DOI: http://dx.doi.org/10.21501/21454086.1957

Lámpsakos | No.17 | pp. 84-91 | enero-junio | 2017 | ISSN: 2145-4086 | Medellín–Colombia

Pruebas de seguridad: estudio de herramientas

Testing security: studies tools

Vianca Vega, PhD.
Universidad Católica del Note
Antofagasta, Chile
vega@ucn.cl

Yahima Hadfeg, MSc.
Universidad Católica del Note
Antofagasta, Chile
yahima.hadfeg01@ucn.cl

(Recibido el 18-05-2016, Aprobado el 20-09-2016, Publicado el 17-01-2017

Estilo de Citación de Artículo:
V. Vega, Y. Hadfeg, "Puebas de seguridad: Estudio de herramientas", Lámpsakos, no. 17, pp 83-90, 2017
DOI: http://dx.doi.org/10.21501/21454086.1957

Resumen

Hoy día con el desarrollo y avances en la tecnología los productos software son parte de nuestra vida cotidiana. Estos productos constituyen un soporte para casi todas nuestras tareas. Estas tareas pueden tener un desempeño fundamental o no, y van desde ejecutar la conducción de un avión a través de un piloto automático hasta posibilitar el funcionamiento de los dispensadores de billetes o cajeros automáticos. Por la criticidad de los procesos en los que ellos se encuentran relacionado, es necesario que se cumplan dos características fundamentales; la primera, que tengan un nivel de calidad aceptado y la segunda, que sean productos seguros.

La seguridad en el software es un atributo no funcional que influye directamente en la calidad del producto. Realizar pruebas a los requisitos no funcionales para constatar su desempeño, tal y como se hace con los requisitos funcionales es una tarea tediosa. Como alternativa a este problema se han desarrollado herramientas que de forma automática o semiautomática realizan pruebas de diferente índole a los sistemas. El objetivo de este trabajo es identificar las herramientas de software existentes para realizar pruebas relacionadas con la seguridad. Para cumplir este objetivo se efectúa un estudio del estado del arte de las herramientas para realizar pruebas de seguridad desde el 2010 a la fecha.

Palabras clave: pruebas de seguridad, calidad de software, herramientas, ataques a la seguridad.

Abstract

Today, due to the development and advancement of technology, software products are part of our daily lives. These products support almost all our tasks. These tasks can be critical or non-performance, and range from piloting a plane with an autopilot to enabling the operation of ticket dispensers or ATMs. By the criticality of the processes in which they are related, it is necessary that two fundamental characteristics be met; first, having achieved a level of quality and second, being safe products.
Software security is a nonfunctional attribute that directly affects product quality. Testing non-functional requirements to verify their performance, as it is done with the functional requirements, is a tedious task. As an alternative to this problem, tools that automatically or semi-automatically perform tests of different types of systems have been developed. The aim of this paper is to identify existing software tools related to testing safety. To achieve this objective a study of the state-of-the-art tools used for security testing is done from 2010 to date.

Keywords: testing security, software quality, tools, security attacks

1 INTRODUCCIÓN

La calidad de software puede parecer un concepto alejado de la vida diaria de la mayoría de las personas; pero nada más lejos de la realidad. Cuando en nuestro ordenador aparece un mensaje de error, estamos ante un problema de calidad del software.

Para las empresas, la calidad del software con que ellas operan es muy valorada, ya que éstos manejan información que debe protegerse de cualquier peligro que afecte la integridad de los datos. Con calidad nos referimos al cumplimiento de las expectativas de un producto.

DOI: http://dx.doi.org/10.21501/21454086.1957

Las normas ISO definen la calidad como *"Grado en el que un conjunto de características inherentes cumple con los requisitos"* y además como un "Conjunto de propiedades o características de un producto o servicio que le confieren aptitud para satisfacer unas necesidades expresadas o implícitas" [1].
Cuando se dice que se tiene un software de calidad se puede entender que se puede modificar, que es confiable, eficaz y se deja usar fácilmente. Una de las tareas para asegurar estos resultados son las Pruebas. Las pruebas de software son las investigaciones empíricas y técnicas cuyo fin es proporcionar información objetiva e independiente sobre la calidad del producto [2].

Existen muchos tipos de pruebas al software [3], [4], [2] que verifican que el software funcione como esperamos. Debe ser verificado el cumplimiento de las especificaciones planteadas al inicio del proyecto por el analista o el propio cliente, y/o eliminar los posibles errores que se puedan haber cometido en cualquier fase del desarrollo. Hasta hace muy pocos años atrás estas pruebas se realizaban de manera relativamente informal, como podría ser la creación de informes que se pasaban de un departamento a otro. Pero hoy es considerada, en muchos aspectos, una de las etapas críticas dentro del ciclo de vida del software.

En la actualidad las empresas de software presenta dificultades para comprobar el buen funcionamiento de los software en cuanto a requerimientos no funcionales, en muchas ocasiones sólo realizan acciones aisladas en este sentido, esto se debe a que en gran medida es priorizado el desarrollo de las funcionalidades del software, siempre teniendo como meta cumplir con las entregas pactadas y minimizar los costos por retrasos. Esto provoca que se le preste poca atención a la realización de estas pruebas y que no sean evaluados los requisitos no funcionales con objetividad. Lo antes descrito genera problemas en cuanto a la utilidad del software, el tiempo de respuesta de las funcionalidades, la seguridad de este, entre otras, ya que es conocido que un error encontrado en tiempo de ejecución es mucho más costoso de solucionar que el descubierto en un ambiente controlado.

Las pruebas realizadas a los requisitos no funcionales de un software son denominadas pruebas no funcionales. Algunas de las pruebas no funcionales encontradas en la literatura están relacionada con las pruebas de usabilidad [5], [6], pruebas de rendimiento [7], [6] y pruebas de seguridad [8], [9], que son a las que nos referiremos en este trabajo.

Las pruebas de seguridad abarcan más allá de lo que es el simple escaneo de puerto, los probadores deben utilizar enfoques basados en el riesgo, basados tanto en la realidad arquitectónica del sistema como en la mentalidad del atacante, para evaluar adecuadamente la seguridad del software.

En muchas ocasiones el personal encargado de la calidad del software dentro de una empresa y el personal responsables de la seguridad de la información de la misma empresa son totalmente diferentes, y no existe un flujo constante de información entre ellos. Siempre que sea posible, los desarrolladores y los probadores de software deberían incluir más a los expertos de seguridad en los nuevos proyectos, evitándose inconvenientes que pudieran llagar demasiado tarde.

En otros casos, no se cuenta en las empresa con un equipo de seguridad, y es un miembro de la parte de sistemas quien asume lo relacionado a la seguridad de la información en la empresa. En casos como este, el equipo de calidad debería colaborar con el equipo de desarrollo y aportar ciertas pruebas de seguridad. También existen casos en los que determinadas herramientas creadas con el fin de realizar pruebas de seguridad, pueden sernos de mucha ayuda a la hora de realizar determinadas pruebas funcionales sobre un software en desarrollo. Por ejemplo, un *Sniffer* como *Wireshark*, puede ser una gran ayuda para verificar si el software está funcionando correctamente.

El objetivo de este trabajo es presentar un estudio del estado del arte de las herramientas existentes para realizar pruebas a la seguridad de los sistemas. Al concluir el estudio se pretende responder la siguiente pregunta de investigación:

¿Qué características tienen las herramientas actualmente para realizar pruebas de seguridad?

En la siguiente sección se indica cuál fue el proceso de investigación y qué criterios se utilizaron para realizar las búsquedas, luego se comparan las propuestas encontradas en la actividad anterior, posteriormente se analizan los resultados obtenidos, y finalmente se presentan las conclusiones del trabajo.

2 PROCESO DE INVESTIGACIÒN

2.1 Fuentes de datos y estrategia de búsqueda

En el proceso de investigación se decidió buscar en bases de datos electrónicas y actas de congresos. Las bases de datos donde se realizó el proceso fueron:

Vianca Vega, PhD
Yahima Hadfeg, MSc
DOI: http://dx.doi.org/10.21501/21454086.1957

- *ACM Digital Library.*
- *Science@Direct.*
- *IEEE Xplore*
- *WileyInterScience.*

Teniendo declarado las fuentes de donde se obtendrá la información se expresan las cadenas de búsqueda. En la Tabla 1 se muestran las cadenas de búsquedas utilizadas en idioma inglés y español. En la Tabla 2 se muestra las cantidades de trabajo obtenidos por cada fuente de información.

TABLA 1
CADENAS DE BÚSQUEDA

Cadena de búsqueda en español	Cadena de búsqueda en inglés
(A1) "Pruebas de Seguridad"	(A1) "Security Testing"
(A2) Herramientas	(A2) Tools
A1 and A2	A1 and A2

TABLA 2
CANTIDAD DE DOCUMENTOS TOTALES OBTENIDOS

Fuente	Cantidad de Documentos en español	Cantidad de documentos en inglés
ACM Digital Library	0	30
Science@Direct	0	754
IEEE Xplore	0	86
WileyInterScience	19	165

La Fig. 1 muestra la proporción de los resultados obtenidos según las fuentes utilizadas. Science@Direct fue la que mayores resultados aportó con un 72% del total.

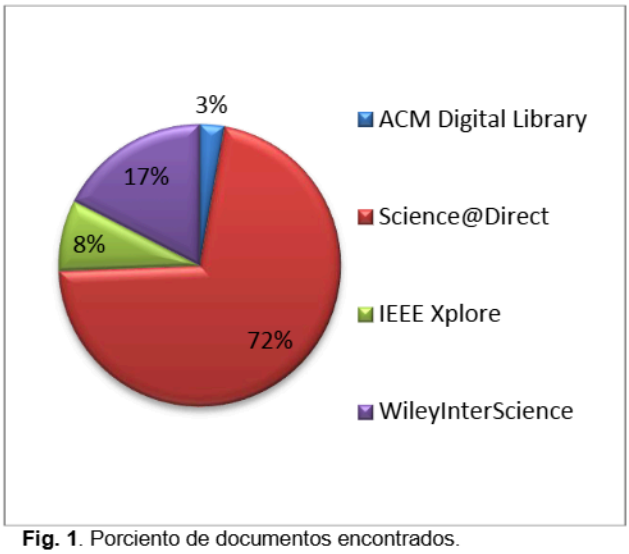

Fig. 1. Porciento de documentos encontrados.

2.2 Selección de los estudios

El criterio para realizar la revisión sistemática fue basado en un modelo iterativo e incremental. Iterativo porque la ejecución (búsqueda, extracción de información y visualización de resultados) de la revisión sistemática se hace primero completamente en una fuente de búsqueda, luego en la siguiente fuente de búsqueda y así hasta cubrirlas todas. Incremental porque el documento (que es el producto) de la revisión sistemática va creciendo y evolucionando en cada iteración hasta convertirse en el definitivo, de esta forma se obtuvo cada una de las herramientas para las pruebas de seguridad y sus características.
A continuación se listan los criterios de inclusión utilizados para la elección de los estudios primarios:

- Estudios presentados en los idiomas Español o Inglés.
- Estudios realizados en los últimos seis años.
- Estudios que contengan en su título las palabras utilizadas en el criterio de búsqueda.
- Estudios que contengan en su resumen las palabras utilizadas en el criterio de búsqueda.
- Estudio cuyas palabras claves concuerden con el criterio de búsqueda definido.
- Estudios que en su cuerpo o desarrollo incluyan referencias o descripciones sobre herramientas para el desarrollo de pruebas de seguridad.

Los criterios de exclusión son listados a continuación:

- Estudio que no corresponda con los idiomas indicados.
- Estudio que no corresponda al período de búsqueda.
- Estudios no indexados en bases de datos de prestigio.
- Estudios que no coincidan con los criterios de búsqueda.

Después de haber aplicado los criterios de inclusión y exclusión nos quedamos con un conjunto de 24 trabajos. La Tabla 3 muestra cómo queda la nueva distribución de documentos por fuentes de datos después de aplicado los filtros.

TABLA 3
CANTIDAD DE DOCUMENTOS A UTILIZAR CLASIFICADOS POR FUENTES

Fuente	Cantidad de Documentos en español	Cantidad de documentos en inglés
ACM Digital Library	0	8
Science@Direct	0	5
IEEE Xplore	0	9
WileyInterScience	0	2

DOI: http://dx.doi.org/10.21501/21454086.1957

3 COMPARACIÒN DE LAS PROPUESTAS

El conjunto de documentos seleccionados incluye artículos de revista, publicaciones en congresos y capítulos de libros.

3.1 Calidad Metodológica

Cada uno de los 24 artículos seleccionados fueron evaluados independientemente de acuerdo con algunos de los criterios propuestos por el programa de habilidades de evaluación crítica de trabajos científicos [10]. A continuación se enumeran en forma de pregunta cada uno de estos criterios adaptados para ésta investigación:

- ¿Están claros los objetivos de la propuesta?
- ¿Hay una descripción adecuada del contexto?
- ¿Se utilizaron fuentes bibliográficas confiables de acuerdo a las temáticas abordadas?
- ¿Se efectúan experimentos?
- ¿Se mencionan claramente los resultados?

A cada uno de los trabajos analizados se le aplicaron los 5 criterios respondiendo *sí* o *no* por cada pregunta planteada. Estas respuestas fueron representadas en la Fig. 2.

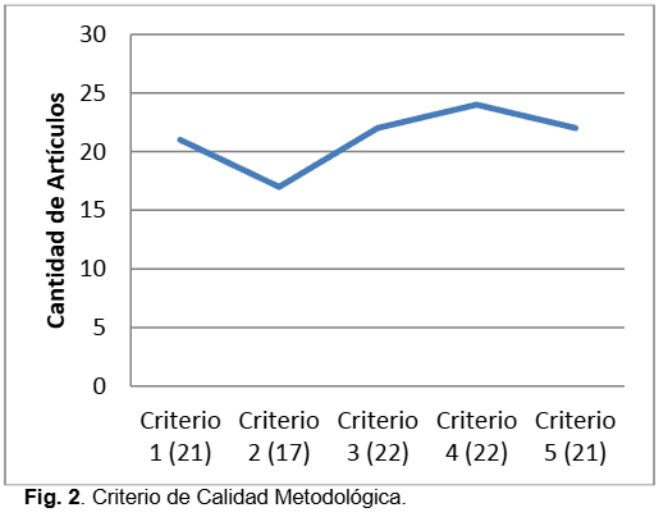

Fig. 2. Criterio de Calidad Metodológica.

Como se aprecia en la gráfica todos los criterios fueron cumplidos por encima de un 68%. Una de las causas de que la este elevado porcentaje en la calidad metodológica es debido a que todos los trabajos fueron extraídos de congresos con alto prestigio internacional y con un buen arbitraje, así como de revistas de alto impacto y capítulos de libros.

El criterio que mejor puntaje alcanzó fue el relacionado con la realización de experimento (criterio 4), los métodos utilizados para realizar la experimentación son diversos van desde la utilización de un caso de estudio hasta el análisis matemático del algoritmo que en algunos caso se propuso como motor de la herramienta de pruebas de seguridad. Por otro lado el criterio que más baja puntuación obtuvo fue el criterio número 2 que evaluaba si había una buena descripción del contexto.
Los 5 trabajos a los que se le asignó un no es por considerar que en el cuerpo del documento no se explicó con la profundidad requerida los fundamentos teóricos de la propuesta.

3.2 Tipo de contenido

Los trabajos fueron agrupados utilizando los siguientes criterios:

- Propuesta de herramientas que atacan a los sistemas para detectar las vulnerabilidades en cuanto a la seguridad.
- Herramientas que automatizan los procesos de prueba.
- Herramientas especializadas en las pruebas de seguridad para dispositivos móviles.
- Herramientas que evalúan los ataques realizados a los sistemas.
- Otras herramientas.

En la Fig. 3 se muestra la clasificación de los documentos siguiendo los criterios definidos. Cada artículo sólo puede pertenecer a una sola categoría.

Como se observa el 37% de los artículos son propuestas de herramientas que atacan la seguridad de los sistemas. Este resultado evidencia que una manera de probar la seguridad de los software es atacándolos con herramientas en ambientes controlados, y no sólo eso sino que en los últimos seis años es el procedimiento más utilizado.

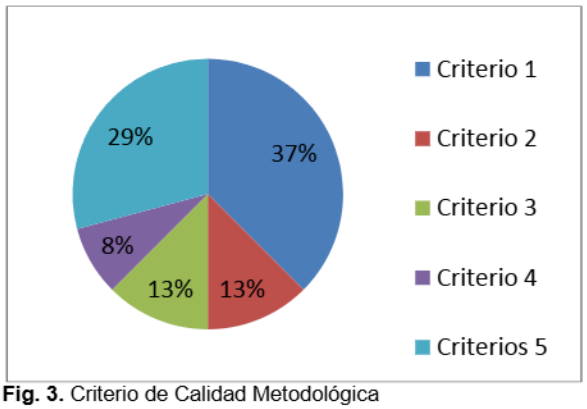

Fig. 3. Criterio de Calidad Metodológica

Vianca Vega, PhD
Yahima Hadfeg, MSc
DOI: http://dx.doi.org/10.21501/21454086.1957

3.3 Relación entre documentos

Las formas y maneras de probar el mismo atributo de seguridad cambian en relación al tipo de sistema y desde la perspectiva que se vea. Ahora veremos algunos puntos de convergencia en las herramientas estudiadas. En primer lugar muchos trabajos se dedican al desarrollo de herramientas que se dedican a probar la seguridad en el marco de servicios web, en este estudio se identificaron un conjunto 7 trabajos [11], [12], [13], [14], [15], [16], [17], donde cada uno de ellos brinda su aporte en cuanto a las pruebas de seguridad.
Existen otras iniciativas relacionada con las pruebas de penetración, ya sea creando herramientas que simulen ataque de este tipo [18], [19] o por herramientas que faciliten la generación de casos de pruebas [20] o automatización de procedimientos para realizar las pruebas de seguridad [21].
Por otra parte, se encontraron artículos que proponen un enfoque de las pruebas de seguridad basado en modelos de amenaza [22], siguiendo ese mismo principio en [23] se propone una evaluación inteligente siguiendo patrones de ataques que no describen más que modelos de amenaza.

Se identificaron además herramientas especializadas en pruebas de seguridad para *Android* [24], [25], [26] estas es particulares son interesantes ya que el número de usuarios con dispositivos móviles ha aumentado significativamente en los últimos años, y las aplicaciones móviles se están convirtiendo en herramientas integrales para la vida diaria.

3.4 Años de publicación

En primer lugar, es importante mencionar que uno de los filtros utilizados en la obtención de los artículos seleccionados para este análisis ha sido que la fecha de publicación, ésta no tenga más de seis años de antigüedad, es decir, para la situación actual los documentos no pueden haber sido publicados antes del año 2010.

La Fig. 4 muestra una gráfica con la cantidad de publicaciones por año. Se evidencia que en el año 2013 y 2015 es dónde más artículos fueron publicados. Este fenómeno puede estar dado porque en esos dos años se realizó el congreso internacional de "Pruebas de software, verificación y validación", impulsando la publicación de muchos trabajos relacionados con el tema que se está analizando.

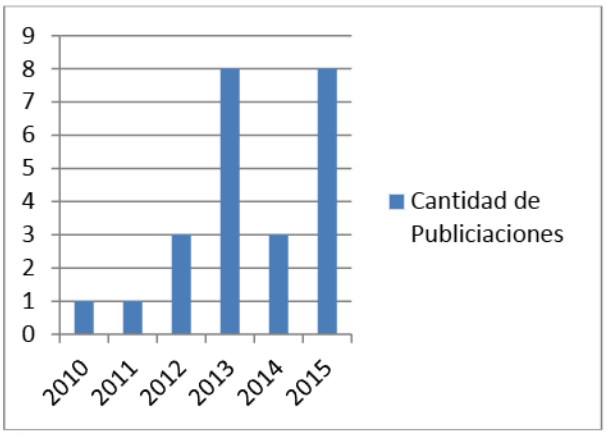

Fig. 4. Cantidad de publicaciones por año

4 RESULTADOS Y DISCUSIÓN

Los documentos seleccionados para el estudio de las herramientas, metodologías y procesos para realizar pruebas, pueden dividirse de forma operativa en tres bandos. En el primer bando podemos agrupar las herramientas que desempeñan el rol de un atacante [17], [16], [19], [15], [13], [27], [12], [28] y tratan de burlar la seguridad del software realizando diferentes ataques. Con la información obtenida trabajan para mejora el software. La Tabla 4 muestra una relación de dichas herramientas, se especifican si están disponibles gratuitamente y los ataques fundamentales que implementan.

TABLA 4
RELACIÓN DE HERRAMIENTAS QUE ACTÚAN COMO ATACANTES

Nombre de la herramienta	Herramienta gratuita	Ataque que implementa
VERA	No	Inyección de código
PURITY	Si	Ofrece al usuario planear sus ataques, cuenta con amplio catálogo
Kali Linux	Si	Penetración, Denegación de Servicio
TFTP RRQ	No	Denegación de servicio
WSFAggressor	Si	Combinatorios
mHealth apps	No	Integridad de los datos

En el segundo bando, las herramientas juegan el rol de defensores, son dotadas con mecanismos capaces de identificar un ataque y tratar de defender el software, o mínimo reportar lo ocurrido [29], [30]. En la Tabla 5 se muestran las dos herramientas encontradas que clasifican en este bando.

TABLA 5
RELACIÓN DE HERRAMIENTAS QUE ACTÚAN COMO DEFENSORAS

Nombre de la herramienta	Herramienta gratuita
SOA-Scanner	Si (algunas versiones con funcionalidades limitadas)
Tool support for secure programming	No

DOI: http://dx.doi.org/10.21501/21454086.1957

En el tercer bando agruparemos todas aquellas investigaciones que no clasifican en las anteriores.

Del contenido de los trabajos que abordan cada uno de los bandos identificados se extraen una serie de ideas importantes. Los ataques que más han sido implementados son:

- Ataques de penetración. Este ataque es el más utilizado en la literatura, está incluido en las herramientas propuestas en [12], [22], [23], [24] donde cada una de ellas juegan el ron del atacante. En [14] también se utiliza el principio del ataque de penetración, pero en esta ocasión el software trata de defenderse de él.

- Ataque inyección de código, es utilizado por [31] [12], [17], [23]. En [12] se implementa como parte de la herramienta utilizando código XML, en [17] el código que se pretende inyectar es SQL, y en [23] se analiza el patrón de este ataque para protegerse de él en una red que los autores denominan inteligente.

- Ataque de entidades externas, ya sea utilizando XML [12] u otro mecanismo [25].

- Ataques contra la integridad de los datos [26] [19], [17].

- Ataque de descripciones de pestaña [12].

- Ataque a la denegación de servicio [13], [32].

- Otros ataques [12].

Durante la investigación se identificaron herramientas que no sólo se limitaban a realizar ataque previamente desarrollados, ellas dan la posibilidad de programar ataques específicos a software mediante un módulo de desarrollo [17], [31], esta característica le da un valor agregado a la herramienta. Una afirmación recurrente en la mayoría de los trabajos es que para poder implementar herramientas que aseguren de una manera u otra la seguridad de los sistemas hay que conocer los modelos de amenazas para diferentes dominios. En varios artículos [22], [28] [17] se ha identificado que al base de su propuesta se centra en los en esta área. Las herramientas de pruebas de software pueden ser muy útiles a los proveedores para evaluar la seguridad de sus plataformas de servicios, y también para los desarrolladores para divulgar cuestiones potencialmente graves antes del despliegue.

Las principales estrategias que utilizan las propuestas que juegan el rol de defensoras son:

- Principio de monitoreo, detectando si algo pasa fuera de lo común, por lo general estas herramientas se basan en un proceso iterativo [20], [21].
- Generación de casos de pruebas y ejecución de en determinados sitios web [16], [33].
- Automatización de los procedimientos de prueba utilizando los requisitos de seguridad [21].
- Análisis de vulnerabilidades [30]
- Integración basada en modelos [34].
- Análisis de árboles de amenazas y patrones de ataque [23], [22], [34].

Hay trabajos que proponen métodos para hacer una evaluación de riesgo [28] y para ello pueden usar los enfoques de patrón de ataque [31]. Los procedimientos de pruebas pueden ser propuestos automáticamente tomando como base la forma tradicional de realizar las pruebas de software.

5 CONCLUSIONES

La búsqueda de la literatura utilizada para la realización de este estudio ha sido complicada de reunir y analizar, ya que hay muchas propuestas de métodos y procedimientos que pueden o no terminar en una herramienta para realizar pruebas de seguridad.

La mayoría de los trabajos realizados no declaran explícitamente que requisito o requisitos de seguridad están trabajando. Hay una marcada tendencia a abordar el problema de la denegación de servicio. Sin embargo no se encontró artículos que hagan referencia al requisito de no repudio.

Se identificaron herramientas que de forma automática generan casos de pruebas de seguridad, pero estas deben tener identificado un modelo de amenaza, que por lo general es característico de cada tipos de software. La tendencia a tratar las pruebas de seguridad son mediante la simulación de ataques.

La mayoría de los esfuerzos en este campo están dedicados al desarrollo web específicamente a los servicios. Este trabajo sirve de base para conocer herramientas que están disponibles para realizar las pruebas de seguridad. Se encontró pocas herramientas robustas para el análisis de las pruebas de seguridad que se encuentren de forma gratuita. Se cuenta con algunas herramientas extensibles y configurables mediante de código.

DOI: http://dx.doi.org/10.21501/21454086.1957

REFERENCIAS

[1] ISO, "ISO 9000 - Quality management". 2009. [Online]. Available at http://www.iso.org/iso/home.html

[2] G. Myers, C. Sandler and T. Badgett, "The art of software testing" John Wiley & Sons, p. 256. 2011.

[3] P. Hamill, "Unit Test Frameworks: Tools for High-Quality Software Development". O'Reilly Media, Inc. p. 304. 2004.

[4] A. Black, "Critical Testing Process: Plan, Prepare, Perform, Perfect" Addison-Wesley Longman Publishing Co., Inc., p. 608. 2003.

[5] J. Rubin, and D. Chisnell, "Handbook of usability testing: how to plan, design and conduct effective tests" John Wiley & Sons, p. 384. 2008.

[6] L. Manzari, and J. Trinidad-Christensen, "User-centered design of a web site for library and information science students: Heuristic evaluation and usability testing". Information technology and libraries, vol. 25, no. 3, pp. 163-169, 2013

[7] Dumas, J. & J. Redish, "A practical guide to usability testing" Intellect Books, p. 404. 1999.

[8] G. McGraw, "Software security". Security & Privacy, IEEE, vol. 2, no. 2, pp. 80-83, 2004

[9] McGraw, G., "Software security: building security in" Addison-Wesley Professional, 448 p. 2006.

[10] T. Greenhalgh, T. and R. Taylor, "How to read a paper" BMJ Publishing Group London, pp 1-2. 2002.

[11] B. Garn, I. Kapsalis, D.E. Simos and S. Winkler. "On the applicability of combinatorial testing to web application security testing: a case study". In Proceedings of the 2014 Workshop on Joining AcadeMiA and Industry Contributions to Test Automation and Model-Based Testing. pp. 16-21, 2014

[12] R. A. Oliveira, N. Laranjeiro and M. Vieira. "WSFAggressor: an extensible web service framework attacking tool". In Proceedings of the Industrial Track of the 13th ACM/IFIP/USENIX International Middleware Conference. pp. 2, 2012

[13] B. Sieklik, R. Macfarlane and W. J. Buchanan, "TFTP DDoS amplification attack". Computers & Security, vol.57, No. pp 67-92. 2016

[14] D. M. Duchesne, "Using CABECTPortal as a Case Study to Extend the Capabilities of Penetration Testing Tools". In Proceedings of the 46th ACM Technical Symposium on Computer Science Education. pp. 715-715, 2015

[15] M. Salas and E. Martins, "Security testing methodology for vulnerabilities detection of xss in web services and ws-security". Electronic Notes in Theoretical Computer Science, vol. 302. pp. 133-154, 2014

[16] J. Bozic, and F. Wotawa. "PURITY: A Planning-based secURITY Testing Tool". In Software Quality, Reliability and Security-Companion (QRS-C), 2015 IEEE International Conference on. pp. 46-55, 2015

[17] A. Blome, M. Ochoa, K. Li, M. Peroli and M. T. Dashti. "Vera: A flexible model-based vulnerability testing tool". In Software Testing, Verification and Validation (ICST), 2013 IEEE Sixth International Conference on. pp. 471-478, 2013

[18] J. Yeo, "Using penetration testing to enhance your company's security". Computer Fraud & Security, vol. 2013, no. 4, pp. 17-20, 2013

[19] L. Allen, T. Heriyanto and S. Ali, "Kali Linux–Assuring Security by Penetration Testing" Packt Publishing Ltd, p. 454. 2014.

[20] L. H. Chen, F. H. Hsu, Y. Hwang, M. C. Su, W. S. Ku and C. H. Chang, "ARMORY: An automatic security testing tool for buffer overflow defect detection". Computers & Electrical Engineering, vol. 39, no. 7, pp. 2233-2242, 2013

[21] V. Manetti, and L. M. Petrella. "FITNESS: a framework for automatic testing of ASTERIX based software systems". In Proceedings of the 2013 International Workshop on Joining AcadeMiA and Industry Contributions to testing Automation. pp. 71-76, 2013

[22] A. Marback, H. Do, K. He, S. Kondamarri and D. Xu, "A threat model-based approach to security testing". Software: Practice and Experience, vol. 43, no. 2, pp. 241-258, 2013

[23] S. Kondakci, "Intelligent network security assessment with modeling and analysis of attack patterns". Security and Communication Networks, vol. 5, no. 12, pp. 1471-1486, 2012.

[24] S. Jadhav, T. Oh, Y.H. Kim and J.N. Kim. "Mobile device penetration testing framework and platform for the mobile device security course". In Advanced Communication Technology (ICACT), 2015 17th International Conference on. pp. 675-680, 2015

[25] S. Salva and S. R. Zafimiharisoa. "Data vulnerability detection by security testing for Android applications". In Information Security for South Africa, 2013. pp. 1-8, 2013

[26] K. Knorr and D. Aspinall. "Security testing for Android mHealth apps". In Software Testing, Verification and Validation Workshops (ICSTW), 2015 IEEE Eighth International Conference on. pp. 1-8, 2015

DOI: http://dx.doi.org/10.21501/21454086.1957

(ICSTW), 2015 IEEE Eighth International Conference on. pp. 1-4, 2015

[27] A. Morais, A. Cavalli and E. Martins. "A model-based attack injection approach for security validation". In Proceedings of the 4th international conference on Security of information and networks. pp. 103-110, 2011

[28] E. Gutesman and A. Waissbein. "The impact of predicting attacker tools in security risk assessments". In Proceedings of the Sixth Annual Workshop on Cyber Security and Information Intelligence Research. pp. 75, 2010

[29] Antunes, N. & M. Vieira. "SOA-Scanner: An Integrated Tool to Detect Vulnerabilities in Service-Based Infrastructures". In Services Computing (SCC), 2013 IEEE International Conference on. pp. 280-287, 2013

[30] Li, K., C. Hebert, J. Lindemann, M. Sauter, H. Mack, T. Schroer & A. Tiple. "Tool support for secure programming by security testing". In Software Testing, Verification and Validation Workshops

[31] Bozic, J. & F. Wotawa. "Xss pattern for attack modeling in testing". In Proceedings of the 8th International Workshop on Automation of Software Test. pp. 71-74, 2013

[32] Smith, C. & G. Francia III. "Security fuzzing toolset". In Proceedings of the 50th Annual Southeast Regional Conference. pp. 329-330, 2012

[33] Aouadi, M.H., K. Toumi & A. Cavalli. "An Active Testing Tool for Security Testing of Distributed Systems". In Availability, Reliability and Security (ARES), 2015 10th International Conference on. pp. 735-740, 2015

[34] Xu, D., W. Xu, M. Kent, L. Thomas & L. Wang, "An Automated Test Generation Technique for Software Quality Assurance". Reliability, IEEE Transactions on, Vol. 64, No. 1, pp. 247-268, 2015