CASO DE ESTUDIO: CONSTRUCCIÓN DE LA HERRAMIENTA MVPS PARA LA VERIFICACIÓN DE POLÍTICAS DE SEGURIDAD EN COMERCIO ELECTRÓNICO B2C UTILIZANDO COMPONENTES XPCOM

STUDY CASE: BUILDING OF MVPS TOOL TO VERIFY THE SECURITY POLICIES IN ELECTRONIC TRADE B2C USING XPCOM COMPONENTS

AUTOR
ROBERTO CARLOS NARANJO CUERVO
Ms (c) en Ingeniería de Sistemas y Computación
Profesor Asociado Tiempo Completo
*Universidad del Cauca
Departamento de Sistemas
rnaranjo@unicauca.edu.co
COLOMBIA

AUTOR
FREDDY MINA GRUESO
Ing(c) de Sistemas
*Universidad del Cauca
Departamento de Sistemas
frmina@unicauca.edu.co
COLOMBIA

AUTOR
ERWIN DAZA RENDÓN
Ing(c) de Sistemas
*Universidad del Cauca
Departamento de Sistemas
edaza@unicauca.edu.co
COLOMBIA

INSTITUCIÓN
*UNIVERSIDAD DEL CAUCA
UNICAUCA
Institución de Educación Superior
Calle 5 No 4-70 Sector Tulcán of 444
vri@unicauca.edu.co
COLOMBIA

RECEPCIÓN: Julio 16 de 2009

ACEPTACIÓN: Mayo 14 de 2010

TEMÁTICA: Ingeniería del software y Seguridad Computacional

TIPO DE ARTÍCULO: Artículo de investigación científica y tecnológica

RESUMEN ANALÍTICO

La construcción de una herramienta que permita verificar las políticas de seguridad básicas durante la realización de una transacción Business to Consumer (B2C) hace necesario que esta sea incorporada a un navegador, entre estos, los de mayor uso son Firefox e Internet Explorer. Firefox ofrecen el mecanismo de extensiones o sobreposiciones para extender su funcionalidad y el propio navegador y que Internet Explorer ya no tiene disponible en sus últimas versiones. En la actualidad es muy común que las extensiones o sobre posiciones desarrolladas para Firefox no incluyan componentes desarrollados en C++ y mas aun que estos sean utilizados en sistemas Windows. La necesidad de incluir este tipo de componentes es debido a que aunque el Framework de Mozilla cuenta con más de 1300 componentes, algo mas de 100 componentes se encuentran en estado frozen y no son suficientes para crear las herramientas con las políticas mínimas requeridas para garantizar un nivel de seguridad aceptable. Esto se establece como un objetivo a cumplir. Para completar este objetivo se hace necesario crear un componente para la comunicación con el firewall y para verificar si se tiene o no instalado un software antivirus. La construcción del componente incluye el diseño del mismo, el cual debe servir como modelo en la creación de nuevos componentes dependiendo de si existe la necesidad de implementar nuevas políticas, además se debe tener en cuenta el diseño como tal de dichas políticas, el cual debe garantizar que se puede hacer un seguimiento de la misma a lo largo de su ciclo de vida. Como resultado se obtuvo una herramienta software que verifica el grado de seguridad de las transacciones B2C y cuyo objetivo es aumentar el nivel de confianza de los usuarios a la hora de enfrentarse a este tipo de transacciones.

PALABRAS CLAVES: Componentes XPCOM, Transacciones B2C, Diseño de Políticas de Seguridad, Diseño Módulo Software

ANALYTICAL SUMMARY

Building of a tool that allows verifying the basic security policies during a Business to Consumer(B2C) transaction requires that this will be incorporated into a browser, the most used are Firefox and Internet Explorer. Firefox provides the extension mechanism or overlaps to extend their performance and the browser since Internet Explorer does not have this option available even in its last versions. Nowadays, it is very common that the extensions and overlaps do not include components done in C++ developed to Firefox and even so, they are used by Windows systems. The need to include these components is despite Mozilla framework consists of more than 1.300 components, approximately 100 components are frozen and they are not enough to create the required minimum policy tools to ensure an acceptable level of security and this is established as an essential goal. To get this, it is necessary to create a component for communication with the firewall and to check whether or not you have antivirus software installed. The component includes the construction of the design itself which should serve as a model for the creation of new components depending on if there is necessary to implement new policies. Moreover, it also must take into account the design of such policies which should ensure that you can keep track of it along its life cycle. A software tool was obtained as a result. It verifies the level of security in B2C transactions and its objective is to increase the level of user confidence when they face these kinds of transactions.

KEYWORDS

XPCOM Components, B2C Transactions, Security Policies design, Software module design.

INTRODUCCIÓN

Aunque existen pronósticos alentadores en cuanto al crecimiento del comercio electrónico para América Latina, hay aun algunos aspectos que impide que este pueda crecer significativamente, la privacidad de la información, integridad de los datos enviados vía internet y la autenticidad de los sitios de comercio electrónico son los factores que más preocupan a la hora de realizar transacciones electrónicas, y aunque existen las herramientas tecnológicas para solucionar estos problemas, hay un gran porcentaje de usuarios que no hace uso de ellas por desconocimiento de las mismas o de los procedimientos que pueden garantizarles un nivel aceptable de seguridad a la hora de realizar una transacción. Algunos antecedentes, tales como *las políticas de seguridad del Banco de Bogotá* las

cuales son: la certificación de la página Web, máxima encriptación de datos, protocolos de comunicación seguros, identificación de usuarios y protección de los datos y que son cubiertos por las aplicaciones disponibles para los usuarios a través de su portal en internet [20]. *Norton confidential de Symantec*, es una herramienta software propietaria que provee desenmascarar el phishing (suplantación de identidad) y el pharming (reorientación hacia otras páginas), descubrir el crimeware (software malicioso), autenticación de sitios web, proteger contraseñas, estrategias de seguridad a mayor escala (integración de todos los componentes de la gama Norton de seguridad) [21]. Todas estas políticas y herramientas se deben acompañar de un conjunto de buenas prácticas por parte de los usuarios, el problema es que estos buenos hábitos no se encuentran implementados en ninguna herramienta que le ayude al usuario a detectarlos y terminan por convertirse en una amenaza para las transacciones que realiza el usuario el no tenerlos en cuenta, tales como realizar sus conexiones y transacciones en sitios encriptados, usar sitios autenticados (para distinguir entre páginas falsas), monitorear sus cuentas, proteger su identidad, adoptar nombres de usuarios seguros y contraseñas difíciles, entre otros. Otros antecedentes como *Google Checkout* [22], es un servicio de pagos por internet, este servicio no le garantiza protección al cliente contra ataques tipo phishing, ni le garantiza la autenticidad del sitio en el cual está comprando, además provee al sitio los datos de la tarjeta de crédito del cliente para la compra eximiéndolo de este paso, lo cual no garantiza aumento de la seguridad en la transacción sino que aumenta los riesgos para el cliente. Debido a esto se propone la creación de una herramienta que permita a partir de un grupo de políticas garantizar un nivel de seguridad aceptable a la hora de realizar transacciones B2C que busca los siguientes objetivos:

- Diseñar un grupo de políticas que permitan un nivel aceptable de seguridad a la hora de realizar una transacción B2C, este diseño debe permitir gestionar el grupo de políticas durante todo su ciclo de vida.
- Diseñar y construir el modulo software que va a automatizar las políticas propuestas.
- Integrar el software al navegador Firefox.

Para lograr estos objetivos podría haberse utilizado estándares como el de la Plataforma de Preferencias de Privacidad (P3P)[1], que han contribuido en la creación de herramientas como PrivacyBird, útil a la hora de gestionar políticas de privacidad, desafortunadamente, este tema ha sido retirado "Temporalmente" de Firefox e Internet Explorer. Por tal motivo se desarrollo completamente en el lado del cliente usando componentes XPCOM [8].

CONTENIDO DEL DOCUMENTO

A continuación presentaremos la metodología empleada en el proceso de investigación, el análisis de riesgos encontrados, el diseño de las políticas de seguridad propuestas, el diseño e implementación de la herramienta software, y su respectiva instalación y pruebas, al final tenemos conclusiones y bibliografía.

1. METODOLOGÍA

Para la etapa de investigación se siguió inicialmente la metodología propuesta en el Modelo para la Investigación Científica [2]: durante la etapa de investigación sugerida en este se exploraron diferentes sitios en busca de políticas implementadas, en la etapa de síntesis de la información, se escogió la metodóloga Magerit [3] con el fin de realizar el análisis de riesgos, el porqué de realizar un análisis de riesgos viene de la mano de algunas guías como la ISO 17799 [4] y RFC2196 [5], las cuales tienen orientaciones para la creación de políticas de seguridad. En cuanto a la creación del diseño de las políticas se optó por seguir la metodología sugerida por la Universidad Nacional de Colombia, en su guía para la elaboración de políticas [6]. Para la creación del componente software, la metodología de desarrollo del software escogida fue el Proceso Unificado de desarrollo de software (UP)[7] de la mano con la guía expuesta en [8] útil a la hora de crear componentes XPCOM.

2. RESULTADOS OBTENIDOS EN LA ETAPA DE INVESTIGACIÓN

El objetivo de esta primera etapa fue el descubrir políticas y/o recomendaciones existentes que sirvieran como base para la creación de las políticas que son la base sobre la cual se implementa el módulo software para resolver el problema de la seguridad en transacciones B2C, esta etapa permitió descubrir, primero, que existen políticas de seguridad, pero estas son las creadas por los sitios para garantizar su propia seguridad, segundo, las políticas de privacidad son las únicas políticas que exponen algunos sitios para informarle a sus clientes de cual es el uso que se va hacer de los datos brindados a ellos, tercero y último, algunos sitios, como los de algunas casas antivirus y otras comprometidas con la seguridad, exponen sólo algunos hábitos que deben implementar los clientes para garantizar su seguridad.

Estos resultados evidenciaron que no existía ningún tipo de política que pueda ser utilizada e implementada para lograr un nivel de seguridad aceptable a la hora de realizar transacciones B2C.

3. RESULTADOS OBTENIDOS EN EL ANÁLISIS DE RIESGOS.

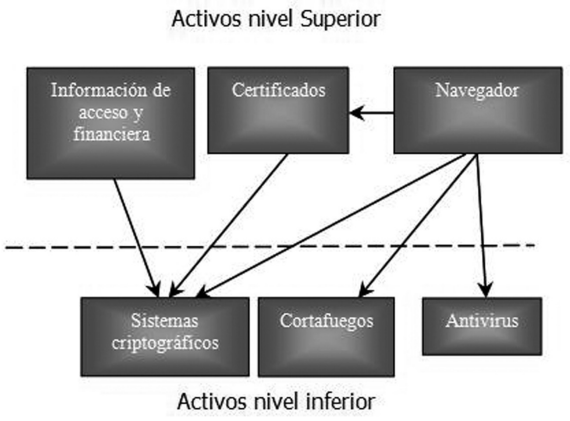
Para llegar hasta el diseño de las políticas se hizo necesario, inicialmente, hacer una búsqueda de los componentes y amenazas presentes en transacciones B2C, se pudo concluir que los principales componentes involucrados en una transacción eran: el navegador web, el cortafuegos, software antivirus, protocolos (entre los que se pueden mencionar (https y TLS)) y certificados digitales para garantizar la autenticidad de los sitios. Entre las amenazas más importantes tenemos el Pharming, troyanos y keyloggers, conexiones sin cifrar, contenido activo [9] (java Applets, java script, pluggins, controles Activex, etc.), desactualizaciones y suplantación de identidad. Con esta información inicial y gracias al análisis de riesgo para el cual se tomaron datos provenientes del Instituto Nacional de Tecnologías de la Comunicación (INTECO)[10], se logro obtener los siguientes datos necesarios para definir posteriormente cuales eran las amenazas más significativas y las salvaguardas para disminuir su impacto y degradación del activo.

Tabla 1: Valoración global de los activos [10]

Activos	*Uso*
Información de acceso y financiera (Número cuenta, clave tarjeta, Contraseñas, etc.)	***54.5%***
Certificados digitales	***2.3%***
Navegador	***100%***
Antivirus	***97.2%***
Sistemas criptográficos	***14.9%***
firewall	***80.3%***

Partiendo de que la metodología nos permite saber cuan valiosos son los activos involucrados en una empresa o actividad comercial y la cantidad de valor que se encuentra en riesgo cuando una amenaza o vulnerabilidad tiene incidencia sobre estos, lo primero que se hizo fue encontrar el valor de los activos más significativos(ver tabla 1). Uno de los resultados mas importantes es encontrar el Modelo de valor en el cual se incluyen las dependencias entre activos (figura 1), estas dependencias son útiles ya que los activos mostrados en el nivel inferior son los pilares de los niveles superiores y cualquier amenaza que tenga influencia en estos se ve repercutido hacia los niveles superiores.

Figura 1: Dependencias entre activos



Cabe destacar entre estos y otros resultados los datos mostrados en la siguiente tabla (tabla 2) en la cual se establecieron relaciones de prioridad por grupos de activos y orden de impacto.

Tabla 2: Interpretación de resultados del análisis de riesgos.

Activo: Certificados Digitales		
Amenazas	**Priorización**	**Valor acumulado**
Suplantación de identidad.	1	7.6
Activo: Navegador Web		
Amenazas	**Priorización**	**Valor acumulado**
Conexiones a sitios sin utilizar cifrado o cifrado deficiente.	1	58.72
Keyloggers + troyanos	2	42.16
Phishing, pharming y similares	3	23.92
Cookies	4	8
Activo: Información de acceso y financiera (Claves y contraseñas)		
Amenazas	**Priorización**	**Valor acumulado**
Keyloggers + troyanos	1	10.54

Obtención fraudulenta	2	2.92
Virus en general	3	1.72
Activo: Antivirus		
Amenazas	**Priorización**	**Valor acumulado**
Ataques del día cero	1	48.3
Nuevos virus	2	25.13
Activo: Firewall		
Amenazas	**Priorización**	
Virus en general	1	6

En la tabla 2 se priorizaron cada una de las amenazas, la priorización de cada una de ellas se realiza en base al valor del riesgo asociado; para el caso de los activos, que tengan dependencias, en los que estas amenazas actúan se tiene en cuenta el valor acumulado asociado, a este valor se llega, primero, midiendo el riesgo, se calcula mediante la multiplicación del valor del impacto económico por el valor de la probabilidad de ocurrencia que resultan al materializarse un riesgo en respuesta a una vulnerabilidad dada, este cálculo se hace para todos los activos con sus correspondientes amenazas y vulnerabilidades, el valor acumulado asociado es el máximo valor de medición del riesgo tomando en cuenta aquellos activos en niveles inferiores de dependencia. Gracias al valor acumulado obtenido se pueden priorizar los riesgos, entre más alto sea el valor acumulado más importante y requiere mayor atención. Como mecanismo escogido para la reducción del riesgo se seleccionó el uso de políticas de seguridad, las cuales se implementaron para aquellas relaciones activos-amenaza cuyo valor acumulado sea mayor, las seleccionadas fueron:

- ***Certificados digitales-Suplantación de identidad:*** cuyo riesgo se puede disminuir con una política que obligue a la verificación de la autenticidad del sitio por medio de certificados digitales, comprobando que se tenga uno y haciendo la validación del mismo.
- ***Navegador web-Conexiones a sitios sin cifrar:*** En este caso la disminución del riesgo se puede dar al verificar que las conexiones a los sitios se hagan a través de conexiones cifradas.
- ***Navegador - Keyloggers + troyanos/ phishing, pharming y similares:*** actualmente los navegadores poseen soporte antikeylogger y antiphishing por lo cual el riesgo se ve disminuido con una política que garantice la utilización de los últimos navegadores y actualización de los mismos.

Para las relaciones entre la información de acceso y financiera y todas sus amenazas, además de las relaciones antivirus-ataques del día cero y antivirus-nuevos virus se puede disminuir el riesgo utilizando una política que garantice el uso de software antivirus. En cuanto a la relación firewall-virus en general se puede reforzar no solo utilizando el antivirus si no que se puede establecer una política que garantice el uso del antivirus al igual que el uso y activación del firewall.

Por último para las relaciones activos-amenaza cuyo valor sea más bajo o su implementación requiere un esfuerzo mayor, se decide aceptar el riesgo y brindar una recomendación, el uso de recomendaciones surge de aquellas prácticas que son difíciles de controlar o de hacer obligatorias debido a que tienen un costo relativamente alto o su implementación resulta casi que imposible.

4. DISEÑO DE LAS POLÍTICAS.

Para una política, como por ejemplo la de verificar la identidad del sitio con la cual se quiera hacer una transacción, se deben definir los siguientes puntos los cuales constituyen el diseño de dicha política: ***la definición formal de la política*** (esta definición debe contener de forma clara la definición de la política), ***las amenazas que pretende atacar dicha política, su descripción*** (esta descripción debe mostrar lo que se espera con dicha política), ***Responsable de la política*** (persona encargada de revisar la política durante su ciclo de vida y de buscar nuevas relaciones amenazas-vulnerabilidades y realizar un análisis de riesgo cuando se requiera), ***Información con la identificación de la política:*** (fecha de creación, fecha de revisión, estado de la política (en uso u obsoleta), e identificador), ***Excepciones*** (En este punto se deben gestionar las situaciones donde la implementación no es posible, la gestión incluye el nombre de la persona responsable de la documentación y vigilancia de la excepción, y el tiempo de duración de la misma), ***Anotaciones*** (son anotaciones especiales relacionadas con la política), ***Estándares u orientaciones*** (son orientaciones que son obligatorias, y ganan este carácter para lograr el cumplimiento de la política), ***Mejor practica*** (estas deben describir la forma como se configuran los sistemas utilizados para garantizar un nivel de seguridad aceptable), ***Guías*** (en estas se especifican los paso necesarios para hacer cumplir las políticas, esto incluye también lo que se debe hacer para cumplir los estándares u orientaciones. Estos ítems, según nuestro criterio, permiten mantener y hacerle seguimiento a las políticas durante todo su ciclo de vida. Las políticas seleccionadas en el análisis de riesgo para ser implementadas se describen a continuación:

- Se debe verificar la identidad del sitio de comercio electrónico en el cual se realizan compras.
- Verificar que durante la transacción se haya establecido una conexión segura entre el navegador web y el sitio de comercio electrónico.
- Verificar el soporte del equipo (software antivirus y firewall).
- Verificar la actualización del navegador.

En cuanto a las amenazas cuya implementación se hace costosa, ya sea por factores técnicos, económicos, etc., o tienen un nivel de riesgo bajo se describen a continuación las recomendaciones desarrolladas para cubrir dicho riesgo (tabla 3):

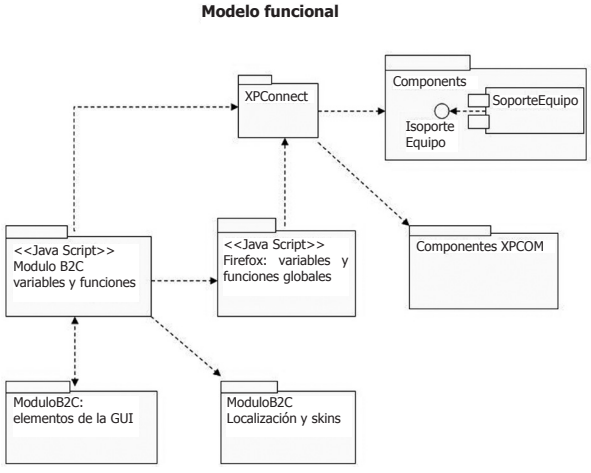
Tabla 3: Relación amenaza-recomendaciones

Amenaza	Recomendación
Nuevos virus	Se debe mantener el Antivirus actualizado. Mantener el sistema operativo actualizado con todos los parches de seguridad.
Ataques del día cero	Están enterados de los últimos boletines acerca de nuevos virus. Actualizar el antivirus constantemente.
Virus en general	Instalación de un software antivirus. Actualizar el antivirus constantemente
Cookies	Verificar las políticas de privacidad de los sitios antes de permitirles utilizar cookies. Utilizar tecnologías como P3P que le permiten verificar sus preferencias de privacidad con las políticas de privacidad de los sitios. Configurar el navegador para no aceptar cookies. Si por alguna razón permite el uso de cookies configure su navegador para que las elimine al cerrarse este.
Phishing pharming y similares	Utilizar navegadores con soporte antiphishing. Utilizar antivirus con soporte antiphishing
Alojamiento de Scripts	Permitir solo Script de páginas reconocidas. Permitir solo Scripts que contengan certificados
Obtención fraudulenta de claves y contraseñas	No envía sus datos por medio de conexiones sin cifrar. Instale un software antivirus y manténgalo actualizado
Ataques para descifrar la información	No envía sus datos por medio de conexiones sin cifrar.

5. DISEÑO DE LA HERRAMIENTA SOFTWARE.

Para la consecución del diseño de la herramienta se tuvieron en cuenta algunas consideraciones iniciales referentes al diseño de extensiones en Firefox, estas consideraciones deben incluir la relación que debe existir entre la extensión y Firefox la cual es mostrada en el diseño funcional de la figura 2, y la estructura del Chrome de la extensión mostrada en la figura 3. El diseño funcional, figura 2, muestra componentes como: xpconnet [11] que es la capa que hace posible que se puedan acceder elementos de Framework desde JavaScript, las variables y funcionales globales de Firefox las cuales se pueden acceder desde nuestra aplicación, las variables y funciones definidas en nuestra extensión, los elementos de la interfaz grafica (archivos xul [12]) los componentes del framework y nuestro propio componente.

Figura 2: Diseño funcional de la relación entre la extensión y Firefox



En cuanto a los requerimientos funcionales que se deben cumplir con el diseño:

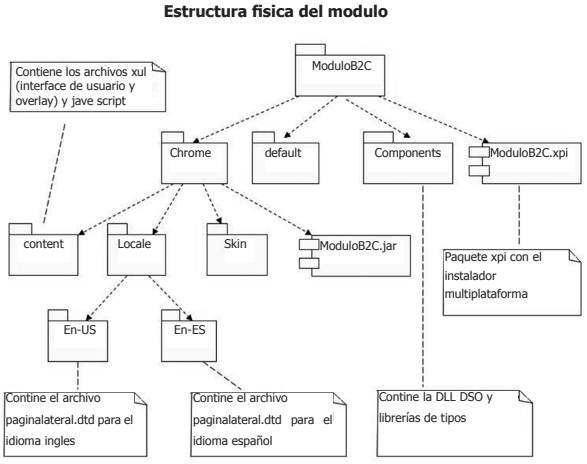
- Verificar las políticas de seguridad mínimas para la realización de transacciones B2C (políticas ya seleccionadas).
- Mostrar el nivel de cumplimiento de cada política.
- Mostrar detalles acerca del cumplimiento de las políticas.
- Mostrar el nivel de seguridad para la transacción.

- Mostrar recomendaciones de acuerdo al nivel de cumplimiento de las políticas.
- Mostrar recomendaciones generales.
- Desplegar alertas de acuerdo a la seguridad en el momento de la transacción.

A continuación se muestra la estructura general que toda extensión implementada para Firefox o cualquier otro producto de Mozilla debería tener. Esta estructura (figura 3) permite ubicar los archivos de la extensión de forma organizada y en las ubicaciones donde el explorador busca por defecto los archivos que necesita. La carpeta principal modulob2c cuenta con las siguientes carpetas:

- ***Chrome:*** esta carpeta contiene en su interior tres carpetas: content, locale y skin, dentro de content se ubican los archivos xul de la extensión, overlay. xul y paginaLateral.xul, el primero permite adicionar la extensión a cualquier punto del navegador y el segundo es la interface grafica para el usuario, dentro de esta misma carpeta también se encuentra los archivos java script de la aplicación. Locale contiene definiciones de idioma para español e ingles por medio de los respectivos DTD [13] y Skin con las imágenes y temas de la extensión.
- ***Default:*** en esta carpeta se ubica preferencias de usuario y variables globales.
- ***Components:*** dentro de esta carpeta se deben colocar las DLL [14] y/o DSO, dependiendo del sistema operativo que se use, y las librerías de tipos para poder acceder los componentes XPCOM que sean de desarrollo propio.

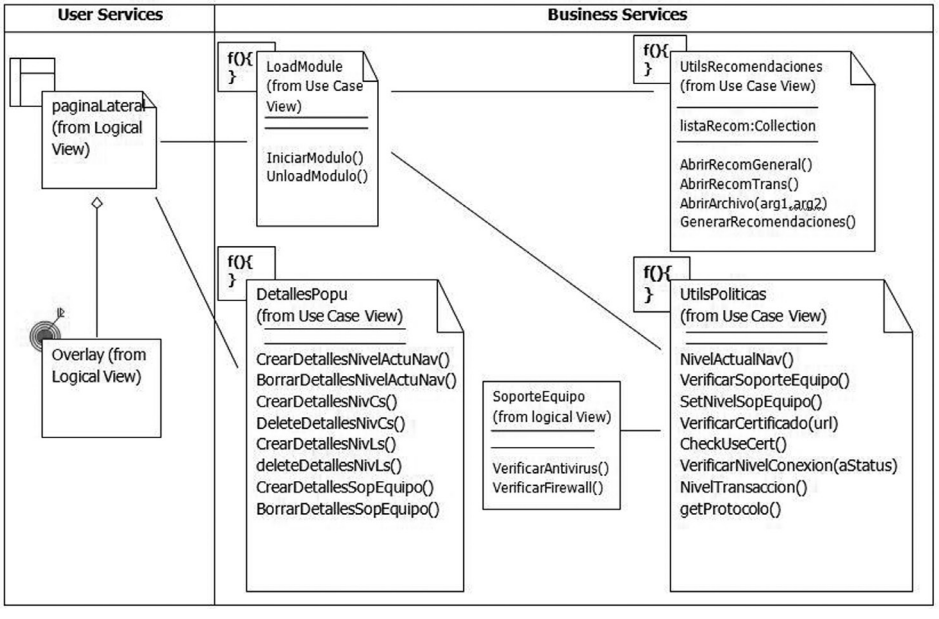
Figura 3: Estructura de la extensión



En cuanto a la arquitectura utiliza no hay ningún tipo de restricción, se puede utilizar la que más convenga, en este caso se utilizo una arquitectura en dos capas (figura 4).

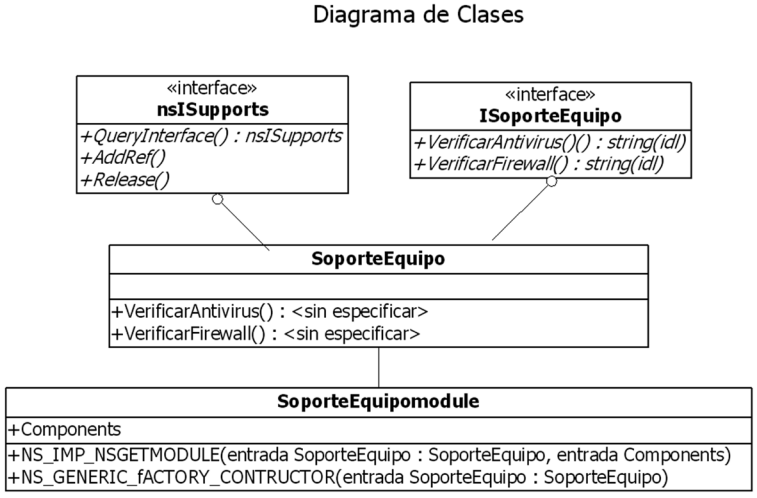
Para la arquitectura de la herramienta se exponen dos capas, la de presentación (User Service) con los archivos Xul necesarios y la capa lógica (Business Services) con los archivos Java Script útiles para darle la funcionalidad requerida a la herramienta y para acceder y crear los componentes del Framework.

Figura 4: Arquitectura de la herramienta



En cuanto al diseño del componente XPCOM, que también hace parte de la herramienta, la figura 5 muestra las interfaces y clases necesarias para implementar el componente. Este componente se debe compilar por medio de una archivo .mak [15] para crear la DLL(DLL en sistemas Windows o DSO en Unix) y para poder ser utilizado desde la extensión esta DLL se debe colocar dentro de la carpeta components junto a la librería de tipos, archivo .xpt, que permite tener acceso al componente desde otros lenguaje como (Phyton, JavaScript, Ruby, etc), la librería de tipos se obtiene por medio de la interface idl y la herramienta xpidl [16] disponible en el gecko-sdk de mozilla.

Figura 5. Diagrama de clases del componente para verificación del soporte del equipo



De lo mostrado en la figura 5, la interface nsISupports es necesaria para acceder las funciones: QueryInterface, la cual es la encargada de preguntarle a un componente si implementa cierta interface, AddRef es llamada cuando se crea una nueva instancia del componente y Release, útil cuando se destruye, esta interface es de implementación obligatoria pues es gracias a ella que se puede tener una referencia de los objetos creados. La interface IsoporteEquipo por otro lado es la que expone la funcionalidad básica para verificar el soporte del equipo, la función verificarAntivirus se encarga de buscar el antivirus instalado y retorna su nombre, y la función VerificarFirewall que devuelve el estado del firewall, activado o no. La clase SoporteEquipo es la encargada de implementar la interface IsoporteEquipo: primero buscando en el registro de Windows los antivirus instalados y posteriormente utilizando una librería de terceros [17] para lograr la comunicación con el firewall. SoporteEquipoModule utiliza macros como NS_GENERIC_FACTORY_CONSTRUCTOR y *NS_IMPL_NSGETMODULE* que le permiten implementar las interfaces nsIFactory y nsIModule, las cuales son accesibles por medio del método nsGetModule, gracias a este método, se puede obtener información acerca del componente(donde se encuentra ubicado, tipo de patrón implementado, factory o singleton). La implementación de la interface nsISupports y de la clase SoporteEquipoModule se lograr gracias a la utilización de macros [18], gracias a estas, la reducción en el código es significativa, facilitando el desarrollo del componente, su registro y depuración, entre las macros utilizadas en el componente tenemos:

- ***NS_IMPL_NSGETMODULE (SoporteEquipo, components):*** Esta Macro permite implementar la interface nsIModule, esta es utilizada para registrar de forma automática componentes, sus parámetros son el nombre con el cual se va a implementar la interface y una lista que incluye el classname del componente, el identificador de la clase y el identificador del componente. Gracias a esta macro se puede exportar el método GetModulo, el cual brinda un punto de acceso al componente desde el cual se pueden descubrir interfaces, sus funciones y clases.

- ***NS_GENERIC_FACTORY_CONSTRUCTOR (SoporteEquipo):*** Esta macro implementa el patrón factoría, cuyo objetivo es ocultar la inicialización e implementación del componente

a los clientes, gracias a esto se puede instanciar un componente solo con conocer su Contract ID o ClassID.

- ***NS_DECL_ISUPPORTS:*** Esta macro es útil ya que implementa funciones como AddRef, Release, útiles a la hora de gestionar el número de interfaces creadas, y QueryInterface necesaria para descubrir interfaces.

- **NS_IMPL_ISUPPORTSn:** Esta macro permite implementar la interfaces nsISupports, que es útil ya que todo componente debe implementar esta interfaces y la macro hace. La letra n al final hace referencia al número de interfaces que expone el componente.

- ***NS_INIT_ISUPPORTS ():*** Esta macro implementa el constructor de nsISupports.

6. INSTALACIÓN

Inicialmente la extensión necesita suministrar dos archivos que controlan como se instalan y donde se encuentran los recursos de la extensión, estos archivos son: chrome.manifiest e install.rdf. Estos archivos deben ir dentro de la carpeta modulob2c. Para luego empaquetar la extensión en un archivo xpi, el cual es un archivo Zip con la extensión renombrada a xpi. Este instalador es multiplataforma y para su consecución primeramente se deben empaquetar los archivos dentro de la carpeta Chrome en un .jar, que al igual que el xpi es un Zip renombrado a jar. Finalmente La instalación de la extensión se realiza abriendo el archivo xpi con el navegador Firefox.

7. CASO DE EJECUCION

En la figura 6, se puede ver la interface de la extensión, la cual está ubicada como un panel lateral en Firefox, este panel contiene dos pestañas, la principal, gestión de seguridad en donde se muestra la información referente a las políticas verificadas (botón ver detalles), su nivel de cumplimiento (bajo, medio o alto) y el nivel de la transacción. La pestaña recomendaciones en donde se pueden ver las recomendaciones generales y de la transacción en curso. Este panel es accesible desde la barra de navegación, ver, panel lateral, MVPS para transacciones B2C. Esta figura presenta el estado del nivel de seguridad cuando se accede a realizar una transacción de B2C a la página del ***BANCO DE BOGOTÁ (http:// www.bancodebogota.com)***, vemos que la primera política ***Actualización del Navegador***, se encuentra como de alto cumplimiento, vemos también que el sitio cuenta con ***Certificado de autenticidad***, presenta también un nivel de conexión cifrado y por lo tanto una ***Conexión segura*** para el usuario, además verifica la ***Seguraridad del equipo*** como alta en cuanto a actualización y funcionamiento del antivirus y del firewall, y por último la extensión calcula el nivel de seguridad que para este caso es ***Alta***, por lo tanto se sugiere realizarla. En caso que no se cumplan todas estas verificaciones la extensión genera un conjunto de recomendaciones que el usuario puede revisar referente a la transacción que se va a realizar.

8. PRUEBAS DE SOFTWARE

Las pruebas realizadas a la extensión se centraron en dos niveles; pruebas de bajo nivel y pruebas de alto nivel. Las pruebas de bajo nivel se usaron para comprobar la lógica interna del programa para lo cual se utilizó técnicas de diseño de casos de prueba de caja blanca. Por otro lado, el nivel más alto es útil para comprobar los requerimientos funcionales por medio de técnicas de diseño de casos de prueba de caja negra [19].

- ***Pruebas de caja blanca realizadas:*** ya que estas se basan en detalles procedimentales, comprobando los caminos lógicos del código, bucles y/o condicionales, se debe comprobar que en varios puntos los estados correspondan con los esperados en ese punto. Para lograr esto, técnicas como la de caminos básicos se hacen muy útiles [19], esta técnica permite encontrar y usar una medida para verificar el conjunto de caminos de ejecución básico, los casos de prueba garantizan que la prueba ejecuta cada sentencia del programa por lo menos una vez, la medida de la cual se habla aquí es la complejidad ciclomática, la cual es una métrica del software que proporciona una medida cuantitativa de la complejidad lógica de un programa y define el número de caminos independientes en un proceso, un camino independiente es aquel que induce a un nuevo conjunto de sentencias o condición dentro de un proceso y se representan por medio de grafos, en la figura 7 se puede observar el grafo del proceso utilizado por la extensión para obtener y verificar el certificado de una página web.

Figura 6: Interface de la extensión (políticas verificadas para la página principal del Banco de Bogotá)

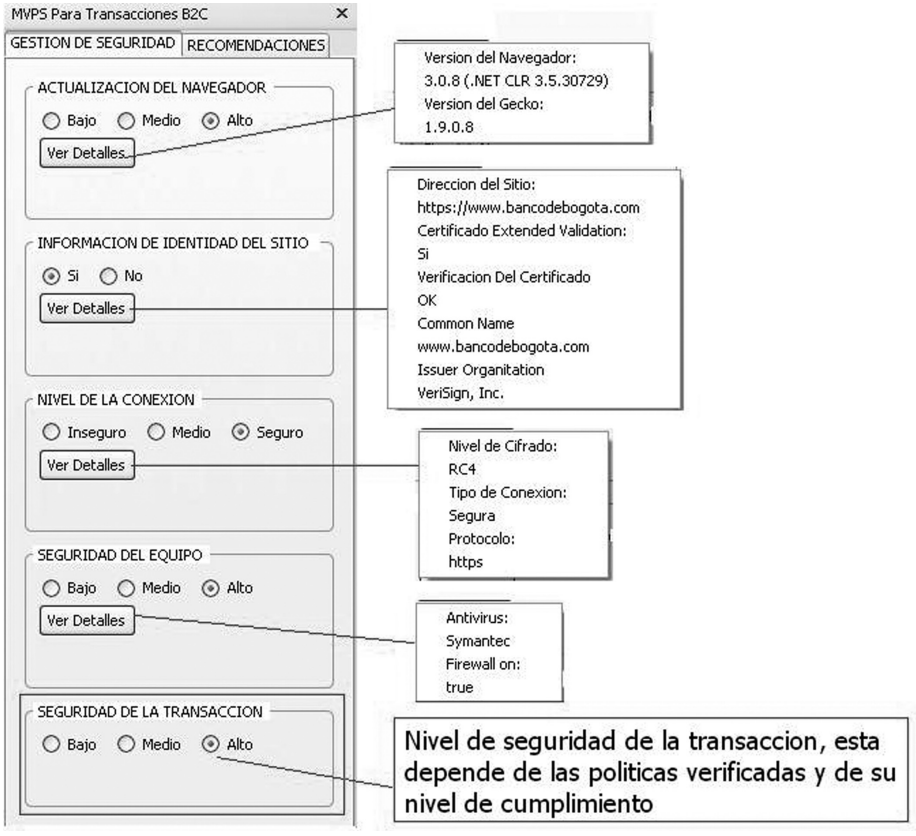
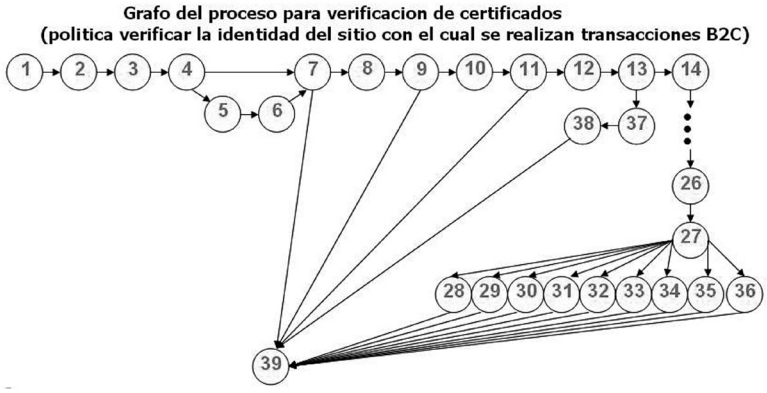


Figura 7: Grafo de proceso utilizado para verificar certificados



Para calcular la complejidad ciclomática del grafo anterior se debe tener en cuenta lo siguiente:

- Aquellos nodos que contiene una condición y se caracteriza porque dos o más aristas emergen de él se les denomina nodo predicado.
- La complejidad ciclomática de un camino sin nodos predicados se calcula tomando:

  V(G) = A – N + 2, donde A es el número de aristas del grafo y N es el número de nodos del mismo.

- La complejidad ciclomática de un camino con nodos predicados se calcula de la siguiente manera:
  V(G) = P + 1, donde P es el número de nodos predicados contenidos en el grafo de flujo G.

Los resultados obtenidos revelan que el número de pruebas mínimas a realizar son 10, y que la complejidad ciclomática es V(G) = 1 + 1 = 2, los nodos 4, 7 ,9 y 11 no se consideran nodos predicados ya que en estos puntos existen condicionales que solo evalúan que exista una condición, el total de caminos independientes entonces es de 10.

- ***Pruebas de caja negra realizadas:*** Estas pruebas deben ser complementarias a las pruebas de caja blanca y con ellas se pretende encontrar errores de los siguientes tipos: funciones incorrectas o ausentes, errores de interfaz de usuario, errores de rendimiento y errores de inicialización y terminación. Estas pruebas se llevan a cabo sobre la interfaz grafica para comprobar que los resultados obtenidos y entradas de datos son los correctos.

CONCLUSIONES Y RECOMENDACIONES

- El análisis de riesgos se convierte en un factor primordial a la hora de conocer los requerimientos de seguridad de toda organización o proceso dentro de esta.
- La implementación de cualquier política de seguridad exige la utilización de estándares, guías, y /o procedimientos que permitan gestionar todo su ciclo de vida.
- La utilización de guías y estándares además garantiza que las políticas sean tenidas en cuenta y que su diseño sea coherente con los objetivos que se quieren alcanzar al implementar la política.
- Tener políticas implementadas permite describir las acciones a tomar para garantizar que los activos no se degraden por las amenazas a los que están expuestos.
- El diseño de la política debe permitir revisiones que permitan verificar la efectividad de los controles o salvaguardas impuestos para disminuir los riesgos que puedan afectar el activo a los que haga referencia la política.
- Cuando el usuario instala el componente en su navegador, al ingresar a un página de comercio electrónico, chequea según las políticas de seguridad definidas: actualización del navegador, autenticación del sitio, seguridad de la conexión, seguridad del equipo (antivirus y firewall), y dependiendo, determina el grado de seguridad y le presenta sugerencias al usuario de acuerdo a si es baja, media o alta, que le permitirán tomar decisiones sobre la conveniencia de realizarla.
- Si se desea acceder componentes desde JavaScript tenga en cuenta que estos no son accesibles si su javascript no se encuentra registrado en el Chrome y que los componentes son creados usando el patrón Factory (los cuales deben ser instanciados utilizando el método createInstance) o Singleton (los cuales deben ser instanciados utilizando el método getServices).
- Utilizar las últimas versiones de gecko o versiones libres de vulnerabilidades para crear sus componentes.
- Implementar la interfaces nsISupport ya que todo componente XPCOM debe hacerlo, gracias a esta interface se puede preguntar por las interfaces implementadas por el componente y gestionar la creación y destrucción del mismo. Utilice macros para implementar esta interface.
- Configurar los path necesarios para poder utilizar las herramientas desde una ubicación especifica.
- XPCOM gestiona que el llamado y retorno de las funciones se haga con los parámetros adecuados, para esto utiliza, primero la macro NS_IMETHODIMP que es un método que se encarga de retornar y pasar el tipo adecuado, y por último nsresult para controlar errores que puedan ocurrir, para poder retornar cualquier tipo de dato se debe utilizar el siguiente código.

```
//////////**********/////////
*_retval = (char*) nsMemory::Clone(myResult,
    sizeof(char)*(strlen(myResult)+1));
        return *_retval ? NS_OK :
    NS_ERROR_OUT_OF_MEMORY;
//////////**********/////////
```

Esta es la forma de controlar el mapeo al tipo dato que se desea retornar.

- Hay código que es común cuando se desarrollo un componente XPCOM y que puede ser desarrollado utilizando macros, gracias a ellas se reducen el tiempo de desarrollo por lo cual es una buena razón su utilización. Entre las más importantes a la hora de la implementación de componentes tenemos; NS_IMPL_ISUPPORTS, NS _INIT_ISUPPORTS, NS_DECL_ISUPPORTS, NS_GENERIC_FACTORY_CONSTRUCTOR, NS_IMPL_NSGETMODULE
- Aunque desarrollar componentes desde C++ permite alcanzar todos las interfaces del Framework, es recomendable no utilizar aquellas interfaces que se encuentren en estado no congelado, puesto que estas pueden desaparecer en un futuro o cambiar drásticamente poniendo en riesgo el buen funcionamiento de nuestro componente.

REFERENCIAS BIBLIOGRAFICAS

[1] López Cañón, Manuel. Plataforma de Preferencias de Privacidad: Un caso práctico, el mundo (online). Octubre 2006. http://www.di.uniovi.es/~cueva/asignaturas/doctorado/2006/trabajos/p3p.pdf

[2] Serrano, C. Modelo Integral para el profesional en ingeniería. Popayán: Editorial Universidad del Cauca, 2005, pp.56-65.

[3] Ministerio de Administraciones Públicas. Metodología de Análisis y Gestión de riesgos de los Sistemas de Información. I-Método, NIPO 326-05-047-X. Magerit versión 2(online). Madrid: junio 2006. 154p. http://www.csi.map.es/csi/pdf/magerit_v2/metodo_v11_final.pdf

[4] Instituto argentino de normalización (IRAM). Information technology, code of practice for information security management. ISO 17799:2005. Segunda edición. Buenos Aires (Argentina): IRAM, 2002. 82p.

[5] Internet Engineering Task Force. Site Security Handbook. RFC2196 (online). Pittsburgh: septiembre 1997. http://translate.google.com.co/translate?hl=es&sl=en&u=http://www.faqs.org/rfcs/rfc2196.html&sa=X&oi=translate&resnum=1&ct=result&prev=/search%3Fq%3Drfc%2B2196%26hl%3Des

[6] Universidad Nacional de Colombia Vicerrectoria General Dirección Nacional de Informática y Comunicaciones. Guía para elaboración de políticas de seguridad (online). 2003. 13 p. http://www.unal.edu.co/seguridad/documentos/guia_para_elaborar_politicas_v1_0.pdf

[7] Jacobson I, Booch G. Proceso unificado de desarrollo de software. Addison Wesley.

[8] Rick Parrish. An introdution a xpcom (online). febrero 2001. http://www.ibm.com/developerworks/webservices/library/co-xpcom.html

[9] Mindi MacDoweell. Browsing Safely: Understanding Active Content and Cookies. National Cyber Alert System, Ciber Security Tip ST04-012 (online). Cer: junio 20 de 2007. http://www.us-cert.gov/cas/tips/ST04-012.html

[10] Instituto Nacional de Tecnologías de la Comunicación. Estudio sobre la seguridad de la información y eConfianza en los hogares españoles (online). Abril 2008. 79 p. http://www.inteco.es/Seguridad/Observatorio/Estudios_e_Informes/Estudios_e_Informes_1/estudio_seguridad_hogares_oleada3

[11] Nigel McFarlane. Rapid Application Development with Mozilla. Prentice Hall: 2004, pp 175-176.

[12] Neil Deakin. Xul Tutorial (online). 2006. https://developer.mozilla.org/en/XUL_reference.

[13] W3C. DTDS (online). http://www.w3.org/TR/xhtml1/dtds.html

[14] Microsoft. Archivos DLL (online). Noviembre de 2007. http://msdn.microsoft.com/es-es/library/1ez7dh12.aspx[15] Lucent Technologies. Nmake User´s Guide (online). Septiembre 1998. 295 p. http://www1.bell-labs.com/project/nmake/download/manual/3.1.2/nmake_user_1098.pdf

[15] Lucent Technologies. Nmake User´s Guide (online). Septiembre 1998. 295 p. http://www1.bell-labs.com/project/nmake/download/manual/3.1.2/nmake_user_1098.pdf

[16] Rick Parrish. XPCOM Component basics (online). febrero 2001. http://www.ibm.com/developerworks/webservice[17] Sunde Peter. Programmatically disable/enable Windows firewall (online). Enero de 2007. http://www.codeproject.com/KB/IP/enable_disable_firewall.aspx

[17] Sunde Peter. Programmatically disable/enable Windows firewall (online). Enero de 2007. http://www.codeproject.com/KB/IP/enable_disable_firewall.aspx

[18] Doug, Turner; Ian, Oeschger. Creating XPCOM Components. 2003. 261p.

[19] Pressman Roger S. Ingeniería del software un enfoque practico (quinta edición). Mc Graw Hill. España: 2002 . 640p.

[20] Portal del Banco de Bogotá. 2010 Junio. Políticas de Seguridad. URL: http://www.bancodebogota.com.co/portal/page?_pageid=793,4212425&_dad=portal&_schema=PORTAL

[21] Norton from Symantec. Junio 2010. Una estrategia de seguridad para las transacciones en línea. Norton. URL http://www.symantec.com/es/mx/home_homeoffice/library/article.jsp?aid=transaction_security_plan

[22] Echarri, A. & Guillermo, S., (2006, jun). Google presenta Google Checkout, un nuevo servicio de pago online seguro, rápido y eficaz. Centro de prensa Google. URL http://www.google.com/press/pressrel/checkout.html