Revista Cubana de Ciencias Informáticas
Vol. 15, No. Especial UCIENCIA I, Septiembre, 2021
ISSN: 2227-1899 | RNPS: 2301
http://rcci.uci.cu
Pág. 333-349

Tipo de artículo : Artículo original
Temática : Pruebas de Software
Recibido: 30/06/2021| Aceptado: 01/10/2021

Procedimiento para evaluar seguridad a productos de software

Procedure for evaluating security of software products

Roberto Menejías García [1*] https://orcid.org/0000-0003-3482-5898
Noel Harrinso Hidalgo Reyes [1] https://orcid.org/0000-0001-6517-2303
Aymara Marín Díaz [1] https://orcid.org/0000-0001-5101-7804
Yaimí Trujillo Casañola [1] https://orcid.org/0000-0002-3138-011x

[1] Universidad de las Ciencias Informáticas, Cuba. Carretera San Antonio Km 2 1/2. {rmenejias, nhhidalgo, amarin, yaimi}@uci.cu

*Autor para la correspondencia: (amarin@uci.cu)

RESUMEN

En la industria del software, la realización de pruebas de calidad constituye la principal forma para la detección de errores y vulnerabilidades, sin embargo, muchas investigaciones y tendencias evidencian que se realizan luego de finalizado el producto y muchas veces solo se ejecutan pruebas funcionales. Esto supone un problema ya que en muchas ocasiones en los resultados de la realización de las pruebas se detectan problemas de tipo: vulnerabilidad, fallos en la integridad de los datos, disponibilidad, pérdidas y costo mediante la manipulación y robo de información. Para garantizar un mayor nivel de seguridad en los sistemas, se realizan las pruebas de seguridad para evaluar específicamente estos elementos fundamentales. En el presente artículo se describe un procedimiento para realizar pruebas no funcionales para evaluar la característica de calidad del producto de seguridad. Es independiente del negocio, del tipo de producto y de

Revista Cubana de Ciencias Informáticas
Vol. 15, No. Especial UCIENCIA I, Septiembre, 2021
ISSN: 2227-1899 | RNPS: 2301
http://rcci.uci.cu
Pág. 333-349

la metodología de desarrollo de software. El procedimiento tiene en cuenta buenas prácticas documentas en modelos, normas y estándares reconocidos internacionalmente, que a su vez fueron enriquecidas y particularizadas por expertos de organizaciones cubanas. Se describe el qué probar y el cómo hacerlo, y se muestran los resultados de la valoración de la propuesta por expertos.

Palabras clave: pruebas; vulnerabilidad; seguridad; procedimiento.

ABSTRACT

In the software industry, the performance of quality tests is the main way to detect errors and vulnerabilities, however many investigations and trends show that they are carried out after the product is finished and many times only functional tests are executed. This is a problem since in many cases, in the results of the tests, problems of type are detected: vulnerability, failures in the integrity of the data, availability, losses and cost through the manipulation and theft of information. To ensure a higher level of security in systems, security tests are performed to specifically evaluate these critical elements. This article describes a procedure for performing non-functional tests to evaluate the quality characteristic of the security product. It is independent of the business, the type of product and the software development methodology. The procedure takes into account good documented practices in internationally recognized models, norms and standards, which in turn were enriched and individualized by experts from Cuban organizations. What to try and how to do it is described, and the results of the evaluation of the proposal by experts are shown.

Keywords: testing; vulnerability; security; procedure.

Introducción

En la actualidad, se vive en una sociedad donde cada vez se hace más evidente el uso de las tecnologías de información en todo ámbito (Proaño et al., 2017). El software se ha convertido en un producto vital, tanto

para empresas, organismos, servicios y tareas cotidianas de los ciudadanos como para la toma de decisiones, el intercambio de información y la gestión del conocimiento (Marín et al., 2020). La ingeniería del software (ISW), según Pressman, permite la construcción de productos de software de alta calidad mediante un conjunto de procesos, colección de métodos y arreglos de herramientas. Desde sus inicios ha aplicado un enfoque sistemático, disciplinado y cuantificable al desarrollo, operación y mantenimiento del software, con el objetivo de alcanzar un software de alta calidad (Marín et al., 2018). El principal objetivo de la ingeniería de software es mejorar la calidad del producto final desde la calidad del proceso y del propio producto (Marín et al., 2018).

Según diversos autores en el área de calidad de software como Humphrey, Larman, Pressman, la calidad está altamente relacionada con los defectos en los productos y coinciden en que hay que invertir más en las actividades de control y aseguramiento de la calidad desde los inicios del desarrollo de software (Marín et al., 2020).

La industria del software reconoce hoy la importancia de llevar a cabo pruebas de software, como instrumento para asegurar la calidad de los productos desarrollados (Rojas et al., 2015). El concepto de pruebas de calidad de software permite en las empresas con áreas afines a los sistemas, la computación, la informática brindar productos con altos estándares de calidad y con una disminución de fallos en estos (Mera, 2016).

La implementación de un proceso de pruebas brinda las pautas para definir objetivos, analizar y viabilizar los requerimientos, diseñar, detallar, programar, implementar y asegurar la calidad de un producto de desarrollo software (Mera, 2016). Mediante las pruebas de software se puede garantizar la ejecución factible del producto, disminuyendo así la cantidad de posibles errores y vulnerabilidades que pueden existir.

En la actualidad es común que existan errores de programación que den origen a problemas de seguridad o vulnerabilidades. Los fallos de seguridad pueden afectar vidas humanas y causar daños en infraestructura industrial y social, poniendo en riesgo la confidencialidad y privacidad de la información y socavando la viabilidad de sectores completos de negocio (Yepes, 2017). La ligera realización y actualización de sistemas, conlleva al mal manejo de las pruebas de seguridad, exponiendo al cliente/usuario a un posible robo o manipulación de información, datos vinculados a su economía y/o a la sociedad.

La existencia de vulnerabilidades en las aplicaciones representa riesgos que afectan los objetivos de negocio de las organizaciones. Cuando existe una vulnerabilidad que permite que un atacante (amenaza) comprometa un activo de información (información valiosa para el negocio), se habla de riesgo o, en otros términos, la probabilidad de ocurrencia de un evento adverso y el impacto asociado en caso de que ocurra (Yepes, 2017; Vásquez, 2020). Los riesgos sobre la seguridad de la información pueden afectar sobremanera aspectos como integridad, confiabilidad y disponibilidad de la misma (Proaño et al., 2017; Casas, 2020). Para detectar estos riesgos se utilizan generalmente procedimientos, técnicas y herramientas que hacen de la seguridad informática un proceso automatizado y eficiente.

En la revisión bibliográfica realizada se pudo constatar que los autores consultados plantean la necesidad de las pruebas (Board (ISTQB), 2018; David Flores Mendoza, 2019; Fernández Pérez, 2018; Pressman, 2010) y de su correcto diseño, enmarcando las buenas prácticas que deben tenerse en cuenta para la realización exitosa de pruebas, pero no se hace una propuesta de qué probar y algunos elementos del cómo probar. Por tanto, se establece como problema a resolver para esta investigación: ¿Cómo evaluar la seguridad como característica de calidad del producto teniendo en cuenta las buenas prácticas recomendadas en los modelos, normas y estándares más utilizados y las experiencias de investigadores?

El objetivo de la investigación consiste en diseñar un procedimiento de pruebas para evaluar la seguridad como característica de calidad del producto que permita disminuir los riesgos de falla en operación.

Métodos o Metodología Computacional

Para el desarrollo de esta investigación se utilizaron los métodos que se mencionan a continuación. Además, se brinda una breve explicación de los fines para los que fueron utilizados.

Métodos teóricos:

1. Método dialéctico para el estudio crítico de los trabajos anteriores y para usar estos como fuente de referencia y comparación de los resultados.

2. El método analítico- sintético se utilizó para el estudio de la bibliografía acerca de los modelos de calidad más usados internacionalmente
3. El hipotético deductivo para la identificación de la situación problemática y de las soluciones.

Métodos empíricos:

1. Entrevista para obtener informaciones en pos de argumentar la situación problemática y la validación de los resultados.
2. La encuesta para obtener las experiencias de las organizaciones.
3. La observación participante para obtener la información necesaria para el planteamiento del problema, así como realizar la confrontación de los resultados obtenidos.
4. Métodos estadísticos para valorar el efecto de la propuesta.
5. El método experimental para comprobar la utilidad de los datos obtenidos a partir de la implementación del procedimiento para realizar pruebas de seguridad.

Un procedimiento es definido por la Real Academia Española como: acción de proceder; método de ejecutar algunas cosas. En el contexto de la Ingeniería de software es definido como:

El procedimiento para ejecutar las pruebas de seguridad cuenta con tres fases y está diseñado para mejorar y guiar el proceso con el objetivo de ejecutar unas pruebas más eficientes al software. Permite evaluar la seguridad de cualquier tipo de sistema, o sea, puede ser utilizado para sistemas sujetos a pruebas (SSP) con diferentes arquitecturas tecnológicas. Se recomienda comenzar el procedimiento luego de finalizada la primera evaluación a la característica de calidad del producto de adecuación funcional. Esto se recomienda debido a que es necesario establecer una estabilidad en el producto a evaluar para evitar que se realicen cambios grandes en el código, posteriores a la evaluación, que pueden insertar vulnerabilidades. El procedimiento se nutre de los requisitos no funcionales que se proponen para tener en cuenta la seguridad desde el inicio del desarrollo y de las experiencias. Los elementos que se proponen en el procedimiento son: fases, actividades y herramientas a utilizar.

Revista Cubana de Ciencias Informáticas
Vol. 15, No. Especial UCIENCIA I, Septiembre, 2021
ISSN: 2227-1899 | RNPS: 2301
http://rcci.uci.cu
Pág. 333-349

Las fases del procedimiento contienen dentro de cada una de ellas las diferentes actividades que se definieron para la ejecución del proceso de pruebas de seguridad de la DC. Las fases son las siguientes:

Fases del diseño de pruebas de seguridad

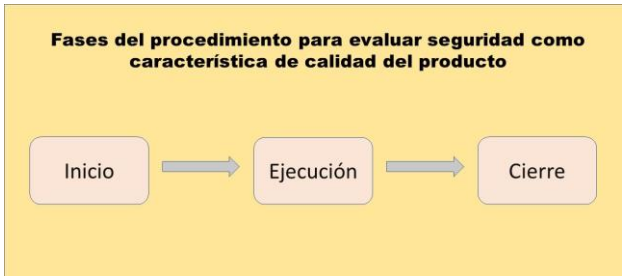

Fig. 1 - Fases del procedimiento para evaluar seguridad como característica de calidad del producto. Confección propia.

En las dos primeras fases están presentes el equipo de desarrollo y el equipo de pruebas, así el proceso se convierte en una tarea de comunicación constante entre ambos. Esta relación permite un mejor aprovechamiento del fondo del tiempo planificado para esta etapa. Siendo este elemento una de las principales fortalezas de este proceso.

Durante cada fase se realiza un grupo de actividades en correspondencia con la misma. Las actividades del procedimiento de manera general son las siguientes:

1. Identificar los datos necesarios de prueba y caracterizar los sistemas.
2. Preparar el entorno de prueba e identificación de cualquier infraestructura necesaria y las herramientas.
3. Registrar el resultado de la ejecución de pruebas y registrar la identidad y las versiones del software. Debemos saber exactamente la versión del software, tenemos que informar defectos con versiones específicas, y el registro de las pruebas para un informe final.
4. Evaluar como resultaron las actividades de pruebas y analizar las lecciones aprendidas.

Revista Cubana de Ciencias Informáticas
Vol. 15, No. Especial UCIENCIA I, Septiembre, 2021
ISSN: 2227-1899 | RNPS: 2301
http://rcci.uci.cu
Pág. 333-349

A continuación, se realiza la definición de cada fase, y la especificación de las actividades dentro de cada una de ellas, modelando la forma de ejecutarlas:

Fase de inicio

En esta fase es donde se recopila información, se analizan los datos necesarios para las pruebas y se definen cuáles se le aplicaran al sistema, de forma que sean más eficaces sobre la aplicación de acuerdo con las características y especificaciones del sistema en prueba. Las actividades que se realizan en esta fase son: Identificar el sistema.

1. El equipo de desarrollo del software provee la información necesaria sobre el sistema, mediante una tabla con las tecnologías que fueron usados en el desarrollo de la versión actual del sistema en cuestión. La tabla de información puede variar de acuerdo con el tipo de aplicación (web, escritorio, video juego o aplicación para móvil). La información recopilada se ajusta al diseño de las pruebas con el objetivo de que sean más eficientes. (Ver Tabla 1).

Tabla 1- Tabla de información de un sistema web.

Nombre/Versión	Tecnología	Entorno de desarrollo (IDE)	Navegador	Gestor de BD/ Servidor Web
Proyecto X	Drupal 8.0 PHP 5.0	PHP Storm	Chrome 65.0 Firefox 64.4 +	Postgrees 5.0
Proyecto Y	Drupal 8.9 Java 7.0	WordPress	Mozilla 59.0 +	Apache 2.4.29

2. Para terminar la recopilación de información se realiza un escaneo de la red con la herramienta Nmap y su interfaz Zenmap, con el objetivo de identificar qué puertos y servicios utiliza e inicia el sistema. El uso de Nmap automatiza esta actividad y hace posible encontrar vulnerabilidades en múltiples sistemas para la exploración de dominios, puertos, direcciones IP, país, región, entre otros datos.

Revista Cubana de Ciencias Informáticas
Vol. 15, No. Especial UCIENCIA I, Septiembre, 2021
ISSN: 2227-1899 | RNPS: 2301
http://rcci.uci.cu
Pág. 333-349

Monitoreo tecnológico

Se realiza un estudio de las tecnologías en busca de vulnerabilidades que no han sido parcheadas o resueltas en las versiones utilizadas en desarrollo de los sistemas. Esta actividad genera un reporte de posibles vulnerabilidades y hallazgos a los cuales se puede enfrentar el equipo de probadores. El reporte se registra y almacena para posibles sugerencias y recomendaciones técnicas para el desarrollo seguro de un sistema, priorizando los principales aspectos de la Seguridad Informática (integridad, disponibilidad y confidencialidad). Permite tener registrados fallos y vulnerabilidades conocidas por tecnologías y las versiones de estas que pueden parchear o actualizar.

Preparar entorno de prueba

Esta actividad recibe la información recopilada de la caracterización, mediante la cual prepara el entorno de prueba, selecciona el tipo de prueba y las herramientas para ejecutarlas. Con la información de la tabla de caracterización y el escaneo de la red previo se realizarán las configuraciones específicas en las herramientas de pruebas, de tal forma que el análisis sea profundo, basándose principalmente en las tecnologías que se utilizaron en el desarrollo de la aplicación en prueba.

Tipo de prueba

Reactivas o basadas en experiencia:

Evaluación de vulnerabilidades: como prueba, es una evaluación de seguridad informática realizada por escáneres de vulnerabilidades web, que tienen como objetivo la detección y análisis de fallos y vulnerabilidades que afecten la confidencialidad, la integridad y la disponibilidad de los sistemas web.

1. Escaneo autorizado
2. Escaneo no autorizado

Revista Cubana de Ciencias Informáticas
Vol. 15, No. Especial UCIENCIA I, Septiembre, 2021
ISSN: 2227-1899 | RNPS: 2301
http://rcci.uci.cu
Pág. 333-349

En el procedimiento se propone utilizar las dos clasificaciones de escaneo de evaluación de vulnerabilidades. Hasta ahora se han descrito las actividades a realizar que indican el qué hacer y el cómo se trabaja proponiendo herramientas específicas para llevar a cabo el procedimiento.

Escáneres de vulnerabilidades

Herramientas diseñadas para realizar análisis en sitios web a nivel de servidores. Proponemos utilizar para el procedimiento: Nikto, Owas Zap y Acunetix para encontrar vulnerabilidades en los sitios.

Fase de ejecución

La segunda fase del procedimiento es donde se ejecutan las pruebas y se recogen los resultados para el análisis posterior de las vulnerabilidades con el equipo de desarrollo. Las actividades en esta fase son las siguientes:

1. Ejecutar las pruebas

   Se ejecutan las pruebas escogidas con las herramientas seleccionadas y configuradas de acuerdo a la caracterización de las aplicaciones en cuestión. Mientras se ejecutan las pruebas las herramientas van mostrando reportes sobre las vulnerabilidades encontradas en el camino.

2. Registrar vulnerabilidades

   Finalizada la actividad de ejecución de las pruebas, se generan los reportes con las vulnerabilidades detectadas en las aplicaciones. Estas vulnerabilidades son guardadas para un análisis posterior y para evaluar con el equipo de desarrollo el impacto de las mismas en el sistema.

3. Evaluar resultados de reportes

   El probador genera una descripción de las vulnerabilidades detectadas para analizar con los programadores o miembros de equipo de desarrollo del sistema. Se analiza el impacto, tipo de vulnerabilidad y una posible recomendación de solución.

Revista Cubana de Ciencias Informáticas
Vol. 15, No. Especial UCIENCIA I, Septiembre, 2021
ISSN: 2227-1899 | RNPS: 2301
http://rcci.uci.cu
Pág. 333-349

Fase de cierre

Es la fase donde se verifican que se haya cumplido satisfactoriamente todo el proceso de pruebas. Se realiza una evaluación del impacto de las vulnerabilidades detectadas con el equipo de desarrollo y se informan de forma oficial, teniendo en cuenta los principios y valores con los que debe contar un probador como es la ética profesional y la sinceridad y honestidad. Los probadores deben ser capaces de informar las vulnerabilidades de forma constructiva.

1. Reconciliar resultados

   En esta actividad, con los resultados obtenidos en las pruebas y los reportes de vulnerabilidades de que cada herramienta se comprueban la cantidad de estas incidencias que pasan la validación o no. Son definidos los Falsos positivos o Falsos negativos con el equipo de desarrollo y el impacto que pueden tener en el software si son explotados por ataques externos.

1. Falso positivo: vulnerabilidades detectadas por las herramientas, pero no representan una amenaza para el sistema.
2. Falso negativo: incidencias que no fueron detectadas por los escáneres pero que están presente en el sistema y son verdaderas amenazas para el software.

Tabla 2 -Ejemplo de clasificación de vulnerabilidades.

Vulnerabilidades	Clasificación	Impacto
Existen formularios HTML sin protección CSRF implementada.	Falso positivo	Medio
Hay formularios HTML sin XXS-protección correctamente configurado	Falso positivo	Medio
Las cabeceras seguras HTTPS no están configuradas	Falso negativo	Alta

Revista Cubana de Ciencias Informáticas
Vol. 15, No. Especial UCIENCIA I, Septiembre, 2021
ISSN: 2227-1899 | RNPS: 2301
http://rcci.uci.cu
Pág. 333-349

El impacto está dado por el daño que puede causar un incidente de seguridad explotando un aguajero y materializando una amenaza sobre alguna vulnerabilidad detectada en el sistema sujeto a prueba. Esta clasificación del impacto la emiten las herramientas de pruebas que se ejecutan.

1. Registrar No conformidad

Las vulnerabilidades detectadas que sean analizadas con el equipo de desarrollo y sean aceptadas como fallos en el software, pasan a ser registradas en la herramienta de gestión de proyectos que se decida utilizar, como No conformidades de tipo seguridad. Estas No conformidades deben ser monitoreadas y revisadas por un conjunto de pruebas de regresión (pueden ser manuales o automatizadas) en la siguiente iteración de las pruebas con el objetivo de verificar la solución de las vulnerabilidades en el sistema.

Resultados y discusión

Para la valoración de la propuesta en la solución del problema planteado, fue necesario realizar una encuesta para obtener los criterios de expertos. La selección de los expertos se hizo a través del análisis curricular. Participaron 17 expertos con más de diez años en la industria del software y de diferentes organizaciones desarrolladoras de software a nivel nacional.

Proceso de selección de expertos

Dada la variedad de instituciones que desarrollan software de la industria cubana y las características que esta posee, se hizo necesario tomar una muestra de expertos que tuviesen conocimientos amplios relacionados con la calidad, la gestión de la calidad y la evaluación de seguridad como característica de calidad del producto, en proyectos de software. Se realizó una valoración inicial de los posibles expertos para la validación del procedimiento, considerando la experiencia práctica como el principal factor en esta investigación. Los criterios iniciales para la selección de expertos se listan a continuación:

1. Experiencia laboral en la industria de software de 10 años.

2. Producción científica enfocada al objeto a evaluar.
3. Haber desempeñado roles relacionados con las pruebas de software.

Al tener en cuenta estos criterios se realizó un cuestionario para el resumen curricular. Como resultado se seleccionaron 37 expertos a nivel nacional, de instituciones como: DESOFT, XETID, UCI y CUJAE. Incrementando los que han participado en la investigación en etapas anteriores con expertos internacionales con más de diez años en la industria del software y más de cinco años como consultores de la mejora de procesos de software.

Luego de seleccionados los expertos, teniendo en cuenta los conocimientos que poseen y el origen de los mismos y con el objetivo de validar la propuesta, se aplicó el método Delphi, el cual tiene un amplio uso en varias áreas del conocimiento a nivel internacional y a nivel nacional, ha sido empleado fundamentalmente en investigaciones educativas y médicas, aunque en los últimos años se ha empleado en investigaciones de Ciencias Informáticas con el objetivo de socializar, externalizar y combinar el conocimiento de los expertos. El método aprovecha los elementos comunes en el grupo de expertos, preserva el anonimato mediante el uso de flujos de comunicación y permite la participación de expertos que se encuentren geográficamente dispersos (Trujillo, 2014).

A partir de los resultados arrojados se pudo contrastar que todas las categorías son evaluadas de Muy altas o Altas, validando la contribución del procedimiento en la solución del problema de investigación. Para todas las categorías se obtuvo una moda[1] de Alta o Muy Alta. Los expertos no emitieron votos en la escala de Baja (2) o Ninguna (1). A partir de los votos emitidos por los expertos se obtiene una relevancia de 94.7, pertinencia de 94, coherencia de 88.3, comprensión de 96.7 y una exactitud de 64.7.

Teniendo en cuenta estos resultados se puede decir que los expertos coinciden en que el procedimiento propuesto incorpora las buenas prácticas propuestas en los modelos, normas y estándares más utilizados. Adicionalmente se recibieron sugerencias por parte de los expertos:

1. Retroalimentar el procedimiento a partir de su empleo en los proyectos.

2. Incorporar el diseño de cada una de las pruebas a ejecutar para evaluar la seguridad como característica de calidad del producto.

Conclusiones

1. La realización de pruebas de software es importante para reducir el riesgo de fallo en operación, sin embargo, muchas veces se realizan luego de finalizado el producto y solo se ejecutan pruebas funcionales.
2. El procedimiento propone fases, actividades y herramientas que integra el qué hacer a partir de las actividades con el cómo probar a partir de proponer herramientas específicas para realizar las pruebas y tipos de pruebas.
3. Los resultados reafirman la necesidad de complementar el proceso de pruebas con un procedimiento para evaluar la seguridad como característica de calidad del producto y se obtienen criterios positivos de los expertos.

Referencias

Proaño Escalante, Rodrigo Arturo, Saguay Chafla, Ciro Napoleón, Jácome Canchig, Segundo Bolívar, Sandoval Zambrano, Fanny. Sistemas basados en conocimiento como herramienta de ayuda en la auditoría de sistemas de información. 2017. http://ingenieria.ute.edu.ec/enfoqueute/

Marin Diaz, Aymara, Trujillo Casañola, Yaimí, Buedo Hidalgo, Denys. Estrategia de pruebas para organizaciones desarrolladoras de software. 2020. ISSN: 2227-1899 | RNPS: 2301 http://rcci.uci.cu

Marin Diaz, Aymara. Marco de trabajo con un enfoque por componentes para gestionar actividades de calidad. La Habana: s.n., 2018.

Yepes Guevara, Ricardo. Un lenguaje para especificar pruebas de seguridad de caja negra automatizadas para sistemas Web. Universidad de Antioquia, Facultad de Ingeniería, 2017, Medellín, Colombia. http://bibliotecadigital.udea.edu.co/bitstream/10495/13266/1/YepesRicardo_2017_Lenguajespecificarpruebas.pdf

Mera Paz, Julián Andrés. Análisis del proceso de pruebas de calidad de software. 2016. http://dx.doi.org/10.16925/in.v12i20.1482

Rojas-Montes, Martha Lucía, Pino-Correa, Francisco José, Mauricio Martínez, James. Proceso de pruebas para pequeñas organizaciones desarrolladoras de software. 2015. Revista Facultad de Ingeniería (Fac. Ing.), Mayo-Agosto 2015, Vol. 24. No. 39

Casas, Valentina Hernández. Dashboard con indicadores para la gestión de riesgos de seguridad de la información de una empresa en Medellin. 2020.

Agarwal, n. And s. Z. Hussain a closer look on Intrusion Detection System for web applications. arXiv preprint arXiv:1803.06153, 2018.

Amit, i. I. Ptes: Penetration Testing Execution Standard. In.: The Penetration Testing Execution Standard, 2012.

Antunes, n. And m. Vieira Assessing and Comparing Vulnerability Detection Tools for Web Services: Benchmarking Approach and Examples. IEEE Transactions on Services Computing, 2015, 8(2), 269-283.

Bajovic, v. Criminal Proceedings in Cyberspace: The Challenge of Digital Era. In E.C. VIANO ed. Cybercrime, Organized Crime, and Societal Responses: International Approaches. Washington. EE.UU: Springer International Publishing Switzerland, 2017, p. 87-101.

Baş seyyar, m., f. Ö. Çatak and e. Gül Detection of attack-targeted scans from the Apache HTTP Server access logs. Applied Computing and Informatics, 2018/01/01/ 2018, 14(1), 28-36.

Bhandari, s., w. B. Jaballah, v. Jain, v. Laxmi, et al. Android inter-app communication threats and detection techniques. Computers & Security, 2017, 70, 392-421.

Calzavara, s., r. Focardi, m. Squarcina and m. Tempesta Surviving the Web: A Journey into Web Session Security. ACM Computing Surveys, 2017, 50(1), 13.

Dadkhah, m., m. Lagzian and g. Borchardt Academic Information Security Researchers: Hackers or Specialists? Science and Engineering Ethics, 2018, 24(2), 785-790.

Dalalana bertoglio, d. And a. F. Zorzo Overview and open issues on penetration test. Journal of the Brazilian Computer Society, 2017, 23(1), 2.

Dong, y., y. Zhang, h. Ma, q. Wu, et al. An adaptive system for detecting malicious queries in web attacks. Science China Information Sciences, 2018, 61(3), 032-114.

Franklin, j., c. Wergin and h. Booth cvss implementation guidance. National Institute of Standards and Technology, NISTIR-7946, 2014.

Huang, h. C., z. K. Zhang, h. W. Cheng and s. W. Shieh Web Application Security: Threats, Countermeasures, and Pitfalls. Computer, 2017, 50(6), 81-85.

Jhaveri, m. H., o. Cetin, c. Ga, t. Moore, et al. Abuse Reporting and the Fight Against Cybercrime. ACM Computer Surveys, 2017, 49(4), 1-27.

Knowles, w., a. Baron and t. Mcgarr the simulated security assessment ecosystem: Does penetration testing need standardisation? Computers & Security, 2016, 62, 296-316.

Mansfield-devine, s. Open-source software: determining the real risk posed by vulnerabilities. Network Security, 2017, 2017(1), 7-12.

Martínez, s., v. Cosentino and j. Cabot Model-based analysis of Java EE web security misconfigurations. Computer Languages, Systems & Structures, 2017, 49, 36-61.

Meucci, m. And a. Muller owasp Testing Guide 4.0. Edtion ed. EE.UU: OWASP Foundation, 2014. 224 p.

Montesino perurena, r., w. Baluja garcía and j. Porvén rubier Gestión automatizada e integrada de controles de seguridad informática. Ingeniería Electrónica, Automática y Comunicaciones, 2013, 34(1), 40-58.

Morrison, p., b. H. Smith and l. Williams 2017. Surveying Security Practice Adherence in Software Development. In Proceedings of the Proceedings of the Hot Topics in Science of Security: Symposium and Bootcamp, Hanover, MD, USA2017 ACM, 3055312, 85-94.

Nazir, s., s. Patel and d. Patel Assessing and augmenting SCADA cyber security: A survey of techniques. Computers & Security, 2017, 70, 436-454.

Rahalkar, s. A. Certified Ethical Hacker (CEH) Foundation Guide. Edtion ed. Pune, Maharashtra: Springer, 2016. 207 p.

Sandhya, s., s. Purkayastha, e. Joshua and a. Deep. Assessment of website security by penetration testing using Wireshark. In 2017 4th International Conference on Advanced Computing and Communication Systems (ICACCS). 2017, p. 1-4.

Seacord, r. C. Java Deserialization Vulnerabilities and Mitigations. In 2017 IEEE Cybersecurity Development (SecDev). 2017, p. 6-7.

Shugrue, d. Fighting application threats with cloud-based WAFs. Network Security, 2017, 2017(6), 5-8.

Singh, a. And k. Chatterjee Cloud security issues and challenges: A survey. Journal of Network and Computer Applications, 2017, 79, 88-115.

Stock, a. V. D., b. Glas, n. Smithline and t. Gigler owasp Top 10 2017. The Ten Most Critical Web Application Security Risks. Edtion ed. EE.UU: The OWASP Foundation, 2017. 50 p.

Topper, j. Compliance is not security. Computer Fraud & Security, 2018, 2018(3), 5-8.

Vásquez ojeda, Agustín Wilmer. Diseño de un Sistema de Gestión de Seguridad de Información para la empresa Neointel SAC basado en la norma ISO/IEC 27001: 2013. 2020

Wang, r., g. Xu, x. Zeng, x. Li, et al. TT-XSS: A novel taint tracking based dynamic detection framework for DOM Cross-Site Scripting. Journal of Parallel and Distributed Computing, 2017.

Contribuciones de los autores

1. Conceptualización: Aymara Marin Diaz

2. Curación de datos: Noel Harrinso Hidalgo Reyes
3. Análisis formal: Roberto Menejías García
4. Adquisición de fondos: Aymara Marin Diaz
5. Investigación: Roberto Menejías García
6. Metodología: Yaimí Trujillo Casañola
7. Administración del proyecto: Aymara Marin Díaz
8. Recursos: Yaimí Trujillo Casañola
9. Software: Roberto Menejías García
10. Supervisión: Aymara Marin Díaz
11. Validación: Roberto Menejías García
12. Visualización: Noel Harrinso Hidalgo Reyes
13. Redacción – borrador original: Noel Harrinso Hidalgo Reyes
14. Redacción – revisión y edición: Aymara Marin Díaz