Dom. Cien., ISSN: 2477-8818
Vol. 3, núm. 3, junio, 2017, pp. 433-460

Número Publicado el 20 de junio de 2017

http://dx.doi.org/10.23857/dom.cien.pocaip.2017.3.3.jun.433-460
URL:http://dominiodelasciencias.com/ojs/index.php/es/index

Ciencias Informáticas

Artículo Científico

Políticas de en la comunicación de la información en los Web Services del Sistema MY-SOUL para el proyecto PROMEINFO de la Universidad de Guayaquil

Security policies in the communication of the information in the Web Services system MY-SOUL for the PROMEINFO project of the University of Guayaquil

Políticas de segurança em comunicação de informações sobre serviços da Web de MY-SOUL Sistema para projeto PROMEINFO da Universidade de Guayaquil

Eleanor A. Varela-Tapia [I]
Universidad de Guayaquil
Guayaquil, Ecuador
eleanor.varelat@ug.edu.ec

Edison J. Navarro-Briones [II]
Universidad de Guayaquil
Guayaquil, Ecuador
johed_nb@hotmail.com

Iván L. Acosta-Guzmán [III]
Universidad de Guayaquil
Guayaquil, Ecuador
ivan.acostag@ug.edu.ec

Recibido: 30 de enero de 2017 * **Corregido:** 20 de febrero de 2017 * **Aceptado:** 20 mayo de 2017

I. Magister en Sistemas de Información Gerencial; Magister en Docencia y Gerencia en Educación Superior; Diplomado en Diseño Curricular por Competencias; Ingeniera en Computación; Docente Titular en la Universidad de Guayaquil.
II. Ingeniero en Sistemas Computacionales; Universidad de Guayaquil
III. Magister en Sistemas de Información Gerencial; Magister en Administración de Empresas; Ingeniero en Computación; Docente Titular en Universidad de Guayaquil

Resumen.

La implementación de Web Services ha tenido gran acogida por parte de las empresas que tienen algún sistema informático, estos Web Services no fueron creados con el objetivo de ser privados, sino públicos por lo que surge un problema de seguridad debido a que en ellos viaja información importante y estos pueden ser accedidos fácilmente por toda clase de personas desde diferentes lugares con acceso a Internet.

El Hospital Universitario de Guayaquil, en conjunto con la carrera de Ingeniería en Sistemas Computacionales, realizó una aplicación llamada MY-SOUL, la cual consiste en una red social médica con el objetivo de innovar la forma en que se realizan las consultas entre paciente y doctor.

Dicha aplicación se desarrolló sin tomar en consideración ningún estándar, norma o política de seguridad, por lo que la información se encuentra vulnerable a toda clase de ataque informático.

Este proyecto plantea una solución que minimiza el impacto en el desarrollo y los riesgos de seguridad existentes, por lo cual, se implementó políticas y estándares de seguridad en la comunicación e información que se genera a través de los Web Services con el objetivo de garantizar la integridad y la confidencialidad de la información.

Palabras Clave: Ataque informático; políticas de seguridad; red social my – soul; web services.

Abstract.

The implementation of Web Services has had large welcomed by companies that have some computer system, these Web Services were not created to be private, but public by what emerges a security problem since they travel important information and these can be accessed easily by all sorts of people from different places with Internet access.

The University Hospital of Guayaquil, in conjunction with the career of computer systems engineering, carried out an application called MY-SOUL, which consists of medical social network with the objective of innovating the way in which consultations between patient and doctor are made.

This project proposes a solution that minimizes the impact on the development and existing security risks, therefore, implemented policies and safety standards in communication and information generated through the Web Services to ensure the integrity and confidentiality of the information.

Keywords: Computer attack; security policies; social network - my soul; web services.

Resumo.

Implementar serviços da Web tem sido bem recebido pelas empresas que têm um sistema de computador, estes Web Services não foram criados com o objetivo de ser privado, mas público de modo surge um problema de segurança porque neles informações de viagem importante e estes podem ser facilmente acessados por todos os tipos de pessoas de diferentes lugares com acesso à Internet.

Hospital Universitário de Guayaquil, em conjunto com a Engenharia em Sistemas de Computação, fez uma aplicação chamada MY-SOUL, que consiste em uma rede social médica com o objetivo de inovar a forma de consulta entre paciente e médico são feitas .

Esta aplicação foi desenvolvida sem levar em consideração qualquer política padrão, regra ou segurança, assim que a informação é vulnerável a todos os tipos de ataque computador.

Este projeto propõe uma solução que minimize o impacto sobre os riscos de desenvolvimento de segurança e, portanto, políticas e normas de segurança implementadas em comunicação e informação gerada através de Web Services, a fim de garantir integridade e confidencialidade das informações.

Palavras chave: ataque computador; políticas de segurança; minha rede social - alma; Web Services.

Introducción.

Los Web Services nos facilitan el desarrollo de las aplicaciones Web y son usados por la mayoría de empresas e instituciones que cuentan con sistemas informáticos para acceder a la información que se necesite, sin embargo su implementación puede representar un riesgo en la seguridad debido a que la información viaja por Internet y en la mayoría de los casos se transmite información sensible, estas aplicaciones por sí sola no incorporan aspectos de seguridad informática vista desde los siete principios de seguridad: integridad, confidencialidad, no repudio, auditoria, autenticación, autorización y disponibilidad, esto estimula a los atacantes a interceptar la comunicación para hacer uso de su contenido.

Según (Areitio Bertolín, 2008), las organizaciones empresariales soportan su actividad las reglas del negocio en las TIC, por lo se debe dotar a los sistemas informáticos en red de políticas y medidas de protección garantizando el desarrollo y sostenibilidad de la actividad del negocio para presentes y futuras amenazas.

(Ramos, 2011), define la seguridad informática como un conjunto de procedimientos, dispositivos y herramientas encargadas de asegurar la integridad, disponibilidad y privacidad de la información en un sistema informático o intentar reducir las amenazas que pueden afectar al mismo. El objetivo de la seguridad informática comprende 3 aspectos fundamentales: confidencialidad, integridad y disponibilidad.

Según (Moisés Benítez, 2013), las políticas de seguridad informática tienen por objeto establecer las medidas de índole técnica y de organización, necesarias para garantizar la seguridad de las tecnologías de información y personas que interactúan haciendo uso de los servicios asociados a ellos y se aplican a todos los usuarios de cómputo de las empresas.

El laboratorio de redes y seguridad de la UNAN, nos indica que los estándares de seguridad son una herramienta que apoya la gestión de la seguridad informática, ya que los ambientes cada vez más complejos requieren de modelos que administren las tecnologías de manera integral, sin embargo, existen distintos modelos aplicables en la administración de la seguridad. (UNAM, 2014)

Según 27001 Academy, ISO 27001 es una norma internacional emitida por la Organización Internacional de Normalización (ISO) y describe cómo gestionar la seguridad de la información en una empresa. La revisión más reciente de esta norma fue publicada en 2013 y se actualizo el nombre a ISO/IEC 27001:2013. La primera revisión se publicó en 2005 y fue desarrollada en base a la norma británica BS 7799-2. (27001 Academy, 2014)

Entre los beneficios del uso del estándar ISO 27001: 2013 se tienen: cumplir con los requerimientos legales, obtener una ventaja comercial, mejores costos y mejores organizaciones,

En el 2014 en Ecuador en un estudio realizado por la compañía Kaspersky las cifras fueron de casi 60.090.173 detecciones, dejando como resultado más de dos millones de dólares en robos. Las cifras podrían ir en aumento debido al crecimiento económico del país, que lo convierte en un blanco interesante para los ataques cibernéticos. Hay hackers que lanzan ataques desde Perú, Colombia o Europa a las instituciones bancarias del Ecuador, lo que antes no sucedía, pues se trata es un delito importado. (Kaspersky, 2014)

Es por eso que no tener políticas de seguridad definida, implica vulnerabilidad a accesos no autorizados lo que causa muchos problemas, así como también el robo de información sensible o confidencial, lo que ocasiona daños en la imagen institucional e incluso el cierre de la misma. Por lo tanto, se debe disponer de medidas de seguridad para garantizar la integridad de la información que se transmite vía Web Services.

Según (Molina, 2014) indica "Que este tipo de incidentes podrían incrementarse si no se toma la precaución de dictar políticas para proteger los sistemas informáticos desde el Estado, que es el responsable de controlar los mandos de la economía del país y de los sistemas de tránsito, electricidad, agua potable".

La Carrera de Ingeniería en Sistemas en conjunto con la Facultad de Ciencias Médicas de la Universidad de Guayaquil realizaron la aplicación llamada MY-SOUL, esta aplicación consiste en una red social médica cuya finalidad es mejorar e innovar las consultas entre pacientes y médicos. Esta aplicación está desarrollada para que sea utilizada por millones de personas a nivel nacional e internacional ya que podrá ser accedida desde cualquier computadora con acceso a internet. (Aray Dominguez, 2015)

Para el desarrollo no se consideró un estándar de seguridad tampoco existen políticas de seguridad definidas, de aquí surge la necesidad de aplicar estándares y políticas de seguridad para la creación de Web Services y la transmisión de la comunicación que en este se realiza, con el objetivo de cuidar la información de los usuarios o pacientes que utilicen este servicio.

Situación Conflicto Nudos Críticos

La aplicación MY-SOUL es usada por un alto número de usuarios o pacientes, lo que genera un gran número de transacciones y en ellas viaja información sensible de los pacientes y la institución, al no contar con ningún tipo de seguridad esta información puede ser interceptada fácilmente por los hackers. Por lo que surge la necesidad de implementar medidas de seguridad en la comunicación, con el objetivo de garantizar que la comunicación se realice de manera íntegra y confiable.

Formulación del Problema

¿Cómo ayudará la implementación del diseño de políticas y normas de seguridad en la comunicación de la información para establecer una comunicación segura en el sistema del Hospital Universitario de Guayaquil?

Objetivo de la investigación.

Este trabajo de investigación tiene como objetivo el desarrollo de políticas y la utilización de un estándar de seguridad informática para los Web Services que se utilizan en la aplicación MY-SOUL con la finalidad de garantizar un nivel de seguridad en la aplicación y así cubrir los principios básicos de seguridad, además de esto, se pretende que estas políticas de seguridad sirvan de base para futuras innovaciones en esta área de conocimiento. Para evaluar el problema fue necesario realizar un análisis del código fuente de la aplicación MY-SOUL proporcionado por el Hospital Universitario de Guayaquil.

Materiales y métodos.

Tipo de investigación

Esta investigación es documental ya que se basa en la recolección de información mediante la lectura y referencias bibliografías que contiene información sobre el tema a tratar.

También es una investigación de proyecto factible ya que consiste en el desarrollo de una propuesta viable para solucionar un problema.

El proyecto factible se divide en tres etapas:

Diagnóstico: Es una reconstrucción del objeto de estudio para detectar algunas situaciones en donde se vea la necesidad de implementar una solución.

Factibilidad: Indica la necesidad de desarrollar un proyecto, tomando en cuenta las necesidades que se detectaron.

Elaboración de la propuesta: Desarrollo de la propuesta con el objetivo de satisfacer una necesidad detectada.

Técnica de recolección de información

Se realiza encuestas a 50 personas entre los cuales se encuentra personal de PROMEINFO, estudiantes de la Carrera de Ingeniería en Sistemas y personal encargado del proyecto que son profesionales con el objetivo de conocer que les parece la implementación de políticas de seguridad en los Web Services que se utilizan en las aplicaciones de PROMEINFO.

Metodología del proyecto

Según (Ikujiro Nonaka e Hirotaka Takeuchi, 2013) “Scrum es una metodología ágil de desarrollo de proyectos que toma su nombre y principios de los estudios realizados sobre nuevas prácticas de producción”.

Figura 1. Metodología Scrum

Fases de Scrum

Comprende las siguientes fases:

1.- Planificación de la iteración

Planificación: Definición de una nueva versión basada en el Sprint actual, junto con una estimación de coste y agenda. Arquitectura: Diseño de la implementación de las funcionalidades del Sprinta. Esta fase incluye la modificación de la arquitectura y diseño generales.

Levantamiento de Información
Capacitaciones del Proyecto con los usuarios
Capacitación del uso y funcionamiento de la Aplicación

Cuadro 1. Planificación de la iteración

2.- Ejecución de la iteración

Ejecución de Sprints: Desarrollo de la funcionalidad de la nueva versión con respeto a las variables de tiempo, requisitos, costo y competencia. La interacción con estas variables define el final de esta fase. El sistema va evolucionando a través de múltiples iteraciones de desarrollo o Sprints.

Análisis de la información del proyecto MY-SOUL
Revisión de la documentación del proyecto
Análisis de las fuentes que utiliza el proyecto
Investigación de estándares de seguridad en Web Services
Creación de ambiente de desarrollo

Cuadro 2. Ejecución de la iteración

3.- Inspección y adaptación

Preparación para el lanzamiento de la aplicación y pruebas antes del lanzamiento.

Diseño e Implementación de la solución
Levantamiento de los web services en ambiente de desarrollo
Implementación de web services Restfull getPublicacion y pruebas
Implementación de web services Restfull guardarLike y pruebas
Implementación de web services Restfull eliminarLike y pruebas
Implementación de web services Restfull addComentario y pruebas
Implementación de web services Restfull addPost y pruebas
Implementación de tokens de seguridad y pruebas

Cuadro 3. Inspección y adaptación

4.- Entregables del proyecto

De acuerdo a la metodología que se utiliza en el desarrollo de la solución, en la etapa de inspección y adaptación se deben entregar los siguientes documentos:

Pruebas y documentación
Creación de manual de implantación de web services Rest
Entrega de código fuente
Entrega de proyecto ejecutable
Manual de Políticas de seguridad en la creación de web services
Manual de usuario del a aplicación

Cuadro 4. Entregables del proyecto

Diseño de la Arquitectura General

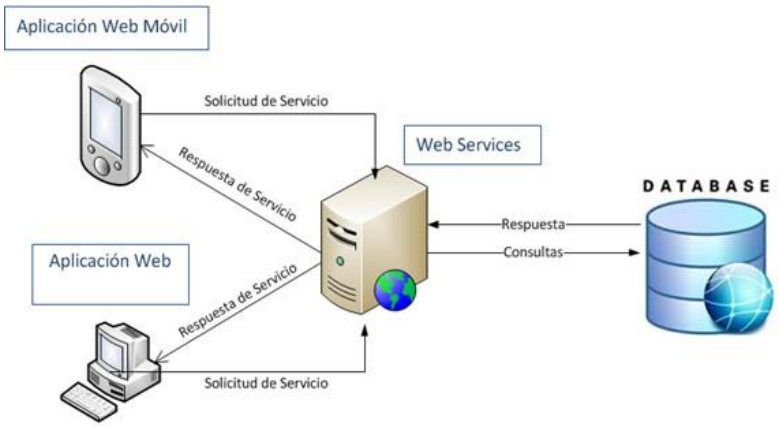


Figura 2. Arquitectura general

La solución está construida en un modelo de 3 capas:

- Capa1: Cliente (Aplicación Web Móvil y Aplicación Web Administrador).

- Capa2: Servicio (Web Services)
- Capa3: Base de Datos

Web Services

El servicio web está construido en la plataforma de programación JAVA en un modelo de capas dividido por paqueterías:

- Capa Acceso Datos (paquete com.accesoDatos)
- Capa Controlador (paquete com.manejadores)
- Capa Servicio (com.servicio)

Capa de Acceso a Datos (AD)

Es la capa encargada de las consultas CRUD (créate/read/update/delete) a la base de datos.

La capa está conformada por las siguientes clases:

- LoginDB.java
- PostBeanDB.java

Capa Controlador (Manager)

Es la capa encargada de gestionar las conversiones de datos y los accesos a las clases AD. La capa está conformada por las siguientes clases:

- LoginManager.java
- PublicacionManager.java

Capa de Servicio

Es la capa encargada de exponer los diferentes servicios y se encarga de acceder a la capa de controladores (Manager). La capa está conformada por las siguientes clases:

- ResponseHeader.java
- ResponseManager.java
- WsApplication.java
- WS_Principal.java

Servicios Web disponibles

A continuación, se detalla los diferentes servicios web que se encuentran disponibles para el funcionamiento de las aplicaciones de PDA:

Clase	Nombre del Servicio	Método Java	URL	Método
WS_Principal.java	WsRed	GetPublicaciones	/WsRed/getPublicaciones	GET
WS_Principal.java	WsRed	getcontactos	/WsRed/getContactos	GET
WS_Principal.java	WsRed	getAmigos	/WsRed/getAmigos	GET
WS_Principal.java	WsRed	guardarLike	/WsRed/guardarLike	POST
WS_Principal.java	WsRed	eliminarLike	/WsRed/eliminarLike	POST
WS_Principal.java	WsRed	addComentario	/WsRed/addComentario	POST
WS_Principal.java	WsRed	addPost	/WsRed/addPost	POST
WS_Principal.java	WsRed	addContact	/WsRed/addPost	POST
WS_Principal.java	WsRed	login	/WsRed/login	POST
WS_Principal.java	WsRed	addUser	/WsRed/addUser	POST
WS_Cliente.java	WsRed	editUser	/WsRed/editUser	POST
WS_Cliente.java	WsRed	changePass	/WsRed/changePass	POST
WS_Cliente.java	WsRed	bloquearAmigos	/WsRed/bloquearAmigos	POST
WS_Cliente.java	WsRed	updatePerfil	/WsRed/updatePerfil	POST

Cuadro 5. Servicios web disponibles

Aplicación Web

La aplicación web móvil esta estructura con el modelo MVC.

- Vista páginas ZUL
- Controlador .java y JavaScript

- Base de datos

Vista

Aquí van todas las páginas .ZUL; las páginas disponibles son:

Página	URL	Descripción
Login.zul	/redS /login.zul	Página inicial para el ingreso de la aplicación.
Registro.zul	/redS /registro.zul	Página para realizar el registro de un nuevo usuario
Pass.zul	/redS /pass.zul	Página para realizar el cambio de contraseña
ForgetPass.zul	/redS /forgetPass.zul	Página para generar una nueva contraseña
Index.zul	/redS /index.zul	Página principal de la aplicacion
editDatos.zul	/redS /index.zul	Popup para editar los datos del usuario
editImagen.zul	/redS /index.zul	Popup para cambiar la imagen de perfil del usuario
Publicación.zul	/redS /index.zul	Popup para agregar una nueva publcacion
Dinamic_Tree.zu l	/redS /index.zul	Popup para visualizar la lista de amigos
Drag.zul	/redS /index.zul	Popup para agregar nuevos usuarios a nuestra lista de amigos
Comment.zul	/redS /index.zul	Popup para ingresar un nuevo comentario

Cuadro 6. Vista de páginas .ZUL

Controlador

Como controlador tenemos clases java y librerías JavaScript.

ZK: utiliza el concepto de MVVC, esto ayuda a gestionar pantallas de una manera más sencilla, mapea a los componentes de las páginas .ZUL y gestiona los datos y actualizaciones bajo eventos, proporcionando una programación más limpia.

Los archivos principales donde se encuentran los métodos y validaciones de la aplicación:

- validacion.java
- FormValidator.java
- ecg.js
- wz_jsgraphics.js

Base de Datos

Esta aplicación tiene como base de datos SqlServer 2008Rh ya que esta es la base de datos que utiliza el hospital universitario.

Metodología de la Evaluación

Se utiliza la metodología de evaluación de riesgo Owasp, es un modelo sencillo de usar, con el fin de estimar la gravedad de los riesgos y tomar decisiones sobre qué hacer con esos riesgos. (Salgado Yánez, 2014)

Esta metodología se basa en 6 pasos los cuales se detallan a continuación:

1. Identificación de un Riesgo
2. Factores de Riesgo Estimación
3. Factores para estimar el impacto
4. Determinación de la severidad del impacto
5. Decidir que arreglar:
6. Personalización de riesgo valoración de modelo

Se propone evaluar los riesgos de la siguiente manera:

Agente de amenaza	Vectores de ataque	Prevalencia de debilidades	Detección de debilidades	Impacto técnico	Impacto al negocio
Específico de la aplicación	Fácil	Difundido	Fácil	Severo	Específico de la aplicación / negocio
	Promedio	Común	Promedio	Moderado	
	Difícil	Poco común	Difícil	Menor	

Cuadro 7. Metodología de la evaluación Owast

Resultados.

Evaluación

A continuación, se analiza punto por punto, los análisis realizados a la aplicación, y cómo han sido solventados los posibles ataques, como se muestra en cuadro 8.

Criterio	Resultado	Comentario
Inyección	No Detectado	El código ofrece seguridades que solventa esta vulnerabilidad.
Autenticación	Parcial	Debe implementarse protocolo seguro. Sin embargo, no es urgente debido a que todo corre en la intranet.
XSS	No Detectado	El código ofrece seguridades que solventan esta vulnerabilidad.
Referencia Directa Insegura	No Detectado	El código ofrece seguridades que solventan esta vulnerabilidad.
Defectuosa Configuración	No evaluado	Esta evaluación debe ser hecha por parte de Infraestructura.
Exposición de Datos Sensibles	Parcial	Debe implementarse protocolo seguro.
Ausencia Control Funciones	Parcial	El código ofrece seguridades parciales. Se sugiere que en una nueva iteración, se oculten las url's de los web services.
CSFR	No Detectado	El código ofrece seguridades que solventan esta vulnerabilidad.
Componentes Vulnerables	Parcial	Hay que implementar esquemas bajo el cual se actualicen los componentes open source que han sido utilizados.
Redirecciones no validadas	No Detectado	El código no ofrece esta vulnerabilidad.

Cuadro 8. Resumen de la evaluación

Implementación de Token de Seguridad

Autenticación con tokens

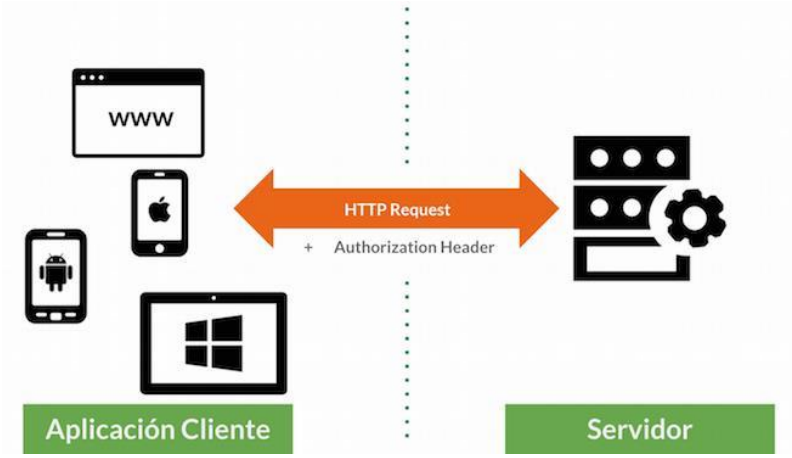


Figura 3. Autentificación con tokens

El funcionamiento es el siguiente: El usuario se autentica en nuestra aplicación, bien con un par usuario/contraseña, o a través de un proveedor como puede ser Twitter, Facebook o Google por ejemplo. A partir de entonces, cada petición HTTP que haga el usuario va acompañada de un Token en la cabecera. Este Token no es más que una firma cifrada que permite a nuestro API identificar al usuario. Pero este Token no se almacena en el servidor, si no en el lado del cliente (por ejemplo en localStorage o sessionStorage) y el API es el que se encarga de descifrar ese Token y redirigir el flujo de la aplicación en un sentido u otro.

Como los tokens son almacenados en el lado del cliente, no hay información de estado y la aplicación se vuelve totalmente escalable. Podemos usar el mismo API para diferentes aplicaciones (Web, Mobile, Android, iOS) solo debemos preocuparnos de enviar los datos en formato JSON y generar y descifrar tokens en la autenticación y posteriores peticiones HTTP a través de un middleware

También nos añade más seguridad. Al no utilizar cookies para almacenar la información del usuario, podemos evitar ataques CSRF (Cross-Site Request Forgery) que manipulen la sesión que se

envía al backend. Por supuesto podemos hacer que el token expire después de un tiempo lo que le añade una capa extra de seguridad.

Descripción de Clases java usadas para la implementación de tokens de seguridad

SessionFilter.java

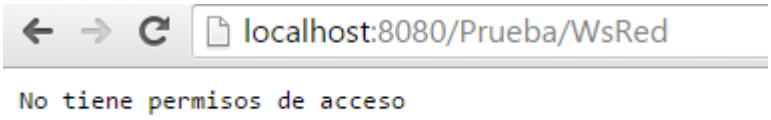
```
public void doFilter(ServletRequest req, ServletResponse res,
            FilterChain chain) throws IOException, ServletException {

        HttpServletRequest request = (HttpServletRequest) req;
        HttpServletResponse response = (HttpServletResponse) res;
        boolean allowedRequest = false;
        String url = request.getRequestURI();
        if (url.contains("error.jsp")) {
            allowedRequest = true;
        }
        if (!allowedRequest) {
             String h=request.getHeader(ResponseManager.cookieName);
            String value=ResponseManager.validaToken(request.getHeader(ResponseManager.cookieName));
            if(!"OK".equals(value)){
              PrintWriter out=response.getWriter();
              out.println(value);
              out.close();
            }
            else{
              chain.doFilter(req, res);
            }
        }
        else{
             chain.doFilter(req, res);
        }
    }
```

1. Esta clase se encarga de filtrar la consulta a los web services
2. Para que sea agregada esta clase como filtro se debe agregar estas líneas de código en el wed.xml

```
<filter>
<filter-name>SessionFilter</filter-name>
<filter-class>org.ws.util.SessionFilter</filter-class>
</filter>
<filter-mapping>
<filter-name>SessionFilter</filter-name>
<url-pattern>/*</url-pattern>
</filter-mapping>
```

3. Para que un servicio web (puede ser el caso de login) no se deba validar el token reemplazar el siguiente código **url.contains("error.jsp") "error.jsp"** por el nombre del método de consulta del web services que no se necesite ser filtrado (puede ser web services de login)
4. **Stringh=request.getHeader(ResponseManager.cookieName);**
   recupera el token que envía el cliente, luego valida el tiempo de sesión
5. ResponseManager.validaToken(request.getHeader(ResponseManag er.cookieName))**;**
6. Si genera algún error o no envía el token presenta un mensaje



7. Si el tiempo excede lo configurado en la clase **ResponseManager**. **tiempoSession** saldrá un mensaje de sesión caducada.
8. El token debe generase en el ingreso de sesión.
9. El token consta de 15 caracteres más la hora y día que se generó el token expresado en valores tipo**long.**
10. Este token es encriptado mediante la clase UtilCryptography.encriptar(palabra)

11. La clase que genera el token es **ResponseManager.generarToken()**
12. Por cada consulta a un método de servicio web se genera un nuevo token.
13. El token viaja en la cabecera de la petición.
14. La clase que agrega el token a la cabecera es **ResponseManager.**
    ResponseHeader (String) este método debe utilizarse para cada respuesta del web services.

Responseheader.java

```java
package org.jboss.samples.rs.webservices;

import javax.ws.rs.core.Response;

public class ResponseHeader
{
    public static Response addHeader(String obj,String token)
    {
        return Response.status(200).entity(obj)
       .header("Access-Control-Allow-Origin", "*")
       .header("Access-Control-Allow-Methods", "GET, POST, DELETE, PUT, OPTIONS, HEAD")
       .header("Access-Control-Allow-Headers", "Content-Type, Accept, X-Requested-With")
       .header(ResponseManager.cookieName,token)
       .build();
    }

    public static Response addHeader(String obj, String origin, String methods, String headers)
    {
       return Response.status(200).entity(obj).header("Access-Control-Allow-Origin", origin)
                .header("Access-Control-Allow-Methods", methods)
                .header("Access-Control-Allow-Headers", headers)
                .build();
    }

}
```

1. **header("Access-Control-Allow-Origin", "*")**

   Permite que el servicio web sea accedido desde otra aplicación que no esté en el mismo servidor donde se encuentra desplegado.

2. ***header("Access-Control-Allow-Methods", "GET, POST, DELETE, PUT, OPTIONS, HEAD")***

   Permite la repuesta se haga solo bajo estos métodos de HTTP.

3. ***header("Access-Control-Allow-Headers", "Content-Type, Accept, X- Requested-With")***

   Permite mantener en tipo de contenido de la solicitud.

4. ***header(ResponseManager.cookieName, token)***

   se agrega el token a la cabecera para enviarlo al cliente.

Módulo WEB ZK

2. En ZK recupera el token de la cabecera y lo manda a sesión en la clase **ApacheHttpClientGet** en ambos métodos **GET y POST**

```
Header token=response.getFirstHeader("autenticar"); if(token!=null){
session.setAttribute("TokenWS", token.getValue());
}
```

3. Cada que realiza una petición al servicio web envía el token que guarda en sesión

```
Header ts= new Header() {
@Override
public String getValue() {
return (String) session.getAttribute("TokenWS");
}
@Override
public String getName() { return
"autenticar";
}
@Override
public HeaderElement[] getElements() throws ParseException { return null;
}
};
getRequest.addHeader(ts);
```

Políticas de seguridad para la creación de Web Services seguros

Las políticas de seguridad que se deben seguir al momento de crear un Web Services en cualquiera de las aplicaciones que utiliza el Hospital Universitario de Guayaquil, representa la posición que tiene el hospital con respecto a la protección de la información que se envía o recibe vía Web Services. Además, se da a conocer el apoyo a la seguridad de la información, el compromiso que se tiene por velar la seguridad de la información y el control de los riesgos.

A continuación, se detallan los objetivos de la política de seguridad:

- Minimizar los riesgos existentes en la comunicación de datos vía Web Services.
- Proteger la información.
- Apoyar la innovación tecnológica.
- Establecer las políticas e instructivos de seguridad.
- Mantener la confianza de los usuarios con la institución.
- Crear una cultura de seguridad.

Alcance/Aplicabilidad

Esta política aplica a todos los desarrollos que se realizan en la institución, sus desarrolladores y practicantes.

Nivel de cumplimiento

Se espera que la política se adhiera al 100% de las personas involucradas.

Las políticas de seguridad para la creación de Web Services del Hospital Universitario de Guayaquil son las siguientes:

Web services Restfull

- Se creará un token de seguridad cuando se realice una autenticación.

- Los identificadores de sesión sólo se encuentra en el encabezado de la petición.
- No incluya información confidencial en los parámetros de la petición HTTP GET.
- La aplicación sólo debe reconocer identificadores válidos de sesión.
- Requerir la autenticación de todas las páginas, excepto los destinados específicamente a ser pública

Web services Soap

- Utilizar estándar de seguridad Ws-Security
- Implementar Xml-Encryption para encriptar la información sensible que viaje en las consultas.
- Usar Xml-Signature para enviar una firma digital en el mensaje, este puede ser emitido por alguna entidad certificadora.

Programación General

- Debe existir el botón “Logout” en todas las páginas.
- Determinar un tiempo de espera de inactividad de la sesión. En la mayoría de los casos, debe ser no más de 15 minutos.
- No permitir conexiones simultáneas con el mismo id usuario
- Enviar contraseñas debidamente encriptada o como datos cifrados, como por ejemplo en un mensaje cifrado.
- Establecer políticas de creación de contraseñas.
- No realizar consultas directas a la base de datos, sino utilizar consultas parametrizadas.
- La cadena de conexión no debe ser codificado dentro de la aplicación, deben ser almacenadas en un archivo de configuración independiente.

- Cerrar la conexión a la base de datos tan pronto como sea posible.
- Cifrar la información almacenada altamente sensible, como datos de verificación de autenticación, incluso en el lado del servidor
- Evitar que el código Fuente pueda ser descargado por personal no autorizado.
- Eliminar los comentarios en el código de producción ya que puede revelar información confidencial.

Resultados de las Encuestas

Pregunta 1

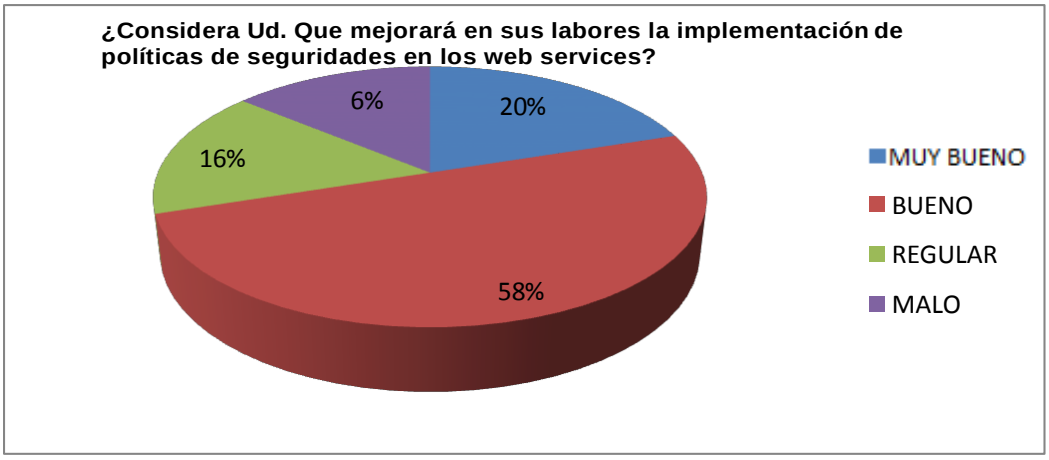


Gráfico 1. Mejoramiento de labores con políticas de seguridad

Análisis: El 58% respondió que le parece muy buena, al 20% respondió buena al 16% le parece regular y al 6% una mala solución por lo que se concluye que la implementación de políticas de seguridad si aportara en la mejora de las actividades ya que esto servirá como guía para futuros desarrollos.

Pregunta 2

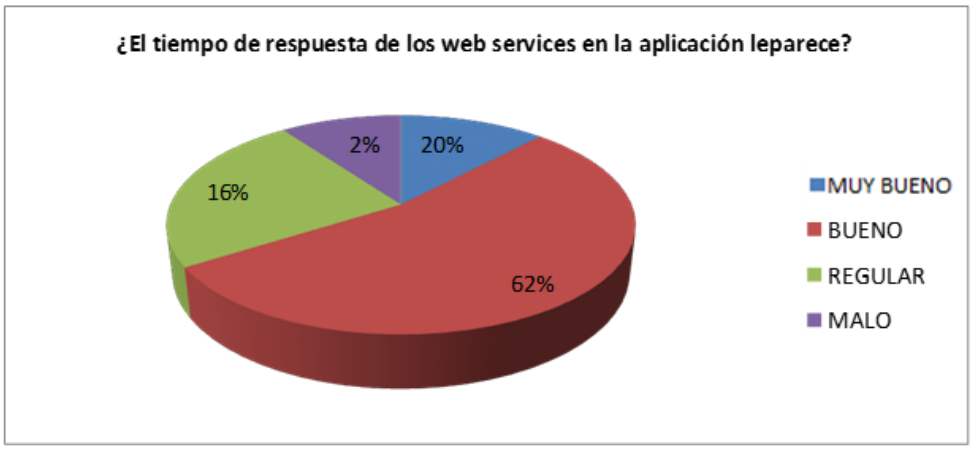


Gráfico 2. Tiempo de respuesta de los Web Services

Análisis: De 50 personas entrevistadas el 62% de los encuestados respondió que le parece muy buena, al 20% le pareció buena, el 16% respondió que el tiempo es regular y que se podría mejorar y el 2% respondió que no están de acuerdo con los tiempos de respuesta y creen que se puede mejorar.

Pregunta 3

Gráfico 3. Propuesta se ajusta a sus necesidades

Análisis: Se puede observar que del total de personas entrevistadas al 32% respondió que muy buena, el 56% respondió bueno, el 12% le parece regular solución planteada, pero a ninguna persona de las encuestadas le pareció una mala idea. Lo que concluye que la propuesta planteada cubre las necesidades de seguridad existente.

Pregunta 4

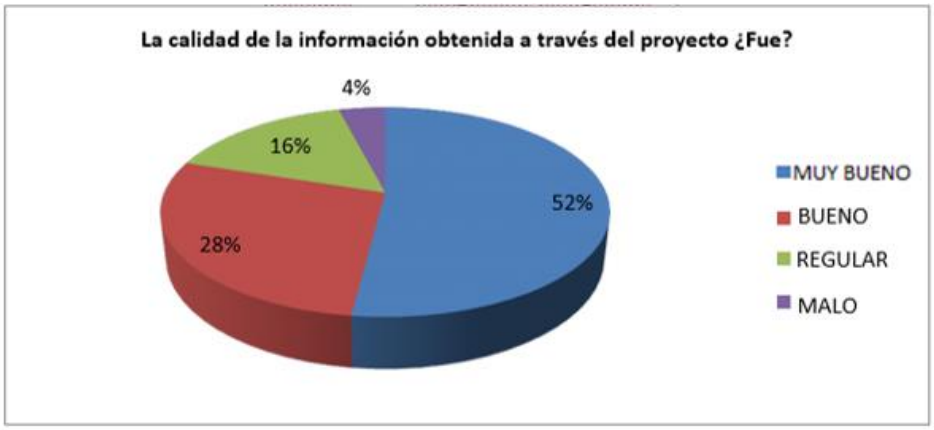


Gráfico 4. Calidad de la información

Análisis: Se puede observar que de 50 personas entrevistadas 26 de ellas que corresponde al 52% respondió que la calidad de la información que devuelve el Web Services es muy buena, otras 14 personas que representan el 28% respondieron que le parece buena, el 16% le parece regular, al 4% le parece mala, por lo que se concluye que el 80% de los encuestados están de acuerdo con la calidad de la información.

Pregunta 5

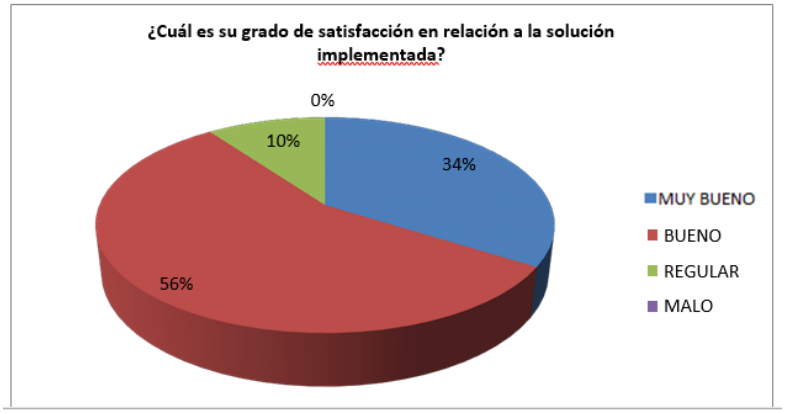


Gráfico 5. Satisfacción a la solución brindada

Análisis: Se puede observar que de 50 personas entrevistadas 23 de ellas que corresponde al 56% respondió que la satisfacción con relación a la solución implementada es excelente otro 34% indico que su satisfacción es buena, solamente el 10% no está conforme con la solución implementada.

Conclusiones.

Este trabajo fue revisado y aprobado por los médicos cardiólogos de la Universidad de Guayaquil y autoridades de PROMEINFO.

El informe se muestra favorable con respecto a las vulnerabilidades más comunes al momento de implementar aplicaciones web.

Para nadie es un secreto la implementación de seguridades, sin embargo, esto no garantiza la seguridad de la información, lo más recomendable es usar una metodología comprobada para garantizar la seguridad.

Las organizaciones requieren una política de seguridad para cumplir con sus requerimientos de seguridad de la información.

Redactar una política de seguridad puede ser fácil, lo difícil es lograr que su cumplimiento.

Se recomienda implementar la aplicación en un protocolo seguro.

Hay que considerar, para futuras evaluaciones, los siguientes riesgos, no necesariamente de seguridad pero que sí corresponden a las buenas prácticas:

- Anti-automatización insuficiente
- Denegación de Servicio
- Ejecución de archivos maliciosos
- Fallas de concurrencia
- Filtración de información y manejo inapropiado de errores
- Inyección de expresiones de lenguaje
- Privacidad de Usuario
- Registro y responsabilidad insuficientes
- Vulnerabilidad de asignación masiva.

Agradecimiento

A la Universidad de Guayaquil por su contribución en patrocinar el proyecto de investigación FCI de PROMEINFO.

Bibliografía.

27001 Academy, 27001. (2014). ISO 27001 and ISO 22301 – Documentation and Professional Guidance | 27001Academy. Retrieved June 8, 2017, from https://advisera.com/27001academy/

Aray Dominguez, J. (2015). Administración De Los Sistemas De Gestión De Proyectos Promeinfo. Universidad de Guayaquil. Facultad de Ciencias Matemáticas y Físicas. Carrera de Ingeniería en Sistemas Computacionales. Retrieved from http://repositorio.ug.edu.ec/handle/redug/10433

Areitio Bertolín, J. (2008). Seguridad de la información : redes, informática y sistemas de información. Paraninfo Cengage Learning.

Benítez Moisés. (2013). Gestion Integral. Obtenido de <http://www.gestionintegral.com.co/wp-content/uploads/2013/05/Pol%C3%ADticas-de-Seguridad- Inform%C3%A1tica-2013-GI.pdf>

Ikujiro Nonaka e Hirotaka Takeuchi. (2013). Ecured. Obtenido de ecured: <http://www.ecured.cu/index.php/Metodolog%C3%ADa_Scrum>

Kaspersky. (2014). Kaspersky Lab | Software Antivirus y Seguridad en Internet. Retrieved June 8, 2017, from https://www.kaspersky.es/

Ramos, María, A. G.C. (2011). SEGURIDAD INFORMATICA ED.11 Paraninfo. Editorial Paraninfo.

Salgado Yánez, Á. L. (2014). Artículo Científico. - Análisis de las aplicaciones web de la Superintendencia de Bancos y Seguros, utilizando las recomendaciones Top Ten de OWASP para determinar los riesgos más críticos de seguridad e implementar buenas prácticas de seguridad para el de. Retrieved from http://repositorio.espe.edu.ec/handle/21000/8246

UNAM. (2014). Redes y seguridad. Material y tutoriales. Retrieved June 8, 2017, from http://www.unamenlinea.unam.mx/recurso/82079-tutorial-de-seguridad-informatica