Módulo de instalación y módulo de seguridad para *SQL Injection*, para el Juez en Línea OJ+.

Universidad De Carabobo

Facultad Experimental de Ciencias y Tecnología

Licenciatura en Computación

Módulo de instalación y módulo de seguridad para *SQL Injection*, para el Juez en Línea OJ+.

Tutores:

Prof. Daniel H. Rosquete

Prof. Amadís A. Martínez

Autor:

Br. Moisés A. Alvarado C.

Trabajo Especial de Grado presentado ante la Universidad de Carabobo cómo Credencial de Mérito para optar al Título de Licenciado en Computación.

Bárbula, Octubre 2015

Universidad De Carabobo

Facultad Experimental de Ciencias y Tecnología

Licenciatura en Computación

Módulo de instalación y módulo de seguridad para *SQL Injection*, para el Juez en Línea OJ+.

Resumen

El Juez en Línea OJ+ es un sistema que tiene como finalidad servir de apoyo a las materias que hacen uso de la programación como pilar fundamental de enseñanza y a los maratones de programación, los cuales son aspectos que requieren de un desarrollo previo de habilidades específicas para así poder obtener resultados óptimos y satisfactorios. El presente trabajo está enfocado en el desarrollo de un módulo de seguridad que comprende encriptado de claves, un segundo módulo, también de seguridad, que abarca el blindaje contra ataques basados en *SQL Injection* y un tercer módulo de instalación que permitirá posicionar adecuadamente los archivos críticos del sistema en lugares específicos y con los permisos bien establecidos basados en roles, con la finalidad de evitar problemas de intrusión por parte de un usuario no deseado.

Palabras Claves: *Script*, Instalación, *SQL Injection*, Juez en Línea, Automatizado, *Hash*.

Tutores:

Prof. Daniel H. Rosquete

Prof. Amadis A. Martinez

Autor:

Br. Moisés A. Alvarado C.

Bárbula, Octubre 2015

Dedicatoria

Dedicado a mis padres, Marco Alvarado y Luz Gisela Claro, por todo el apoyo que me han brindado a lo largo de mi vida. Eternamente agradecido con ustedes.

Dedicado a Iveth Chávez, por compartir tantos momentos agradables juntos, por darme ánimos en los momentos bajos y por simplemente ser y estar.

Dedicado a todas aquellas personas que hicieron de mí un mejor ser humano y que de alguna manera u otra influyeron en la realización de esta meta. ¡Muchas Gracias!

Agradecimientos

Para todos aquellos seres humanos que influyeron en la realización de este Trabajo Especial de Grado: ¡Muchas Gracias!

Índice

Índice de Figuras

Introducción

El presente trabajo de investigación está enfocado en el desarrollo de los módulos que se encargan de la instalación y de algunos aspectos de seguridad del Juez en Línea OJ+, aportando un grado de protección adecuado para una herramienta de esta índole y, además, un proceso de instalación rápido y sencillo.

Son tres (3) los módulos a desarrollar, los cuales son: instalador, capa de seguridad contra ataques basados en *SQL Injection*¸ y un algoritmo *Hash* para codificar las contraseñas de inicio de sesión de los usuarios. Dada la naturaleza de este proyecto especial, el cual es un trabajo que requiere del desarrollo de *software* y de llevar a cabo pruebas para obtener un producto final lo más refinado posible, se propone encarar esta investigación haciendo uso de las metodologías de desarrollo iterativas e incrementales del Modelo en Espiral junto al Modelo Evolutivo.

El presente trabajo de investigación se encuentra estructurado en cuatro (4) capítulos como se describe a continuación:

1. **Capítulo I:** En este capítulo es presentado el planteamiento del problema, objetivo general, objetivos específicos y justificación de la investigación.
2. **Capítulo II:** Son detalladas las bases teóricas de la investigación y los antecedentes de la misma.
3. **Capítulo III:** Se encuentra especificada la metodología de desarrollo utilizada, las evidencias del logro de los objetivos específicos y las actividades realizadas durante la implementación de la metodología de desarrollo elegida.
4. **Capítulo IV:** Se presentan las conclusiones, recomendaciones y trabajos futuros de esta investigación.

Capítulo I

El Problema

Planteamiento del problema

El desarrollo de habilidades es sumamente importante para obtener resultados óptimos en una labor. Realizar constantes prácticas es necesario para establecer un dominio sobre una habilidad determinada, tal es el caso de la programación, que es una actividad en la cual el planteamiento anteriormente expuesto se aplica con el fin de poder lograr la agilidad y naturalidad requerida al momento de desarrollar e implementar una solución algorítmica a un problema.

La realización de ejercicios prácticos de programación es una de las formas en la que, los estudiantes, pueden lograr adquirir las habilidades y técnicas necesarias al momento de obtener resultados óptimos y satisfactorios. Realizar problemas prácticos le permitirá a los estudiantes un entendimiento concreto de estructuras de datos, técnicas algorítmicas y tópicos avanzados de programación (Skiena & Revilla, 2003).

A nivel universitario, específicamente en la Facultad Experimental de Ciencias y Tecnología de la Universidad de Carabobo, se presenta la necesidad de realizar

prácticas para las asignaturas que usan la programación como pilar fundamental de enseñanza, como lo son las materias que comprenden diseño de algoritmos, en las que el estudiante tiene la necesidad de realizar ejercicios prácticos y se encuentra con inconvenientes como lo son la falta de un repositorio de ejercicios para resolver, ausencia de ejercicios corregidos con los cuales, dadas unas entradas (*inputs*) comparar sus resultados (o*utputs*), incompatibilidad de horarios por parte de los profesores con sus alumnos para realizar las correcciones, entre otros (Quirós, 2009).

Otro ambiente en el cual la problemática previa hace presencia son las maratones de programación. Un maratón de programación es una actividad que proporciona a los participantes la oportunidad de poner en uso sus habilidades mediante la resolución algorítmica de un conjunto de problemas dados, en un lenguaje determinado por el competidor, entre varios que propone el organizador (Programación, 2007).

Existen diversas maratones de programación de clase mundial organizadas por diferentes organizaciones, como lo son la ICPC (*International Collegiate Programming Contest)* realizada por la *Association for Computing Machinery* (ACM, s.f.), *International Olympiad in Informatics* (IOI) (Piele, s.f.), entre otras. A la fecha de este trabajo de investigación, las maratones de programación son utilizadas por

diversas empresas (Appirio, s.f.), (Google, s.f.), (Hackerrank, 2014), (Facebook, s.f.), entre otras como método de reclutamiento para la selección adecuada de personal.

A lo largo del tiempo han sido desarrollados diversos sistemas de corrección automática (Eldering, Johnson, Kinkhorst, & Tobias, 2014), (Labs, s.f.)**,** (Sharif Judge, 2014)**,** (Revilla & Garcia de Celis, s.f.), principalmente enfocados hacia el área de maratones de programación, dejando por fuera el ámbito didáctico o académico. Lo antes expuesto es el motivo por el cual, los repositorios de problemas que presentan estos sistemas no tienen un buen criterio de organización, debido a que están agrupados por años, pero no por dificultad, tópico o propósitos específicos.

Una herramienta de esta índole, tiende a sufrir ataques de seguridad de diversos tipos, ya que al ser competencia de programadores, siempre hay personas que intentan obtener el código de los otros participantes para conseguir también puntos por cada problema resuelto. Es por ello que la presente herramienta implementa técnicas para mitigar o eliminar el impacto que los ataques de índole informático puedan tener sobre la misma. Un ataque común a este tipo de herramienta viene siendo aquellos que están basados en *SQL Injection*, los cuales tienen como finalidad vulnerar la base de datos del sistema para así exponer información sensible y comprometer la integridad de los datos que se encuentran almacenados en la misma. Otro tipo de ataque al que se

encuentra expuesta una herramienta como esta es el *cracking* (Press, 2014) de las contraseñas de inicio de sesión de los usuarios del sistema, ataque que consiste en aplicar diversas técnicas dirigidas a romper, descifrar o borrar las contraseñas o cualquier mecanismo de seguridad de las mismas (González & Benítez, 2004).

Objetivos

Objetivo General

Desarrollar módulo de instalación y módulo de seguridad para *SQL Injection*, para el Juez en Línea.

Objetivos Específicos

1. Investigar sobre los estudios previamente realizados referentes a los módulos, con la finalidad de establecer una base teórica.
2. Delimitar la permisología de los usuarios en el sistema.
3. Indagar referente a la construcción de *scripts* de sistema en plataforma de software libre.
4. Diseñar los *scripts* que permitirán la instalación del sistema, haciendo uso de un lenguaje de programación adecuado.
5. Implementar los *script*s que permitirán la instalación del sistema, haciendo uso de un lenguaje de programación adecuado.

6. Examinar documentación sobre la seguridad *web*, enfocada al área de la prevención de ataques basados en *SQL Injection*.
7. Determinar las vulnerabilidades del sistema enfocadas a *SQL Injection.*
8. Corregir las vulnerabilidades del sistema enfocadas a *SQL Injection.*
9. Investigar sobre la seguridad *web*, enfocada al área de encriptado de datos.
10. Determinar un algoritmo de encriptado de datos a usar.
11. Implementar un algoritmo de encriptado de datos.

Justificación e Importancia de la investigación

El Departamento de Computación de la Facultad Experimental de Ciencias y Tecnología (FaCyT) de la Universidad de Carabobo, cuenta con un sistema basado en DomJudge (Eldering, Johnson, Kinkhorst, & Tobias, 2014) para correcciones automáticas y organización de los problemas, el cual es utilizado para maratones de programación únicamente, mas no cuenta con un sistema automatizado de corrección de código fuente propio, por lo tanto, al momento de realizar las correcciones de proyectos, talleres o tareas de programación, surge una situación particular. Dicha situación viene dada por la corrección manual de los códigos fuentes por parte de los profesores, trayendo como consecuencia tiempos de espera prolongados, entre otros.

Se presenta una herramienta conocida como Juez en Línea OJ+, pensada para la corrección de código fuente e ideal para ser utilizada en las maratones de programación,

como también en diversas materias que hacen uso de la programación como pilar fundamental de enseñanza. Esta herramienta cuenta con una base de datos la cual almacena, de forma organizada, ejercicios junto a sus respectivos casos de entrada (*inputs*), casos de salida (*outputs*) y códigos fuente.

Dicha herramienta cuenta con un instalador, el cual fue desarrollado utilizando el lenguaje de programación Python en su versión 3.4. El instalador anteriormente mencionado permite posicionar adecuadamente los archivos críticos del sistema en lugares específicos y con los permisos bien establecidos basados en roles, con la finalidad de evitar problemas de intrusión por parte de un usuario no deseado. El objetivo del instalador con el cual cuenta esta herramienta, es permitir que la misma no solo atienda necesidades de la Facultad Experimental de Ciencias y Tecnología (FaCyT), sino también hacer de ella un código abierto, fácilmente utilizable y configurable.

"La seguridad es un tema que debe inquietar a cualquier organización que hoy día decida conectar su red a Internet debido a las estadísticas para tomar conciencia del riesgo que se corre: el número de incidentes contra sistemas conectados casi se duplica cada año, según el *Computer Emergency Response Team Coordination Center* (CERT-CC). No debe extrañar, si se tiene en cuenta el vertiginoso crecimiento de Internet en

los últimos años, que implica, por una parte, nuevas redes susceptibles de ser atacadas y por otra, nuevos atacantes en potencia, además como la ley de Moore expresa que aproximadamente cada dos años se duplica el número de transistores en un circuito integrado, lo cual permite que las computadoras se hagan eficientes al momento de violentar estos sistemas de seguridad" (Ibáñez Martínez, Gómez Skarmeta, & Martínez Barberá, 1977). Por lo que al tener computadores más veloces, un *Cracker* (Press, 2014), puede violentar estos sistemas de seguridad más rápido.

Es importante hacer énfasis en que una gran parte de los sistemas *web* tienen implementadas bases de datos, las cuales son una herramienta altamente utilizada para almacenar información persistente. Esta información debe resguardarse de usuarios malintencionados.

Existen diversos ataques que tienen como finalidad la extracción de datos o causar daños a las bases de datos, tal es el caso de ataques basados en *SQL Injection*, los cuales hacen uso de una técnica donde un atacante crea o altera comandos *SQL* existentes para exponer datos ocultos, sobreponerse a los que son importantes, o peor aún, ejecutar comandos peligrosos a nivel de sistema en el equipo donde se encuentra alojada la base de datos (php, 2014). Es por esto que la presente herramienta implementa técnicas de protección que ayudan a mitigar o eliminar los riesgos de

ataques basados en *SQL Injection*, como por ejemplo *Parameterized Queries* (Wichers, Manico, & Seil, 2015).

Mantener seguras las contraseñas que utilizan los usuarios para iniciar sesión es una tarea de gran importancia. No solo basta con resguardar la base de datos contra posibles ataques, sino también se hace necesario codificar dichas contraseñas antes de almacenarlas, para evitar que las mismas, en caso de ser robadas, comprometan la integridad de la herramienta o las cuentas de otros servicios de los usuarios, siempre y cuando no utilicen contraseñas distintas (PHP, 2015).

Diversos métodos son utilizados para codificar las contraseñas de inicio de sesión de los usuarios en un sistema. Uno de ellos consiste en aplicar algoritmos de encriptado tipo *one way* o irreversibles, los cuales se conocen como algoritmos *Hash* (Halderman, Waters, & Felten, 2005).

Hacer uso de un algoritmo *Hash* para codificar las contraseñas antes de almacenarlas en la base de datos supone un método para dificultar el *cracking* (Press, 2014) de las mismas, ya que le impide al atacante conocer de manera fácil la contraseña original (PHP, 2015).

Es por lo anteriormente expuesto que esta herramienta implementa un algoritmo de *Hash* de contraseñas para brindar este tipo de seguridad y, además, hacer del Juez en Línea OJ+ una herramienta más segura y confiable para los usuarios y administradores.

Capítulo II

Marco Teórico

El marco teórico o referencial se define como soporte principal del estudio, donde se amplía la descripción del problema y se integra la teoría con la investigación, permite ubicar, dentro del contexto de ideas y planteamiento.

Antecedentes de la Investigación

A fin de sustentar esta investigación fueron revisados diversos estudios previos que aportaron una base sobre los temas en cuestión y permitieron sustentar ideas para este trabajo investigativo.

(Quirós, 2009) En su investigación titulada "Sistema *Web* basado en la Filosofía de Software Libre para la Corrección de Ejercicios de Programación", cuyo objetivo principal era ofrecer a los estudiantes una herramienta práctica para poder complementar sus estudios teóricos de una manera eficaz y afianzar los conocimientos adquiridos que le permitieran la solución algorítmica de un problema. La misma se relaciona con el actual trabajo en la propuesta de la creación de los jueces de corrección automática en línea, los cuales son presentados como una solución efectiva y eficiente para cubrir, en gran parte, la necesidad de una herramienta de asistencia para la ejercitación práctica.

(Yánez, 2011) En su investigación titulada "Entorno de Casos de Prueba para la Resolución de Problemas de Programación" en la cual el objetivo principal era la

creación de una herramienta que permitiera la realización de búsqueda eficiente de problemas de programación utilizando Ajax como técnica de desarrollo. Esta se relaciona con el trabajo actual en la implementación de un sistema de búsqueda el cual permita el fácil acceso a diversos problemas de programación para la posterior verificación de los resultados obtenidos, permitiendo así disponer de una herramienta rápida y eficaz la cual ayude a mejorar los conocimientos en el desarrollo de software.

Por otra parte, cabe destacar el estudio realizado por Ricardo Acosta (Acosta, 2015) titulado "Base de datos, Seguridad e Integración *Web* para el Juez en Línea". Este trabajo de investigación, tuvo como objetivo principal el modelado de la base de datos, seguridad e integración *web* para el Juez en Línea OJ+ a través del diseño del *Frontend* y *Backend*. Dicha investigación se relaciona con el presente trabajo de investigación en el aporte de la base de datos del Juez en Línea OJ+ y del sistema de archivos que representan la interfaz *web* del mismo. El aporte anteriormente mencionado, resulta de utilidad para el proceso de instalación.

Bases Teóricas

Automatización

Es un método donde se trasfieren tareas de producción, realizadas habitualmente por operadores humanos a un conjunto de elementos tecnológicos (Sistema automatizado, 2011).

Entre las características de un sistema automatizado se encuentran: Mejorar la productividad de la empresa, reduciendo los costes de la producción y mejorando la

calidad de la misma, mejorar las condiciones de trabajo del personal, suprimiendo los trabajos pesados e incrementando la seguridad, realizar las operaciones imposibles de controlar intelectual o manualmente, mejorar la disponibilidad de los productos, pudiendo proveer las cantidades necesarias en el momento preciso, simplificar el mantenimiento de forma que el operario no requiera de grandes conocimientos para la manipulación del proceso productivo e integrar la gestión y producción global. (Sistema automatizado, 2011).

Bash

Bash es el intérprete de comandos de consola propio del Proyecto GNU. *Bash* está destinado a ajustarse a los estándares IEEE, POSIX, P1003.2/ISO y 9945.2 (Free Software Foundation, s.f.).

Diseño

El Diseño es una actividad abstracta que implica programar, proyectar, coordinar una larga lista de factores materiales y humanos, traducir lo invisible en visible, en definitiva, comunicar. Incluye juicios de valor, aplicaciones de conocimientos, adquisición de nuevos conocimientos, uso de intuiciones y toma de decisiones. El diseño es una actividad creativa cuyo propósito es establecer las cualidades multifacéticas de objetos, procesos, servicios en su ciclo completo de vida. (The International Council of Societies of Industrial Design. , 2015).

Encriptado

El cifrado (encriptado) es la ciencia matemática de códigos, cifras y mensajes secretos. A lo largo de la historia, la gente ha utilizado el cifrado para enviar mensajes entre sí que no podían ser leídos por cualquier persona además del destinatario (The Electronic Frontier Foundation, 2014).

Hash de contraseñas

El *Hash* de contraseñas es una de las consideraciones de seguridad más elementales que se deben llevar a la práctica al diseñar una aplicación que acepte contraseñas de los usuarios. Sin *Hashing*, cualquier contraseña que se almacene en la base de datos de la aplicación podrá ser robada si la misma se ve comprometida, con lo que inmediatamente no sólo estaría expuestas las cuentas de la aplicación, sino también las de otros servicios de nuestros usuarios, siempre y cuando no utilicen contraseñas distintas (The PHP Group, 2015).

Instalación

Proceso por el cual los archivos de programas son transferidos a un dispositivo con el fin de ser configurados, y preparados para ser ejecutados en el sistema del dispositivo, para cumplir las funciones por las cuales fueron desarrollados en primera instancia (IEEE, 1990).

Interfaz gráfica de usuario (GUI)

Es un tipo de interfaz que se caracteriza por permitir que el usuario interactúe con un dispositivo electrónico a través de imágenes, en vez de líneas de comando. Las interfaces de usuario pueden estar en computadoras, dispositivos móviles como celulares y tabletas, o incluso, en dispositivos más simples, como un reproductor mp3, entre otros. Una interfaz de usuario representa la información y acciones disponibles al usuario en forma de botones, íconos, entre otros elementos (Redondo, 2013).

Internet

Internet es un conjunto descentralizado de redes de comunicación interconectadas, que utilizan la familia de protocolos TCP/IP, garantizando que las redes físicas heterogéneas que la componen funcionen como una red lógica única, de alcance mundial. Sus orígenes se remontan a 1969, cuando se estableció la primera conexión de computadoras, conocidas como ARPANET, entre tres universidades en California y una en Utah, Estados Unidos (Redondo, 2013).

Internet incluye aproximadamente 5000 redes en todo el mundo y más de 100 protocolos distintos basados en TCP/IP, que se configura como el protocolo de la red. Los servicios disponibles en la red mundial de PC, han avanzado mucho gracias a las nuevas tecnologías de trasmisión de alta velocidad, se ha logrado unir a las personas con videoconferencias, ver imágenes por satélite, observar el mundo por cámaras *web*, hacer llamadas telefónicas gratuitas, o disfrutar de un juego multijugador en 3d, un libro en formato PDF, o álbumes y películas para descargar (Redondo, 2013).

Juez en Línea

Es un sistema de corrección automática utilizado en competiciones de programación así como también en prácticas para dichas competiciones, y cuya característica principal es su implementación como una plataforma *web* (Quirós, 2009).

Maratón de programación

Los Maratones de Programación son eventos de tipo competitivo que proporcionan a estudiantes universitarios la oportunidad de trabajar en equipo para demostrar sus habilidades en la resolución de problemas algorítmicos, a través del desarrollo de programas en un determinado lenguaje, en el menor tiempo posible. Una de las habilidades fundamentales requeridas en este evento es la programación, la cual se adquiere con actividades como, por ejemplo, prácticas frecuentes de ejercicios de programación. (JIMENEZ & L., 2012)

Un referente histórico de las maratones de programación es la *Association for Computer Machinery* (*ACM*), una de las principales organizaciones educativas en cuanto a informática en todo el mundo y que ha promovido este tipo de competencias desde 1977. Por medio de las maratones de programación, tanto la *ACM* como la industria y la academia animan y centran la atención del público en la próxima generación de profesionales de la algoritmia mientras que incentivan a sus participantes en la búsqueda por la excelencia. (JIMENEZ & L., 2012).

Permisos

Consisten en asignar a un determinado usuario ciertos privilegios, bien sea en archivos o directorios.

Los permisos son reglas asociadas a los objetos de un equipo o red, como archivos y carpetas. Los permisos determinan si se puede obtener acceso a un objeto y lo que se puede hacer con él (Microsoft, 2014).

PHP

PHP (PHP: HyperText Pre-processor) es un lenguaje de programación incrustado en HTML; la mayor parte de su sintaxis es tomada de otros lenguajes, como C, Java y Perl, en conjunto con funciones específicas. El objetivo de este lenguaje es permitirle a los desarrolladores la creación de páginas dinámicas, rápidamente. (The PHP Group, 2015).

PyQt

Es un conjunto de *bindings* para Python v2 y v3 del *framework* QT y es compatible con todas las plataformas en las cuales es compatible dicho *framework*. Los mencionados *bindings* son implementados como un conjunto de módulos de Python y contienen cerca de mil clases (Limited, 2015).

Python

Python es un lenguaje de programación de alto nivel, interpretado y Orientado a Objetos con semántica dinámica. Su alto nivel en las estructuras de datos, junto con tipado dinámico y unión dinámica, lo hacen muy atractivo para el desarrollo rápido de aplicaciones, así como para su uso como un lenguaje de *script*ing (Python Software Foundation, 2015).

QT

Biblioteca multiplataforma utilizada para la creación de aplicaciones que requieran de una GUI, así como también para el desarrollo de programas, como por ejemplo consolas de servidores, que no tienen la necesidad de una interfaz (QT, 2015).

QT utiliza el lenguaje de programación C++ de forma nativa, sin embargo puede ser usado en otros lenguajes, como por ejemplo Python, a través de *bindings* (QT, 2015).

Script

En programación, un *script* es un programa o secuencia de instrucciones que se interpretan o son llevadas a cabo por otro programa y no por el procesador de la computadora (Rouse, 2005).

Seguridad

La seguridad en la *web* es un conjunto de procedimientos, prácticas y tecnologías para proteger a los *servidores* y usuarios de la *web* y las organizaciones que los rodean. La seguridad es una protección contra el comportamiento inesperado, es el conjunto de procedimientos, estrategias y herramientas que permitan garantizar la integridad, la disponibilidad y la confidencialidad de la información de una entidad. (JEREZ, 2004).

Sentencias Parametrizadas (Prepared Statements)

Son un método recomendado para construir consultas en *SQL*. Este método obliga a definir previamente la sentencia *SQL* para luego incrustar cada uno de los parámetros correspondientes dentro la misma, y de esta manera, sin importar cuál sea el *input* suministrado, se construye una consulta *SQL* segura de ejecutar por el manejador de la base de datos (OWASP, 2014).

Las sentencias parametrizadas aseguran que un atacante no pueda ser capaz de cambiar la intención de las consultas *SQL* existentes, aun cuando código malicioso haya sido suministrado como parámetro (OWASP, 2014).

Sistema

Un sistema es un conjunto de partes o elementos organizados y relacionados que interactúan entre sí para lograr un objetivo. Los sistemas reciben (entrada) datos, energía o materia del ambiente y proveen (salida) información, energía o materia. (Diccionario de Informática y Tecnología, 2015)

Un sistema puede ser físico o concreto (una computadora, un televisor, un humano) o puede ser abstracto o conceptual (un *software*). Cada sistema existe dentro de otro más grande, por lo tanto un sistema puede estar formado por subsistemas y elementos, y a la vez puede ser parte de un supersistema. Un Supersistema es aquel sistema que se encuentra conformado por todos los sistemas con los cuales se relaciona éste (Carmona, 1996). Los sistemas tienen límites o fronteras, que los diferencian del ambiente. Ese límite puede ser físico o conceptual. Si hay algún intercambio entre el sistema y el ambiente a través de ese límite, el sistema es abierto, de lo contrario, el sistema es cerrado. El ambiente es el medio en externo que envuelve física o conceptualmente a un sistema. El sistema tiene interacción con el ambiente, del cual recibe entradas y al cual se le devuelven salidas. (Diccionario de Informática y Tecnología, 2015)

SQL Injection

Es una técnica donde el atacante crea o altera comandos *SQL* existentes para exponer datos ocultos, sobreponerse a los que son importantes, o peor aún, ejecutar comandos peligrosos a nivel de sistema en el equipo donde se encuentra alojada la base de datos (php, 2014).

Web

Uno de los servicios que más éxito ha tenido en Internet ha sido la *World Wide Web* (WWW, o la "*web*"), hasta tal punto que es habitual la confusión entre ambos términos, la WWW es un conjunto de protocolos que permite, de forma sencilla, la consulta remota de archivos de hipertexto, ésta fue un desarrollo posterior (1990) y utiliza Internet como medio de transmisión (Redondo, 2013).

Algunos de los servicios disponibles en Internet, aparte de la *web*, son el acceso remoto a otras máquinas (SSH y telnet), la transferencia de archivos (FTP), el correo electrónico (SMTP y POP), los boletines electrónicos (*Feeds* o RSS), las conversaciones en línea (IRC y *chats*), la mensajería instantánea, compartir archivos (P2P, P2M, Descarga Directa), la radio a la carta (*Podcast*), el visionado de video a la carta (P2PTV, *Videocast*) y los juegos en línea (Redondo, 2013).

Wrapper Script

Es un tipo de *script* de sistema, especialmente utilizado para preparar el ambiente para la ejecución de otro *script*. Tienen como finalidad hacer transparente la ejecución, configuración y paso de parámetros a otro *script* (Cooper, 2014).

Capítulo III

Marco Metodológico

Metodología de la Investigación

El trabajo de investigación científica propuesto, tiene como objetivo el diseño y desarrollo de un módulo de un sistema o *software*, por lo cual, debe estar fundamentado en los diversos procesos y métodos para la creación y desarrollo de este tipo de producto, los cuales permiten diseñar, implementar y evaluar el mismo.

Para llevar a cabo este trabajo se seleccionó, entre varias metodologías consultadas, como modelo de desarrollo de *software* el modelo en espiral, propuesto en 1988 por Barry Boehm (Boehm, 1988), el cual plantea el desarrollo de *software* como un proceso iterativo en el que varios conjuntos de cuatro fases suceden antes de obtenerse el sistema final (Santos & Rocha, 2006).

Modelo en Espiral

El modelo en espiral, como afirma (Sommerville, 2005), permite que a lo largo de cada iteración se obtengan versiones del sistema, cada vez más completas y con menos riesgos, como también, le proporciona al cliente observar resultados desde temprano, ya que el Modelo en Espiral se basa en el desarrollo de prototipos.

Llama la atención, que este modelo se fundamenta principalmente, en el análisis de riesgo y dependiendo de ellos se determinan las actividades que se van a ir realizando durante el desarrollo, es decir, no existen fechas fijas, sino que dependen del desarrollo de las fases anteriores.

Cada iteración de este método consta de cuatro actividades a realizar, como se muestra en la figura:

Figura 1: Funcionamiento del Modelo en Espiral.
Fuente: Yánez, 2010.

1. Determinación de Objetivos, Alternativas y Restricciones

En esta primera actividad se definen los objetivos, se definen las restricciones del proceso, así como del producto. Además, comprende la identificación de riesgos del proyecto y las estrategias alternativas para evitarlos, la predicción de personal, esfuerzo y costo que se requerirán para terminar las actividades y productos conocidos asociados con el proyecto, la planificación de las actividades, tareas a realizar y la definición de los requerimientos del sistema.

2. Análisis de Riesgo

Esta segunda actividad consiste en realizar un análisis detallado de todos los riesgos encontrados del proyecto en la primera actividad para con ello valorar si su presencia es o no admisible. En el caso de no serlo, desarrollar un plan que contenga una o varias alternativas para reducirlos o eliminarlos (Del Carpio Gallegos, 2006).

3. Desarrollo, Verificación y Validación

En esta tercera actividad se desarrolla el producto o prototipo acorde a las condiciones de la actividad anterior y su evaluación respectiva con respecto a los requerimientos especificados en la primera actividad. Con base en el resultado de la evaluación de riesgos, se elige un modelo para el desarrollo, que puede ser cualquiera de los existentes, tales como: formal, evolutivo, cascada, entre otros.

4. Planificación

La cuarta y última actividad consiste en la revisión y evaluación de las fases realizadas, para decidir si es necesario continuar realizando iteraciones y, en caso afirmativo, planificar las próximas actividades a realizar.

En el desarrollo de este trabajo investigativo, específicamente durante la actividad de desarrollo del modelo espiral, se va a utilizar el modelo evolutivo. A continuación se explicará aspectos importantes del mismo.

Modelo Evolutivo

El modelo de desarrollo evolutivo consiste en desarrollar una implementación del sistema inicial, exponerla a la evaluación del usuario, refinarla tantas versiones como sean necesarias hasta que se desarrolle un sistema adecuado. Una de las ventajas de este modelo es que se obtiene una rápida retroalimentación del usuario, ya que las actividades de especificación, desarrollo y pruebas se ejecutan en cada iteración (Letelier, s.f.).

Este método permite la reducción de los riesgos, ya que provee visibilidad sobre el progreso a través de sus nuevas versiones y así permite atacar los mayores riesgos desde el inicio (ALLSOFT, 2008).

Como se muestra en la Figura, el Modelo de desarrollo Evolutivo se comprende de tres actividades concurrentes:

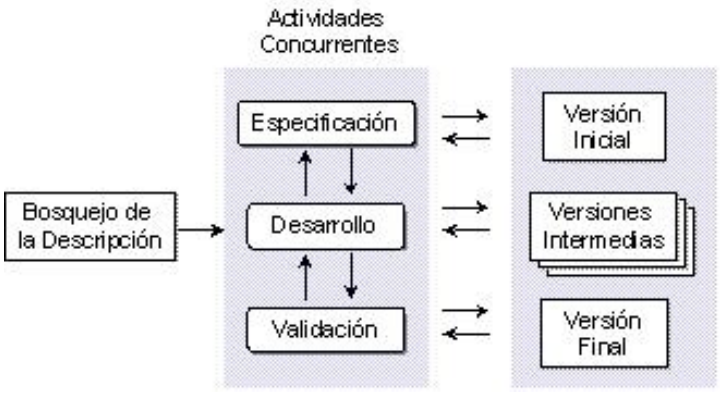

Figura 2: Modelo de Desarrollo Evolutivo.

Fuente: Sommerville, 2005.

Las actividades concurrentes se pueden definir de la siguiente manera:

Especificación

Esta etapa comprende la definición del problema y especificación inicial tomando en cuenta los requerimientos definidos, resultantes de la comunicación con el cliente. En este punto se obtiene una versión inicial del sistema que sirve de base para el desarrollo y mejora del mismo (Pardo, s.f.).

Desarrollo

En esta etapa se realiza el desarrollo del *software* en base a un proceso con énfasis en la rapidez de la liberación, de manera tal que se obtenga una versión rápida intermedia del sistema (Pardo, s.f.).

Validación

Consiste en utilizar el *software* en un ambiente de exploración, para realizar pruebas y monitorear los nuevos requerimientos. En esta etapa se obtiene una versión final de la iteración, que luego de ser evaluada, permite comprobar si el sistema está terminado, o en caso contrario, admite un replanteamiento del problema en base a los nuevos requerimientos y la iniciación de una nueva iteración (Pardo, s.f.).

Evidencias del logro de los Objetivos de la Investigación

A continuación, y en forma secuencial, se muestran las evidencias del logro de cada uno de los Objetivos Específicos, así como el Objetivo General de la presente investigación. En efecto:

1- **Investigar sobre los estudios previamente realizados referentes a los módulos, con la finalidad de establecer una base teórica.**

El resultado indica la realización de una investigación, bien sea en *papers*, libros e internet, sobre estudios previamente realizados referentes a los módulos pertinentes de este trabajo, puede ser constatada, y dichos resultados pueden ser observados en el Capítulo II del presente documento, en el cual se expresa un conjunto de basamentos teóricos pertinentes y antecedentes de gran utilidad. Estos aspectos conforman la base teórica de la presente investigación.

2- **Delimitar la permisología de los usuarios en el sistema.**

Para concretar este objetivo, se estudiaron las necesidades del sistema *web* y fueron tomados en cuenta los actores principales que hacen vida en el Juez en Línea OJ+. Dichos actores quedaron definidos de la siguiente manera:

- Usuario: Es aquel que tiene permisos para ver la lista de problemas, resolver un problema, ver las maratones en las que se encuentra, ver la tabla de posiciones. También puede gestionar su equipo, ver los miembros, cambiar el logo y editar el equipo.

- Jurado: Es un rol importante dentro del sistema, el cual puede realizar las mismas acciones que el rol previamente especificado. Además tiene permisos para juzgar en una maratón, gestionar problemas y gestionar repositorios. En cuanto a las labores de juzgar una maratón, puede: Ver la lista de maratones, ver la tabla de posiciones, ver la lista de *submits*, ver la lista de casos de prueba para cada *submit,* re-juzgar, ver y responder las clarificaciones. Para la gestión de problemas, tiene disponible las siguientes funciones: Crear, editar, borrar, agregar y eliminar casos de pruebas para un problema. Para el manejo de los repositorios, puede realizar las siguientes acciones: Agregar, listar y borrar repositorios.
- Administrador: Otro rol importante dentro del sistema, y puede realizar las funciones de los roles previamente especificados. Además puede llevar a cabo la gestión de las maratones de programación. Para cada maratón de programación tiene permisos de crear, eliminar, listar y editar una maratón de programación. Además puede gestionar el lenguaje de programación permitido en una maratón, pudiendo agregar, listar o borrar el lenguaje. Adicionalmente, tiene permisos para crear, listar, editar o eliminar los repositorios de ejercicios disponibles para una maratón.
- *Root*: Es un rol especial, ya que tiene disponibles todas las acciones de los roles previamente especificados. Adicionalmente, puede realizar las siguientes labores: Editar un rol de un usuario y ver el estado del servidor.

A fin de complementar los roles previamente descritos y aprovechando el manejo de permisos que ofrece Linux, se decidió crear un rol especial dentro del sistema operativo bajo el seudónimo "**adminjuez**", con la finalidad de establecer un nivel de seguridad adecuado para los archivos y directorios críticos del Juez en Línea OJ+.

Para efectos del Juez en Línea OJ+, los roles dentro del sistema operativo se clasifican de la siguiente manera:

- adminjuez: Este rol es el propietario de todos los archivos y directorios del Juez en Línea OJ+. Sobre dichos archivos y directorios, este rol tiene permisos para leer, escribir, modificar, eliminar y ejecutar.
- Otros: Este rol representa a cualquier otro usuario dentro del sistema operativo. Para los archivos y directorios del Juez en Línea OJ+, sólo tendrá permisos de ejecución.

3- Indagar referente a la construcción de *scripts* de sistema en plataforma de software libre.

Para cumplir con este objetivo, se realizó una investigación sobre los lenguajes de programación más comunes usados para *scripting* en sistemas basados en GNU/Linux. Se determinó que, tanto *Bash* como Python, son lenguajes de programación ideales para la realización de *scripts* bajo ambiente Linux.

Para ambos lenguajes de programación, se plantearon múltiples pruebas. Dichas pruebas consistieron en: descomprimir archivos, crear directorios,

interactuar con el Sistema Operativo mediante comandos *Shell*, manejar hilos, entre otras. Al realizar las pruebas anteriormente mencionadas, se determinó que Python es un lenguaje de programación más flexible en comparación con *Bash* para este desarrollo, por lo tanto fue elegido como el lenguaje para desarrollar el *script* de instalación.

4- **Diseñar los *script*s que permitirán la instalación del sistema, haciendo uso de un lenguaje de programación adecuado.**

En cumplimiento con este objetivo, se llegó a la conclusión de que son requeridos dos (2) *scripts* para el debido proceso de instalación. El primer *script*, se conoce como *"Wrapper Script",* y tiene como finalidad preparar el ambiente del Sistema Operativo para la correcta ejecución del *script* de instalación y, realizar ciertas verificaciones previas a la ejecución del mencionado *script*. Algunas de las verificaciones llevadas a cabo por el *"Wrapper Script"* son, como por ejemplo, asegurar que el lenguaje de programación Python esté instalado en la máquina, entre otras. En consecuencia de las labores de verificación que dicho *script* debe llevar a cabo, se decidió utilizar *Bash* como lenguaje de programación para la construcción del mismo.

El segundo *script,* el cual se decidió desarrollar usando el lenguaje de programación Python en su versión 3.4, viene siendo el *script de* instalación. Este segundo *script*, tiene como finalidad la instalación y puesta en marcha de todos los módulos que corresponden al Juez.

Para el *script* de instalación, se requirió la creación de una *GUI*. Para el diseño de la misma, se decidió utilizar el *framework* QT, sin embargo

debido a que el lenguaje previamente seleccionado para crear el *script* de instalación fue Python, se tuvo que utilizar una variante del *framework* QT la cual se conoce como PyQT, ya que dicho *framework* nativamente no es compatible con el lenguaje de programación elegido.

5- Implementar los *script*s que permitirán la instalación del sistema, haciendo uso de un lenguaje de programación adecuado.

Para el logro de este objetivo, se implementó, tanto el *Wrapper Script* como el *script* de instalación para el Juez en Línea OJ+.

El *Wrapper Script* fue implementado usando el lenguaje de programación *Bash*, y se encarga de descargar e instalar el *framework* PyQT5, verificar que Python se encuentre instalado, de no estarlo procede a instalarlo, descargar los archivos que conforman al Juez en Línea OJ+ y ejecutar el segundo *script* conocido como el *script* de instalación.

Para el s*cript* de instalación se utilizó el lenguaje de programación Python 3.4, con el cual fue implementada la lógica del mismo. Para la *GUI* del *script* anteriormente mencionado, se empleó el *framework* PyQT5.

La estrategia utilizada para el logro del diseño de la *GUI* fue la de producir prototipos de cada una de las pantallas del instalador que fueron identificadas a lo largo del desarrollo de las diferentes iteraciones de las actividades metodológicas.

Como resultado se obtuvo cuatro (4) prototipos finales, los cuales representan la *GUI* del instalador, estos prototipos son:

Figura 3: Pantalla inicial
Fuente: Autor

Figura 4: Pantalla de Configuración de Cuenta

Fuente: Autor

Figura 5: Pantalla de Verificación
Fuente: Autor

Figura 6: Pantalla de Instalación
Fuente: Autor

6- **Examinar documentación sobre la seguridad *web*, enfocada al área de la prevención de ataques basados en SQL Injection.**

Para cumplir con este objetivo, se realizó una investigación, en *papers*, libros e internet, con la finalidad de determinar los métodos para mitigar o eliminar riesgos de ataques basados en *SQL Injection*, y pudo determinarse que existen cuatro (4) métodos recomendados para dicha tarea:

1. El uso de Sentencias Parametrizadas.
2. Conversión de caracteres especiales a caracteres literales (*Escape Characters*).
3. Chequear los datos mediante patrones (*REGEX*).
4. Limitar los permisos de la base de datos dependiendo del tipo de consulta y la tabla objetivo de la consulta.

7- **Determinar las vulnerabilidades del sistema enfocadas a *SQL Injection.***

Para el logro de este objetivo, fue realizada una investigación, en *papers*, libros e internet, sobre los medios por los cuales los ataques basados en *SQL Injection* violentan la integridad de los sistemas. Se pudo determinar que los campos del tipo *input*, mediante los cuales un usuario envía datos al sistema para su posterior procesado, representan el medio para llevar a cabo este tipo de ataques, por lo que se determinó que cualquier campo de dicho tipo, representa un punto vulnerable para el sistema *web* del Juez en Línea OJ+.

8- **Corregir las vulnerabilidades del sistema enfocadas a *SQL Injection*.**

Para la realización de este objetivo, se cuenta con el conocimiento de que existen cuatro (4) métodos recomendados para mitigar o eliminar los riesgos de ataques basados en *SQL Injection*, dichos métodos son:

1. El uso de Sentencias Parametrizadas.

2. Conversión de caracteres especiales a caracteres literales (*Escape Characters*).
3. Chequear los datos mediante patrones (*REGEX*).
4. Limitar los permisos de la base de datos dependiendo del tipo de consulta y la tabla objetivo de la consulta.

Teniendo en cuenta lo anteriormente expuesto, se determinó que de los cuatro (4) métodos recomendados, dos (2) de ellos (1 y 2) son implementados por el *framework* elegido para el desarrollo del sistema *web* del Juez en Línea OJ+, el método número cuatro (4) se escapa del alcance de este trabajo investigativo y por último, el método tres (3) no es factible su implementación dada su sencillez. Por lo tanto, se recomienda el uso de los métodos 1 y 2 para la prevención de ataques basados en *SQL Injection*.

9- Investigar sobre la seguridad *web,* enfocada al área de encriptado de datos.

El resultado indica la realización de una investigación, en *papers*, libros e internet, enfocada al área de encriptado de contraseñas de usuario en sistemas *webs*. En dicha investigación se determinó que la técnica ideal de encriptar las contraseñas de los usuarios de un sistema *web*, es haciendo uso de un algoritmo de encriptado de tipo "*one way"*, mejor conocido como algoritmo *Hash*.

La investigación también permitió conocer que existen dos técnicas ideales para el encriptado de contraseñas de usuarios en sistemas *webs*. La primera técnica indica la realización de múltiples iteraciones, aplicando en cada una de ellas un algoritmo *Hash* a la contraseña que se desea codificar.

La segunda técnica, hace referencia a concatenar a la contraseña un valor conocido como "s*alt"* (PHP, 2015), el cual tiene como objetivo hacer a la contraseña significativamente más robusta, ya que elimina la posibilidad de que la misma pueda ser buscada en diccionarios pre-calculados (PHP, 2015), y como paso final aplicar una función *Hash* a dicha contraseña.

10- Determinar un algoritmo de encriptado de datos a usar.

Este objetivo, estuvo enfocado en determinar cuál de las dos (2) técnicas de encriptado de contraseñas, especificadas en el objetivo número 9 de la presente sección, sería la ideal para codificar las claves de inicio de sesión de los usuarios del Juez en Línea OJ+. En vista de que ambas técnicas son recomendadas para tal fin, se decidió desarrollar un algoritmo *Hash* híbrido, que implemente el enfoque de utilizar un *"salt",* junto a la técnica de aplicar una función *Hash* a la contraseña en múltiples iteraciones.

11- Implementar un algoritmo de encriptado de datos.

En cumplimiento con este objetivo específico, fue implementado un algoritmo de encriptado de contraseñas del tipo "*one way*" o *Hash* híbrido utilizando el lenguaje de programación PHP. Dicho algoritmo utiliza un *"salt"* de 128 caracteres, el cual se concatena con la contraseña que se desea encriptar. Luego de ello, se realizan seis (6) iteraciones. En cada iteración, se divide la contraseña entre un número predeterminado, obteniendo como resultado, a la iteración número seis (6), una cadena de caracteres codificada de longitud 32.

Actividades realizadas durante la implementación de la metodología de desarrollo

En este orden de ideas, a continuación se muestran los productos o resultados obtenidos en cada una de las actividades de la metodología de desarrollo de software conocida como Modelo en Espiral, utilizada durante la elaboración de la presente investigación. Es importante destacar que la misma es iterativa e incremental, por lo que sólo se muestran los aspectos considerados más relevantes de su implementación.

Cabe destacar que en la tercera actividad de la metodología utilizada, se hace, a su vez, uso de una metodología de desarrollo de software llamada Modelo Evolutivo, que consta de sus propias actividades.

Modelo en Espiral

Para el módulo de instalación de la herramienta fueron realizadas varias iteraciones, las cuales son descritas a continuación:

Primera Iteración del Modelo en Espiral:

1- **Determinar Objetivos:**

   a. **Objetivos:**

      - Desarrollar el *wrapper script*.

   b. **Restricciones:**

      - El equipo debe contar con el intérprete de comandos de consola *Bash* instalado.

   c. **Riesgos:**

      - Que el equipo no cuente con conexión a internet.

2- Análisis de Riesgos:

El riesgo detectado en la precedente actividad fue considerado inadmisible, ya que representa una fatalidad para el correcto funcionamiento del *wrapper script.* No es posible ofrecer un plan para eliminar o mitigar este riesgo, debido a que no se encuentra directamente relacionado con el *wrapper script*, sino con el equipo computacional en el cual se ejecute el mismo.

3- Desarrollo, verificación y Validación:

Modelo Evolutivo

I. Primera Iteración:

a. Especificación:

Diseñar el *wrapper script.*

b. Desarrollo:

Se determinaron los objetivos a cumplir por el *script* y se realizó una lista de tareas para el mismo.

c. Validación:

Se determinó que era necesario la realización de una nueva iteración, para así proceder a la implementación del *script* objetivo.

II. Segunda Iteración:

a. Especificación:

Implementar el *wrapper script.*

b. Desarrollo:

Se desarrolló un primer prototipo del *wrapper script*, el cual cumplió con la lista de tareas establecida.

c. Validación:

Se detectó la necesidad de una nueva iteración, con la finalidad de incluir nuevas tareas que no habían sido tomadas en cuenta en la iteración anterior.

III. Tercera Iteración:

a. Especificación:

Integrar al *wrapper script* tareas adicionales.

b. Desarrollo:

Tomando como punto de partida al prototipo ya obtenido, se procedió a incluir nuevas tareas al mismo, como por ejemplo:

- Detección de conexión a internet.
- Índice de progresos de descarga.
- Validación de paquetes basado en MD5.

c. Validación:

El *script* fue sometido a pruebas de funcionamiento, llegando a la conclusión de que cumple con los objetivos planteados. No fueron requeridas iteraciones adicionales.

4- **Planificación:**

Habiendo culminado la actividad de desarrollo de esta primera iteración del Modelo en Espiral, fue estudiado el estado de progreso del *script* de instalación para ese entonces, lo cual permitió llegar a la conclusión de la necesidad de realizar, al menos, dos (2) iteraciones más, con la finalidad de implementar el resto de las funcionalidades para dicho *script* de instalación.

Segunda Iteración del Modelo en Espiral:

1- **Determinar Objetivos:**

a. **Objetivos:**

- Diseñar e implementar las Interfaces Gráficas de Usuario para el *script* de instalación.

b. **Restricciones:**

- El equipo debe contar con el *framework* PyQT5 instalado.

2- **Análisis de Riesgos:**

No fueron identificados riesgos.

3- **Desarrollo, verificación y Validación:**

Modelo Evolutivo

I. **Primera Iteración:**

a. **Especificación:**

Desarrollar las diferentes Interfaces Gráficas de Usuario requeridas para el *script* instalador.

b. Desarrollo:

Fueron diseñas e implementadas las cuatro (4) Interfaces Gráficas de Usuario requeridas para el *script* de instalación.

c. Validación:

Se determinó que no era requerida alguna iteración adicional, ya que se cumplieron los objetivos planteados.

4- Planificación:

Habiendo culminado la actividad de desarrollo de esta iteración del Modelo en Espiral, fue estudiado el estado de progreso del *script* de instalación para ese momento, lo cual permitió llegar a la conclusión de la necesidad de realizar, al menos, dos (2) iteraciones más, con la finalidad de implementar las funcionalidades requeridas para las Interfaces Gráficas de Usuario y para el resto del *script* de instalación.

Tercera Iteración del Modelo en Espiral:

1- Determinar Objetivos:

a. Objetivos:

- Implementar función de chequeo de dependencias requeridas para el correcto funcionamiento del Juez en Línea OJ+.
- Integrar las Interfaces Gráficas de Usuario al *script* de instalación.

b. Riesgos:

- Que el equipo no cuente con conexión a internet.

2- Análisis de Riesgos:

El riesgo encontrado en la precedente actividad fue considerado inadmisible, por ser vital para el correcto funcionamiento del Juez en Línea OJ+. No es posible ofrecer un plan para mitigar o eliminar este riesgo, ya que el mismo no depende directamente de la herramienta, sino del equipo computacional en el cual se lleve a cabo la instalación de la misma. Fue planteado el desarrollo de una validación dentro de la función de chequeo de dependencias, la cual permita obtener información sobre la existencia de una conexión a internet. La mencionada validación tiene como finalidad detener el proceso de instalación en caso de que no sea posible acceder a la red de internet, notificándole al administrador de la ausencia del recurso.

3- Desarrollo, verificación y Validación:

Modelo Evolutivo

I. Primera Iteración:

a. Especificación:

Desarrollar una función que permita validar las dependencias requeridas para el correcto funcionamiento del Juez en Línea OJ+.

b. Desarrollo:

Fue implementada una función dentro del *script* de instalación, la cual se encarga de validar que se encuentren instaladas las dependencias necesarias para el correcto funcionamiento del Juez en Línea OJ+.

c. Validación:

Fue tomada la decisión de que es necesaria otra iteración.

De igual manera, se continuó iterando hasta el momento que se consideró que los objetivos de la primera metodología utilizada, es decir, el modelo en espiral, fueron alcanzados o cumplidos.

4- Planificación:

Habiendo culminado la actividad de desarrollo de esta iteración del Modelo en Espiral, fue estudiado el estado de progreso del *script* de instalación para ese momento, lo cual permitió llegar a la conclusión de la necesidad de realizar otras iteraciones que permitiesen implementar funcionalidades adicionales, como por ejemplo, posicionamiento y configuración de los archivos del Juez en Línea OJ+, creación y configuración de la base de datos, así como también mejoras en las interfaces, validaciones menores en los diferentes *scripts*, entre otras. Las mismas no serán descritas en esta sección por ser, en su núcleo, muy similares a las anteriormente expuestas.

Para el módulo de seguridad contra ataques basados en *SQL Injection* y para el módulo de encriptado de contraseñas, fueron realizadas diversas iteraciones. Las mismas se describen a continuación:

Primera Iteración Modelo en Espiral:

1- **Determinar Objetivos:**

   a. **Objetivos:**

      - Desarrollar e implementar una capa de seguridad contra ataques basados en *SQL Injection.*
      - Desarrollar e implementar un algoritmo *Hash* para codificar las contraseñas de inicio de sesión de los usuarios.

   b. **Restricciones:**

      - El equipo debe contar con PHP 5 o mayor, instalado.

   c. **Riesgos:**

      - Que las contraseñas de inicio de sesión de los usuarios no se encuentren codificadas al momento de almacenarlas en la base de datos.
      - Vulnerabilidades a nivel de base de datos, por el riesgo de ataques basados en *SQL Injection*.

2- **Análisis de Riesgos:**

   Los riesgos encontrados en la precedente actividad fueron considerados inadmisibles, ya que representan graves vulnerabilidades para la herramienta. Teniendo como objetivo mitigarlos o eliminarlos, se planteó el desarrollo de dos (2) módulos de seguridad, los cuales tienen como finalidad implementar las técnicas recomendadas para tal objetivo.

3- Desarrollo, verificación y Validación:

Modelo Evolutivo

I. Primera Iteración

a. Especificación:

Desarrollar e implementar una capa de seguridad para mitigar o eliminar riesgos de ataques basados en *SQL Injection*.

b. Desarrollo:

Se llegó a la conclusión de que no es factible desarrollar algo novedoso, puesto que las técnicas recomendadas para tal fin ya se encuentran implementadas en el *framework* PHP *CodeIgniter*, el cual fue utilizado para la construcción de la interfaz *web* del Juez en Línea OJ+.

c. Validación:

Se toma la decisión de que es requerida una iteración adicional.

II. Segunda Iteración

a. Especificación:

Desarrollar e implementar un algoritmo *Hash*, que tenga la tarea de encriptar las contraseñas de inicio de sesión de los usuarios.

b. Desarrollo:

Fue desarrollada una librería en PHP, la cual implementa un algoritmo *Hash* híbrido. Dicho algoritmo se encarga de encriptar las contraseñas de inicio sesión de los usuarios del Juez en Línea OJ+ antes de ser almacenadas en la base de datos.

c. Validación:

Se toma la decisión de no realizar alguna otra iteración, ya que los objetivos de la primera metodología, Modelo en Espiral, fueron cubiertos.

4- Planificación:

Al culminar de desarrollar todas las actividades de esta iteración, se llevó a cabo una revisión del progreso de los módulos de seguridad, teniendo como resultado el requerimiento de realizar más iteraciones para agregar validaciones menores a los diferentes módulos. Las mismas no serán descritas en esta sección por ser, en su núcleo, muy similares a las anteriormente expuestas.

Capítulo IV

Conclusiones y Recomendaciones

Una vez analizados los datos, se procedió a elaborar las conclusiones tomando como punto de partida los objetivos planteados, expresando en forma resumida los resultados del análisis de la investigación.

En cuanto a las recomendaciones, las mismas están enfocadas a ofrecer sugerencias sobre posibles pasos a seguir partiendo de lo presentado como proyecto de investigación.

Conclusiones

Con la finalización de este trabajo de investigación y desarrollo, se logra el añadido de módulos al sistema de corrección en línea de código fuente conocido como Juez en Línea OJ+, los cuales permiten establecer un grado mayor de seguridad y contar con un proceso de instalación rápido y sencillo.

El desarrollo de esta herramienta, representa una solución a la problemática que se vive en la Facultad Experimental de Ciencias y Tecnología (FaCyT), permitiendo a los estudiantes que deseen aumentar sus habilidades en cualquier materia que utilice la programación como pilar fundamental de enseñanza, tener a su disposición una herramienta que cuente con un repositorio de ejercicios organizado, junto a sus respectivos archivos de entrada y salida.

Este trabajo especial de grado fue tomado como una continuación de la herramienta de corrección automática de código fuente conocida como Juez en Línea OJ+, mediante la construcción de tres (3) módulos que llevarían a cabo el proceso instalación y puesta en marcha de la herramienta, así como también ofrecieran un grado mayor de seguridad a nivel de base de datos y contraseñas de inicio de sesión de los usuarios. Para el desarrollo de dichos módulos, los cuales son parte del mismo sistema, pero diferentes unos de otros, fue necesario establecer una base teórica adecuada, la cual fue consultada mediante diversas referencias, accediendo y recopilando de distintos sitios la información requerida para fundamentar el desarrollo de los mismos.

Es relevante destacar que las metodologías empleadas para la realización exitosa de este trabajo especial de grado, se ajustaron perfectamente al desarrollo del mismo, ya que de manera organizada, permitieron avanzar en la finalización de cada uno de los módulos, tomando en cuenta que se debía llevar un seguimiento sobre el avance de cada uno de ellos.

Recomendaciones

Dada la naturaleza de esta herramienta y la importancia que supone para la Facultad Experimental de Ciencias y Tecnología (FaCyT), se recomienda lo siguiente:

- Se debe reconocer que los módulos desarrollados en este trabajo especial de grado, representan una primera versión de los mismo, lo cual los hace altamente mejorables, con muchas funcionalidades adicionales, es por ello que se recomienda a la Universidad de Carabobo, en especial a la Facultad Experimental de Ciencias y Tecnología (FaCyT) incentivar la investigación de este tipo de trabajos que promuevan la innovación.

- Se recomienda impulsar la implementación del sistema dentro del recinto universitario, pudiendo llegar a ser una herramienta de estudio para las diferentes facultades que conforman la Universidad de Carabobo.

- Se aconseja hacer uso del API de *Hash* de contraseñas que ofrece el lenguaje PHP, puesto que es una librería que cuenta con años de desarrollo, pruebas y validaciones.

- Continuar el desarrollo de esta herramienta, añadiendo los módulos faltantes y de esta manera complementar lo ya desarrollado.

La idea de esta herramienta es que se mantenga en constante cambio y crecimiento, con la finalidad de que llegue a ser lo suficientemente aceptada para ser utilizada no solo en la FaCyT, sino también en las diferentes facultades que conforman la Universidad de Carabobo, así como en otros recintos universitarios.

Trabajo Futuros

Como módulos o actividades pendientes para trabajos de ampliación o mejoramiento de la herramienta a futuro se puede destacar:

- Desarrollar un instalador multiplataforma.

Referencias

ACM. (s.f.). *acm ICPC*. Obtenido de International collegiate programing contest: http://icpc.baylor.edu/

Acosta, R. (2015). Base de datos, Seguridad e Integración Web para el Juez en Línea. Carabobo, Venezuela.

ALLSOFT. (24 de Enero de 2008). *Modelos de Desarrollo*. Recuperado el 26 de Febrero de 2013, de ALLSOFT de C.V, Monterrey, N.L: http://www.slideshare.net/inventa2/modelos-de-desarrollo

Appirio. (s.f.). *Topcoder*. Obtenido de http://www.topcoder.com/

Boehm, B. (1988). *A Spiral Model of Software Development and Enhancement*. USA. Obtenido de http://portal.ou.nl/documents/114964/2986739/T24331_02.pdf

Carmona, D. H. (1996). *Teoría General de Sistemas: Un enfoque hacia la ingeniería de Sistemas.*

Cooper, M. (2014). *Advanced Bash-Scripting Guide.* Obtenido de http://www.tldp.org/LDP/abs/html/wrapper.html

Del Carpio Gallegos, J. (Mayo de 2006). *Análisis del riesgo en la administración de proyectos de tecnología de Información*. Recuperado el 26 de Febrero de 2013, de Sistemas de Bibliotecas Universidad Nacional Mayor de San Marcos: http://sisbib.unmsm.edu.pe/BibVirtualData/publicaciones/indata/vol9_n1/a13.pdf

Diccionario de Informática y Tecnología. (2015). Obtenido de http://www.alegsa.com.ar/Dic/sistema.php

Eldering, J., Johnson, K., Kinkhorst, T., & Tobias, W. (18 de Octubre de 2014). *DOMjudge*. Obtenido de http://www.domjudge.org/

Facebook. (s.f.). *Facebook*. Obtenido de https://www.facebook.com/

Foundation., T. L. (2015). *Linux Foundation*. Obtenido de http://www.linuxfoundation.org/what-is-linux

Free Software Foundation. (s.f.). *GNU Operating System*. Obtenido de http://www.gnu.org/software/bash/manual/html_node/What-is-Bash_003f.html#What-is-Bash_003f

González, M. B., & Benítez, J. G. (2004). *La web de Sistemas Operativos (SOPA).* Obtenido de Universidad de las Palmas de Gran Canaria (ULPGC).: http://sopa.dis.ulpgc.es/ii-aso/portal_aso/leclinux/seguridad/crack/crack.pdf

Google. (s.f.). *Google Code Jam 2014*. Obtenido de https://code.google.com/codejam

Hackerrank. (2014). *HackerRank for Work*. Obtenido de https://www.hackerrank.com/

Halderman, J. A., Waters, B., & Felten, E. W. (Mayo de 2005). A Convenient Method for Securely Managing Passwords.

Ibáñez Martínez, J., Gómez Skarmeta, A., & Martínez Barberá, H. (1977). *Seguridad en Internet. Parte I: Ataques, Tecnicas y Firewalls*. Obtenido de A Non sTop workerS Research Group, Univesidad de Murcia, España: http://ants.dif.um.es/~humberto/papers/1997-ati-1.pdf

IEEE. (1990). *IEEE Standard Glossary of Software Engineering Terminology.* Obtenido de http://dis.unal.edu.co/~icasta/ggs/Documentos/Normas/610-12-1990.pdf

JEREZ, C. (2004). *Seguridad para lograr Confiabilidad y Calidad de los Servicios Digitales en Internet.* México: Universidad de las Américas Puebla.

JIMENEZ, & L. (2012). *Herramienta de estudio para las maratones de programación promovidas en el programa de Ingeniería de sistemas y computación de la Universidad tecnológica de Pereira. Colombia.*

Labs, S. R. (s.f.). *Sphere Online Judge*. Obtenido de http://www.spoj.com/

Letelier, P. (s.f.). *Proceso de Desarrollo de Software*. Recuperado el 26 de 02 de 2013, de Departamento de Sistemas Informáticos y Computación, Universidad Politécnica de Valencia, España: http://www.dsic.upv.es/asignaturas/facultad/lsi/doc/IntroduccionProcesoSW.doc

Limited, R. C. (2015). *Riverbank Software*. Obtenido de https://www.riverbankcomputing.com/software/pyqt/intro

Marin, H. (Octubre de 2014). Desarrollar e integrar Evaluaciones en línea utilizando Políticas de Seguridad para el Sistema de Videoconferencia ARGOS. Barbula, Carabobo, Venezuela.

Microsoft. (2014). *Microsoft*. Obtenido de http://windows.microsoft.com/es-419/windows/what-are-permissions#1TC=windows-7

OWASP. (21 de 11 de 2014). *The Open Web Application Security Project*. Obtenido de Query Parameterization Cheat Sheet: https://www.owasp.org/index.php/SQL_Injection_Prevention_Cheat_Sheet

P. G. (2015). *PHP*. Obtenido de http://php.net/manual/es/faq.passwords.php

Pardo, P. (s.f.). *Modelo de Procesos*. Recuperado el 26 de Febrero de 2013, de Universidad Tecnológica de Chile: http://www.slideshare.net/pilypardo/modelos-de-procesos

php. (2014). Obtenido de http://php.net/manual/es/security.database.sql-injection.php

Piele, D. T. (s.f.). *International Olympiad in Informatics*. Obtenido de http://www.ioinformatics.org/index.shtml

Press, O. U. (2014). *Oxford Dictionaries*. Obtenido de http://www.oxforddictionaries.com/es/definicion/ingles_americano/cracker

Programación, C. d. (2007). *Maraton de programación FaCyT*. Obtenido de http://maraton.facyt.uc.edu.ve/

Python Software Foundation. (2015). *Python.org*. Obtenido de https://www.python.org/doc/essays/blurb/

QT. (2015). *QT - Home*. Obtenido de http://wiki.qt.io/About_Qt

Quirós, A. (2009). *Sistema Web de Corrección Automatizada de Programas basada en el Método de Pruebas de Caja Negra.* Facultad Experimental de Ciencias y Tecnología, Universidad de Carabobo, Venezuela.

Redondo, G. (2013). Argos, Sistema de Videoconferencias basado en el protocolo RTSP, para el apoyo académico a los cursos de capacitación en la Universidad de Carabobo. Carabobo, Venezuela.

Revilla, M., & Garcia de Celis, C. (s.f.). *UVa Online Judge*. Obtenido de http://uva.onlinejudge.org/

Rouse, M. (2005). *http://searchenterpriselinux.techtarget.com/*. Obtenido de http://searchenterpriselinux.techtarget.com/definition/script

Santos, M., & Rocha, V. (2006). *Modelo Em Espiral*. Recuperado el 26 de Febrero de 2013, de Faculdade de Ciência e Tecnologia, Universidade do Algarve: http://cin.ufpe.br/~cadcn/files/Monitoria/Monitoria%20-%20Gradua%E7%E3o/material%20de%20apoio/esperial.pdf

Sharif Judge. (20 de Agosto de 2014). Obtenido de http://docs.sharifjudge.ir/

Sistema automatizado. (2011).

Skiena, S. S., & Revilla, M. A. (2003). *The Programming Contest Training Manual.*

Sommerville, I. (2005). *Ingeniería del Software*. Recuperado el 26 de Febrero de 2013, de PEARSON EDUCACIÓN S.A., España: http://www.scribd.com/doc/17480126/Ian-Sommerville-Ingenieria-de-Software-Septima-Edicion

The Electronic Frontier Foundation. (03 de 11 de 2014). *Surveillance Self-Defense*. Obtenido de https://ssd.eff.org/en/module/what-encryption

The International Council of Societies of Industrial Design. . (2015). Obtenido de http://www.icsid.org/resources/design_index.htm

The PHP Group. (2015). *PHP: Hypertext Preprocessor*. Obtenido de https://secure.php.net/

Wichers, D., Manico, J., & Seil, M. (2015). *The Open Web Application Security Project*. Obtenido de https://www.owasp.org/index.php/SQL_Injection_Prevention_Cheat_Sheet

Yánez, M. A. (2011). *Entorno de casos de pruebas para la resolución de problemas de programación.*