ReCIBE, Año 4 No. 1, Febrero 2015

Guía de ataques, vulnerabilidades, técnicas y herramientas para aplicaciones web

Ana Laura Hernández Saucedo
Centro de Investigación en Matemáticas (CIMAT) Unidad Zacatecas
ana.hernandez@cimat.mx

Jezreel Mejia Miranda
Centro de Investigación en Matemáticas (CIMAT) Unidad Zacatecas
jmejia@cimat.mx

Resumen: En la actualidad el riesgo para los sistemas informáticos ha aumentado debido a un crecimiento en la complejidad en las tecnologías de la información. Hoy en día cualquier computadora conectada a internet está expuesta a diversas amenazas. Una consecuencia es el aumento en el número de ataques informáticos. Una manera de prevenirlo es actuar anticipadamente, detectando las vulnerabilidades potenciales que pueden ser aprovechadas por los atacantes. De esta manera se disminuye la probabilidad de éxito de los ataques realizados. Este trabajo revisa algunas de las técnicas y herramientas utilizadas actualmente para la detección de vulnerabilidades, se presenta una matriz de trazabilidad entre ataques, vulnerabilidades, técnicas y herramientas que determinarán cuales vulnerabilidades y ataques pueden ser mitigados con la utilización de dichas técnicas y herramientas.

Palabras clave: Seguridad, vulnerabilidades, aplicaciones web, ataques, técnicas, herramientas, detección de vulnerabilidades.

Guide of attacks, vulnerabilities, techniques and tools for web application

Abstract: Currently the risk for computer system has increased due to an increase in complexity in information technology. Today any computer connected to internet is exposed to diverse threats. As a result, the number of attacks has increased around the world. One way to prevent it is to act in advance detecting potential vulnerabilities that can be exploited by attackers. Thus the probability of successful attacks decreased. This paper reviews some of the techniques and tools currently used to detect vulnerabilities, presenting a traceability matrix between attacks, vulnerabilities, techniques and tools that determine which vulnerabilities and attacks can be mitigated with the use of these techniques and tools.

Keywords: Security, vulnerabilities, web application, attacks, techniques, tools, vulnerability detection.

1. Introducción

En la actualidad el nivel de complejidad de las Tecnologías de la Información y Comunicación (TICs) ha aumentado, agregando un mayor riesgo para los sistemas informáticos, teniendo como consecuencia el aumento en el número de ataques aprovechando las vulnerabilidades o fallos de seguridad (McAfee An Intel Company, 2014). Dentro de los principales ataques basados en vulnerabilidades se encuentran por ejemplo los de inyección en Lenguaje de consulta estructurado (SQL por sus siglas en ingles) o de Sistemas Operativos (SO); secuencia de comandos en sitios cruzados; falsificación de petición en sitios cruzados (CSRF por sus siglas en ingles) entre otros (Landau, 2013; OSVDB, 2014; UNAM-CERT, 2014).

Una manera de evitar este tipo de ataques informáticos es la prevención utilizando técnicas y herramientas que permitan detectar vulnerabilidades. En cuanto a técnicas existen varios enfoques para la detección de vulnerabilidades, algunos de ellos son Black-box y White-box (Sreenivasa & Kuman, 2012).Existen más enfoques como es el análisis estático y dinámico (Sreenivasa & Kuman, 2012), de ellos existen más técnicas como *passive testing* (AmelMammar, 2011), *faul injection* (AmelMammar, 2011), *fuzz testing* (Xiao-song Zhang, 2008), *penetration testing* (Thompson, 2005), entre otros. Con la utilización de diferentes técnicas como el análisis estático, análisis dinámico, así como, la utilización de herramientas se puede determinar si los sistemas informáticos son vulnerables a ataques. En cuanto a herramientas existen diversas herramientas comerciales por ejemplo McAfee Vulnerability Manager (McAfee An Intel Company, 2013-2014), QualysGuard Web Application Scanning WAS (Qualys Continuous Security, 1999-2014), así como open source como Nessus Vulnerability Scanner en su versión Home (Tenable Network Security , 2014), entre otras, que cumplen este propósito. Por lo tanto, En este trabajo se presenta una propuesta de técnicas y herramientas para la detección de vulnerabilidades actuales en sistemas de información. Además de permitir conocer el estado actual en esta área. Para lograr esto, se ha implementado el protocolo de la revisión sistemática (Kitchenham, 2004), además de la utilización de la herramienta (Mejia, 2014).

Este trabajo está estructurado de la siguiente manera, en la sección 2 se presenta el proceso llevado a cabo durante la realización de la revisión sistemática. En la sección 3 se presentan los resultados obtenidos de la revisión, además de una matriz de trazabilidad realizada con el análisis de los resultados de la revisión y por último la realización de una propuesta para la utilización de herramientas para cada ataque basado en vulnerabilidades. Por último, en la sección 4 se presenta la conclusión de este trabajo, así como el trabajo futuro.

2. Revisión Sistemática

Una revisión sistemática es un método que permite a los especialistas obtener resultados relevantes y cuantificados. Esto puede llevar a la identificación, selección y presentación de pruebas en relación con la investigación en un tema en particular. (Jorgensen & Shepperd, 2007; Kitchenham, 2004).

El proceso de desarrollo de la revisión sistemática se divide en tres fases. En la primera fase se realiza la planeación de la revisión sistemática, incluyendo tareas como identificar la necesidad de realizar la revisión, especificar la pregunta o preguntas de investigación, entre otras. La segunda fase es la conducción de la revisión sistemática, incluyendo tareas como selección de los estudios primarios, evaluación de la calidad del estudio, entre otros. Y finalmente la tercera fase de reporte de la revisión como se muestra en la Figura 1.

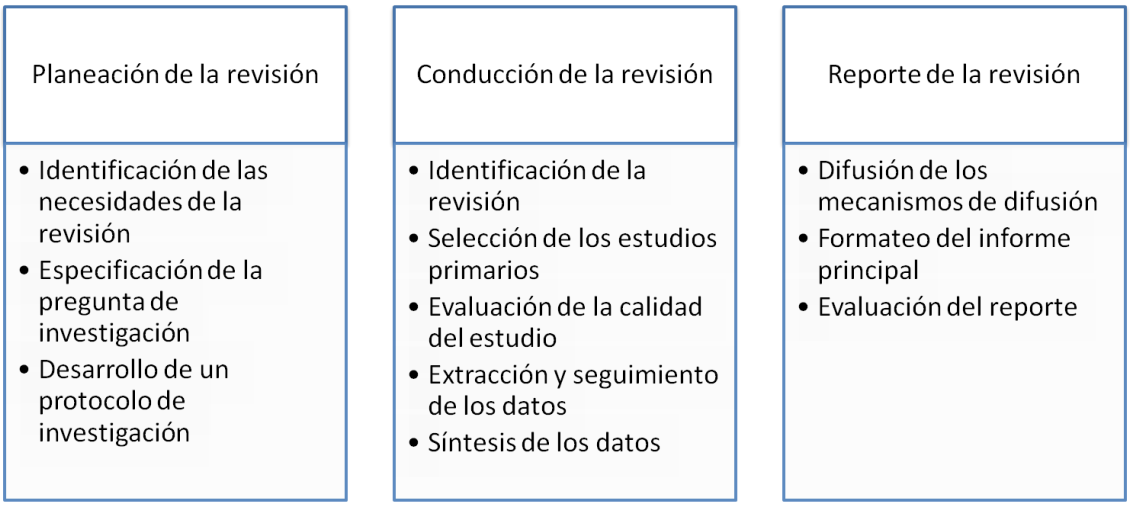

Figura 1. *Proceso de desarrollo de revisión sistemática.*

Unas de las principales actividades dentro del proceso de revisión sistemática es el establecimiento de las preguntas de investigación. Las preguntas que se establecieron para esta investigación fueron las que se indican en la Tabla 1.

Tabla 1. **Preguntas de investigación**

No.	**Preguntas de investigación**
1	¿Cuáles son las herramientas y técnicas utilizadas para la detección de vulnerabilidades para garantizar el buen funcionamiento de los sistemas y la integridad de la información?
2	¿Cuáles son los tipos de vulnerabilidades en tecnologías de información?
3	¿Cuáles son las herramientas y técnicas utilizadas para la detección de vulnerabilidades para aplicaciones web?
4	¿Cuáles son las herramientas open source para detección de vulnerabilidades?

Como resultado del proceso de revisión sistemática se encontraron 23 artículos en IEEExplorer, 22 en ACM Digital Library, 72 en Google Scholar y 58 en Citiseer Library. Teniendo 175 artículos en total, reduciéndolos aplicando los criterios de inclusión y exclusión se llegó a un total de 42 artículos primarios como se muestra en la Figura 2.

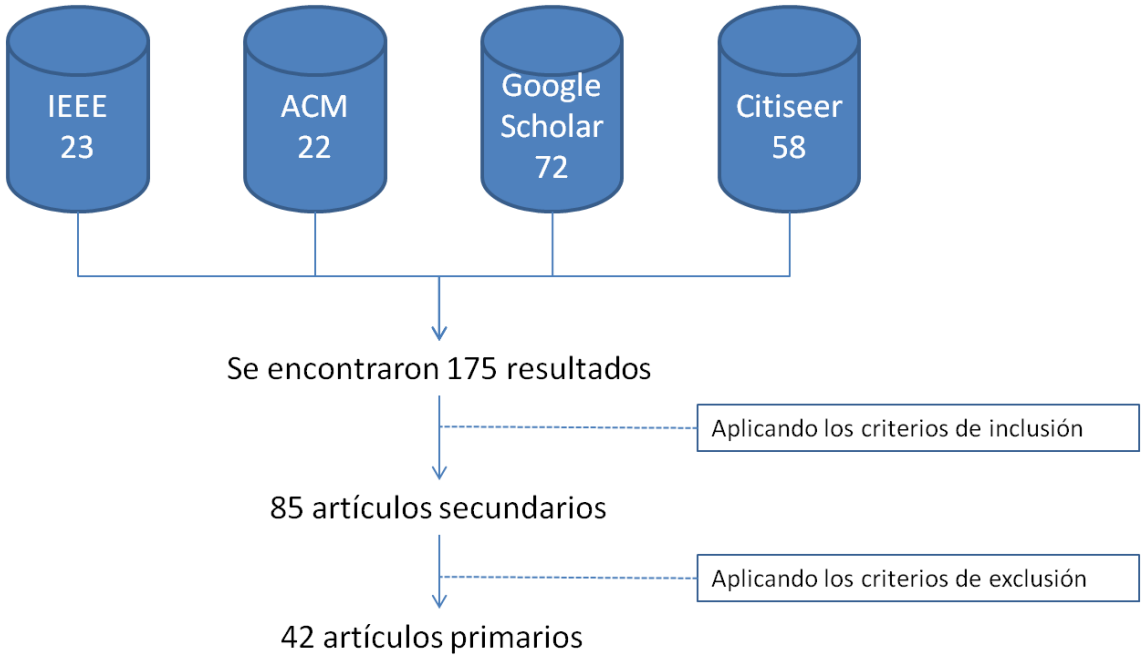

Figura 2. *Resultados de la revisión sistemática.*

3. Análisis de resultados

A partir de la información extraída de los estudios primarios, se realizó un análisis para mostrar descubrimientos relevantes de la revisión sistemática acerca de ataques basados en vulnerabilidades, además de las técnicas y herramientas que son utilizadas actualmente para detectar vulnerabilidades en aplicaciones web, esta información servirá de base para la realización de la propuesta. Los resultados que se obtuvieron son mostrados a continuación.

3.1. Ataques basados en vulnerabilidades

El proyecto abierto de seguridad en aplicaciones web (OWASP por sus siglas en inglés) emite el top 10 de las vulnerabilidades más graves de aplicaciones web (Lai, Grad. Inst. of Inf. & Comput. Educ., Wu, Chen, & Wu, 2008). El objetivo principal es educar a las organizaciones que hacen uso de las TICs sobre las consecuencias de las vulnerabilidades de seguridad en aplicaciones web más importantes(Landau, 2013). Los principales 5 ataques son:

- **Inyección:** Las fallas de inyección, tales como SQL, OS, LDAP, ocurren cuando datos no confidenciales son enviados a un interprete como parte de un comando o consulta, tratando de engañar al intérprete en ejecutar comandos no intencionados o acceder datos no autorizados.
- **Secuencia de Comandos en Sitios Cruzados:** Las fallas XSS ocurren cada vez que una aplicación toma datos no confidenciales 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 victima los cuales pueden secuestrar las sesiones de usuario, destruir sitios web, o dirigir al usuario hacia un sitio malicioso.
- **Configuración de Seguridad Incorrecta:** Una buena seguridad requiere tener definidas e implementada una configuración segura para la aplicación, marcos de trabajo, servidores de aplicación, servidores web, base de datos, y plataformas. Todas estas configuraciones deben ser definidas, implementadas, y mantenidas ya que por lo general no son seguras por defecto.

- **Exposición de datos sensibles:** Muchas aplicaciones web no protegen adecuadamente datos sensibles tales como números de tarjetas de crédito, o credenciales de autenticación. Los datos sensibles requieren de métodos de protección adicionales tales como el cifrado de datos, así como también de precauciones especiales en un intercambio de datos con el navegador.
- **Falsificación de Petición en Sitios Cruzados (CSRF):** 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.

3.2. Técnicas para detección de vulnerabilidades

De acuerdo a los resultados obtenidos de la revisión sistemática sobre las herramientas y técnicas más utilizadas actualmente para detección de vulnerabilidades, se pueden establecer las siguientes:

- **Black-box:** Es una técnica basada para descubrir vulnerabilidades en aplicaciones web, probando la aplicación desde el punto de vista del atacante (Sreenivasa & Kuman, 2012).
- **White-box:** Está del lado del servidor. En este tipo de enfoque se tiene acceso a información relevante de la organización (Sreenivasa & Kuman, 2012).
- **Análisis estático de código (auditoria de código fuente):** Es un método en el que no se requiere ejecutar el programa, este realiza un análisis de código fuente directo para determinar huecos en la seguridad (Sreenivasa & Kuman, 2012).
- **Análisis dinámico de código:** Se comunica con la aplicación web a través de front-end de la aplicación en orden de identificar vulnerabilidades de seguridad potenciales y debilidades en la arquitectura de la aplicación web (Sreenivasa & Kuman, 2012).
    - **Pruebas de penetración:** Consiste en la simulación de un ataque de los maliciosos outssiders (que no tienen un medio autorizado de acceder a

los sistemas de la organización) y de maliciosos insiders (que tienen algún nivel de acceso autorizado). El proceso implica un análisis activo del sistema en busca de posibles vulnerabilidades que podrían resultar de configuración deficiente o inadecuada del sistema, fallos de hardware o software, ya sea conocidos y desconocidos, o fallos operativos en proceso o contramedidas técnicas (Thompson, 2005).

- **Pruebas pasivas:** Las pruebas pasivas están diseñadas para el análisis del tráfico de telecomunicaciones. Permite detectar fallas y defectos de seguridad mediante el examen de los paquetes capturados (livetrafficor log files) (Mammar, Cavalli, & Jimenez, 2011).
- **Pruebas activas:** Utiliza un programador de subprocesos asignados al azar para verificar si las advertencias comunicadas por un análisis predictivo de programa son errores reales (Xiao-song Zhang, 2008).
- **Fuzz testing (pruebas de caja negra):** Consiste en estimular el sistema bajo prueba, utilizando datos aleatorios o mutados queridos, con el fin de detectar comportamientos no deseados como violación de confidencialidad (Xiao-song Zhang, 2008).

3.3. Herramientas para la detección de vulnerabilidades

Las herramientas que se obtuvieron como resultado de la revisión sistemática se describen a continuación:

- **QualysGuard Web Application Scanning WAS:** Es una herramienta en la nube que permite realizar pruebas funcionales con selenium para aplicaciones web, además de pruebas de penetración. Permite encontrar vulnerabilidades del top 10 de OWASP (Qualys, 2014).
- **WebSite Security Audit- WSSA:** Permite examinar páginas web, aplicaciones y servidores web para encontrar vulnerabilidades de seguridad. Realiza pruebas de vulnerabilidades de código conocidas como: SQL Injection, XSS (Cross Site Scripting), entre otras (BeyondSecurity, 2014).
- **Retina Web Security Scanner:** Es una solución de escaneo de sitios web, aplicaciones web complejas para hacer frente a las vulnerabilidades

de aplicaciones. Prioriza las vulnerabilidades por su nivel de riesgo (Beyontrust, 2014).

- **WEBAPP 360: Enterprise Class web application scanning:** Evalúa de manera completa la infraestructura de aplicaciones web, incluyendo aplicaciones web, sistemas operativos subyacentes y aplicaciones subyacentes en entorno de producción. Utiliza el Top 10 de OWASP para cerrar las brechas de seguridad en aplicaciones web (Tripwire, 2014).
- **Frame-C:** Es un software Open Source que permite analizar código fuente escrito en C. Reúne varias técnicas de análisis estático en una sola herramienta. (Frama-C, 2014).
- **Parasoft C/C++ Test:** Es una solución de pruebas para aplicaciones basadas en C y C++. Ayuda a desarrolladores a prevenir y eliminar defectos. Ayuda a eliminar problemas de seguridad, además vigila el cumplimiento de OWASP Top 10, CWE/SANS, FDA, entre otros (Parasoft, 2014).
- **Fortify Static Code Analyzer:** Proporciona análisis de código estático automatizado para ayudar a los desarrolladores a eliminar las vulnerabilidades y crear software de seguridad. Analiza el código fuente, identifica las causas originarias de las vulnerabilidades de la seguridad del software y correlaciona y prioriza los resultados (HP, 2014).
- **MCAfee Vulnerability Manager:** Realiza monitorización activa y pasiva, además de realizar pruebas de penetración. Permite conocer los puntos en los que se debe centrar los esfuerzos de programación. Cubre las categorías de OWASP top 10 y CWE-25 (McAfee, 2014).
- **Nessus Vulnerability Scanner:** Permite realizar escaneo de vulnerabilidades en servidores web, servicios web, además de las vulnerabilidades de OWASP. Además de verificar la configuración erróneas del sistema y parches faltantes. Muestra informes personalizados en formato XML, CVS, PDF nativo y HTML (Tenable, 2014).
- **Nexpose Vulnerability Manager:** Es una solución de gestión de vulnerabilidades que combina la evaluación de vulnerabilidades y controles, la validación de vulnerabilidades y la planificación de remediación. Maneja estándares de riesgo, vulnerabilidades y gestión de

la configuración como PCI DSS, NERC CIP, FISMA, entre otros (Rapid7, 2014).

- **Whatweb:** Identifica el sitio web, reconoce tecnologías web, incluyendo los sistemas de gestión de contenidos (CMS por sus siglas en inglés), plataformas de blog, bibliotecas de JavaScript, servidores web. También identifica los números de versiones de correo electrónico, errores de SQL y más (MorningStartSecurity, 2014).

4. Propuesta de Trazabilidad

Finalmente con respeto a los resultados obtenidos se realizó una matriz de trazabilidad de las herramientas utilizadas para la detección de vulnerabilidades con las técnicas, ataques y vulnerabilidades existentes: esto con el objetivo de mostrar cuales ataques y vulnerabilidades son cubiertos con que técnica y herramienta para ser mitigados. A continuación se muestra la matriz de trazabilidad en la Tabla 2.

Tabla 2. **Matriz de trazabilidad de ataques, vulnerabilidades, técnicas y herramientas**

Ataque	**Vulnerabilidad**	**Técnica para detección de vulnerabilidades**	**Herramienta para detección de vulnerabilidades**
Inyección SQL	Inyección	Análisis estático de código	QualysGuard Web Application Scanning WAS
		Análisis dinámico de código	WebSite Security Audit- WSSA
		Pruebas de penetración	WEBAPP 360: Enterprise Class web application scanning
			Retina Web Security Scanner
	Perdida de autenticación y		QualysGuard Web Application Scanning WAS

Ataque de fijación de sesiones	manejo de sesión	Utilización de estándar de manejo de sesiones	WebSite Security Audit- WSSA
			Retina Web Security Scanner
			WEBAPP 360: Enterprise Class web application scanning
Ataque XSS	Secuencia de comandos en sitios cruzados XSS	Análisis estático de código	QualysGuard Web Application Scanning WAS
		Pruebas de penetración	WebSite Security Audit- WSSA
			Retina Web Security Scanner
			WEBAPP 360: Enterprise Class web application scanning
Referencia directa insegura a objetos	Referencia directa insegura a objetos	Análisis estático de código	WEBAPP 360: Enterprise Class web application scanning
			Frame-C
			Parasoft C/C++ Test
			ITS4
			SCA
Configuración de seguridad incorrecta	Configuración de seguridad incorrecta	Pruebas de penetración	MCAfee Vulnerability Manager
			QualysGuard Web Application Scanning WAS
			Nessus Vulnerability Scanner
			Nexpose Vulnerability Manager
			Retina Web Security Scanner
Ataque "Man in the middle"	Exposición a datos sensibles		Nessus Vulnerability Scanner
			Retina Web Security Scanner

Control de acceso a nivel de funcionalidades	Inexistente control de acceso a nivel de funcionalidades	Pruebas de proxy	SCA
		Análisis estático de código sobre el control de acceso	Parasoft C/C++ Test
Ataque de falsificación de peticiones en sitios cruzados (CSRF)	Falsificación de peticiones en sitios cruzados (CSRF)	Análisis estático de código	QualysGuard Web Application Scanning WAS
Uso de componentes con vulnerabilidades conocidas	Uso de componentes con vulnerabilidades conocidas		Whatweb
Phising	Redirección y reenvíos no válidos	Análisis estático de código	SCA
		Análisis estático de código	

Como puede observarse, tras el análisis de la propuesta mostrada en la Tabla 2, la herramienta más utilizada para detección de vulnerabilidades es QualysGuard Web Application Scanning WAS, seguido de Retina Web Security Scanner y WEBAPP 360: Enterprise Class web application scanning. Así también se puede determinar que las menos utilizadas son Frame-C, Nexpose, esto debido a que son herramientas para funcionalidades más específicas.

5. Conclusiones y trabajos futuros

Después de la ejecución de la revisión sistemática y del análisis de los resultados acerca de herramientas y técnicas utilizadas para la detección de

vulnerabilidades, se concluye que existen muchas herramientas que proporcionan la detección para diferentes propósitos, es decir, algunas herramientas cubren desde escaneo de vulnerabilidades en aplicaciones web, hasta escaneo de vulnerabilidades en dispositivos móviles, un ejemplo de este tipo de herramientas es Nessus Vulnerability Scanner (Tenable, 2014), además de muchas otras funcionalidades.

De igual manera existen herramientas muy específicas para la detección de problemas de seguridad muy específica, como por ejemplo WhatWeb(MorningStartSecurity, 2014) en el que solamente se enfoca en el escaneo de sitios web.

Sin embargo, aún con la existencia de las herramientas antes mencionadas las organizaciones continúan con un desconocimiento de cuándo deben ser utilizadas, por lo tanto, la matriz de trazabilidad propuesta se considera una aportación muy importante para las organizaciones ya que proporciona una guía para la utilización de técnicas y herramientas sobre una vulnerabilidad en específico, que permitirá prevenir un tipo de ataque.

Como trabajo futuro se plantea realizar un estudio con la finalidad de establecer una categorización de las diferentes herramientas existentes dependiendo del tipo de dominio. También se realizará un estudio en las organizaciones locales sobre el uso de herramientas que les permiten detectar vulnerabilidades. Además de desarrollar una herramienta que permita detectar vulnerabilidades en aplicaciones web, de acuerdo a la información que se obtendrá de los estudios realizados.

Referencias

BeyondSecurity. (2014). Web Site Security Audit - WSSA by Beyond Security. Retrieved December 19, 2014, from http://www.beyondsecurity.com/vulnerability-scanner.html

Beyontrust. (2014). Web Vulnerability Management Software | Assessment Software. Retrieved December 19, 2014, from http://www.beyondtrust.com/Products/RetinaWebSecurityScanner/

Frama-C. (2014). Frama-C. Retrieved December 19, 2014, from http://frama-c.com/what_is.html

HP. (2014). análisis estáticos, prueba de seguridad de aplicaciones estáticas, SAST | HP® México. Retrieved December 20, 2014, from http://www8.hp.com/mx/es/software-solutions/software.html?compURI=1338812#.VJS9SF4AM

Jorgensen, M., & Shepperd, M. (2007). A systematic review of software development cost estimation studies. Software Engineering, IEEE …, 33(1), 33–53. Retrieved from http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=4027147

Kitchenham, B. (2004). Evidence-based software engineering. Software Engineering. Retrieved from http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=1317449

Landau, L. (2013). OWASP Top 10 - 2013 Los diez riesgos más críticos en Aplicaciones Web. Zhurnal Eksperimental'noi i Teoreticheskoi Fiziki. Retrieved from http://scholar.google.com/scholar?hl=en&btnG=Search&q=intitle:No+Title#0

Mammar, A., Cavalli, A., & Jimenez, W. (2011). Using testing techniques for vulnerability detection in C programs. Testing Software and …, 80–96. Retrieved from http://link.springer.com/chapter/10.1007/978-3-642-24580-0_7

McAfee. (2014). McAfee Vulnerability Manager | Soluciones de McAfee. Retrieved December 19, 2014, from http://www.mcafee.com/mx/products/vulnerability-manager.aspx

Miranda, J. M., Muñoz, M., Uribe, E., Márquez, J., Uribe, G., & Valtierra, C. (2014). New Perspectives in Information Systems and Technologies, Volume 1, 275, 171–181. doi:10.1007/978-3-319-05951-8

MorningStartSecurity. (2014). WhatWeb. Retrieved December 19, 2014, from http://www.morningstarsecurity.com/research/whatweb

OSVDB. (2014). OSVDB: Open Sourced Vulnerability Database. Retrieved December 07, 2014, from http://osvdb.org/

Parasoft. (2014). Static Analysis, static code analysis | Parasoft static analysis tools. Retrieved December 20, 2014, from http://www.parasoft.com/static-analysis

Qualys. (2014). Qualys Web Application Scanning (WAS) | Qualys, Inc. Retrieved December 19, 2014, from https://www.qualys.com/enterprises/qualysguard/web-application-scanning/

Rapid7. (2014). Vulnerability Management & Risk Management Software | Rapid7. Retrieved December 19, 2014, from http://www.rapid7.com/products/nexpose/

Sreenivasa, R., & Kuman, N. (2012). International Journal of Enterprise Computing and Business Systems ISSN (Online) : 2230-8849 WEB APPLICATION VULNERABILITY DETECTION USING DYNAMIC ANALYSIS International Journal of Enterprise Computing and Business Systems ISSN (Online) : 2230-8849, 2(1).

Tenable. (2014). Nessus. Retrieved December 19, 2014, from http://www.tenable.com/products/nessus

Tripwire. (2014). Tripwire WebApp 360 | Vulnerability Management | Tripwire. Retrieved December 20, 2014, from http://www.tripwire.com/it-security-software/enterprise-vulnerability-management/web-application-vulnerability-scanning/

UNAM-CERT. (2014). Estadísticas - UNAM-CERT -. Retrieved December 20, 2014, from http://www.cert.org.mx/estadisticas.dsc

Notas biográficas:

Ana Laura Hernández Saucedo Ingeniera en Computación, egresado de la Universidad Autónoma de Zacatecas (UAZ), actualmente estudia la Maestría en Ingeniería de Software en el Centro de Investigación en Matemáticas (CIMAT) Unidad Zacatecas. Su interés es seguridad informática, así como la utilización de técnicas y herramientas que permitan detectar vulnerabilidades en aplicaciones web.

Jezreel Mejia Miranda Doctor en Informática por la Universidad Politécnica de Madrid (UPM), España con mención de "Doctorado Europeo". Realizó una estancia de investigación para obtener el doctorado europeo en la Universidad Fernando Pessoa en Oporto, Portugal. Previamente, en el Instituto Tecnológico de Orizaba, Veracruz, cursó la maestría en Ciencias de la Computación y la licenciatura en Informática. Actualmente es investigador del Centro de Investigación en Matemáticas, A.C. (Cimat), Unidad Zacatecas, en el área de Ingeniería de Software. Es miembro del grupo de investigación Cátedra de Mejora de Procesos Software en el Espacio Iberoamericano (MPSEI), donde participa en proyectos internacionales de investigación con entidades educativas y de vinculación con la industria. Es miembro del comité científico de diversos congresos. Ha publicado diversos artículos técnicos en temas relacionados con la gestión de proyectos, entornos multi-modelo, modelos y estándares de calidad y temas relacionados en entornos outsourcing. También ha participado en proyectos de la empresa multinacional everis consulting. Además, forma parte del equipo oficial de traducción al español del libro CMMI-DEV v1.2 y 1.3, versiones reconocidas por el prestigioso Software Engineering Institute (SEI) de la Carnegie Mellon University. Como investigador, sus áreas de interés son: entornos multi-modelo, gestión de proyectos software, modelos y estándares de calidad (CMMI, ISO, TSP, PSP, etc.), metodologías ágiles, métricas, mejora de procesos en entornos

outsourcing y entornos de desarrollo tradicional. Cuenta con certificación en CMMI e ISO 20000.