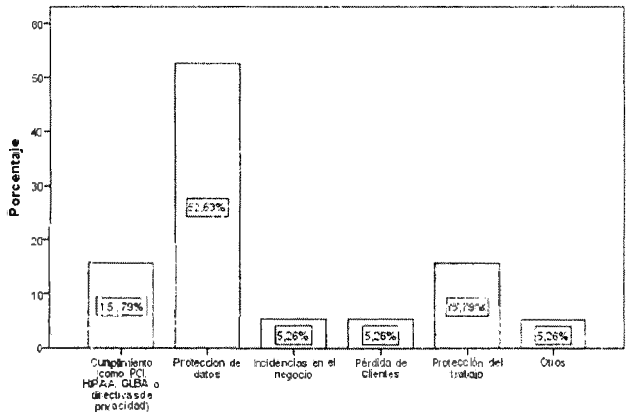


Figura 4.11: Razones importantes para asegurar las aplicaciones web (Fuente: elaboración propia basado en encuestas, 2014)

4.2.1.2. Resultados de las preguntas específicas

Identificación del riesgo de inyección SQL

Preg.9: Cuando desarrolla una aplicación web ¿Valida rigurosamente todas los campos de los formularios (incluido los comandos SQL y código javascript), campos escondidos, cabeceras, cookies y cadenas de petición?

Interpretación: Del cuadro 4.11 se observa que en el 57.9 % de los encuestados no validan rigurosamente todas los campos de los formularios, campos escondidos, cabeceras, cookies y cadenas de petición, mientras que en el 42.1 % si lo hace.

Cuadro 4.11: Validación de los formularios, campos escondidos, cabeceras, cookies y cadenas de petición (Fuente: elaboración propia basado en encuestas, 2014)

	Frecuencia	Porcentaje	Porcentaje válido	Porcentaje acumulado
si	8	42.1	42.1	42.1
no	11	57.9	57.9	100.0
Total	19	100.0	100.0	

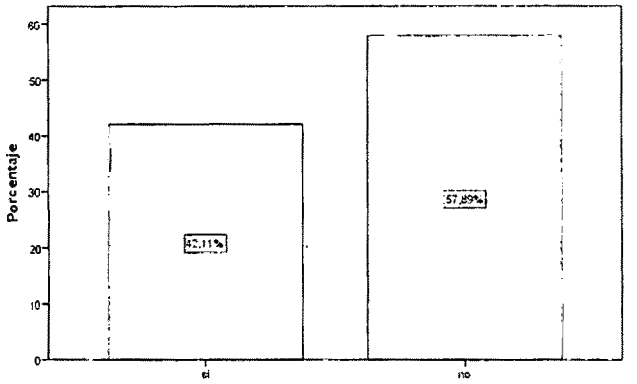


Figura 4.12: Validación de los formularios, campos escondidos, cabeceras, cookies y cadenas de petición (Fuente: elaboración propia basado en encuestas, 2014)

Preg.11: ¿Verifica que todos los errores producidos como resultado del rechazo de acceso por la aplicación web están correctamente manejados por la aplicación?

Interpretación: Del cuadro 4.12 se observa que en el 52.6% de las organizaciones no se verifican que todos los errores producidos como resultado del rechazo de acceso por la aplicación web, están correctamente manejados por la aplicación, mientras que el 47.4 % si lo hace.

Cuadro 4.12: Errores correctamente manejados por la aplicación (Fuente: elaboración propia basado en encuestas, 2014)

	Frecuencia	Porcentaje	Porcentaje válido	Porcentaje acumulado
si	9	47.4	47.4	47.4
no	10	52.6	52.6	100.0
Total	19	100.0	100.0	

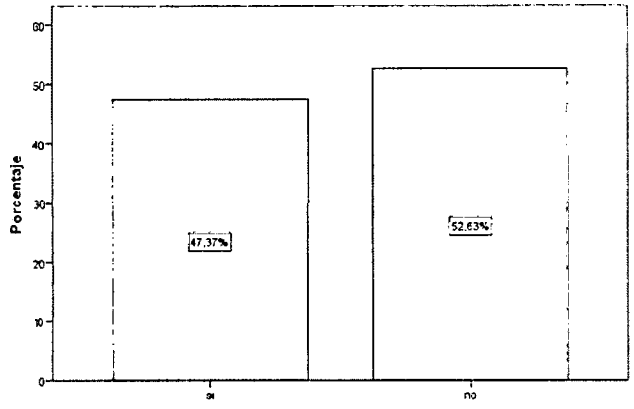


Figura 4.13: Errores correctamente manejados por la aplicación (Fuente: elaboración propia basado en encuestas, 2014)

Preg.12: ¿Hace uso de procedimientos almacenados en lugar de consultas dinámicas?

Interpretación: Del cuadro 4.13 se observa que en el 63.2 % de las organizaciones se usa procedimientos almacenados como mecanismo de solución frente al riesgo de inyección SQL, mientras que en el 36.8 % no usa procedimientos almacenados como mecanismo de solución.

Cuadro 4.13: Uso de procedimientos almacenados (Fuente: elaboración propia basado en encuestas, 2014)

	Frecuencia	Porcentaje	Porcentaje válido	Porcentaje acumulado
si	12	63.2	63.2	63.2
no	7	36.8	36.8	100.0
Total	19	100.0	100.0	

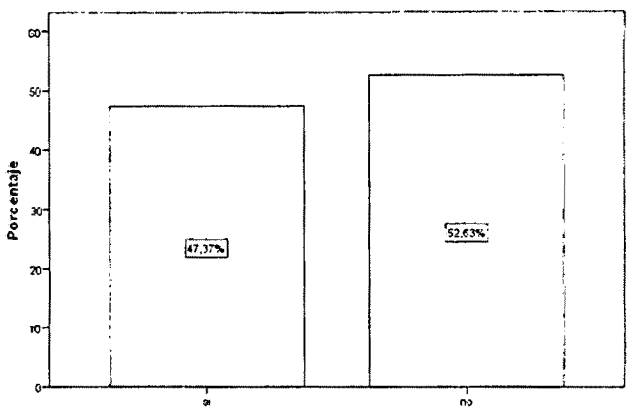


Figura 4.13: Errores correctamente manejados por la aplicación (Fuente: elaboración propia basado en encuestas, 2014)

Preg.12: ¿Hace uso de procedimientos almacenados en lugar de consultas dinámicas?

Interpretación: Del cuadro 4.13 se observa que en el 63.2 % de las organizaciones se usa procedimientos almacenados como mecanismo de solución frente al riesgo de inyección SQL, mientras que en el 36.8 % no usa procedimientos almacenados como mecanismo de solución.

Cuadro 4.13: Uso de procedimientos almacenados (Fuente: elaboración propia basado en encuestas, 2014)

	Frecuencia	Porcentaje	Porcentaje válido	Porcentaje acumulado
si	12	63.2	63.2	63.2
no	7	36.8	36.8	100.0
Total	19	100.0	100.0	

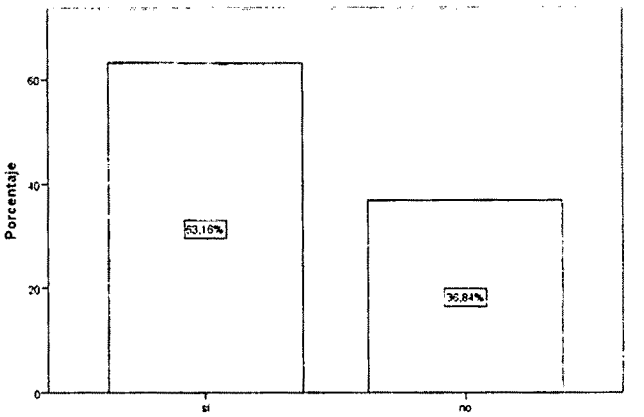


Figura 4.14: Uso de procedimientos almacenados (Fuente: elaboración propia basado en encuestas, 2014)

Conclusión: del análisis de las preguntas 9, 11 y 12 del cuestionario se concluye que el riesgo de inyección SQL está contemplado en un 63.2 % mediante el uso de procedimientos almacenados.

Identificación del riesgo de pérdida de autenticación y gestión de sesiones

Identificación del riesgo de pérdida de autenticación

Preg.13: ¿Comprueba si los campos de ingreso de usuarios y contraseñas en sus aplicaciones web están protegidos contra un ataque de fuerza bruta?

Interpretación: Del cuadro 4.14 se observa en el 57.9% de la organizaciones no se comprueba si los campos de ingreso de usuarios y contraseñas en las aplicaciones web están protegidos contra un ataque de fuerza bruta, mientras que en el 42.1 % si lo hacen.

Cuadro 4.14: Protección de ataques de fuerza bruta (Fuente: elaboración propia basado en encuestas, 2014)

	Frecuencia	Porcentaje	Porcentaje válido	Porcentaje acumulado
si	8	42.1	42.1	42.1
no	11	57.9	57.9	100.0
Total	19	100.0	100.0	

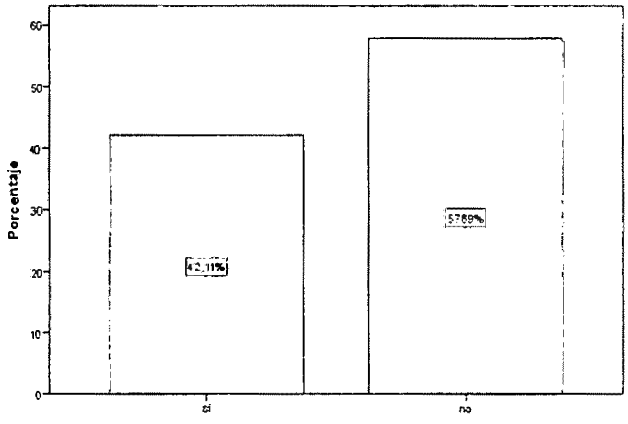


Figura 4.15: Protección de ataques de fuerza bruta (Fuente: elaboración propia basado en encuestas, 2014)

Preg.14: ¿Sus aplicaciones web cuentan con funciones para proteger contra el uso de contraseñas comúnmente elegidos?

Interpretación: Del cuadro 4.15 se observa que en el 52.6% de las organizaciones las aplicaciones web no cuentan con funciones para proteger contra el uso de contraseñas comúnmente elegidos, mientras que en el 47.4 % si cuenta con este mecanismo de protección.

Cuadro 4.15: Protección del uso de contraseñas comúnmente elegidos (Fuente: elaboración propia basado en encuestas, 2014)

	Frecuencia	Porcentaje	Porcentaje válido	Porcentaje acumulado
si	9	47.4	47.4	47.4
no	10	52.6	52.6	100.0
Total	19	100.0	100.0	

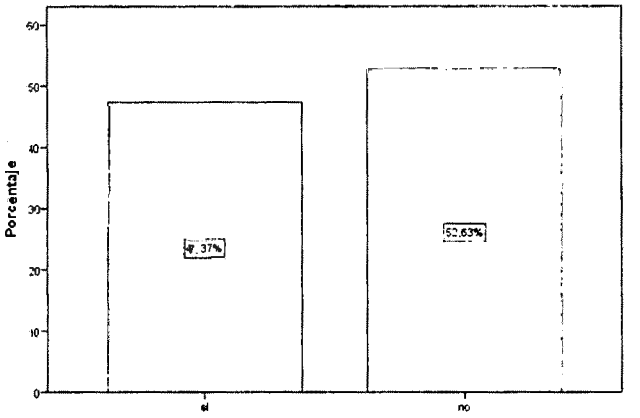


Figura 4.16: Protección del uso de contraseñas comúnmente elegidos (Fuente: elaboración propia basado en encuestas, 2014)

Preg.15: ¿Verifica que todos las credenciales de autenticación se guardan y se transmiten siempre cifradas con hashes no reversibles y no así en el código fuente?

Interpretación: Del cuadro 4.16 se observa que en el 52.6 % de las organizaciones no verifican que todos las credenciales de autenticación se guardan y se transmiten siempre cifradas con hashes no reversibles y no así en el código fuente, mientras que en el 47.4 % si lo hacen.

Cuadro 4.16: Cifrado de credenciales de autenticación (Fuente: elaboración propia basado en encuestas, 2014)

	Frecuencia	Porcentaje	Porcentaje válido	Porcentaje acumulado
si	9	47.4	47.4	47.4
no	10	52.6	52.6	100.0
Total	19	100.0	100.0	

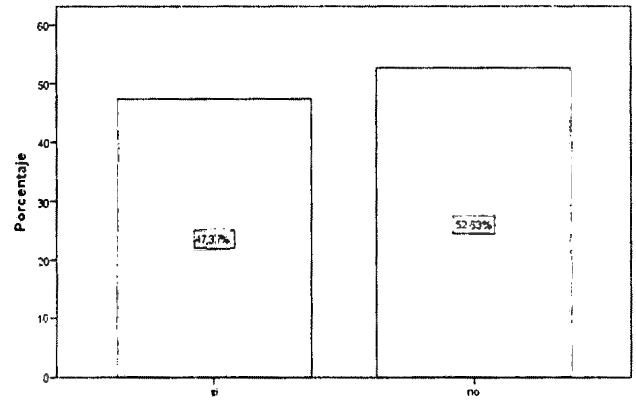


Figura 4.17: Cifrado de credenciales de autenticación (Fuente: elaboración propia basado en encuestas, 2014)

Conclusión: del análisis de las preguntas 13, 14 y 15 del cuestionario se concluye que el riesgo de pérdida de autenticación está contemplado en un 47.4 % mediante el cifrado de las credenciales de autenticación.

Identificación del riesgo de gestión de sesiones

Preg.16: ¿Comprueba que sólo los identificadores de sesión generados por la aplicación web son reconocidos como válidos?

Interpretación: Del cuadro 4.17 se observa que en el 57.9 % de las organizaciones no se comprueban que sólo los identificadores de sesión generados por la aplicación web son reconocidos como válidos, mientras que en el 42.1 % si lo hacen.

Cuadro 4.17: Comprobación de los identificadores de sesión (Fuente: elaboración propia basado en encuestas, 2014)

	Frecuencia	Porcentaje	Porcentaje válido	Porcentaje acumulado
si	8	42.1	42.1	42.1
no	11	57.9	57.9	100.0
Total	19	100.0	100.0	

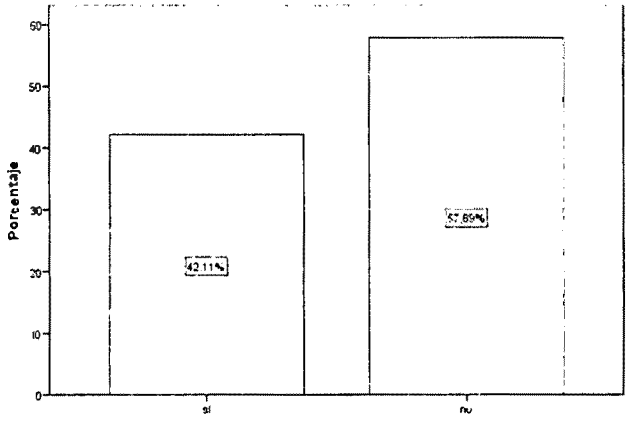


Figura 4.18: Comprobación de los identificadores de sesión (Fuente: elaboración propia basado en encuestas, 2014)

Preg.17: ¿Verifica que las sesiones en las aplicaciones web caducan después de un período de tiempo de inactividad?

Interpretación: Del cuadro 4.18 se observa que en el 68.4 % de las organizaciones verifican que las sesiones en las aplicaciones web caducan después de un período de tiempo de inactividad, mientras que en el 31.6 % no lo hacen.

Cuadro 4.18: Caducidad de sesiones (Fuente: elaboración propia basado en encuestas, 2014)

	Frecuencia	Porcentaje	Porcentaje válido	Porcentaje acumulado
si	13	68.4	68.4	68.4
no	6	31.6	31.6	100.0
Total	19	100.0	100.0	

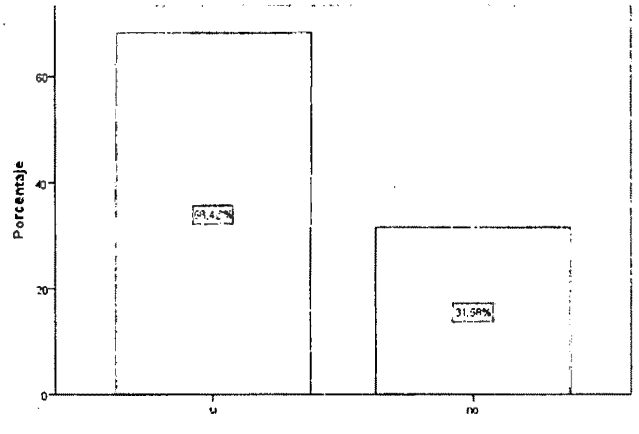


Figura 4.19: Caducidad de sesiones (Fuente: elaboración propia basado en encuestas, 2014)

Preg.18: ¿Comprueba que la aplicación web no permite que usuarios concurrentes dupliquen sesiones?

Interpretación: Del cuadro 4.19 se observa que en el 57.9 % de las organizaciones comprueban que la aplicación web no permite que usuarios concurrentes dupliquen sesiones, mientras que en el 42.1 % no lo hacen.

Cuadro 4.19: Gestión de usuarios concurrentes (Fuente: elaboración propia basado en encuestas, 2014)

	Frecuencia	Porcentaje	Porcentaje válido	Porcentaje acumulado
si	11	57.9	57.9	57.9
no	8	42.1	42.1	100.0
Total	19	100.0	100.0	

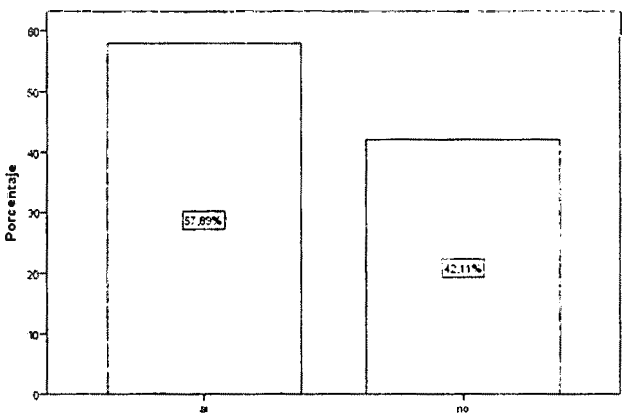


Figura 4.20: Gestión de usuarios concurrentes (Fuente: elaboración propia basado en encuestas, 2014)

Conclusión: del análisis de las preguntas 16, 17 y 18 del cuestionario se concluye que el riesgo de gestión de sesiones está contemplado en un 68.4 % mediante la verificación de caducidad de las sesiones después de un periodo de tiempo de inactividad.

Identificación del riesgo de secuencia de comandos en sitios cruzados (XSS)

Preg.19: ¿Comprueba que sus aplicaciones web no son susceptibles a ataques de Cross Site Scripting (XSS) o que los controles de seguridad impiden XSS?

Interpretación: Del cuadro 4.20 se observa que en el 84.2 % de las organizaciones no se comprueban que las aplicaciones web son susceptibles a ataques de Cross Site Scripting (XSS) o que los controles de seguridad impiden XSS, mientras que en el 15.8 % si lo hacen.

Cuadro 4.20: Comprobación de ataques de Cross Site Scripting (Fuente: elaboración propia basado en encuestas, 2014)

	Frecuencia	Porcentaje	Porcentaje válido	Porcentaje acumulado
si	3	15.8	15.8	15.8
no	16	84.2	84.2	100.0
Total	19	100.0	100.0	

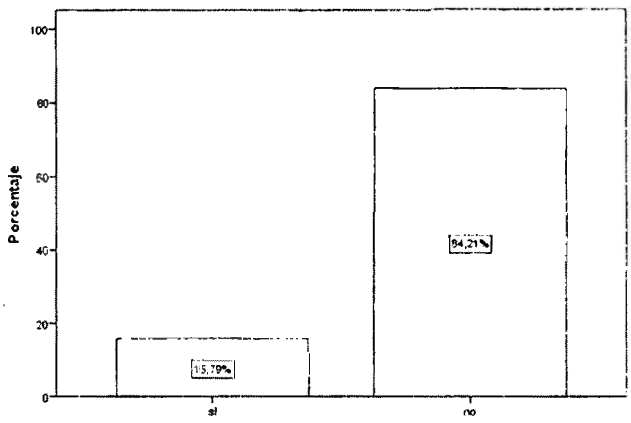


Figura 4.21: Comprobación de ataques de Cross Site Scripting (Fuente: elaboración propia basado en encuestas, 2014)

Preg.20: ¿Utiliza algún tipo de biblioteca que no permite que el riesgo de Cross Site Scripting (XSS) se produzca ?

Interpretación: Del cuadro 4.21 se observa que en todas las organizaciones no utilizan ningún tipo de biblioteca que no permite que el riesgo de Cross Site Scripting (XSS) se produzca.

Cuadro 4.21: Uso de bibliotecas contra el riesgo de Cross Site Scripting (Fuente: elaboración propia basado en encuestas, 2014)

	Frecuencia	Porcentaje	Porcentaje válido	Porcentaje acumulado
no	19	100.0	100.0	100.0

Preg.21: ¿Comprueba que todos los datos ingresados por un usuario que son la salida a HTML (incluyendo elementos HTML, atributos HTML, Java script, valores de datos, bloques de CSS y atributos URI) están debidamente validados en la aplicación web?

Interpretación: Del cuadro 4.22 se observa que en el 52.6 % de las organizaciones comprueban que todos los datos ingresados por un usuario que son la salida a HTML (incluyendo elementos HTML, atributos HTML, Java script, valores de datos, bloques de CSS y atributos URI) están debidamente validados en la aplicación web, mientras que en el 47.4 % no lo hacen.

Cuadro 4.22: Comprobación de datos ingresados que son la salida a HTML (Fuente: elaboración propia basado en encuestas, 2014)

	Frecuencia	Porcentaje	Porcentaje válido	Porcentaje acumulado
si	10	52.6	52.6	52.6
no	9	47.4	47.4	100.0
Total	19	100.0	100.0	

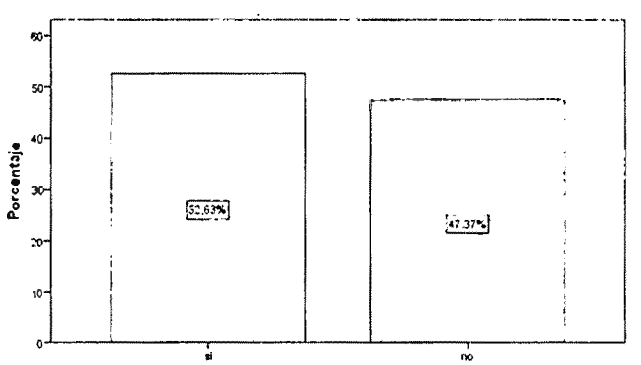


Figura 4.22: Comprobación de datos ingresados que son la salida a HTML(Fuente: elaboración propia basado en encuestas, 2014)

Conclusión: del análisis de las preguntas 19, 20 y 21 del cuestionario se concluye que el riesgo de secuencia de comandos en sitios cruzados (XSS) está contemplado en un 52.6 % mediante la comprobación de los datos ingresados que son la salida a HTML.

Identificación del riesgo de configuración de seguridad incorrecta

Preg.22: ¿Cuenta con una guía de hardening (como los de OWASP, CERT, SANS, NSA, Microsoft, etc. que ayudan a configurar correctamente los sistemas) para personalizar los componentes de sus aplicaciones web?

Interpretación: Del cuadro 4.23 se observa que en el 94.7 % de las organizaciones no cuentan con una guía de hardening (como los de OWASP, CERT, SANS, NSA, Microsoft, etc. que ayudan a configurar correctamente los sistemas) para personalizar los componentes de sus aplicaciones web, mientras que el 5.3 % si cuenta con estas guías.

Cuadro 4.23: Uso de guías de hardening (Fuente: elaboración propia basado en encuestas, 2014)

	Frecuencia	Porcentaje	Porcentaje válido	Porcentaje acumulado
si	1	5.3	5.3	5.3
no	18	94.7	94.7	100.0
Total	19	100.0	100.0	

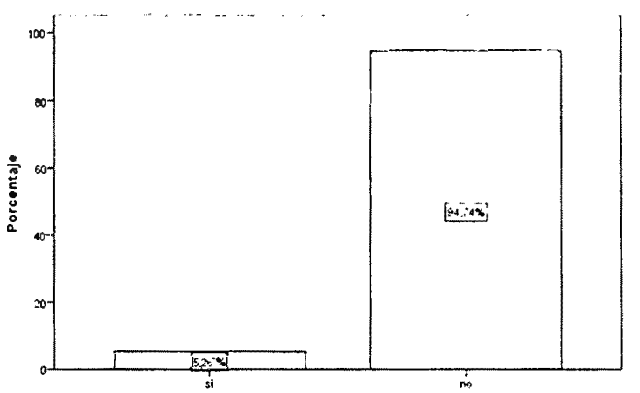


Figura 4.23: Uso de guías de hardening (Fuente: elaboración propia basado en encuestas, 2014)

Preg.23: ¿Conoce usted, todos los número de puertos, servicios, páginas, cuentas y privilegios que están configurados actualmente en su servidor web)?

Interpretación: Del cuadro 4.24 se observa que en el 68.4 % de las organizaciones conocen todos los número de puertos, servicios, páginas, cuentas y privilegios que están configurados actualmente en su servidor web, mientras que el 31.6 % no conocen.

Cuadro 4.24: Puertos, servicios, páginas y cuentas conocidas (Fuente: elaboración propia basado en encuestas, 2014)

	Frecuencia	Porcentaje	Porcentaje válido	Porcentaje acumulado
si	13	68.4	68.4	68.4
no	6	31.6	31.6	100.0
Total	19	100.0	100.0	

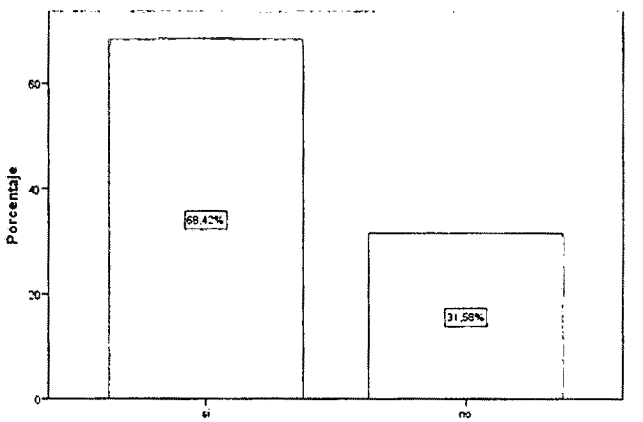


Figura 4.24: Puertos, servicios, páginas y cuentas conocidas (Fuente: elaboración propia basado en encuestas, 2014)

Conclusión: del análisis de las preguntas 22 y 23 del cuestionario se concluye que el riesgo de configuración de seguridad incorrecta está contemplado en un 68.4 % mediante la administración de los número de puertos, servicios, páginas, cuentas y privilegios.

Identificación del riesgo de exposición de datos sensibles

Preg.24: ¿Indique los algoritmos criptográficos que utiliza?

Interpretación: En la tabla 4.25 se observa que en el 21.1 % de las organizaciones se usa algoritmos criptográficos DES, en la actualidad considerado inseguro, además 21.1 % no usa ningún algoritmo de cifrado, y sólo el 15.8% usa AES conciderado actualmente el más seguro.

Cuadro 4.25: Algoritmos criptográficos utilizados (Fuente: elaboración propia basado en encuestas, 2014)

	Frecuencia	Porcentaje	Porcentaje válido	Porcentaje acumulado
DES	4	21.1	21.1	21.1
N.A	4	21.1	21.1	42.1
3DES	1	5.3	5.3	47.4
RC5	2	10.5	10.5	57.9
AES	3	15.8	15.8	73.7
RSA	4	21.1	21.1	94.7
Otros	1	5.3	5.3	100.0
Total	19	100.0	100.0	

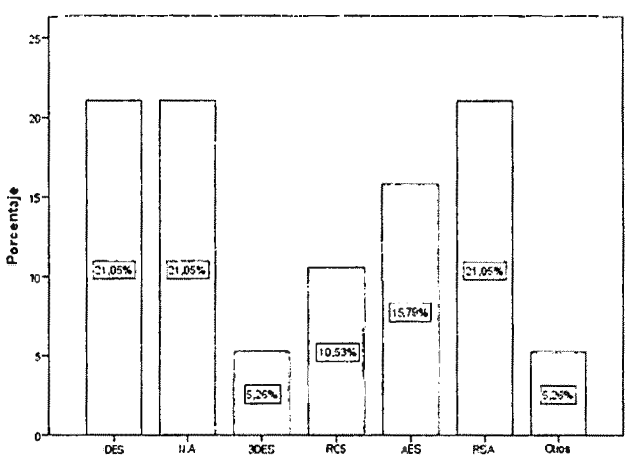


Figura 4.25: Algoritmos criptográficos utilizados (Fuente: elaboración propia basado en encuestas, 2014)

Preg.25: ¿Comprueba que los datos confidenciales procesados por la aplicación web son cifrados tanto en el almacenamiento como en la transmisión de datos?

Interpretación: Del cuadro 4.26 se observa que en el 57.9 % de las organizaciones no comprueban que los datos confidenciales procesados por la aplicación web son cifrados tanto en el almacenamiento como en la transmisión de datos, mientras que el 42.1 % si lo hace.

Cuadro 4.26: Cifrado de datos confidenciales (Fuente: elaboración propia basado en encuestas, 2014)

	Frecuencia	Porcentaje	Porcentaje válido	Porcentaje acumulado
si	8	42.1	42.1	42.1
no	11	57.9	57.9	100.0
Total	19	100.0	100.0	

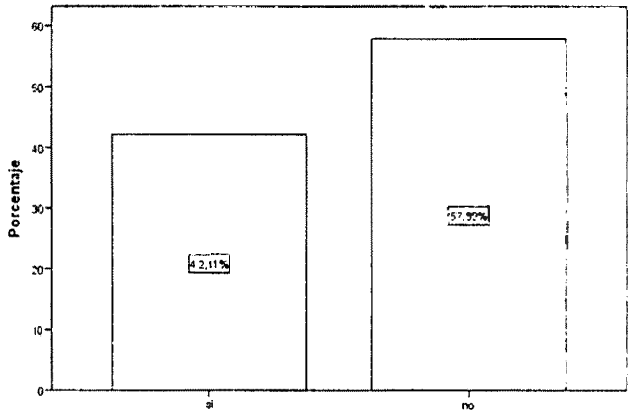


Figura 4.26: Cifrado de datos confidenciales (Fuente: elaboración propia basado en encuestas, 2014)

Preg.26: ¿Utiliza TLS (Seguridad de la Capa de Transporte) para todas las conexiones (incluyendo tanto internas como externas) en las que impliquen datos sensibles o funciones especiales?

Interpretación: Del cuadro 4.27 se observa En el 63.2 % de las organizaciones no utiliza TLS (Seguridad de la Capa de Transporte) para todas las conexiones (incluyendo tanto internas como externas) en las que impliquen datos sensibles o funciones especiales, mientras que en el 63.8 % si lo hacen.

Cuadro 4.27: Uso de TLS (Fuente: elaboración propia basado en encuestas, 2014)

	Frecuencia	Porcentaje	Porcentaje válido	Porcentaje acumulado
si	7	36.8	36.8	36.8
no	12	63.2	63.2	100.0
Total	19	100.0	100.0	

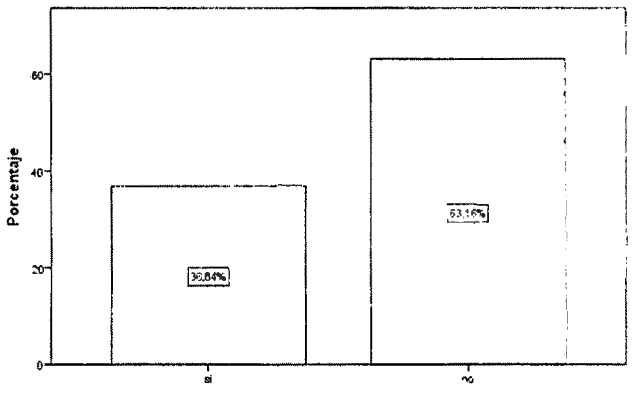


Figura 4.27: Uso de TLS (Fuente: elaboración propia basado en encuestas, 2014)

Conclusión: del análisis de las preguntas 24, 25 y 26 del cuestionario se concluye que el riesgo de exposición de datos sensibles está contemplado en un 15.8 % mediante el uso del algoritmo criptográfico AES.

Identificación del riesgo de falsificación de peticiones en sitios cruzados (CSRF)

Preg.27: ¿Utiliza algún tipo de librería que proporcione construcciones que hacen más fácil de evitar el riesgo de CSRF (Falsificación de Peticiones en Sitios Cruzados)?

Interpretación: Del cuadro 4.28 se observa que en el 89.5 % de las organizaciones no utilizan algún tipo de librería que proporcione construcciones que hacen más fácil de evitar el riesgo de CSRF (Falsificación de Peticiones en Sitios Cruzados), mientras que en el 10.5 % si utilizan.

Cuadro 4.28: Uso de librerías contra el riesgo de CSRF (Fuente: elaboración propia basado en encuestas, 2014)

	Frecuencia	Porcentaje	Porcentaje válido	Porcentaje acumulado
si	2	10.5	10.5	10.5
no	17	89.5	89.5	100.0
Total	19	100.0	100.0	

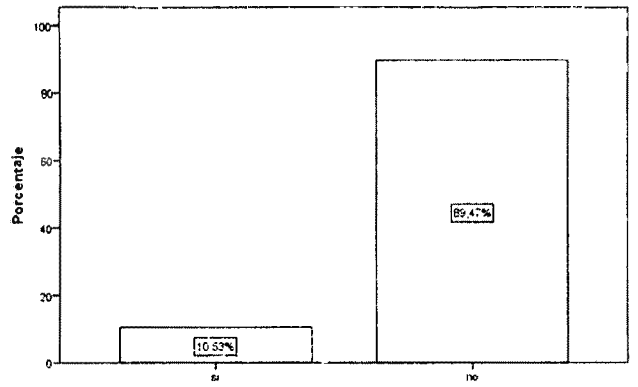


Figura 4.28: Uso de librerías contra el riesgo de CSRF (Fuente: elaboración propia basado en encuestas, 2014)

Preg.28: ¿Implementa usted funciones de reautenticación de usuarios para realizar operaciones críticas?

Interpretación: Del cuadro 4.29 se observa que en el 68.4 % de las organizaciones no implementan funciones de reautenticación de usuarios para realizar operaciones críticas, mientras que en el 31.6 % si lo hacen.

Cuadro 4.29: Implementación de funciones para la reautenticación de usuarios (Fuente: elaboración propia basado en encuestas, 2014)

	Frecuencia	Porcentaje	Porcentaje válido	Porcentaje acumulado
si	6	31.6	31.6	31.6
no	13	68.4	68.4	100.0
Total	19	100.0	100.0	

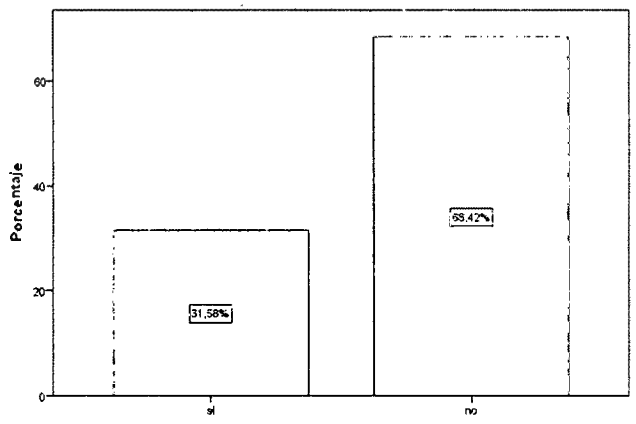


Figura 4.29: Implementación de funciones para la reautenticación de usuarios (Fuente: elaboración propia basado en encuestas, 2014)

Preg.29: ¿Implementa Ud. mecanismos como los captcha para comprobar que las solicitudes provenientes son de un usuario real?

Interpretación: Del cuadro 4.30 se observa que en el 52.6 % de las organizaciones implementan mecanismos como los captcha para comprobar que las solicitudes provenientes son de un usuario real, mientras que en el 47.4 % no lo hacen.

Cuadro 4.30: Uso de captcha (Fuente: elaboración propia basado en encuestas, 2014)

	Frecuencia	Porcentaje	Porcentaje válido	Porcentaje acumulado
si	10	52.6	52.6	52.6
no	9	47.4	47.4	100.0
Total	19	100.0	100.0	

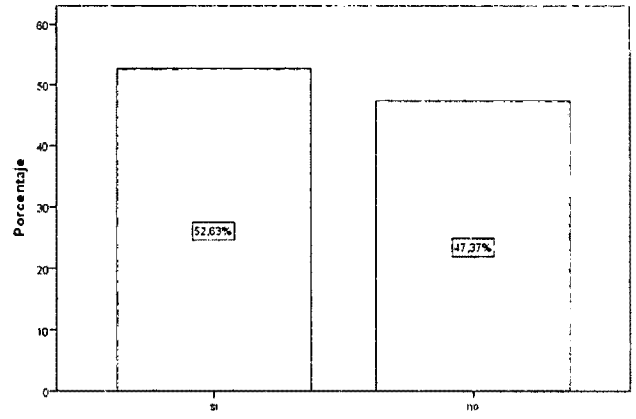


Figura 4.30: Uso de captcha (Fuente: elaboración propia basado en encuestas, 2014)

Conclusión: del análisis de las preguntas 27, 28 y 29 del cuestionario se concluye que el riesgo de falsificación de peticiones en sitios cruzados(CSRF) está contemplado en un 52.6 % mediante la implementación de mecanismos como los captcha.

Identificación del riesgo de uso de componentes con vulnerabilidades conocidas

Preg.30: ¿Indique los nombres y las versiones de los componentes de sus aplicaciones web?

Observación: No se obtuvo suficiente información para su tratamiento estadístico, razón por la cual se decide no analizar este item.

Preg.31: ¿Revisa, actualiza y parcha constantemente los componentes de la aplicación (servidor web, servidor de aplicación, etc) y actualiza las guías?

Interpretación: Del cuadro 4.31 se observa que en el 52.6 % de las organizaciones no revisan, actualizan y parchan constantemente los componentes de la aplicación (servidor web, servidor de aplicación, etc) y actualiza las guías, mientras que en el 47.4 % si lo hacen.

Cuadro 4.31: Actualización de componentes de aplicaciones web (Fuente: elaboración propia basado en encuestas, 2014)

	Frecuencia	Porcentaje	Porcentaje válido	Porcentaje acumulado
si	9	47.4	47.4	47.4
no	10	52.6	52.6	100.0
Total	19	100.0	100.0	

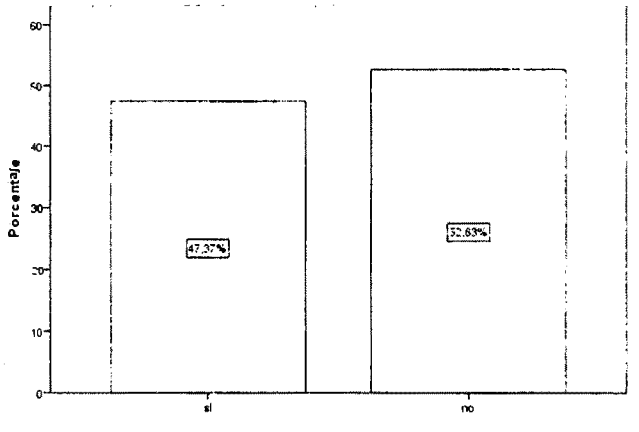


Figura 4.31: Actualización de componentes de aplicaciones web (Fuente: elaboración propia basado en encuestas, 2014)

Preg.32: ¿Revisa la seguridad de los componentes de sus aplicaciones web en listas de correo de seguridad tales como: securityfocus, bugtraq, CERT, etc. y los mantiene actualizados?

Interpretación: Del cuadro 4.32 se observa que en el 89.5 % de las organizaciones no revisan la seguridad de los componentes de sus aplicaciones web en listas de correo de seguridad tales como: securityfocus, bugtraq, CERT, etc. para mantenerlos actualizados, mientras que en el 10.5 % si lo hace.

Cuadro 4.32: Revisión de la seguridad de los componentes de las WebApps en listas de correo de seguridad (Fuente: elaboración propia basado en encuestas, 2014)

	Frecuencia	Porcentaje	Porcentaje válido	Porcentaje acumulado
si	2	10.5	10.5	10.5
no	17	89.5	89.5	100.0
Total	19	100.0	100.0	

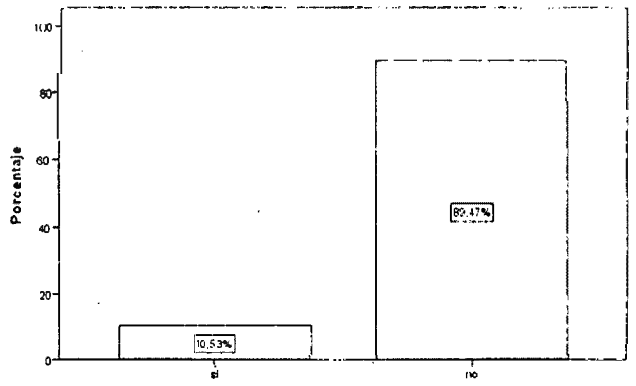


Figura 4.32: Revisión de la seguridad de los componentes de las WebApps en listas de correo de seguridad (Fuente: elaboración propia basado en encuestas, 2014)

Conclusión: del análisis de las preguntas 31 y 32 del cuestionario se concluye que el riesgo de uso de componentes con vulnerabilidades conocidas está contemplado en un 47.4 % mediante la revisión y actualización de los componentes de la aplicación.

4.2.2. Resultados de la guía de entrevista

Preg.1: ¿Qué riesgos de seguridad o vulnerabilidades en aplicaciones web conoce Ud.?

Interpretación: Del cuadro 4.33 se observa que los riesgos de seguridad en webapps que se conocen en las organizaciones píblicas y privadas son como sigue: el 9.1 % conoce Inyección SQL, 18.2 % Configuración de Seguridad Incorrecta, 18.2 % Exposición de Datos Sensibles, 9.1 % Componentes con Vulnerabilidades Conocidas, 36.4 % N.A y 9.1 % otros(phishing, DoS).

Cuadro 4.33: Riesgos de seguridad conocidos en las organizaciones públicas y privadas (Fuente: elaboración propia basado en entrevistas, 2014)

	Frecuencia	Porcentaje	Porcentaje válido	Porcentaje acumulado
Inyección SQL	1	9.1	9.1	9.1
Configuración de Seguridad Incorrecta	2	18.2	18.2	27.3
Exposición de Datos Sensibles	2	18.2	18.2	45.5
Componentes con Vulnerabilidades Conocidas	1	9.1	9.1	54.5
N.A	4	36.4	36.4	90.9
Otros(phishing, DoS)	1	9.1	9.1	100.0
Total	11	100.0	100.0	

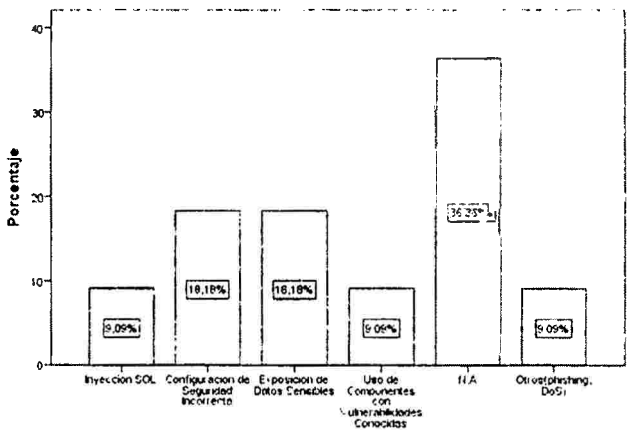


Figura 4.33: Riesgos de seguridad conocidos en las organizaciones públicas y privadas (Fuente: elaboración propia basado en entrevistas, 2014)

Preg.2: ¿Qué información tiene Ud. del riesgo o vulnerabilidad de inyección SQL y qué soluciones conoce para prevenirlo?

Interpretación: Del cuadro 4.34 se observa que la información que tienen del riesgo de inyección SQL son como sigue: el 36.4 % conoce poco, 18.2 % tiene referencias y 45.5 % desconocen el tema.

Cuadro 4.34: Información que se tiene del riesgo de inyección SQL (Fuente: elaboración propia basado en entrevistas, 2014)

	Frecuencia	Porcentaje	Porcentaje válido	Porcentaje acumulado
Conoce poco	4	36.4	36.4	36.4
Tiene referencias	2	18.2	18.2	54.5
Desconoce el tema	5	45.5	45.5	100.0
Total	11	100.0	100.0	

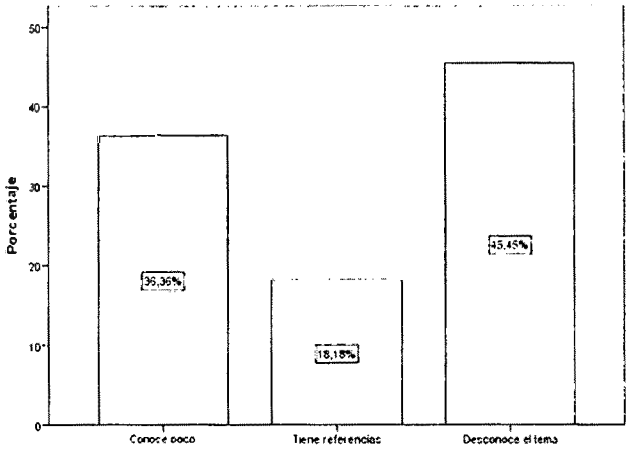


Figura 4.34: Información que se tiene del riesgo de inyección SQL (Fuente: elaboración propia basado en entrevistas, 2014)

Interpretación: Del cuadro 4.35 se observa que las soluciones que se conocen del riesgo de inyección SQL son como sigue: 36.4 % indicó procedimientos almacenados. 36.4% N.A y 27.3% otros.

Cuadro 4.35: Soluciones que conocen del riesgo de inyección SQL(Fuente: elaboración propia basado en entrevistas, 2014)

	Frecuencia	Porcentaje	Porcentaje válido	Porcentaje acumulado
Procedimientos almacenados	4	36.4	36.4	36.4
N.A	4	36.4	36.4	72.7
Otros	3	27.3	27.3	100.0
Total	11	100.0	100.0	

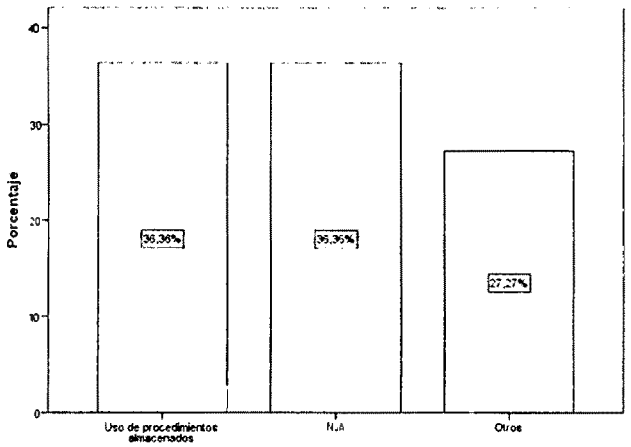


Figura 4.35: Soluciones que conocen del riesgo de inyección SQL(Fuente: elaboración propia basado en entrevistas, 2014)

Preg.3: ¿Qué información tiene Ud. del riesgo o vulnerabilidad de Perdida de Autenticación y Gestión de Sesiones y qué soluciones conoce para prevenirlo?

Interpretación: Del cuadro 4.36 se observa que la información que tienen del riesgo de Perdida de Autenticación y Gestión de Sesiones son como sigue: el 27.3 % conoce el tema, 54.5 % conoce poco, 9.1 % tiene referencias y 9.1 % desconocen el tema.

Cuadro 4.36: Información que se tiene del riesgo de Perdida de Autenticación y Gestión de Sesiones (Fuente: elaboración propia basado en entrevistas, 2014)

	Frecuencia	Porcentaje	Porcentaje válido	Porcentaje acumulado
Conoce el tema	3	27.3	27.3	27.3
Conoce poco	6	54.5	54.5	81.8
Tiene referencias	1	9.1	9.1	90.9
Desconoce el tema	1	9.1	9.1	100.0
Total	11	100.0	100.0	

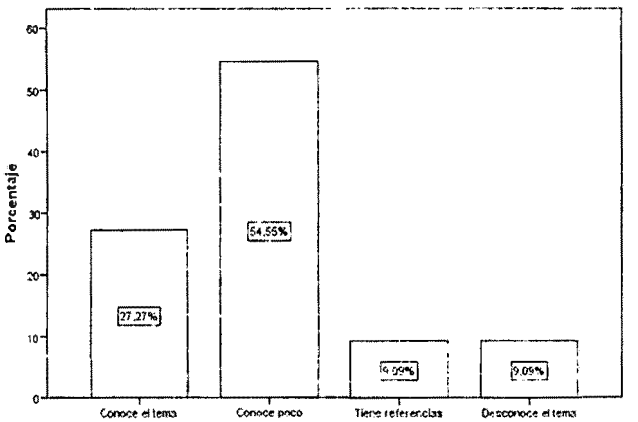


Figura 4.36: Información que se tiene del riesgo de Perdida de Autenticación y Gestión de Sesiones (Fuente: elaboración propia basado en entrevistas, 2014)

Interpretación: Del cuadro 4.37 se observa que las soluciones que se conocen del riesgo de Perdida de Autenticación y Gestión de Sesiones son como sigue: el 9.1 % indicó Cifrado de las credenciales, 9.1 % Caducidad de las sesiones, 27.3 Gestión de usuarios concurrentes, 27.3 % N.A y 27.3 Otros.

Cuadro 4.37: Soluciones que conocen del riesgo de Perdida de Autenticación y Gestión de Sesiones(Fuente: elaboración propia basado en entrevistas, 2014)

	Frecuencia	Porcentaje	Porcentaje válido	Porcentaje acumulado
Cifrado de las credenciales	1	9.1	9.1	9.1
Caducidad de las sesiones	1	9.1	9.1	18.2
Gestión de usuarios concurrentes	3	27.3	27.3	45.5
N.A	3	27.3	27.3	72.7
Otros	3	27.3	27.3	100.0
Total	11	100.0	100.0	

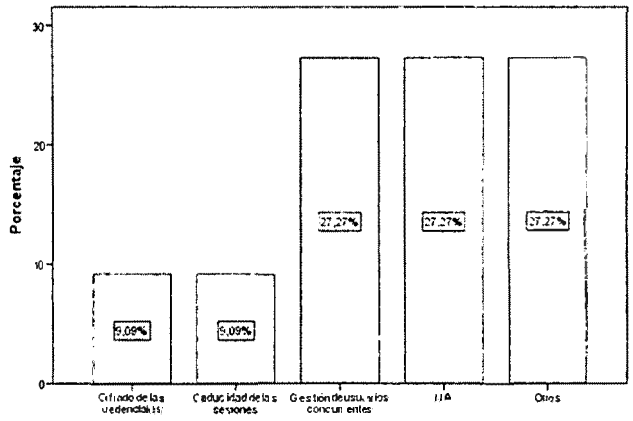


Figura 4.37: Soluciones que conocen del riesgo de Perdida de Autenticación y Gestión de Sesiones (Fuente: elaboración propia basado en entrevistas, 2014)

Preg.4: ¿Qué información tiene Ud. del riesgo o vulnerabilidad de Secuencia de Comandos en Sitios Cruzados (XSS) y qué soluciones conoce para prevenirlo?

Interpretación: Del cuadro 4.38 se observa que la información que tienen del riesgo de Secuencia de Comandos en Sitios Cruzados (XSS) son de la siguiente manera: el 9.1 % conoce poco, 9.1 % tiene referencias y 81.8 % desconocen el tema.

Cuadro 4.38: Información que se tiene del riesgo de Secuencia de Comandos en Sitios Cruzados (XSS) (Fuente: elaboración propia basado en entrevistas, 2014)

	Frecuencia	Porcentaje	Porcentaje válido	Porcentaje acumulado
Conoce poco	1	9.1	9.1	9.1
Tiene referencias	1	9.1	9.1	18.2
Desconoce el tema	9	81.8	81.8	100.0
Total	11	100.0	100.0	

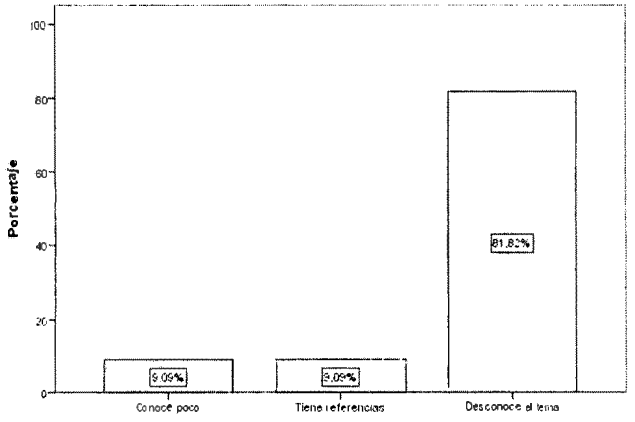


Figura 4.38: Información que se tiene del riesgo de Secuencia de Comandos en Sitios Cruzados (XSS) (Fuente: elaboración propia basado en entrevistas, 2014)

Interpretación: Del cuadro 4.39 se observa que las soluciones que se conocen del riesgo de Secuencia de Comandos en Sitios Cruzados (XSS) son: el 9.1 % indicó el uso de librerías que impiden XSS, 72.7% N.A y 18.2% otros.

Cuadro 4.39: Soluciones que conocen del riesgo de Secuencia de Comandos en Sitios Cruzados (XSS) (Fuente: elaboración propia basado en entrevistas, 2014)

	Frecuencia	Porcentaje	Porcentaje válido	Porcentaje acumulado
Uso de librerías que impiden XSS	1	9.1	9.1	9.1
N.A	8	72.7	72.7	81.8
Otros	2	18.2	18.2	100.0
Total	11	100.0	100.0	

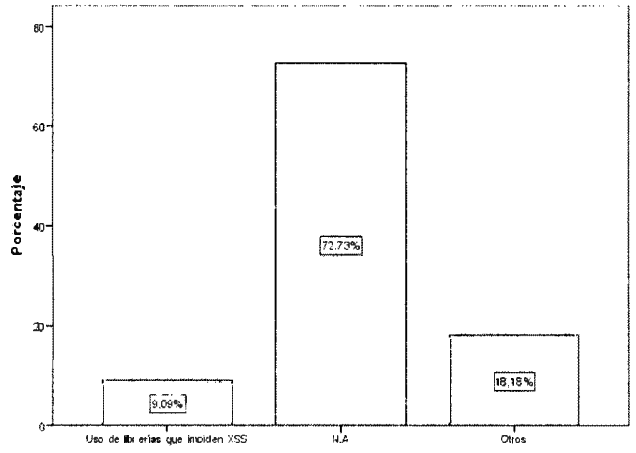


Figura 4.39: Soluciones que conocen del riesgo de Secuencia de Comandos en Sitios Cruzados (XSS) (Fuente: elaboración propia basado en entrevistas, 2014)

Preg.6: ¿Qué información tiene Ud. del riesgo o vulnerabilidad de Exposición de Datos Sensibles qué soluciones conoce para prevenirlo?

Interpretación: Del cuadro 4.40 se observa que la información que tienen del riesgo de Exposición de Datos Sensibles son de la siguiente manera: 9.1 % domina el tema, 63.6 % conoce el tema y 27.3 % conoce poco.

Cuadro 4.40: Información que se tiene del riesgo de Exposición de Datos Sensibles (Fuente: elaboración propia basado en entrevistas, 2014)

	Frecuencia	Porcentaje	Porcentaje válido	Porcentaje acumulado
Domina el tema	1	9.1	9.1	9.1
Conoce el tema	7	63.6	63.6	72.7
Conoce poco	3	27.3	27.3	100.0
Total	11	100.0	100.0	

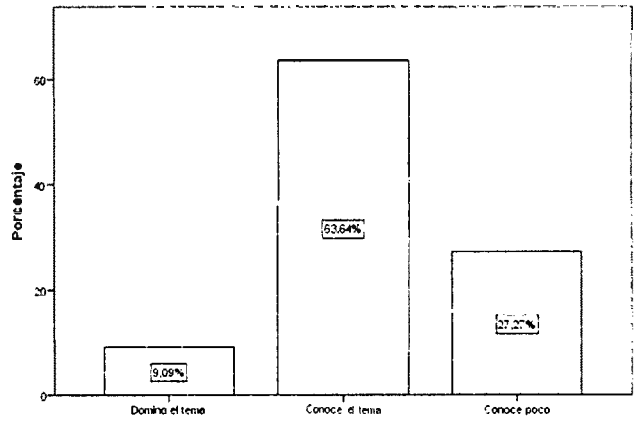


Figura 4.40: Información que se tiene del riesgo de Exposición de Datos Sensibles (Fuente: elaboración propia basado en entrevistas, 2014)

Interpretación: Del cuadro 4.41 se observa que las soluciones que se conocen del riesgo de Exposición de Datos Sensibles son de la siguiente manera: el 81.8 % indicó cifrado de datos confidenciales y 18.2 % uso de TLS.

Cuadro 4.41: Soluciones que conocen del riesgo de Exposición de Datos Sensibles (Fuente: elaboración propia basado en entrevistas, 2014)

	Frecuencia	Porcentaje	Porcentaje válido	Porcentaje acumulado
Cifrado de datos confidenciales	9	81.8	81.8	81.8
Uso de TLS	2	18.2	18.2	100.0
Total	11	100.0	100.0	

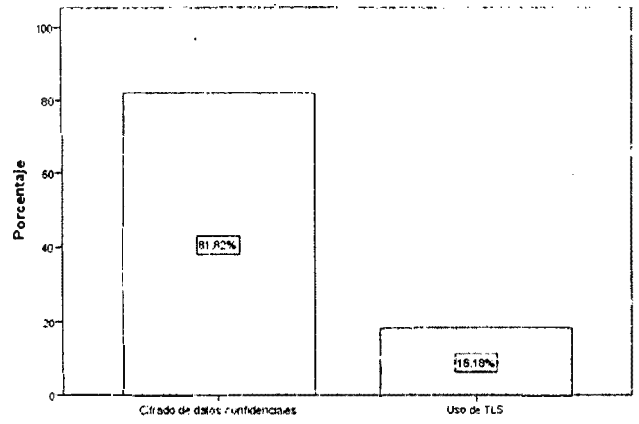


Figura 4.41: Soluciones que conocen del riesgo de Exposición de Datos Sensibles (Fuente: elaboración propia basado en entrevistas, 2014)

Preg.7: ¿Qué información tiene Ud. del riesgo o vulnerabilidad de Falsificación de Petición en Sitios Cruzados (CSRF) y qué soluciones conoce para prevenirlo?

Interpretación: Del cuadro 4.42 se observa que la información que tienen del riesgo de Falsificación de Petición en Sitios Cruzados (CSRF) son de la siguiente manera: el 9.1 % tiene referencias y 90.9 % desconoce el tema.

Cuadro 4.42: Información que se tiene del riesgo de Falsificación de Petición en Sitios Cruzados (CSRF) (Fuente: elaboración propia basado en entrevistas, 2014)

	Frecuencia	Porcentaje	Porcentaje válido	Porcentaje acumulado
Tiene referencias	1	9.1	9.1	9.1
Desconoce el tema	10	90.9	90.9	100.0
Total	11	100.0	100.0	

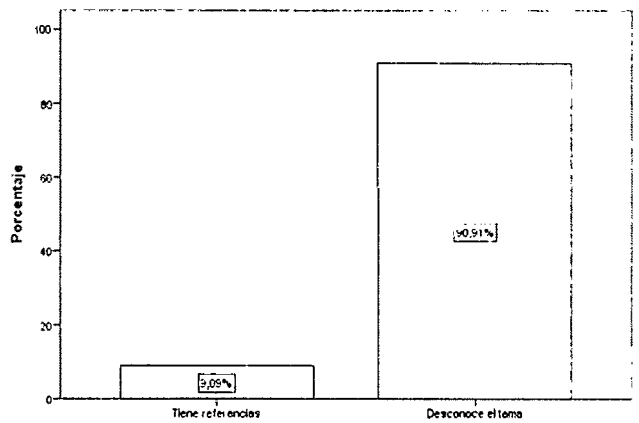


Figura 4.42: Información que se tiene del riesgo de Falsificación de Petición en Sitios Cruzados (CSRF) (Fuente: elaboración propia basado en entrevistas, 2014)

Interpretación: Del cuadro 4.43 se observa que las soluciones que se conocen del riesgo de Falsificación de Petición en Sitios Cruzados (CSRF) son de la siguiente manera: es 90.9% N.A y 9.1 % otros.

Cuadro 4.43: Soluciones que conocen del riesgo de Secuencia de Comandos en Falsificación de Petición en Sitios Cruzados (CSRF) (Fuente: elaboración propia basado en entrevistas. 2014)

	Frecuencia	Porcentaje	Porcentaje válido	Porcentaje acumulado
N.A	10	90.9	90.9	90.9
Otros	1	9.1	9.1	100.0
Total	11	100.0	100.0	

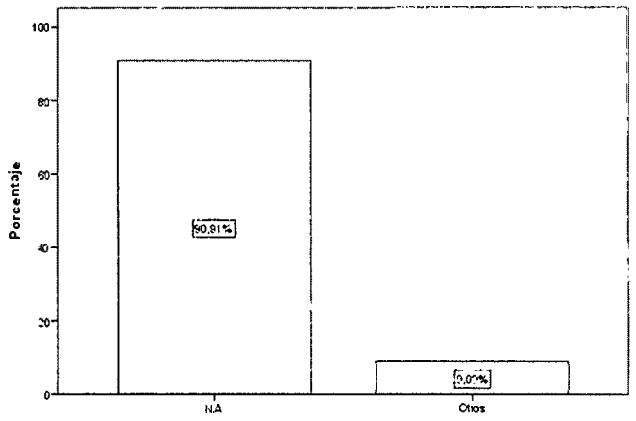


Figura 4.43: Soluciones que conocen del riesgo de Falsificación de Petición en Sitios Cruzados (CSRF) (Fuente: elaboración propia basado en entrevistas. 2014)

4.3. Discusión de Resultados

Los instrumentos utilizados en la investigación (el cuestionario y la guía de entrevista) cumplen con la fiabilidad y validez de instrumentos, estos datos dieron garantía para su uso en el presente estudio.

No se pudo investigar los diez riesgos de seguridad del OWASP top 10 por ser este un tema muy amplio, razón por la cual se trabajó con siete riesgos y se descartaron tres, estos son: referencia directa insegura a objetos, ausencia de control de acceso a las funciones y redirecciones y reenvíos no validados.

De la misma manera, no se tuvo acceso al código fuente de las aplicaciones web de las organizaciones de la muestra seleccionada para la evaluación de los riesgos de seguridad.

Del cruce de las respuestas de los instrumentos de medición se observa que en el cuestionario los encuestados afirman conocer los riesgos de seguridad en aplicaciones

web, en tanto que en las respuestas de la guía de entrevista los entrevistados desconocen los riesgos de seguridad en aplicaciones web.

Seguidamente se discute los resultados con los antecedentes de la investigación.

Cruz (2012). En el año 2012, Cruz Ayala, Javier, en su trabajo de tesis titulado: Técnicas de protección para mejorar la seguridad de una aplicación web. Mediante el lenguaje java, implementó algunas técnicas de protección para mejorar la seguridad de una aplicación web, entre ellos: cifrado de datos, creación de certificados digitales, configuración de apache tomcad para el uso del protocolo https, implementación de un teclado virtual, implementación de captcha y finalmente para la protección de la URL usó el fron controller.

Discrepancias: la tesis mencionada no considera todas los problemas de sus variables de estudio. Por ejemplo, para implementar una solución frente al problema de inyección SQL considera únicamente procedimientos almacenados, pero además existen otras soluciones más importantes como: validación de ingreso de datos, validación de errores de los rechazos de acceso y mínimos privilegios en el acceso a la base de datos, los cuales si están considerados en mi investigación.

Zavaleta (2012), Zavaleta De la Cruz, Yury Daniel, en su trabajo de tesis titulado: Impacto de ataques SQL injection en los portales web interactivos de las empresas del sector TI de la ciudad de Trujillo, 2012. Sostuvo que su investigación "tiene como finalidad proponer una metodología, cuya secuencia de pasos permitirá disminuir el grado de vulnerabilidad de los tipos de ataques llamados SQL injection, en los portales web de Trujillo. Para ello, describe las diferentes técnicas y herramientas que se encuentran disponibles así como las buenas prácticas que dictaminan organismos que son autoridades en el tema de SQL injection, como por ejemplo ISACA, OWASP, RSA, SANS, teniendo en cuenta que SQL injection es uno de los tipos de ataques que tiene mayor impacto en los sistemas."

Discrepancias: la diferencia con esta tesis es que hago un estudio de los riesgos en aplicaciones web publicados por la organización OWASP, OWASP top 10, utilizando básicamente instrumentos como el cuestionario y la guía de entrevista con los cuáles identifico si estos riesgos son contemplados o no en el desarrollo de las aplicaciones web en las organizaciones públicas y privadas más representativas de la ciudad de Ayacucho. Como parte final de la investigación en base a los resultados se realiza una serie de sugerencias para minimizar los riegos más críticos en el desarrollo de aplicaciones web.

Kopman, 2012, En el año 2012, Martin Kompan, en la universidad Masaryk Uni-

versity, facultad de informática, en su tesis de maestría titulado: Enterprise Web Application Security, llegó a la conclusión de que el problema de secuestro de sesión es la vulnerabilidad más grave en las aplicaciones web. Así mismo recomienda implementar políticas de uso de contraseñas más estrictas que no permitan que los usuarios utilicen contraseñas débiles y susceptibles a ataques de diccionarios.

Discrepancias: La diferencia con este trabajo de investigación es que Martin Kompan evalúa los riesgos de seguridad utilizando la metodología OWASP en dos empresas, una con información sensible de sus usuarios y la otra con información financiera. Esta evaluación lo realiza utilizando las diferentes herramientas de auditoría de seguridad sin considerar los detalles existentes de estos problemas. En tanto que mi investigación estudia los riesgos de seguridad contemplados en el desarrollo de WebApps, utilizando los instrumentos del cuestionario y la guía de entrevista, así mismo considero los detalles de estos problemas de seguridad.

Delgado, 2012, En el año 2012, Delgado Caballero, Gerson Geovanny, en su trabajo de investigación titulado: Metodología de pruebas de inyección SQL para entornos web, llegó a la conclusión de que el eslabón más débil en una aplicación web para que permita ataques de inyección SQL es la poca o ninguna validación de las variables de entrada. Así mismo indica que los desarrolladores deben reconocer que la seguridad es un componente fundamental de cualquier producto de software y deben incluir buenas prácticas de programación en el software que están desarrollando.

Discrepancias: La diferencia con este trabajo de investigación, es que el Sr. Delgado Caballero, Gerson Geovanny, investiga detalladamente el problema de riesgo de inyección SQL. En tanto que mi investigación se enfoca a estudiar los riesgos de seguridad en aplicaciones web publicados por la OWASP, utilizando los instrumentos del cuestionario y la guía de entrevista en las organizaciones públicas y privadas de la ciudad de Ayacucbo.

Capítulo 5

Conclusiones y Recomendaciones

5.1. Conclusiones

5.1.1. Conclusión general

Los riesgos de seguridad contemplados en el desarrollo de aplicaciones web, ciudad de Ayacucho, 2014, son: inyección SQL en mayor porcentaje mediante el uso de procedimientos almacenados. Así mismo, pérdida de autenticación y gestión de sesiones mediante el cifrado de las credenciales de autenticación y la verificación de caducidad de las sesiones, secuencia de comandos en sitios cruzados (XSS) mediante la comprobación de los datos ingresados que son la salida a HTML, configuración de seguridad incorrecta mediante la administración de los número de puertos, servicios, páginas, cuentas y privilegios, exposición de datos sensibles mediante el uso del algoritmo criptográfico AES, falsificación de peticiones en sitios cruzados(CSRF) mediante la implementación de los captcha, y uso de componentes con vulnerabilidades conocidas mediante la revisión y actualización de los componentes de la aplicación.

5.1.2. Conclusiones específicas

A. **Conclusiones del riesgo de inyección SQL**

- En el 57.9 % de las organizaciones no se validan rigurosamente todas los campos de los formularios, campos escondidos, cabeceras, cookies y cadenas de petición.
- En el 52.6 % de las organizaciones no se verifican que todos los errores producidos como resultado del rechazo de acceso por la aplicación web, están

correctamente manejados por la aplicación.

- En el 63.2 % de las organizaciones se usan procedimientos almacenados como mecanismo de solución del riesgo de inyección SQL, mientras que en el 36.8 % no se usa este mecanismo de solución.

B. **Conclusiones del riesgo de pérdida de autenticación y gestión de sesiones**

Conclusiones del riesgo de autenticación

- En el 57.9 % de la organizaciones no se comprueban si los campos de ingreso de usuarios y contraseñas en aplicaciones web están protegidos contra un ataque de fuerza bruta.
- En el 52.6 % de las organizaciones las aplicaciones web no cuentan con funciones para proteger del uso de contraseñas comúnmente elegidos.
- En el 52.6 % de las organizaciones se verifican que todas las credenciales de autenticación se guardan y se transmiten siempre cifradas con hashes no reversibles y no así en el código fuente.

Conclusiones del riesgo de gestión de sesiones

- En el 57.9 % de las organizaciones no se comprueban que sólo los identificadores de sesión generados por la aplicación web son reconocidos como válidos.
- En el 68.4 % de las organizaciones se verifican que las sesiones en las aplicaciones web caducan después de un período de tiempo de inactividad, mientras que en el 31.6 % no lo hace.
- En el 57.9 % de las organizaciones comprueban que la aplicación web no permite que usuarios concurrentes dupliquen sesiones, mientras que en el 42.1 % no lo hace.

C. **Conclusiones del riesgo de secuencia de comandos en sitios cruzados (XSS)**

- En el 84.2 % de las organizaciones no comprueban que las aplicaciones web son susceptibles a ataques de Cross Site Scripting (XSS) o que los controles de seguridad impiden XSS.
- En todas las organizaciones no utilizan algún tipo de biblioteca que no permite que el riesgo de Cross Site Scripting (XSS) se produzca.

- En el 52.6 % de las organizaciones comprueban que todos los datos ingresados por un usuario que son la salida a HTML (incluyendo elementos HTML, atributos HTML, Java script, valores de datos, bloques de CSS y atributos URI) están debidamente validados en la aplicación web, mientras que el 47.4 % no lo hacen.

D. **Conclusiones del riesgo de configuración de seguridad incorrecta**

- El 94.7 % de las organizaciones no cuentan con una guía de hardening (como los de OWASP, CERT, SANS, NSA, Microsoft, etc. que ayudan a configurar correctamente los sistemas) para personalizar los componentes de sus aplicaciones web.
- En el 68.4 % de las organizaciones conocen todos los número de puertos, servicios, páginas, cuentas y privilegios que están configurados actualmente en su servidor web, mientras que el 31.6 % desconocen estas características.

E. **Conclusiones del riesgo de exposición de datos sensibles**

- En el 21.1 % de las organizaciones se usa algoritmos criptográficos DES, en la actualidad considerado inseguro, además 21.1 % no usa ningún algoritmo de cifrado y sólo el 15.8 % usa AES conciderado actualmente el más seguro.
- En el 57.9 % de las organizaciones no comprueban que los datos confidenciales procesados por la aplicación web son cifrados tanto en el almacenamiento como en la transmisión de datos.
- En el 63.2 % de las organizaciones no utilizan TLS (Seguridad de la Capa de Transporte) para todas las conexiones (incluyendo tanto internas como externas) en las que impliquen datos sensibles o funciones especiales.

F. **Conclusiones del riesgo de falsificación de peticiones en sitios cruzados (CSRF)**

- En el 89.5 % de las organizaciones no utilizan algún tipo de librería que proporcione construcciones que hacen más fácil de evitar el riesgo de CSRF (Falsificación de Peticiones en Sitios Cruzados).
- En el 68.4 % de las organizaciones no implementan funciones de reautenticación de usuarios para realizar operaciones críticas.

- En el 52.6 % de las organizaciones implementan mecanismos como los captcha para comprobar que las solicitudes provenientes son de un usuario real, mientras que el 47.4 % no implementan este mecanismo.

G. **Conclusiones del riesgo de uso de componentes con vulnerabilidades conocidas**

- En el 52.6 % de las organizaciones no revisan, actualizan y tampoco parchan constantemente los componentes de la aplicación (servidor web, servidor de aplicación, etc), así mismo no actualizan las guías.
- En el 89.5 % de las organizaciones no revisan la seguridad de los componentes de sus aplicaciones web en listas de correo de seguridad tales como: securityfocus, bugtraq, CERT, etc. para mantenerlos actualizados.

5.2. Recomendaciones

A. **Para el riesgo de inyección SQL**

- Validar rigurosamente todas los campos de los formularios, campos escondidos, cabeceras, cookies y cadenas de petición. Así mismo verificar que todos los errores producidos como resultado del rechazo de acceso por la aplicación web, están correctamente manejados por la aplicación. Usar procedimientos almacenados para las consultas a la base de datos. Y finalmente mínimos privilegios para el acceso a la base da datos.

B. **Para el riesgo de pérdida de autenticación y gestión de sesiones**

Para el riesgo de autenticación

- Comprobar si los campos de ingreso de usuarios y contraseñas están protegidos contra un ataque de fuerza bruta. Implementar funciones para proteger contra el uso de contraseñas comúnmente elegidos. Y verificar que todos las credenciales de autenticación se guardan y se transmiten siempre cifradas con hashes no reversibles y no así en el código fuente.

Para el riesgo de gestión de sesiones

- Comprobar que sólo los identificadores de sesión generados por la aplicación web son reconocidos como válidos. Verificar que las sesiones en las aplicaciones web caducan después de un período de tiempo de inactividad. Y comprobar que la aplicación web no permite que usuarios concurrentes dupliquen sesiones.

C. **Para el riesgo de secuencia de comandos en sitios cruzados (XSS)**

- Comprobar que las aplicaciones web no son susceptibles a ataques de Cross Site Scripting (XSS) o que los controles de seguridad impiden XSS. Utilizar algún tipo de biblioteca que no permite que el riesgo de Cross Site Scripting (XSS) se produzca. Y así mismo, todos los datos ingresados por un usuario que son la salida a HTML (incluyendo elementos HTML, atributos HTML, Java script, valores de datos, bloques de CSS y atributos URI) están debidamente validados en la aplicación web.

D. **Para el riesgo de configuración de seguridad incorrecta**

- Usar una guía de hardening (como los de OWASP, CERT, SANS, NSA, Microsoft, etc. que ayudan a configurar correctamente los sistemas) para personalizar los componentes de sus aplicaciones web. Y conocer todos los número de puertos, servicios, páginas, cuentas y privilegios que están configurados en los componentes de las aplicaciones web.

E. **Para el riesgo de exposición de datos sensibles**

- Usar algoritmos criptográficos considerados actualmente seguros como el AES. Comprobar que los datos confidenciales procesados por la aplicación web son cifrados tanto en el almacenamiento como en la transmisión de datos. Y utilizar TLS (Seguridad de la Capa de Transporte) para todas las conexiones (incluyendo tanto internas como externas) en las que impliquen datos sensibles o funciones especiales.

F. **Para el riesgo de falsificación de peticiones en sitios cruzados (CSRF)**

- Utilizar algún tipo de librería que proporcione construcciones que hacen más fácil de evitar el riesgo de CSRF (Falsificación de Peticiones en Sitios Cruzados). Implementar funciones de reautenticación de usuarios para realizar operaciones críticas. E implementar mecanismos como los captcha para comprobar que las solicitudes provenientes son de un usuario real.

G. **Para el riesgo de uso de componentes con vulnerabilidades conocidas**

- Revisar, actualizar y parchar constantemente los componentes de las aplicaciones web (servidor web, servidor de aplicación, etc) y actualizar las guías. Además revisar la seguridad de los componentes en listas de correo de seguridad tales como: securityfocus, bugtraq, CERT, etc. para mantenerlos actualizados.

5.3. Investigaciones Futuras

A. Investigar a profundidad el riesgo de inyección considerando las consultas SQL, LDAP, comandos de SO e intérpretes del XML.

B. Investigar la gestión de sesiones considerando los tres riesgos publicados por la OWASP relacionados con la gestión de sesiones, estos son: Pérdida de Autenticación y Gestión de Sesiones, Secuencia de Comandos en Sitios Cruzados (XSS)y Falsificación de Peticiones en Sitios Cruzados(CSRF).

C. Investigar los riesgos de seguridad en aplicaciones web en las aplicaciones móviles, considerando el riesgo de inyección y gestión de sesiones.

Bibliografía

|AENOR., 2004| AENOR (2004). Especificaciones para los sistemas de gestión de la seguridad de la información (sgsi).

|ASVS, 2013| ASVS (2013). Owasp application security verification standard, recuperado en enero del 2014 de. https://www.owasp.org.

|BBC, 2013| BBC (2013). ¿es rusia un paraíso para hackers?, recuperado en enero del 2014 de. http://www.bbc.eo.uk/.

|CEH, 2013| CEH (2013). *Ethical Hacking and Countermeasures V8. Module 14: SQL injection.* Council CEH.

|Colobran, 2008| Colobran, A. y. M. (2008). *Administración de sistemas operativas en red.* UOC.

|CWE/SANS, 2011| CWE/SANS (2011). Cwe/sans top 25 most dangerous software errors, recuperado en enero del 2014 de, http://cwe.mitre.org.

|Demarco, 1995| Demarco, T. (1995). *Why Does Software Cost so Much and other puzzles of the Information Age 1995 paperback.* Dorset.

|Hansen, 2013| Hansen, R. (2013). Website scanners: the good, the bad and the reality, recuperado en febrero del 2014 de. http://info.whitehatsec.com/rs/whitehatsecurity/images/WebsiteScanners.pdf.

|Herrera, 2012| Herrera, D. (2012). Seguridad en aplicaciones web, recuperado en mayo del 2014 de. http://cryptomex.org, cryptomex.

|Horna, 2012| Horna, D. A. A. V. (2012). *7 Pasos para una tesis exitosa.* Universidad San Martín de Porres, tercera edición edition.

[Informatica64,] Informatica64. Seguridad en servidores e internet, recuperado en marzo del 2014 de. http://www.informatica64.com/ManualesTecnicos.aspx.

[ISO/IEC27000, 2014] ISO/IEC27000 (2014). International standad information technology, recuperado en marzo del 2014 de, http://standards.iso.org/ittf/licence.html. ISO/IEC 27000:2014(E).

[Magerit, 2006] Magerit (2006). Metodología de análisis y gestión de riesgos de los sistemas de información, recuperado en abril del 2014 de. https://www.ccn-cert.cni.es/publico/herramientas/pilar43/magerit/.

[Marañón, 2009] Marañón, G. A. (2009). ¿están seguras sus aplicaciones web? *Revista PC world.*

[Mateu, 2004] Mateu, C. (2004). *Desarrollo de aplicaciones web.* UOC, Primera edición.

[Neff, 2002] Neff, P. (2002). Web application security, recuperado en diciembre del 2013 de, http://www.itsec.gov.cn.

[NTP-ISO/IEC17799, 2007] NTP-ISO/IEC17799 (2007). Tecnología de la información. código de buenas prácticas para la gestión de la seguridad de la información. norma técnica peruana ntp-iso/iec 17799.

[offensive security, 2013] offensive security (2013). Backtrack-linux, recuperado en enero del 2014 de, http://www.offensive-security.com/community-projects/backtrack-linux/.

[OWASP, 2004] OWASP (2004). Las diez vulnerabilidades de seguridad más críticas en aplicaciones web.

[OWASP, 2007] OWASP (2007). Las 10 vulnerabilidades de seguridad más críticasas en aplicaciones web.

[OWASP, 2008] OWASP (2008). *Guía de pruebas OWASP v3.0, Recuperado en diciembre del 2013 de.* https://www.owasp.org/.

[OWASP, 2010] OWASP (2010). Owasp top 10 - 2010. los diez riesgos más importantes en aplicaciones web, recuperado en diciembre del 2013 de, https://www.owasp.org/.

[OWASP, 2013] OWASP (2013). Owasp top 10 - 2013. los diez riesgos más críticos en aplicaciones web, recuperado en enero del 2014 de, https://www.owasp.org/.

[Pressman, 2010] Pressman, R. S. (2010). *Ingeniería de Software (Spanish Edition).* McGraw-Hill Interamericana Editores S.A. de C.V.

[Romero, 2011] Romero, D. A. E. C. (2011). *Metodología integral innovadora para planes y tesis.*

[Scambray et al., 2010] Scambray, J., Liu, V., and Sima, C. (2010). *HACKING EXPOSED WEB APPLICATIONS, 3rd Edition.* McGraw-Hill Osborne Media.

[securityfocus, 2013] securityfocus (2013). Glossary, recuperado en diciembre del 2013 de. http://www.securityfocus.com/glossary/H.

[Tori, 2008] Tori, C. (2008). *Hácking ético.* Metroianni impresiones, primera edición.

[Trejos, 2010] Trejos, C. I. P. (2010). Arquitectura multi-agente adaptativa para la detección de ataques en entornos dinámicos y distribuidos.

[WASC, 2010] WASC (2010). Wasc threat classification, recuperado en enero del 2014 de, http://projects.webappsec.org.

[WASC, 2013] WASC (2013). Web application security consortium, recuperado en noviembre del 2013 de, http://projects.webappsec.org.

[WhiteHat, 2010] WhiteHat (2010). Whitehat website security statistic report, recuperado en noviembre del 2013 de. *https://www.whitehatsec.com/.*

Anexos

Anexo A

Solicitud Para Realizar las Entrevistas y los Cuestionarios

SOLICITO: AUTORIZACIÓN PARA RECABAR INFORMACIÓN PARA UNA INVESTIGACIÓN DE TESIS MEDIANTE UN CUESTIONARIO Y UNA ENTREVISTA.

SEÑOR RESP. DEL ÁREA RELACIONADO A LAS TECNOLOGÍAS DE INFORMACIÓN Y COMUNICACIÓN DE LA COOPERATIVA DE AHORRO Y CRÉDITO SAN CRISTÓBAL DE HUAMANGA.

ROMÁN CANCHARI GUTIÉRREZ, egresado de la Universidad Nacional San Cristóbal de Huamanga, Facultad de Ing. De Minas, Geología y Civil, EFP. de Ing. de Sistemas. Identificado con código de estudiante 27041104 y DNI: 43168883. Ante usted, con el debido respeto me presento y expongo:

- Que, siendo necesario contar con información acerca de los riesgos de seguridad en aplicaciones web del órea de su digno cargo, solicito me autorice para recabar información mediante un cuestionario y una entrevista relacionados con mi trabajo de investigación titulado: "Riesgos de Seguridad en el Desarrollo de Aplicaciones Web, ciudad de Ayacucho, 2014".

POR LO EXPUESTO:
Ruego a usted, se sirva acceder a mi petición.

Ayacucho, 03 de marzo del 2014.

CANCHARI GUTIÉRREZ, Román
DNI Nº 43168883

COOPERATIVA DE AHORRO Y CRÉDITO "SAN CRISTÓBAL DE HUAMANGA"
Enver Alex Añanca Gamboa
Jefe (e) Sistemas e Informática
Recibido
03/03/2014
12:17 p.m.

COOPERATIVA DE AHORRO Y CRÉDITO
SANTA MARÍA MAGDALENA
UNIDAD DE SISTEMAS
RECIBIDO
Registro N°..... Firma:.....
FECHA: 28-02-2014

SOLICITO: AUTORIZACIÓN PARA RECABAR INFORMACIÓN PARA UNA INVESTIGACIÓN DE TESIS MEDIANTE UN CUESTIONARIO Y UNA ENTREVISTA.

SEÑOR RESP. DEL ÁREA RELACIONADO A LAS TECNOLOGÍAS DE INFORMACIÓN Y COMUNICACIÓN DE LA COOPERATIVA DE AHORRO Y CRÉDITO SANTA MARÍA MAGDALENA.

ROMÁN CANCHARI GUTIÉRREZ, egresado de la Universidad Nacional San Cristóbal de Huamanga, Facultad de Ing. De Minas, Geología y Civil, EFP. de Ing. de Sistemas. Identificado con código de estudiante 27041104 y DNI: 43168883. Ante usted, con el debido respeto me presento y expongo:

Que, siendo necesario contar con información acerca de los riesgos de seguridad en aplicaciones web del área de su digno cargo, solicito me autorice para recabar información mediante un cuestionario y una entrevista relacionados con mi trabajo de investigación titulado: "Riesgos de Seguridad en el Desarrollo de Aplicaciones Web, ciudad de Ayacucho, 2014".

POR LO EXPUESTO:
Ruego a usted, se sirva acceder a mi petición.

Ayacucho, 28 de febrero del 2014.

CANCHARI GUTIÉRREZ, Román
DNI N° 43168883

SOLICITO: AUTORIZACIÓN PARA RECABAR INFORMACIÓN PARA UNA INVESTIGACIÓN DE TESIS MEDIANTE UN CUESTIONARIO Y UNA ENTREVISTA.

SEÑOR RESP. DEL ÁREA RELACIONADO A LAS TECNOLOGÍAS DE INFORMACIÓN Y COMUNICACIÓN DE LA DE ENTIDAD MAKIPURA MICROFINANZAS S.A.C.

ROMÁN CANCHARI GUTIÉRREZ, egresado de la Universidad Nacional San Cristóbal de Huamanga, Facultad de Ing. De Minas, Geología y Civil, EFP. de Ing. de Sistemas. Identificado con código de estudiante 27041104 y DNI: 43168883. Ante usted, con el debido respeto me presento y expongo:

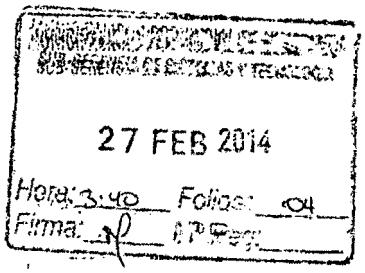
Que, siendo necesario contar con información acerca de los riesgos de seguridad en aplicaciones web del área de su digno cargo, solicito me autorice para recabar información mediante un cuestionario y una entrevista relacionados con mi trabajo de investigación titulado: "Riesgos de Seguridad en el Desarrollo de Aplicaciones Web, ciudad de Ayacucho, 2014".

POR LO EXPUESTO:
Ruego a usted, se sirvo acceder a mi petición.

Ayacucho, 03 de marzo del 2014.

CANCHARI GUTIÉRREZ, Román
DNI N° 43168883

RECIBIDO
Bolle
Firma: Hora: 12:00
N° Reg ..... Folios: 4



SOLICITO: AUTORIZACIÓN PARA RECABAR INFORMACIÓN PARA UNA INVESTIGACIÓN DE TESIS MEDIANTE UN CUESTIONARIO Y UNA ENTREVISTA.

SEÑOR RESP. DEL ÁREA RELACIONADO A LAS TECNOLOGÍAS DE INFORMACIÓN Y COMUNICACIÓN DE LA MUNICIPALIDAD PROVINCIAL DE HUAMANGA.

ROMÁN CANCHARI GUTIÉRREZ, egresado de la Universidad Nacional San Cristóbal de Huamanga, Facultad de Ing. De Minas, Geología y Civil, EFP. de Ing. de Sistemas. Identificado con código de estudiante 27041104 y DNI: 43168883. Ante usted, con el debido respeto me presento y expongo:

Que, siendo necesario contar con información acerca de los riesgos de seguridad en aplicaciones web del área de su digno cargo, solicito me autorice para recabar información mediante un cuestionario y una entrevista relacionados con mi trabajo de investigación titulado: "Riesgos de Seguridad en el Desarrollo de Aplicaciones Web, ciudad de Ayacucho, 2014".

POR LO EXPUESTO:
Ruego a usted, se sirva acceder a mi petición.

Ayacucho, 28 de febrero del 2014.

CANCHARI GUTIÉRREZ, Román
DNI N° 43168883

SOLICITO: AUTORIZACIÓN PARA RECABAR INFORMACIÓN PARA UNA INVESTIGACIÓN DE TESIS MEDIANTE UN CUESTIONARIO Y UNA ENTREVISTA.

SEÑOR RESP. DEL ÁREA RELACIONADO A LAS TECNOLOGÍAS DE INFORMACIÓN Y COMUNICACIÓN DEL SERVICIO DE ADMINISTRACIÓN TRIBUTARIA HUAMANGA.

ROMÁN CANCHARI GUTIÉRREZ, egresado de la Universidad Nacional San Cristóbal de Huamanga, Facultad de Ing. De Minas, Geología y Civil, EFP. de Ing. de Sistemas. Identificado con código de estudiante 27041104 y DNI: 43168883. Ante usted, con el debido respeto me presento y expongo:

Que, siendo necesario contar con información acerca de los riesgos de seguridad en aplicaciones web del área de su digno cargo, solicito me autorice para recabar información mediante un cuestionario y una entrevista relacionados con mi trabajo de investigación titulado: "Riesgos de Seguridad en el Desarrollo de Aplicaciones Web, ciudad de Ayacucho, 2014".

POR LO EXPUESTO:
Ruego a usted, se sirva acceder a mi petición.

Ayacucho, 28 de febrero de 2014.

CANCHARI GUTIÉRREZ, Román
DNI N° 43168883

SAT HUAMANGA
V° B°
Nelmar Moises Paredes
GERENCIA DE INFORMÁTICA
28/02/2014.

MUNICIPALIDAD DISTRITAL DE SAN JUAN BAUTISTA

UNIDAD DE TESORERIA
RUC 20143114911
CANCELADO
NOMBRE:
ROMAN CANCHARI GUTIERREZ
06 MAR 2014
CODIGO: P020401
DOMICILIO
CANCELADO

NUMERO: DPTO:
MANZANA: LOTE:
FECHA:06/03/2014 09:22:54 AM
Pago 0019430

Cant.	Concepto	Total
1	DERECHO DE TRAMITE	1.00
	Sub-Total OT-2014	1.00

CAJA: 01 CAJERO:CAJA01
REF: CAMUCHA
INGRES:***010084643***

PAGO TOTAL: 1.00

LICITO: AUTORIZACIÓN PARA CABAR INFORMACIÓN PARA UNA ESTIGACIÓN DE TESIS MEDIANTE UN ESTIONARIO Y UNA ENTREVISTA.

S TECNOLOGÍAS DE INFORMACIÓN Y STRITAL DE SAN JUAN BAUTISTA.

ANCHARI GUTIÉRREZ, egresado de la l Nacional San Cristóbal de , Facultad de Ing. De Minas, y Civil, EFP. de Ing. de Sistemas. o con código de estudiante DNI: 43168883. Ante usted, con el peto me presento y expongo:

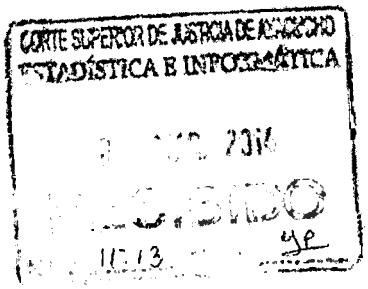
formación acerca de los riesgos de su digno cargo, solicito me autorice n cuestionario y una entrevista relacionados con mi trabajo de investigacion titulado: "Riesgos de Seguridad en el Desarrollo de Aplicaciones Web, ciudad de Ayacucho, 2014".

POR LO EXPUESTO:
Ruego a usted, se sirva acceder a mi petición.

Ayacucho, 05 de marzo del 2014.

CANCHARI GUTIÉRREZ, Román
DNI N° 43168883

MUNICIPALIDAD DISTRITAL DE SAN JUAN BAUTISTA
TRÁMITE DOCUMENTARIO
N° Registro: 1560
Fecha: 06 MAR. 2014
Hora: 9:23
N° de Folios: 06
Recibido por:



SOLICITO: AUTORIZACIÓN PARA RECABAR INFORMACIÓN PARA UNA INVESTIGACIÓN DE TESIS MEDIANTE UN CUESTIONARIO Y UNA ENTREVISTA.

SEÑOR RESP. DEL ÁREA RELACIONADO A LAS TECNOLOGÍAS DE INFORMACIÓN Y COMUNICACIÓN DE LA CORTE SUPERIOR DE JUSTICIA AYACUCHO.

ROMÁN CANCHARI GUTIÉRREZ, egresado de la Universidad Nacional San Cristóbal de Huamanga, Facultad de Ing. De Minas, Geología y Civil, EFP. de Ing. de Sistemas. Identificado con código de estudiante 27041104 y DNI: 43168883. Ante usted, con el debida respeto me presento y expanga:

Que, siendo necesario contar con información acerca de los riesgos de seguridad en aplicacianes web del área de su digno cargo, solicito me autarice para recabar información mediante un cuestionario y una entrevista relacionados con mi trabaja de investigación titulado: "Riesgos de Seguridad en el Desarrollo de Aplicacianes Web, ciudad de Ayacucho, 2014".

POR LO EXPUESTO:
Ruego a usted, se sirva acceder a mi petición.

Ayacucho, 03 de marza del 2014.

CANCHARI GUTIÉRREZ, Román
DNI N° 43168883

SOLICITO: AUTORIZACIÓN PARA RECABAR INFORMACIÓN PARA UNA INVESTIGACIÓN DE TESIS MEDIANTE UN CUESTIONARIO Y UNA ENTREVISTA.

SEÑOR RESP. DEL ÁREA RELACIONADO A LAS TECNOLOGÍAS DE INFORMACIÓN Y COMUNICACIÓN DE LA RENIEC AGENCIA AYACUCHO.

ROMÁN CANCHARI GUTIÉRREZ, egresado de la Universidad Nacional San Cristóbal de Huamanga, Facultad de Ing. De Minas, Geología y Civil, EFP. de Ing. de Sistemas. Identificado con código de estudiante 27041104 y DNI: 43168883. Ante usted, con el debido respeto me presento y expongo:

Que, siendo necesario contar con información acerca de los riesgos de seguridad en aplicaciones web del área de su digno cargo, solicito me autorice paro recabar información mediante un cuestionario y una entrevista relacionados con mi trabajo de investigación titulado: "Riesgos de Seguridad en el Desarrollo de Aplicaciones Web, ciudad de Ayacucho, 2014".

POR LO EXPUESTO:
Ruego a usted, se sirva acceder a mi petición.

Ayacucho, 05 de marzo del 2014.

CANCHARI GUTIÉRREZ, Román
DNI N° 43168883

Ing. Maribel Margarita Vilchaca Rejas
Asistente de Jefatura
Jefatura Regional 7 - Ayacucho
REGISTRO NACIONAL DE IDENTIFICACIÓN Y ESTADO CIVIL

EPSASA - AYACUCHO
Mesa de Partes
-5 MAR. 2014
Nº Reg: 947 Nº Folios: 04
Hora: 12:00 Firma:

SOLICITO: AUTORIZACIÓN PARA RECABAR INFORMACIÓN PARA UNA INVESTIGACIÓN DE TESIS MEDIANTE UN CUESTIONARIO Y UNA ENTREVISTA.

SEÑOR RESP. DEL ÁREA RELACIONADO A LAS TECNOLOGÍAS DE INFORMACIÓN Y COMUNICACIÓN DE LA ENTIDAD PRESTADORA DE SERVICIOS DE SANEAMIENTO AYACUCHO S.A.

ROMÁN CANCHARI GUTIÉRREZ, egresado de la Universidad Nacional San Cristóbal de Huamanga, Facultad de Ing. De Minas, Geología y Civil, EFP. de Ing. de Sistemas. Identificado con código de estudiante 27041104 y DNI: 43168883. Ante usted, con el debido respeto me presento y expongo:

Que, siendo necesario contar con información acerca de los riesgos de seguridad en aplicaciones web del área de su digno cargo, solicito me autorice para recabar información mediante un cuestionario y una entrevista relacionados con mi trabajo de investigación titulado: "Riesgos de Seguridad en el Desarrollo de Aplicaciones Web, ciudad de Ayacucho, 2014".

POR LO EXPUESTO:
Ruego a usted, se sirva acceder a mi petición.

Ayacucho, 05 de marzo del 2014.

CANCHARI GUTIÉRREZ, Román
DNI Nº 43168883