Capacidades de Detección de las Herramientas de Análisis de Vulnerabilidades en Aplicaciones Web

Fernando Román Muñoz, Iván Israel Sabido Cortés, Luis Javier García Villalba

Grupo de Análisis, Seguridad y Sistemas (GASS), Departamento de Ingeniería del Software e Inteligencia Artificial
Facultad de Informática, Despacho 431, Universidad Complutense de Madrid (UCM)
Calle Profesor José García Santesmases, 9, Ciudad Universitaria, 28040 Madrid
Email: {froman, javiergv}@fdi.ucm.es, isc_86@hotmail.com

***Resumen*—Debido al continuo incremento del número de vulnerabilidades en las aplicaciones Web, se han elaborado diversas clasificaciones para mantener organizadas estas vulnerabilidades, y también se han desarrollado herramientas para detectarlas. Hasta este momento, según nuestra información, no se ha realizado ningún estudio sobre las capacidades de estas herramientas en la detección de las vulnerabilidades presentes en las clasificaciones de vulnerabilidades. En este trabajo mapeamos y agrupamos las diferentes clasificaciones para obtener un conjunto lo más completo posible de vulnerabilidades web, y comprobamos si las herramientas mejor valoradas disponen de las características necesarias para detectarlas. Después introducimos la aplicación web vulnerable que hemos desarrollado, intentando incorporar todas las vulnerabilidades web de nuestra lista, con el objetivo de comprobar las capacidades reales de detección de las herramientas de análisis de vulnerabilidades web.**

Palabras clave*—Clasificaciones de vulnerabilidades, precisión de las herramientas de análisis de vulnerabilidades web, vulnerabilidades web. *(Vulnerability classifications, web vulnerability scanners accuracy, web vulnerabilities).

I. Introducción

La opción más habitual para detectar las vulnerabilidades de una aplicación Web es utilizar una herramientas automática de análisis. A estas herramientas se les proporciona un conjunto de URLs y eventualmente unas credenciales, con las que explorar la aplicación. Una vez que la ha recorrido, intentará introducir determinados valores en los campos y cabeceras, para posteriormente analizar el resultado en busca de evidencias de vulnerabilidades.

Para saber que herramienta de las existentes es mejor en la detección de vulnerabilidades se han realizado varios estudios. En ellos básicamente se parte de un conjunto de herramientas comerciales o de código libre, un conjunto de vulnerabilidades a detectar, y una aplicación con esas vulnerabilidades. Se configuran las herramientas para analizar la aplicación vulnerable y se analizan los resultados. El comparar unos estudios con otros es difícil ya que en cada uno de ellos las herramientas analizadas, el conjunto de vulnerabilidades y la aplicación vulnerable son diferentes.

El conjunto de herramientas disponibles cambia a lo largo del tiempo, por lo que no se puede definir un conjunto estático de herramientas. Pero lo que sí existen son clasificaciones de vulnerabilidades Web que incluyen las categorías existentes de vulnerabilidades. Estas clasificaciones aunque se revisan periódicamente, no están en constante actualización.

En este documento los autores describen el proceso seguido para unificar las clasificaciones disponibles de vulnerabilidades Web, agrupádolas en una única lista. A continuación se revisan las capacidades de detección de las herramientas mejor valoradas frente a las vulnerabilidades de esa lista.

Posteriormente se describe el proceso seguido para desarrollar una aplicación Web vulnerable, intentando que incluya todas las categorías de vulnerabilidades web. Esta aplicación será usada en el futuro para comprobar las capacidades reales de detección de las herramientas de análisis, y para realizar acciones de formación en detección de vulnerabilidades.

II. Antecedentes

Esta sección proporciona información sobre los diferentes conceptos relacionados con las vulnerabilidades y las clasificaciones existentes actualmente.

II-A. Conceptos

Aunque algunas listas de vulnerabilidades clasifican según el concepto de vulnerabilidad, otras lo hacen según otros conceptos como: amenaza, debilidad, riesgo o control. El utilizar diferentes conceptos no impide que puedan compararse las clasificaciones, ya que todos ellos están relacionados, como puede verse en la guía NIST Special Publication 800-30 [1] y en el estándar internaciónal ISO/IEC 27001:2005 [2]. Por motivos de claridad en este documento nos referiremos a todos ellos como vulnerabilidad.

II-B. Herramientas de análisis de vulnerabilidades Web

En un artículo anterior [31] se agrupaban las principales carencias de las herramientas de análisis de vulnerabilidades Web, y se proponían soluciones para varias de ellas. Posteriormente en [4] se proponían mejoras de las soluciones indicadas en ese artículo anterior. En otro artículo anterior [5], se analizaban varios estudios relevantes sobre estas herramientas, tanto comerciales como de código libre. Los resultados muestran las herramientas, vulnerabilidades y aplicaciones vulnerables que se usaron en cada estudio. En lo referente a las herramientas analizadas, aunque en cada estudio se comparan al menos siete herramientas, sólo dos de ellas se valoran en todos los

ISBN: 978-84-9717-323-0

estudios, y muchas solamente en uno. En lo que respecta a las vulnerabilidades, entre todas se prueban 35, pero sólo las dos más conocidas se pruebas en todos ellos: Cross Side Scripting e inyección SQL. Sobre la aplicación vulnerable utilizada, ninguna se usa en más de un artículo, en algunas se usan aplicaciones desarrolladas a propósito para sus pruebas, y en otras aplicaciones de uso habitual con vulnerabilidades conocidas. Finalmente, en los resultados, ninguna herramienta aparece en las tres primeras posiciones de todos los artículos, y hay muchas de ellas que aparecen en las primeras posiciones de unos y en las últimas de otros. Agrupando los resultados de estos estudios se obtiene una jerarquía de herramientas, en la que que aparecen las siguientes en los primeros puestos: Appscan [20], Acunetix [21], Webinspect [22] y Burp Suite [23].

II-C. Clasificaciones de vulnerabilidades

En esta sección se indican las clasificaciones de vulnerabilidades más relevantes hasta el momento de la elaboración de este documento. Se incluyen clasificaciones tanto de vulnerabilidades en aplicaciones Web, como otro tipo de vulnerabilidades.

La organización Web Application Security Consortium (WASC) proporciona una clasificación de 49 amenazas en aplicaciones Web (WASC TC) [6]. Divide las amenazas entre debilidades y ataques, e incluye la fuente de las amenazas: diseño, implementación o desarrollo. Contiene descripciones y ejemplos. La última versión es la 2.0 liberada en 2010.

La organización Open Web Application Security Project (OWASP) mantiene actualizada la lista OWASP Top 10 [7], que incluye los riesgos de Seguridad más críticos en las aplicaciones Web. Obtiene sus datos de empresas de consultoría y de fabricantes de herramientas de detección de vulnerabilidades. La versión actual es del 2013. Esta lista incluye la descripción y ejemplos de cada vulnerabilidad, y también métodos para mitigar su impacto. OWASP también desarrolla la Guía de Pruebas de OWASP [8] que describe un conjunto de pruebas que se pueden realizar sobre las aplicaciones Web para detectar vulnerabilidades. Contiene 66 pruebas, y la versión actual es la v3 del 2008. Incluye la descripción de las vulnerabilidades, ejemplos y métodos de detección. No se indican métodos para mitigar su impacto, pero sí incluye referencia a otra información de interés de OWASP. La siguiente versión v4 actualmente está en desarrollo.

Del proyecto NIST SAMATE surge en 2007 la guía NIST Special Publication 500-269 [9] que describe las tareas que debe realizar una herramientas de detección de vulnerabilidades Web. En su anexo A se incluye una lista de 14 vulnerabilidades Web que una de estas herramientas debe ser capaz identificar. Las vulnerabilidades de esta lista se han incluido según su probabilidad de ser explotadas. En el anexo B se suguieren brevemente métodos para mitigarlas.

En 2009 la organización WASC también elabora el Web Application Security Scanner Evaluation Criteria (WASSEC) [10], que incluye una lista de problemas de Seguridad que una herramienta de análisis de aplicaciones Web debe detectar. Los extrae principalmente de WASC TC, e incluye 55 problemas agrupados en varias categorías: autenticación, autorización, ataque del lado del cliente, ejecución de comandos y revelación de información.

La clasificación Common Weakness Enumeration (CWE) [11] desarrollada por la Corporación MITRE, es un conjunto de debilidades software en todo tipo de software, no sólo aplicaciones Web. La versión existente en el momento de este documento, la 2.5 de 2013, incluía 940 debilidades. También se citan ejemplos, métodos de detección y de mitigación, y referencias a otras clasificaciones de vulnerabilidades.

SANS y MITRE han desarrollado la clasificación CWE/SANS TOP 25 de los errores software más peligrosos (SANS CWE/25) [12], que fue actualizada por última vez en 2011. Es un subconjunto de CWE e igualmente incluye vulnerabilidades de todo tipo de aplicaciones. Incluye ejemplos y métodos de detección.

Otra clasificación que incluye vulnerabilidades en todo tipo de software es Common Attack Patterns Enumeration and Classification (CAPEC) [3] que mantiene la corporación MITRE. Contiene patrones de ataque y la última version durante la elaboración de este documento es la 2.1 de 2013. Incluye ejemplos y descripciones de flujos de ataque.

Por ultimo Shay Chen mantiene la clasificación SecToolMarket [13] donde se incluye una clasificación de 33 características de auditoria de las herramientas de análisis de aplicaciones web. Se actualiza normalmente cada año e incluye referencias a otras clasificaciones. En Sectoolmarket también se analizan 62 herramientas, para determinar cuales de esas 33 características de auditoría detecta cada una.

II-D. Relaciones entre clasificaciones

En el apartado anterior se indicaban las principales clasificaciones de vulnerabilidades. Para relacionar las vulnerabilidades de unas clasificaciones con otras se han realizado varios trabajos de mapeo. Suelen incluir todas las vulnerabilidades de una clasificación, relacionando las que sean posibles con vulnerabilidades de otras clasificaciones. A continuación se ofrece una breve descripción de los principales mapeos entre clasificaciones.

Denim Group [14] realizó en 2010 un mapeo entre las vulnerabilidades de WASC TC v1.0, SANS CWE/25, y OWASP Top 10 2004 y 2007.

Jeremiah Grossman [15] mapeó en 2009 las clasificaciones WASC TC v2.0 y OWASP Top 10 2010.

Threat Classification Taxonomy Cross Reference View (Webappsec) [16] es una vista de la clasificación WASC actualizada en 2013, que relaciona las vulnerabilidades en WASC TC v2.0 con las de CWE, CAPEC, OWASP Top Ten (2004, 2007 y 2010) y SANS CWE/25. Para ello usan la información de los dos mapeos indicados antes: Denim Group y Jeremiah Grossmans.

La clasificación que incorpora la guía NIST Special Publication 500-269 en su anexo A incluye un mapeo con CWE, OWASP Top 10 2007 y Common Vulnerabilities and Exposures (CVE) [17].

Securing Telligent Evolution [18] es un documento creado por Telligent en 2012, que describe las amenazas que se prueban en la plataforma Telligent Evolution, incluyendo un mapeo entre la guía de pruebas de OWASP y WASC.

Finalmente SecToolMarket incluye relaciones entre las vulnerabilidades de WASC TC v2.0 y la guía de pruebas de OWASP.

La información de todos estos mapeos se puede resumir para determinar qué clasificaciones están más relacionadas. La clasificación WASC TC v2.0 está relacionada con otras 8 en los mapeos analizados, SANS CWE/25 con 5 clasificaciones, OWASP Top 10 2007, OWASP Top 10 2010 y WASC TC v1.0 con tres, y la guía de pruebas de OWASP, CWE, CVE, y CAPEC sólo con otra clasificación.

II-E. Aplicaciones Web vulnerables

Para probar las herramientas existen multitud de aplicaciones Web vulnerables a propósito, algunas de ellas desarrolladas por organizaciones dedicadas a la seguridad, y otras por los fabricantes de las herramientas de detección. Dentro del primer grupo las más relevantes son las siguientes.

Damn Vulnerable Web Application (DVWA) [28] es una aplicación desarrollada en PHP, con vulnerabilidades basadas en la clasificación OWASP Top 10, que ofrece la opción de cambiar el nivel de seguridad. WebGoat [29] es mantenida por OWASP y está basada en tecnología J2EE. Esta aplicación cuenta con más de 30 lecciones para aprender a detectar las vulnerabilidades y corregirlas. OWASP también mantiene Mutillidae [30], desarrollada en PHP, y como DVWA, se basa en la clasificación OWASP Top 10. También tiene distintos niveles de dificultad e incluye información para detectar las vulnerabilidades. Dentro del grupo de aplicaciones vulnerables desarrolladas por los fabricantes de herramientas de detección podemos indicar, a modo de ejemplo, la aplicación vulnerable de Acunetix [27] desarrollada en PHP, o la de IBM AppScan [26] desarrollada en ASP.NET.

En esta sección se han introducidos los conceptos relacionados con las vulnerabilidades, las principales clasificaciones y la relaciones entre ellas, y las aplicaciones vulnerables. En la siguiente sección se explica el proceso seguido para unificar las clasificaciones.

III. Clasificación de vulnerabilidades Web unificada

A partir de las principales clasificaciones de vulnerabilidades y los mapeos entre ellas se puede desarrollar una nueva clasificación que incluya los tipos de vulnerabilidades de todas ellas, sin vulnerabilidades redundantes. Para ello como primer paso nos quedarnos con las clasificaciones completas que sólo incluyen vulnerabilidades Web, es decir no se tienen en cuenta las clasificaciones que incluyen las vulnerabilidades de todos los tipos de aplicaciones, como CWE, ni las que sólo incorporan las vulnerabilidades más relevantes o frecuentes, como hace OWASP Top 10. El resultado son las clasificaciones WASC TC, la guía de pruebas de OWASP y la clasificación de Sectoolmarket. Los mapeos entre ellas son el de Telligent y el de Sectoolmarket. El segundo paso es unificar los mapeos de Telligent y de Sectoolmarket eliminando la información redundante. De esta forma se obtiene un listado de 29 mapeos entre elementos de WASC TC y la guía de pruebas de OWASP. En tercer lugar las vulnerabilidades restantes de cada una de estas clasificaciones se revisan teniendo en cuenta sus descripciones. De esta forma se obtienen 12 nuevas relaciones entre vulnerabilidades de WASC TC y la guía de pruebas de OWASP. Por último se obtiene la clasificación final añadiendo las vulnerabilidades sin relacionar en WASC TC, la guía de pruebas de OWASP y la clasificación de Sectoolmarket. De WASC TC se obtienen ocho, nueve de la guía de pruebas de OWASP, y cinco de la clasificación de Sectoolmarket. Al final tenemos una clasificación de 63 tipos de vulnerabilidades. Esta nueva clasificación (ULWeV) y los mapeos entre las clasificaciones seleccionadas pueden consultase en [19].

En la tabla I puede verse las vulnerabilidades de la nueva clasificación obtenida agrupadas según su procedencia: WASC TC, guía de pruebas OWASP que no están en WASC TC, y Sectoolmarket que no están en las dos anteriores.

Tabla I
Origen de las vulnerabilidad en la nueva clasificación

Clasificación	Vulnerabilidades
WASC TC	Todas (de WASC-01 a WASC-49)
Guía de OWASP	-Information Gathering -User enumeration -Weak Multiple Factors Authentication -Race Conditions vulnerability -DB Listener weak -Code Injection -Cookie attributes -Exposed sensitive session variables -Web services testing
Sectoolmarket	-EL Injection -Padding Oracle -Server Side Java Script (SSJS/NoSQL) Injection -Unrestricted File Upload and Blind/Time-Based SQL Injection -Source Code Disclosure

IV. Precisión de las herramientas de detección de vulnerabilidades Web

En la sección 2 se indicaban los conceptos relacionados con las vulnerabilidades, las herramientas de detección de vulnerabilidades mejor valoradas, las clasificaciones actuales de vulnerabilidades, y las relaciones entre ellas. En la sección 3 se describía el proceso seguido para unificar estas clasificaciones, y obtener una clasificación de vulnerabilidades lo más completa posible. En esta sección se usará esa nueva clasificación de vulnerabilidades para comprobar las características (capacidades de detección) de estas herramientas.

IV-A. Capacidades de detección de las herramientas

Para revisar las capacidades de detección de las herramientas se ha podido conseguir versiones válidas de AppScan, Webinspect y Acunetix. De Burp Suite no se ha podido

conseguir una versión válida completa, por lo que se ha sustituido por Zaproxy [24].

La clasificación ULWeV tiene 63 vulnerabilidades, que se puede agrupar en tres grupos, como se ha visto anteriormente en la tabla I: (1) 49 de WASC TCv2.0, (2) nueve de la guía de pruebas OWASP que no están en WASC TC v2.0, y (3) cinco de la clasificación de Sectoolmarket que no están en las otras dos. Para saber si una herramientas tiene capacidad de detectarlas (incorpora la característica necesaria) se usan varias fuentes de información: el sistema de gestión de vulnerabilidades Threadfix [25] junto con el mapeo entre clasificaciones Webappsec, la información en Sectoolmarket, y finalmente la revisión manual de las características de las herramientas. Threadfix es un Sistema de gestión de vulnerabilidades capaz de consolidar informes de vulnerabilidades provenientes de Acunetix, Appscan, Webinspect, Zaproxy y otras herramientas, relacionando las vulnerabilidades detectadas con CWE.

Como se ha visto en un apartado anterior Webappsec relaciona vulnerabilidades en WASC TC v2.0 con las de CWE. A partir de esta información y la que proporciona Threadfix se pueden determinar las vulnerabilidades de WASC TC v2.0 que detecta cada una de las herramientas seleccionadas, lo que se corresponde con el primer grupo (1) de vulnerabilidades de la nueva clasificación ULWeV. Como también se ha visto en un apartado anterior en Sectoolmarket se analizan 62 herramientas para ver si detectan las vulnerabilidades de su clasificación. De aquí se obtiene si las cinco vulnerabilidades del grupo (3) las detectan las herramientas. Para el grupo (2) de las nueve vulnerabilidades de la guía de pruebas de OWASP, y las que no ha sido posible localizar en los dos pasos anteriores, se han revisado manualmente para comprobar si las herramientas tienen capacidad de detectarlas.

En la tabla II se indica el número de pruebas que puede realizar cada herramienta analizada, así como las que se corresponden con las de ULWeV según su clasificación de origen.

Tabla II
CAPACIDADES DE DETECCIÓN DE LAS HERRAMIENTAS

	Acunetix	Appscan	Webinspect	Zaproxy	Las 4 herramientas
Número aproximado de pruebas	350	2000	4300	49	-
63 vulnerabilidades en ULWeV:					
49 de WASC TC v2.0	28	31	27	49	49
8 de de la guía de OWASP	5	4	3	0	5
5 de Sectoolmarket	5	23	3	0	5
Total	38	37	33	49	59

IV-B. Aplicación Web vulnerable

En apartados anteriores se ha elaborado una clasificación unificada de posibles vulnerabilidades en aplicaciones Web, y después se han revisado las capacidades de las herramientas para detectar esas vulnerabilidades. De esta forma se tiene un conjunto actualizado de vulnerabilidades que deberían de detectar las herramientas, y las que podrían detectar cada una. Pero para poder determinar si realmente una herramientas es capaz de detectar una vulnerabilidad hay que probarla.

Para determinar las capacidades reales de detección de una herramienta de análisis se hace necesario disponer de aplicaciones que tengan esas vulnerabilidades. Para ello se han analizado las aplicaciones vulnerables existentes, de forma que se pueda seleccionar una o varias de ellas, que incorporen el mayor número de vulnerabilidades de la nueva clasificación. El resultado es que DVWA tiene 17 vulnerabilidades de las 63 de la clasificación ULWeV, WebGoat tiene 32, Mutillidae 34, la aplicación vulnerable de Acunetix cuenta con 18 vulnerabilidades de ULWeV, y la de AppScan cuenta con 14. Tomando las que más vulnerabilidades incorpora, con WebGoat y Mutillidae se cubre un total de 39 vulnerabilidades de las 63. Las 14 restantes se analizan para determinar si es posible incorporarlas en alguna de las aplicaciones seleccionadas.

Con la nueva clasificación de vulnerabilidades, y la aplicación, o conjunto de aplicaciones, vulnerables a ellas, se pueden realizar acciones de formación sobre los futuros programadores, de forma que es cubran dos objetivos: enseñar a detectar las vulnerabilidades, para realizar pruebas de penetración sobre las aplicaciones Web; y enseñar a programar de forma segura, evitando en la medida de lo posible desarrollar aplicaciones con vulnerabilidades. En la figura 1 se explica el proceso que se intenta seguir en este documento para mitigar, usando formación y concienciación, el desarrollo de aplicaciones Web con vulnerabilidades.

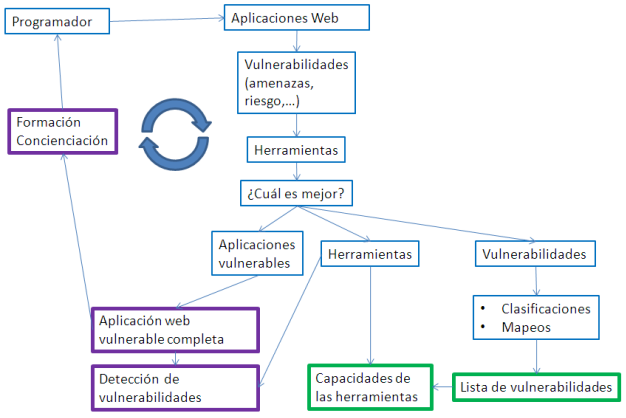

Figura 1. Formación para detectar y mitigar las vulnerabilidades Web

V. ANÁLISIS DE LOS RESULTADOS

Como se ve ninguna de las herramientas analizadas incorpora las capacidades de detección necesarias para detectar las 63

vulnerabilidades, aunque entre las cuatro analizadas sí cubren casi toda la lista de vulnerabilidades.

Aunque Appscan y Webinspect incluyen muchas pruebas de vulnerabilidades, según el análisis realizado son las otras dos Zaproxy y Acunetix las que detectarían más vulnerabilidades deULWeV. Esto es debido a que Appscan y Webinspect incorporan muchas vulnerabilidades de aplicaciones Web específicas y no tanto de tipos de vulnerabilidades. Por ejemplo en las versiones de las herramientas analizadas, Zaproxy incluye solo una característica para detectar Cross-Site Scripting, Acunetix siete, Webinspect más de 500, y Appscan más de 600. En el caso de estas dos últimas, Webinspect y Appscan, solo unas pocas son para detectar la vulnerabilidad en cualquier aplicación, por ejemplo "URL Cross-Site Scripting" la mayoría de ellas son para productos concretos, como "WebSphere Cross-Site Scripting" o "ASP Nuke Cross-Site Scripting Vulnerability". Tanto Sectoolmarket como Threadfix relacionan todas estas vulnerabilidades con el tipo genérico "Cross-Site Scripting". Esto lleva a la conclusión de que herramientas como AppScan o Webinspect son las más adecuadas para analizar aplicaciones cerradas, ya sean comerciales o de software libre, pero que otras herramientas como Acunetix o gratuitas como Zapproxy, deberían de ser al menos igualmente adecuadas para analizar aplicaciones web desarrolladas a medida.

VI. Conclusiones y trabajo futuro

En este artículo se muestra el resultado de unificar las clasificaciones de vulnerabilidades Web más relevantes para obtener una única lista. También se analizan varias de las herramientas mejor valoradas de detección de vulnerabilidades Web, para determinar si incorporan las características de detección necesarias para detectar las vulnerabilidades que incluye la nueva lista. Finalmente se explica la selección de un conjunto de aplicaciones Web vulnerables, y su mejora con vulnerabilidades adicionales, intentando obtener una aplicación que incorpore todas las vulnerabilidades de las clasificaciones. Aunque esta nueva clasificación pueda crecer con el tiempo, sí sirve para determinar que les falta a las herramientas, en que deben mejorar, al dar una visión global de que deben detectar y sus capacidades actuales de detección.

El siguiente paso será analizar estas aplicaciones vulnerables con las herramientas seleccionadas para determinar sus capacidades reales de detección. También esta aplicación vulnerable se podrá utilizar para realizar acciones de formación en desarrollo seguro, al partir de una clasificación completa de tipos de vulnerabilidades, y no solo las más comunes.

Referencias

[1] National Institute of Standards and Technology, "NIST Special Publication 800-30 Revision 1: Guide for Conducting Risk Assessments", 2012.

[2] International Organization for Standardization, "International Standard ISO/IEC 27001", 2005.

[3] The MITRE Corporation, "Common attack patterns Enumeration and Classfication", 2013. Disponible en http://capec.mitre.org/.

[4] F. Román Muñoz, L. J. García Villalba, "Web From Preprocessor for Crawling", en *Multimedia Tools and Applications*, p.p. 1–12, April 2013.

[5] F. Román Muñoz, L. J. García Villalba, "Methods to Test Web Applications Scanners", *Proceedings of the 6th International Conference on Information Technology*, 2013.

[6] Web Application Security Consortium, "The WASC Threat Classification", 2010.

[7] Open Web Application Security Project, "OWASP Top Ten Project", 2013. Disponible en https://code.google.com/p/owasptop10.

[8] Open Web Application Security Project, "OWASP Testing Guide", 2013.

[9] National Institute of Standards and Technology, "Software Assurance Tools: Web Application Security Scanner Functional Specification Version 1.0. NIST Special Publication 500-269", 2008.

[10] Web Application Security Consortium, "Web Application Security Scanner Evaluation Criteria", 2009.

[11] The MITRE Corporation, "Common Weakness Enumeration", 2013. Disponible en http://cwe.mitre.org.

[12] SANS, "CWE/SANS TOP 25 Most Dangerous Software Errors", 2011. Disponible en http://www.sans.org/top25-software-errors.

[13] S. Chen, "General Features Comparison - Web Application Scanners", 2012. Disponible en http://www.sectoolmarket.com.

[14] D. Cornel, "Mapping Between OWASP Top 10 (2004, 2007), WASC 24+2 and SANS CWE/25", 2010.

[15] J. Grossman, "WASC Threat Classification to OWASP Top Ten RC1 Mapping" 2013.

[16] Web Application Security Consortium, "Threat Classification Taxonomy Cross Reference View", 2012.

[17] The MITRE Corporation, "CVE", 2013. Disponible en http://cve.mitre.org.

[18] Telligent, "Telligent Evolution Platform", 2013.

[19] F. Roman, l.-J. Garcia, "Web vulnerability classification mapping", 2013.

[20] IBM, "IBM Security Appscan", 2014. Disponible en http://www-03.ibm.com/software/products/us/en/appscan.

[21] Acunetix, "Acunetix Web Vulnerbility Scanner", 2014. Disponible en http://www.acunetix.com.

[22] HP, "HP Webinspect", 2014.

[23] Portswigger, "Burp Suite",2014. Disponible en http://portswigger.net/burp.

[24] Open Web Application Security Project, "Zaproxy", 2014.Disponible en http://code.google.com/p/zaproxy/.

[25] Denim Group, "Threadfix", 2013. Disponible en https://code.google.com/p/threadfix.

[26] IBM, "IBM Security Appscan", 2014. Disponible en http://demo.testfire.net.

[27] Acunetix, "TEST and Demonstration site for Acunetix Web Vulnerability Scanner", 2014. Disponible en http://testphp.vulnweb.com.

[28] DVWA, "Damn Vulnerable Web Application", 2014. Disponible en http://www.dvwa.co.uk/.

[29] WebGoat, "OWASP WebGoat Project", 2014. Disponible en https://www.owasp.org/index.php/Category:OWASP_WebGoat_Project/.

[30] Mutillidae, "OWASP Mutillidae 2 Project", 2014. Disponible en https://www.owasp.org/index.php/OWASP_Mutillidae_2_Project.

[31] F. Román Muñoz, L. J. García Villalba, "Preproceso de Formularios para el Análisis de Seguridad de las Aplicaciones Web", Actas de la XII Reunión Española sobre Criptología y Seguridad de la Información (RECSI 2012), Donostia-San Sebastián, España, Septiembre 2012.