Original research / Artículo original / Pesquisa original - Tipo 1

Security control for website defacement

Oscar Eduardo Mondragón Maca, MSc / eduardo.mondragon@sikkerdata.co

Andrés Felipe Mera Arcos, MSc / felipe.mera@sikkerdata.co

Sikkerdata SAS, Bogotá-Colombia

Christian Urcuqui, MSc / ccurcuqui@icesi.edu.co

Andrés Navarro Cadavid, Ph.D / anavarro@icesi.edu.co

Grupo de Investigación i2t. Universidad Icesi, Cali-Colombia

ABSTRACT Cyber-attacks to websites are increasing steadily affecting the integrity and availability of information, so the implementation of safeguards to mitigate or reduce to acceptable levels the risks generated are necessary. Computer incidents produce economic and reputational impacts to different organizations. It has identified an increase in computer attacks on different organizations where one of them, and highly reputational impact, is the "Defacement" attack, which consists of unauthorized modification or alteration to the web sites, affecting the integrity of information. The result of this article proposes the development of a model for establishing a security control to perform the containment and reporting of this attack type, which currently have focused on the websites of the government entities. The development model allows online control the attack on Web sites by constant reading of certain parts of the source code making the detection and maintenance of the integrity of information.

KEYWORDS Defacement; web application; security; vulnerability; web security; integrity.

Validación y pruebas de un control de seguridad para defacement en sitios web

RESUMEN Los ataques cibernéticos a sitios web constantemente afectan la integridad y disponibilidad de la información, esto hace necesaria la implementación de salvaguardas capaces de mitigar o reducir a niveles aceptables los riesgos generados. Los incidentes informáticos producen impactos económicos y de reputación para diferentes organizaciones. Se ha identificado un aumento en los ataques informáticos en diferentes organizaciones, uno de ellos con impacto altamente reputacional, el ataque defacement, que consiste en la modificación no autorizada o alteración de los sitios web, lo cual afecta la integridad de la información. Este artículo presenta el desarrollo de un modelo para establecer un control de seguridad para realizar el confinamiento y reporte de este tipo de ataque, el cual actualmente se ha centrado en los sitios web de las entidades gubernamentales. El modelo de desarrollo permite el control en línea del ataque a sitios web mediante la lectura constante de ciertas partes del código fuente, lo que permite la detección y el mantenimiento de la integridad de la información.

PALABRAS CLAVE Defacement; aplicación web; seguridad; vulnerabilidad; seguridad web; integridad.

Validação e testes de um controle de segurança para defacement em sítios web

RESUMO Os ciberataques em sites afetam constantemente a integridade e disponibilidade da informação, sendo necessária a implementação de salvaguardas que possam mitigar ou reduzir para níveis aceitáveis os riscos gerados. Os incidentes informáticos produzem impactos econômicos e de reputação em diferentes organizações. Foi identificado um aumento nos ataques cibernéticos em diferentes organizações, um deles com um alto impacto na reputação, o ataque defacement, que consiste na modificação não autorizada ou alteração de websites, o que afeta a integridade da informação. Este trabalho apresenta o desenvolvimento de um modelo para estabelecer um controle de segurança para fazer o confinamento e relatório deste tipo de ataque, que atualmente está focado nos sites dos órgãos governamentais. O modelo de desenvolvimento permite o controle em linha do ataque a sítios web através da leitura constante de certas partes do código fonte, permitindo a detecção e manutenção da integridade da informação.

PALAVRAS-CHAVE Defacement; aplicação web; segurança; vulnerabilidade; segurança web; integridade.

Received / Recepción: April 21, 2017 - Accepted / Aceptación: May 30, 2017 *doi: 10.18046/syt.v15i41.2442*

I. Introduction

Internet is a technology that connects around 3 billion of users in the world (Cerf & Quaynor, 2014). Furthermore, this technology permits that the people and the companies can make different works, for example: the diffusion of the information in the web pages, communication, connection, and others. On the other hand, when the technology is increasing, the cybercriminals attacks are grown (Dalai & Jena, 2011; Jericho & Munge, 2000). Therefore, is significant to create new solutions and to implement good practices in the software development, because the computer security might increase and protect the websites.

Web pages contain static or dynamic files and this depends of the information flow between the server and the web browser (unidirectional or bidirectional respectively). Usually, the cybercriminals want to modify the files into the server with the objective to deface the website or gain access to system resources for the distribution of "warez" (Stuttard & Pinto, 2011).

The web pages could be compromised with different vulnerabilities; the most representatives are: SQL injection, broken authentication and session management, cross-site scripting, insecure direct object references, security misconfiguration, insecure cryptographic storage, failure to restrict URL access, cross-site request forgery, security misconfiguration, invalidated redirects and forwards, and insufficient transport layer protection (Harper, Harris, Ness, Eagle, Lenkey, & Williams, 2015).

Web Site defacement is the process in which an attacker introduces no authorized modifications into a webpage; for instance, the web site of Gaana was defaced by a hacker through of a SQL Injection (Kumar, 2015), in 2013 a hacktivist group attacked government webpages of different countries (Wei, 2015), and the website of the army US was defaced and downed (Gross, 2015). Finally, the past situations are part of different attacks presented during last years, in fact, some statistics can obtain from the page Zone-H (www.zone-h.org); Zone-H is a security information website which has been getting around two thousands of defaced notifications.

Nowadays, we can found some techniques, methodologies, and systems for defacement attacks analysis. As an illustration, static and dynamic analysis, and the application of artificial intelligence are some techniques for evaluate and found some vulnerabilities or malicious content in the websites. Additionally, some frameworks are proposed for website analysis (Bartoli, Davanzo, & Medvet, 2010; Zhong, Asakura, Taka-

I. Introducción

Internet es una tecnología que permite conectar alrededor de tres billones de usuarios en el mundo (Cerf & Quaynor, 2014). Adicionalmente, esta tecnología permite que las personas y las empresas puedan realizar diferentes trabajos, por ejemplo: la difusión de la información en las páginas web, la comunicación y la conexión, entre otros. Por otra parte, a medida que la tecnología crece, los ataques de los ciberdelincuentes también aumentan (Dalai & Jena, 2011; Jericho & Munge, 2000), por lo que es importante crear nuevas soluciones e implementar buenas prácticas en el desarrollo de software, para aumentar así la seguridad informática y con ello proteger los sitios web.

Las páginas web pueden contener campos estáticos o dinámicos, dependiendo del flujo de información entre el servidor y el buscador web (unidireccional o bidireccional, respectivamente). Por lo general, los ciberdelincuentes desean modificar los archivos en el servidor con el objetivo de destruir el sitio web o acceder a los recursos del sistema para la distribución warez (Stuttard & Pinto, 2011).

Las páginas web podrían verse comprometidas con diferentes vulnerabilidades, las más representativas son: inyección de SQL, pérdida de autenticación y gestión de sesiones, secuencia de comandos en sitios web cruzados (XSS), referencia directa insegura a objetos, configuración de seguridad incorrecta, almacenamiento criptográfico inseguro, falla de restricción de acceso a URL, falsificación de peticiones en sitios cruzados (CSRF), configuración errónea de seguridad, redirecciones y reenvíos no validados y, finalmente, protección insuficiente en la capa de transporte (Harper, Harris, Ness, Eagle, Lenkey & Williams, 2015).

Un *defacement* de un sitio web es el proceso por medio del cual un atacante introduce modificaciones no autorizadas en una página web, por ejemplo, el sitio web de Gaana fue alterado por un hacker a través de una inyección de SQL (Kumar, 2015), en 2013 un grupo "hacktivista" atacó las páginas web gubernamentales de diferentes países (Wei, 2015), y el sitio web del ejército de EE.UU fue "derribado" (Bruto, 2015). Finalmente, las situaciones mencionadas hacen parte de los diferentes ataques que se han presentado durante los últimos años, de hecho, algunas estadísticas pueden obtenerse de la página Zone-H (www.zone-h.org); Zone-H es un sitio web de información de seguridad que ha estado recibiendo alrededor de dos miles de notificaciones destruidas.

Hoy en día es posible encontrar algunas técnicas, metodologías y sistemas para el análisis de ataques que ocasionan el *defacement*. Como ilustración, el análisis estático y dinámico y la aplicación de la inteligencia artificial son algunas técnicas para evaluar y encontrar vulnerabilidades o contenido malicioso en los sitios web. Adicionalmente, se han propuesto marcos de trabajo para el análisis de sitios web (Bartoli, Davanzo, & Medvet, 2010; Zhong, Asakura, Takakura, & Oshima, 2015; Kim & Spafford, 1994). Por otro lado, se han desarrollado algunos sistemas, por

ejemplo: Tripware (Kim & Spafford, 1994), Nagios (Aman, Yamashita, Sasaki y Kawahara, 2014), entre otros.

De acuerdo con Urcuqui, García, Osorio y Navarro (2016), una aplicación web puede estar expuesta a diferentes vulnerabilidades, las cuales ocurren generalmente por las malas prácticas durante el desarrollo del software, la falta de conocimiento en seguridad informática, los ataques cibernéticos, los ataques de día cero y las soluciones de software personalizadas (Stuttard & Pinto, 2011). En conclusión, una solución con un cien por ciento de seguridad y efectividad es demasiado difícil de hallar; por lo tanto, la seguridad informática es una línea que requiere de un trabajo continuo a través de nuevas técnicas o herramientas para reducir los riesgos en un sistema.

Este artículo se expone un control de seguridad basado en la detección de anomalías, es decir, la solución permite supervisar y corregir el *defacement* en páginas web. Este trabajo se ha organizado así: la sección II incluye los trabajos previos en técnicas y sistemas que permiten resolver el problema de *defacement* en la web; la sección III describe el control de seguridad; la sección IV contiene los resultados experimentales y los resultados de la implementación del control en dos entornos de producción; y por último, la sección IV se enfoca en las conclusiones y los trabajos futuros.

II. Revisión de técnicas de detección de ataques web

Los ataques web se pueden evaluar con dos enfoques (Zhong et al., 2015): detección basada en firmas y detección basada en anomalías. El objetivo de la detección basada en firmas es evaluar los ataques a través de una base de datos que contiene información sobre las características de las cargas útiles malintencionadas; la detección basada en anomalías, por su parte, tiene dos partes, la primera actividad crea un perfil del sitio web con características benignas, la segunda se dedica a monitorear y detectar anomalías en la comunicación con el perfil creado.

A. Detección basada en firmas

Roesch (1999) explica la utilidad Snort, un *sniffer* y registrador que puede ser utilizado para el sistema de detección de intrusiones [NIDS]. La solución utiliza unas reglas características para detectar el contenido malicioso (ataques) en el sistema de red, cada regla es una clave que representa los servicios no proporcionados para el sistema. Durante la captura de la información de red, la herramienta extrae y compara todos los paquetes de transferencia con las reglas configuradas, si la herramienta detecta una anomalía, genera una alarma para el administrador del sistema. Snort posee, entre otras, las siguientes reglas (Caswell, Beale, & Baker, 2007): P2P, puertas traseras, ataques de rechazo del servicio, ataques web y virus.

PSigene es un IDS que sirve para reunir y generar automáticamente firmas generales para solicitudes benignas y maliciosas (Howard, Gutiérrez, Arshad, Bagchi, & kura, & Oshima, 2015; Kim & Spafford, 1994). In addition, some systems have been developed, for example: Tripware (Kim & Spafford, 1994), Nagios (Aman, Yamashita, Sasaki, & Kawahara, 2014), and others.

According to state of art, a web application can be exposed to different vulnerabilities, usually it happens because the bad practices during the software development, lack of knowledge in computer security, cybercrimes increase, zero-day attacks, and custom built software solutions (Stuttard & Pinto, 2011). In conclusion, a solution with 100% in security and effectivity is too difficult; therefore, computer security is a line that needs a constant work through new techniques or tools to reduce the risks in a system.

This article contains information about a security control based in anomaly detection, that is, the solution permits to monitor and to correct the defacement in web pages. This work has been organized as follows. Section I includes the previous works in techniques and systems that permits to solve the problem of web defacement. After that, Section II describes our security control. Then, Section III contains the experimental and the results of the implementation control in two different production environments. Finally, Section IV is focused in some conclusions and future works.

II. Revision on web attack detection techniques

Web attacks can be evaluated with two approaches (Zhong et al., 2015): signature based detection and anomaly based detection. Signature based detection has the objective to evaluate the attacks through a database that contains information about the features of malicious payloads. On the other hand, Anomaly based detection has two phases, the first activity creates a profile of the website with benign features, the second phase is dedicated to monitor and to detect anomalies into the communication with the profile created.

A. Signature based detection

Roesch (1999) explains the Snort utility. Snort is a sniffer and logger that can be used for intrusion detection system (NIDS). In fact, the solution uses features rules to detect malicious content (attacks) in the network system. Every rule is a key that represent the services not provided for the system. During the capture of the network information, the tool extracts and compares all transfer packages with the configured rules; if the tool detects an anomaly then it generates an alarm for the system administrator. Snort has the next rules (Caswell, Beale, & Baker, 2007): P2P, back doors, Attacks of denial of service, web attacks, virus, and etc.

PSigene is an IDS to gather and to make automatically general signatures for benign and malicious requests (Howard, Gutierrez, Arshad, Bagchi, & Qi, 2014). The solution develops generalized signatures through four activities. Firstly, it downloads the information of different cybersecurity websites, i.e., Security Focus, Exploit Database, PacketStorm Security, and Open Source Vulnerability Database. Moreover, the collection has 30,000 SQLi examples and 240,000 examples of benign HTTP traffic. Secondly, the activity tries to categorize the information with a clustering algorithm in order to create 159 categories: SQL reserved words, SQLi signatures from Bro, Snort, ModSecurity, and SQLi reference documents. In addition, the process creates a binary sparse matrix of size 30,000 by 159. In the other hand, evaluation process of PSigene integrates a signatures collection of Bro, Snort, and ModSec with three datasets for evaluation (traffic of SQLmap, Arachni, and Vega). Finally, the results show that the tool is more efficient than Bro and Snort (86.53% and 90.52 for 9 signatures), but, the performance not is better than ModSecurity.

B. Anomaly based detection

Bartoli et al., (2010) propose a framework (Goldrake) to analyze websites defacement with high dynamic content. Goldrake uses an anomaly detection technique with a value added; the framework can be integrated with a monitoring service without using any infrastructure installation in the website. Moreover, the training and monitoring phases have 45 sensors to extract the features for every analyzed URL. Each sensor will belong to five collections, the first collection has cardinality sensors to generate numeric data of every object (i.e., number of code lines), the second group has the sensors to compute the relative frequency of each element with their respective class (i.e., ASCII collection and the HTML etiquettes), the third collection of sensors counts the number of times each element does not appear in a lecture, in the four collection each sensor creates a HTML/XML resources tree, and the final collection of sensors searches the attributes and generates the alerts for every webpage lecture. On the other hand, the study developed a prototype with 300 dynamic resources and had a process to monitor every 6 hours during 4 months a collection of 900 defacements. Finally, the results of the evaluation show that the prototype has a good accuracy in the false positive rate and the false negative rate.

Zhong et al., (2015) expose a method which uses an abstract structure of parameters; it can be created from the capture of the HTTP requests. To begin with, the training phase has three activities (transformation, filter, and profile Qi, 2014). La solución desarrolla firmas generalizadas a través de cuatro actividades. En primer lugar, descarga la información de diferentes sitios web de seguridad cibernética, por ejemplo: Security Focus, Exploit Database, PacketStorm Security, y Open Source Vulnerability Database; adicionalmente, la colección tiene 30.000 ejemplos de SQLi y 240.000 ejemplos de tráfico HTTP benigno. En segundo lugar, la actividad intenta categorizar la información con un algoritmo de agrupación para crear 159 categorías: palabras reservadas SQL, firmas SQLi de Bro, Snort, ModSecurity y documentos de referencia SQLi; por otra parte, el proceso de evaluación de PSigene integra una colección de firmas de Bro, Snort y ModSec con tres conjuntos de datos para la evaluación (tráfico de SQLmap, Arachni y Vega). Finalmente, los resultados muestran que la herramienta es más eficiente que Bro y Snort (86.53% y 90.52 para 9 firmas), pero, el rendimiento no supera ModSecurity.

B. Detección basada en anomalías

Bartoli et al., (2010) proponen un marco –Goldrake– para analizar el defacement de sitios web con alto contenido dinámico. Goldrake utiliza una técnica de detección de anomalías con un valor agregado: el marco se puede integrar con un servicio de monitoreo sin utilizar ninguna instalación de infraestructura en el sitio web, además, las fases de entrenamiento y monitoreo tienen 45 sensores para extraer las características de cada URL analizada. Cada sensor pertenecerá a cinco colecciones, la primera tiene sensores de cardinalidad para generar datos numéricos de cada objeto, es decir, el número de líneas de código; la segunda tiene los sensores para calcular la frecuencia relativa de cada elemento con su respectiva clase, es decir, la colección ASCII y las etiquetas HTML; la tercera colección de sensores cuenta el número de veces que cada elemento no aparece en una lectura; en la cuarta, cada sensor crea un árbol de recursos HTML / XML; y la quinta colección de sensores busca los atributos y genera las alertas para cada lectura de la página web. Por otro lado, el estudio desarrolló un prototipo con 300 recursos dinámicos y tuvo un proceso para monitorear cada seis horas durante cuatro meses una colección de novecientos defacements. Finalmente, los resultados de la evaluación muestran que el prototipo tiene una buena precisión en las tasas de falsos positivos y falsos negativos.

Zhong et al., (2015) exponen un método que utiliza una estructura abstracta de parámetros que se puede crear desde la captura de las peticiones HTTP. Para empezar, la fase de entrenamiento tiene tres actividades: transformación, filtro y determinación de perfil. La actividad de transformación adquiere y transforma todas las solicitudes HTTP en una clase de caracteres de secuencia, cada clase se expresará en una expresión regular dentro de las siguientes categorías: alfabetos, números, símbolos, URIs, códigos postales y correos electrónicos. La segunda actividad asigna una frecuencia para todas las clases y clasifica

cada clase en un perfil por reglas de asociación. La tercera actividad detecta las anomalías en las peticiones a través del perfil creado. El rendimiento de la técnica se evaluó con los métodos de Kruegel y Kim, y el resultado fue bueno en la tasa de falsos positivos.

C. Inteligencia artificial para el análisis de defacement de sitios web

Diferentes artículos presentan la aplicación de las técnicas mencionadas, un camino prometedor es la aplicación de inteligencia artificial en ciberseguridad. Un ejemplo es la aplicación del aprendizaje automático para la clasificación de sitios web benignos y maliciosos a través de las clases de vulnerabilidades (Mohaisen, 2015). El proceso de clasificación incluye dos fases. La primera fase utilizó una máquina de vector de soporte con gradiente estocástico para clasificar los sitios web benignos y maliciosos, el modelo fue entrenado con una matriz binaria con 41 por 20, donde cada uno de los datos representa las características de los sitios web maliciosos o benignos, las características se extrajeron de un conjunto de datos compuesto por veinte mil de URLs benignas y maliciosas. En la segunda fase se implementó un clasificador para las doce vulnerabilidades, y cada algoritmo se estableció con la representación de cada vulnerabilidad. El algoritmo de clasificación tuvo un rendimiento del 93% en inyecciones y puertas traseras; sin la muestra de inyección el algoritmo tuvo un rendimiento del 96% y 91% en la detección de objetos maliciosos.

D. Soluciones existentes

Tripware (Kim & Spafford, 1994) es una herramienta de código abierto para entornos UNIX que proporciona la seguridad e integridad de la información en el sistema de archivos. El sistema tiene un proceso para monitorear y detectar cualquier cambio en el sistema de archivos, si la herramienta detecta anomalías aplica los cambios respectivos. Tripware aplica la detección basada en firma. Esta herramienta extrae algunos valores para cada archivo (tamaño, fecha de la última modificación y propietario) para generar una firma que estará en una lista de comprobación (base de datos). Fujimura y Jin (2007) aplicaron la implementación de un sistema de archivos de interpolación, utilizando una modificación de Tripware, de tal manera que el sistema tiene la capacidad de trabajar, tanto con los administradores, como con cualquier usuario del sistema.

Nagios es un sistema para monitorear, identificar y resolver problemas en la infraestructura de TI. La solución se compone de varios proyectos que incluyen, tanto comerciales, como de código abierto. El sistema genera alarmas cuando detecta cualquier modificación no autorizada en la página web (Aman et al., 2014).

III. Control de seguridad contra defacement de sitios web

El diseño del control por computadora para proteger páginas web contra ataques de defacement contempla la activación del control, el cálculo del valor de comprobadetermination). The transformation activity takes and transforms all HTTP requests to a sequence character class, each class will be expressed in a regular expression in the next categories: alphabets, numbers, symbols, URIs, ZIP codes, and emails. Furthermore, the second activity assigns a frequency for all classes and classifies each class in a profile by association rules. The last activity detects the anomalies in the requests through of the profile created. Finally, the performance of the technique was evaluated with the Kruegel and Kim methods, and the result was good in the false positive rate.

C. Artificial intelligence for website defacement analysis

Nowadays, different articles present the application of the previous mentioned techniques, but one promising path is the application of artificial intelligence in cybersecurity. One example is the application of machine learning for the classification of benign and malicious websites through the vulnerabilities classes (Mohaisen, 2015). The classification process includes two phases; the first phase used a support vector machine with stochastic gradient to classify the benign and malicious websites. The model was trained with a binary matrix with 41 by 20; each data represents the features of the malicious or benign websites. The features were extracted from a dataset composed for 20 thousand of the benign and malicious URLs. In the second phase, for the 12 vulnerabilities was implemented a classifier, and each algorithm was trained with the representation of each vulnerability. Last, the classification algorithm had a performance of 93% in injections and backdoors; without the injection sample the algorithm had a performance of 96% and 91% in the detection of malicious objects.

D. Existing solutions

Tripware (Kim & Spafford, 1994) is an open source tool for UNIX environments that supplies the security and integrity of the information in the file system. The system has a process to monitor and to detect any change in the file system, if the tool detects anomalies then it applies the respective changes. Tripware applies signature-based detection. For each file extract some values (size, date of the last modification, and owner) to generate a signature that will be in a checklist (database). Fujimura and Jin (2007) applied the implementation of an interpolation file system, using a modification of Tripware, in such way that the system has the capacity to work with both administrators and any user of the system.

Nagios is a system to monitor, identify, and resolve problems in the IT infrastructure. The solution is composed with

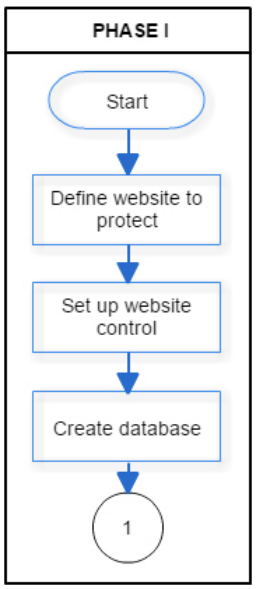

Figure. 1. Control Activation / Activación del control

several projects that include both commercial and open source. Also, the system generates alarms in the case if it detects any non-authorized modification in web page (Aman et al., 2014).

III. Security control for website defacement

The design of computer control to protect web pages against defacement attacks, contemplates the activation of the control, the checksum calculation, and the deactivation of computer control. The following describes the different phases integrated into the process of the control.

Phase I: Control Activation

In this phase, the control is activated and the web page to be protected is defined. It performs the control configuration based on the technical characteristics of the website and defines the necessary parameters to establish communication between servers. Finally, the control administrator creates and configures the database, which will store the hash integrity verification of the website. Figure 1 shows the process.

Phase II: Checksum calculation

In the second phase, the administrator performs the authentication on page management control to perform authorized actions. A script developed in Python activates the control and captures the source code of the website calculating checksums for all website and individual referral links. The hash is stored in the database and a demon starts the verification process, which verifies hash changes.

ción y la desactivación del control por computador. A continuación, se describen las diferentes fases integradas en el proceso del control.

Fase I: activación del control

En esta fase se activa el control y se define la página web a proteger, de esta forma se realiza la configuración de control basada en las características técnicas del sitio web y se definen los parámetros necesarios para establecer la comunicación entre servidores. Finalmente, el administrador de control crea y configura la base de datos, la cual almacenará la verificación de integridad a través de un hash del sitio web. La **Figura 1** muestra el proceso.

Fase II: Cálculo del valor de comprobación (checksum)

En la segunda fase, el administrador realiza la autenticación en el control de administración de páginas para realizar acciones autorizadas, un script desarrollado en Python activa el control y captura el código fuente del sitio web, y calcula los valores de comprobación para todos los sitios web y enlaces de referencia individuales. El hash se almacena en la base de datos y un demonio inicia el proceso que verifica los cambios de hash (ver **Figura 2**).

Fase III: verificación y restauración

En esta última fase se realiza la verificación de integridad del sitio web protegido. Inicialmente se compara el hash almacenado con el nuevo hash del sitio web. Si no hay cambios, el proceso de comparación continúa según lo programado, sin embargo, si los hash no coinciden, se establece una comunicación SSH entre los dos servidores y

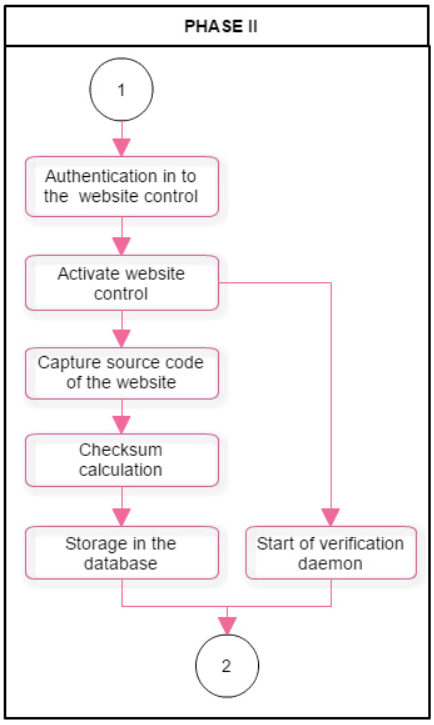

Figure 2. Checksum calculation / Cálculo del valor de comprobación (checksum)

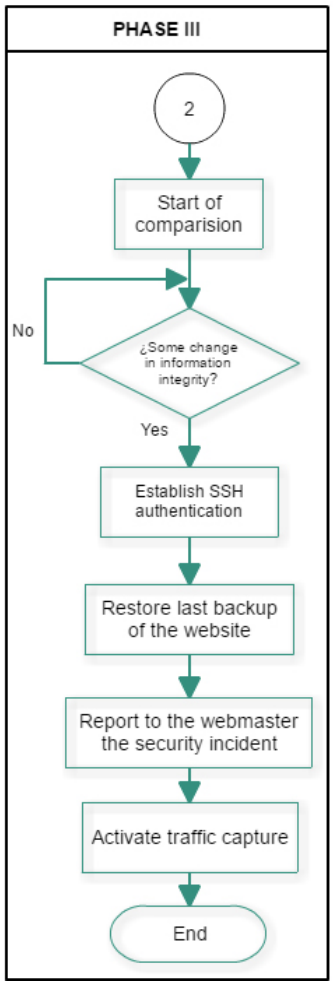

Figure 3. Verification and restore / Verificación y restauración

se restaura la última copia de seguridad almacenada en el sitio web. Inmediatamente, se envía un correo electrónico con el informe del incidente al webmaster y una captura de tráfico comienza a detectar anomalías y consecuentes ataques, lo que le permite al webmaster tomar medidas sobre el incidente. En la **Figura 3** se muestra el proceso de restauración.

IV. Metodología

En esta sección se muestra la metodología utilizada para probar la herramienta desarrollada en dos páginas web reales. Durante la prueba se utilizaron dos entornos reales con diferentes números de atributos (HREF, SRC) y complejidades (en términos de recursos, enlaces, infraestructura, etc.). En primer lugar, se utilizó una página web desarrollada en PHP con 415 atributos, en segundo lugar, se evalúo otra página web con menos atributos (149). La segunda actividad del control consistió en descargar todos los recursos de las páginas web a probar. La tercera actividad evaluó la eficiencia del control en términos de su funcionamiento en las dos páginas web. En particular, se definieron quince muestras y se registró el tiempo generado durante el proceso de cálculo del valor de comprobación.

Phase III: Verification and restore

In the last phase, the integrity check of the protected website is performed. It starts comparing the stored hash with the new hash of the website. If there are no changes, the comparison process continues on schedule. If the hashes do not match, a SSH communication is established between the two servers and the last backup stored on the website is restored. Immediately, it sends an e-mail with the report of the incident to the webmaster and a traffic capture starts to detect anomalies and consequent attacks. Therefore, the webmaster can take action on the incident. In figure 3, the restore process is shown.

IV. Methodology

In this section, we show the methodology used to test the developed tool in two real web pages. The test process integrated the following activities: During the test, we used two real environments with different number of attributes (HREF, SRC) and complexities (in terms of resources, links, infrastructure, etc.), firstly, we used a webpage developed in PHP with 415 attributes, secondly, we evaluated another webpage with less attributes (149); the second activity of the control consists to download all resources of the WebPages that will be tested; the third activity evaluates the efficiency of control in terms of its operation in the two WebPages. Particularly, we defined 15 samples and recorded the time generated during the process of calculating the checksum; and, the final activity evaluates the efficiency of the control response in time when a defacement attack occurs for the two websites. Specifically, we used 15 samples and recorded the time it takes to performed the process control and to detected the attack.

A. Technology

Table 1 shows the technology used during the testing process of the developed tool.

The two servers are located in the same LAN (Local Area Network).

V. The experiment

A. Control efficiency

The experiment evaluated the control efficiency from the beginning to the end of the process of calculating the checksum, stored in the database and comparing existing hash. The experiment is developed for two websites which vary in the number of attributes (HREF, SCR) containing the source code. It defines 15 samples and records the time it takes to perform process control.

Table 1. Resources / Recursos

Web server / *Servidor Web*	Linux Debian 8.5 MySQL 5.5.32 PHP 5.5.30 Apache 2.4.4
Control server / *Servidor de control*	Virtual Machine VmWare: 1GB RAM 45GB DD 1 processor core i7 Debian 8.5 Apache 2.4.4 MySQL 5.5.32 PHP 5.5.30 phpMyAdmin 4.0.4
Programming language / *Lenguaje de programación*	Python 3.0
Web browser / *Visor Web*	Google Chrome 47.0.2526.106 m
SSH Communication / *Comunicación SSH*	Open SSH 6.0
Server administration / *Servidor de administración*	WinSCP

Finally, the results (**Table 2**) show that the time control efficiency is better when a webpage has less attributes in its HTML. In conclusion, the time evaluated might be has a relation with the complexity algorithm that's used in the calculating checksum process.

B. Control efficiency with defacement attack

The experiment evaluates the control efficiency in detecting unauthorized modification to the two website initially defined. It develops 15 samples and records the time it takes to perform process control and detect the attack.

According to the results (**Table 3**), the optimal execution times for the script control of website 1 and 2 are set in 17 and 5 seconds respectively.

VI. Conclusions and future work

The experiment obtained the following conclusions and future works.

Table 2. Control efficiency results / Resultados del control de eficiencia

	Website 1	Website 2
Attributes / *Atributos*	415	149
Samples results (seconds) / *Muestras de resultados (segundos)*	16,7381	4,1917
	17,5476	3,8152
	16,7177	4,2373
	17,4311	5,8618
	15,9561	4,6306
	15,3000	4,2793
	16,2903	4,4191
	15,7444	3,8306
	16,4127	3,7416
	15,8721	3,8401
	16,8771	3,7822
	15,8191	3,7383
	15,7399	4,0987
	16,0878	3,7949
	16,3904	3,8378

Finalmente, la última actividad evaluó la eficacia de la respuesta de control en el tiempo al momento de producirse un ataque de defacement para los dos sitios web. Específicamente, se utilizaron quince muestras y se registró el tiempo estimado en realizar el control de proceso y detectar el ataque.

A. Tecnología

La **Tabla 1** representa la tecnología utilizada durante el proceso de prueba de la herramienta desarrollada.

Los dos servidores están localizados en la misma LAN [Local Area Network].

V. El experimento

A. Eficiencia del control

El experimento evaluó la eficiencia de control desde el principio hasta el final del proceso de cálculo de los valores de comprobación, almacenado en la base de datos y comparándolo con el hash existente. Como se indicó, el experimento se desarrolló para dos sitios web con distinto número de atributos (HREF, SCR) que contiene el código fuente. Se definieron quince muestras y se registró el tiempo estimado para realizar el control del proceso.

Los resultados (**Tabla 2**) muestran una mejor eficiencia del control de tiempo cuando una página web tiene menos atributos en su HTML. En conclusión, el tiempo evaluado podría tener una relación con el algoritmo de complejidad que se utiliza en el proceso de cálculo del valor de comprobación.

B. Eficiencia del control con ataque de defacement

El experimento evalúo la eficiencia del control en la detección de modificaciones no autorizadas para los dos sitios web inicialmente definidos. Como se explicó, se desarrollaron quince muestras y se registró el tiempo estimado para realizar el control del proceso y detectar el ataque.

De acuerdo con los resultados (**Tabla 3**), los tiempos de ejecución óptimos para el control de script de los sitios web 1 y 2 se establecieron en diecisiete y cinco segundos, respectivamente.

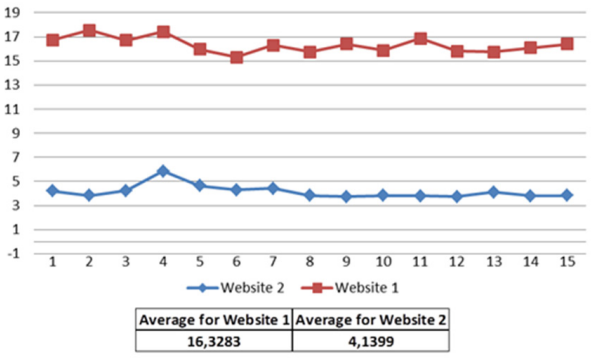

Figure 4. Control effiency in time / Control de eficiencia en tiempo

Table 3. Control efficiency with defacement attack results / Eficiencia del control con resultados de ataque de defacement

	Website 1	Website 2
Attributes / *Atributos*	415	149
Samples results (seconds) / *Muestras de resultados (segundos)*	37,9520	11,3399
	34,8841	12,0189
	35,7323	11,3679
	34,9520	12,3320
	34,8161	10,1840
	37,0160	11,6841
	35,5364	11,1921
	35,5167	11,5812
	35,3230	11,5481
	36,5643	13,1443
	35,6882	11,1164
	35,0441	13,2042
	34,9282	11,5561
	35,1843	12,1564
	35,1124	11,0152

VI. Conclusiones y trabajo futuro

El experimento obtuvo las siguientes conclusiones:

- al reducir el número de atributos (HREF, SCR) al 35,9% de los atributos iniciales del sitio web 1, el tiempo efectivo en que el script realizó la verificación se redujo al 25,3% del tiempo obtenido en el experimento realizado en el sitio web 1;
- de acuerdo con los tiempos de ejecución del control obtenidos para cada sitio web, se establece el tiempo óptimo para que el script de control realice operaciones en cada tipo de sitio web, los tiempos óptimos son diecisiete y cinco segundos para los sitios web 1 y 2, respectivamente;
- se produce un incremento del 35,4% en el tiempo de detección con respecto al tiempo de viaje de la secuencia de comandos en el sitio web 2;
- para el sitio web 1, que es más complejo, hay un aumento del 45,84% en el tiempo de detección con respecto al tiempo de viaje del script.

Como trabajo futuro se debería:

- incrementar el alcance del control para la protección de varios sitios web alojados en un único servidor o remotamente desde una sola interfaz (dash board); y
- aplicar control a páginas web con más de 415 atributos (HREF, SCR) y comprobar si el resultado medio de la eficiencia del control es similar al obtenido en los experimentos descritos en este documento.

Agradecimientos

Gracias a Colciencias y a la Universidad Icesi por su inestimable contribución para llevar a cabo esta investigación, a José Ignacio Claros por sus comentarios, a Luis Jacome por su inestimable ayuda, así como al resto de las personas que contribuyeron con este trabajo. S&T

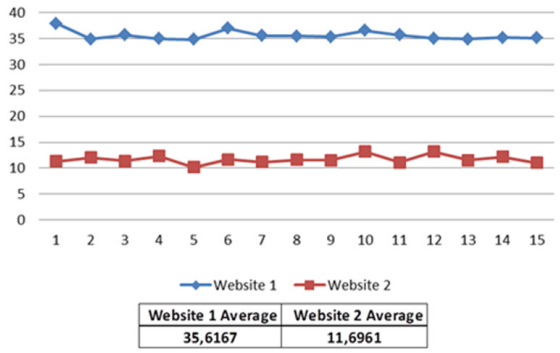

Figure 5. Control responds with defacement attack in time / Respuestas de control con ataque al defacement a tiempo

- By decreasing the number of attributes (HREF, SCR) to 35.9 % of the initial attributes of the website 1, the effective time that the script makes the verification is reduced to 25.3 % of the time gained to the experiment on website 1.
- According to control execution times obtained for each website, the optimum time is set for the control script to perform operations on each type of website. The optimal times are 17 and 5 seconds to website 1 and 2 respectively.
- There is an increase of 35.4 % in the detection time with respect to the travel time of the script on the site 2
- For website 1, which is more complex, there is an increase of 45.84 % in the detection time with respect to the travel time of the script.

As future work, it is suggested to:

- Increase the scope of control for the protection of various websites hosted on a single server or remotely from a single interface (Dash Board).
- Apply control to web pages with more than 415 attributes (HREF, SCR) and check if the average result of the efficiency of control is similar to the results obtained.

Acknowledgment

Thanks to Colciencias and the Universidad Icesi for their invaluable contribution for this work, Jose Ignacio Claros for his comments, Luis Jacome for their invaluable help and the other people that contributed in this work. S&T

References / Referencias

Aman, H., Yamashita, A., Sasaki, T., & Kawahara, M. (2014, August). Multistage growth model for code change events in open source software development: An example using development of Nagios. *In Software Engineering and Advanced Applications (SEAA), 2014 40th EUROMICRO Conference on,* (pp. 207-212). IEEE.

Bartoli, A., Davanzo, G., & Medvet, E. (2010). A framework for large-scale detection of Web site defacements. *ACM Transactions on Internet Technology (TOIT), 10*(3), Art. 10. doi:10.1145/1852096.1852098

Caswell, B., Beale, J., & Baker, A. (2007). *Snort intrusion detection and prevention toolkit.* Burlington, MA: Syngress.

Cerf, V. G. & Quaynor, N. (2014). *The Internet of everyone. IEEE Internet Computing, 18*(3), 96-96.

Dalai, A. K. & Jena, S. K. (2011). Evaluation of web application security risks and secure design patterns. *In Proceedings of the 2011 International Conference on Communication, Computing & Security,* (pp. 565-568). New York, NY: ACM.

Fujimura, N. & Mei, J. (2007). Implementation of file interpolation detection system. *In Proceedings of the 35th annual ACM SIGUCCS fall conference,* (pp. 118-121). New York, NY: ACM.

Gross, G. (2015, June). US Army website defaced, then brought down. Retrieved from: http://www.pcworld.com/article/2932936/us-army-website-defaced-then-brought-down.html

Harper, A., Harris, S., Ness, J., Eagle, C., Lenkey, G., & Williams, T. (2015). Gray hat hacking: The ethical hackers handbook. New York, NY: McGraw-Hill.

Howard, G. M., Gutierrez, C. N., Arshad, F. A., Bagchi, S., & Qi, Y. (2014, June). pSigene: Webcrawling to generalize SQL injection signatures. *In Dependable Systems and Networks (DSN), 2014 44th Annual IEEE/IFIP International Conference on,* (pp. 45-56). IEEE.

Jericho & Munge. (2000). *Hard-core web defacement statistics trends and analysis* [video]. Retrieved from: https://www.youtube.com/watch?v=7nrDoH4GZV0

Kim, G. H. & Spafford, E. H. (1994, November). The design and implementation of tripwire: A file system integrity checker. *In: Proceedings of the 2nd ACM Conference on Computer and Communications Security,* (pp. 18-29). New York, NY: ACM.

Kumar, M. (2015, May). Gaana.com hacked, 10 million user´s details exposed. Retrieved from: http://thehackernews.com/2015/05/gaanacom-hacked-10-million-users.html

Mohaisen, A. (2015, November). Towards automatic and lightweight detection and classification of malicious web contents. *In Hot Topics in Web Systems and Technologies* (HotWeb), 2015 Third IEEE Workshop on, (pp. 67-72). IEEE.

Roesch, M. (1999, November). Snort: Lightweight intrusion detection for networks. *In LISA, 99*(1), 229-238.

Stuttard, D. & Pinto, M. (2011). The web application hacker's handbook: Finding and exploiting security flaws. Indianapolis, IN: John Wiley & Sons.

Wei, W. (2015, November). Rise in website: Defacement attacks by hackers around the world. retrieved from: http://thehackernews.com/2013/11/rise-in-website-defacement-attacks-by.html

Zhong, Y., Asakura, H., Takakura, H., & Oshima, Y. (2015, July). Detecting malicious inputs of web application parameters using character class sequences. *In Computer Software and Applications Conference (COMPSAC), 2015 IEEE 39th Annual,* (Vol. 2, pp. 525-532). IEEE.

Zone-H [Web site]. Retrieved from: http://www.zone-h.org

CURRICULUM VITAE

Oscar Eduardo Mondragón Maca Engineer in Electronics and Telecommunication (Universidad del Cauca, Popayán-Colombia) and Master in Informatics and Telecommunications (Universidad Icesi, Cali-Colombia). He has participated in two projects focused in information security developed by the Universidad Icesi's i2t research group. He is founder partner of Sikkerdata SAS, company dedicated to cyber security / Ingeniero en Electrónica y Telecomunicaciones de la Universidad del Cauca (Popayán, Colombia) y Máster en Informática y Telecomunicaciones de la Universidad Icesi (Cali, Colombia). Ha participado en dos proyectos del área de ciberseguridad implementados por el grupo de investigación i2t de la Universidad Icesi. Es socio fundador de Sikkerdata SAS empresa dedicada a la seguridad informática.

Andrés Felipe Mera Engineer in Electronics and Telecommunication (Universidad del Cauca, Popayán-Colombia) and Master in Informatics and Telecommunications (Universidad Icesi, Cali-Colombia). He has participated in two projects focused in information security developed by the Universidad Icesi's i2t research group. He is founder partner of Sikkerdata SAS, company dedicated to cyber security / Ingeniero en Electrónica y Telecomunicaciones de la Universidad del Cauca (Popayán, Colombia) y Máster en Informática y Telecomunicaciones de la Universidad Icesi (Cali, Colombia). Ha participado en dos proyectos del área de ciberseguridad implementados por el grupo de investigación i2t de la Universidad Icesi. Es socio fundador de Sikkerdata SAS empresa dedicada a la seguridad informática.

Christian Camilo Urcuqui Systems Engineer (emphasis in Management and Computing) and Master in Informatics and Telecommunications from Universidad Icesi (Cali-Colombia). Member of Informatics and Telecommunications research group [i2t]. His areas of interest include: artificial intelligence, machine learning and security applied to informatics / Ingeniero de Sistemas con énfasis en Administración e Informática y Máster en Informática y Telecomunicaciones de la Universidad Icesi (Cali-Colombia). Miembro del grupo de investigación en Informática y Telecomunicaciones [i2t] de la Universidad Icesi. Sus áreas de interés incluyen: inteligencia artificial, aprendizaje de máquina, y seguridad informática.

Andrés Navarro Cadavid Full professor and Director of i2t (Informatics and Telecommunications research group) at the Universidad Icesi (Cali, Colombia). Electronics Engineer and Master in Technology Management (Universidad Pontificia Bolivariana de Medellín (Colombia), and Ph.D. in Telecommunications (Universidad Politécnica de Valencia, España). His main areas of interest are: spectrum management, radio propagation and m-health / Profesor titular y Director del Grupo de Investigación en Informática y Telecomunicaciones (i2T) de la Universidad Icesi de Cali (Colombia). Es Ingeniero Electrónico y Magister en Gestión de la Tecnología de la Universidad Pontificia Bolivariana de Medellín (Colombia) y Doctor Ingeniero en Telecomunicaciones de la Universidad Politécnica de Valencia (España). Sus áreas de interés son: la gestión del espectro radioeléctrico, la radio-propagación y las soluciones móviles aplicadas a salud.