Universidad de las Ciencias Informáticas

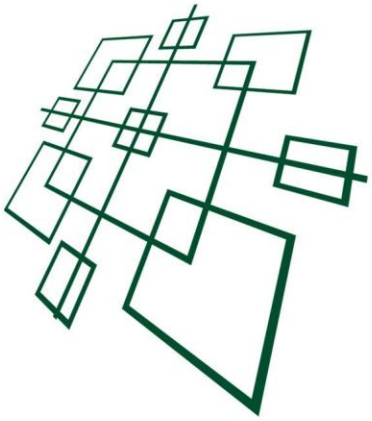
Dirección de Informatización

Akademos
Gestión Académica

SEGURIDAD EN AKADEMOS 2.0

Trabajo de diploma para optar por el título de:

Ingeniero en Ciencias Informáticas



Autor: Orlando Barriel Victorial

Tutor: Ing. Geidis Sánchez Michel

Junio 2008

"Año 50 de la Revolución"

Declaración de Autoría

Declaro que soy el único autor de este trabajo y autorizo a la Dirección de Informatización de la Universidad de las Ciencias Informáticas de hacer uso del mismo en su beneficio.

Para que así conste firmo la presente a los 5 días del mes de junio del año 2008.

_____ _____

Orlando Barriel Victorial Geidis Sánchez Michel

Opinión del tutor del trabajo de diploma

Título: **Seguridad en Akademos 2.0**
Autor: Orlando Barriel Victorial.

El tutor del presente Trabajo de Diploma considera que durante su ejecución el estudiante mostró las cualidades que a continuación se detallan:
Dominio del tema desarrollado, correcta organización del contenido, coherencia en el desempeño de la problemática planteada, logrando independencia, originalidad y creatividad, adecuada utilización de los recursos tecnológicos y los medios, capacidad explicativa y poder de síntesis, con un desempeño positivo en la realización y la finalidad del mismo.
El trabajo cuenta además con una buena calidad científico-técnica y responde a las necesidades del sistema en su nueva versión.
Por todo lo anteriormente expresado considero que el estudiante está apto para ejercer como Ingeniero en Ciencias Informáticas; y propongo que se le otorgue al Trabajo de Diploma la calificación de 5 puntos.

5 días del mes de junio del año 2008.

_____
Geidis Sánchez Michel

Agradecimientos

A mi madre por creer y esperar siempre lo mejor de mi.

A mi abuela por su apoyo en todo momento y por enseñarme a ser mejor cada día.

A mi hermana y a Luis por estar ahí siempre pendientes de mi futuro.

A mi Mónica intranquila, cariñosa, humilde, noble; por alumbrarme cada día con la luz de su alma.

A todos mis amigos por estar juntos en los buenos y malos tiempos.

A mis compañeros de apto por su apoyo en todos estos años.

A Jota por ser mi alumno mas destacado y superar al maestro.

A mis compañeros de batalla en el proyecto: Grisell,Norges, Dianly, Camejo,Molina,Miño, Joe, Darién, Frank,Andry y el Robe.

A Geidis por ayudarme tanto en la confección de este documento.

A la Revolución por darme la oportunidad de estudiar, forjarme como profesional y convertir en realidad mi sueño.

A todo el que de una forma u otra ha contribuido a mi desarrollo como profesional y como persona.

Dedicatoria

A mi madre y abuela por hacerme todo lo que soy.

Resumen

El presente trabajo cuyo título es: **Seguridad en Akademos 2.0,** surge dado la necesidad de realizar un estudio sobre la seguridad del Sistema Automatizado para la Gestión Académica (Akademos) y para el desarrollo de una nueva versión del mismo, teniendo en cuenta que este esta siendo diseñado bajo las políticas de migración a software libre llevadas a cabo por el país y en especial por la universidad.

En el documento se realiza un estudio de la gestión de la seguridad en Akademos, criticando los errores que presenta y reutilizando componentes confiables, además de analizar también las herramientas y protocolos más destacados a nivel global en cuanto a seguridad refiere, teniendo en cuenta su pertenencia al mundo del Open Source. Aparte de la seguridad también se analizan los servicios Web que brinda hoy el sistema, los cuales son muy usados por la comunidad universitaria.

En el trabajo se propone utilizar varios protocolos de seguridad como son SSL y HTTPS, un método de encriptación como es el caso de MD5 y también se definen formas de registrar las incidencias de los usuarios en el sistema, asi como gestionar los permisos de estos.

El presente estudio puede ser utilizado en sistemas con características similares.

Índice

Índice de figuras

Introducción

La gestión académica con todos sus procesos es una de las actividades fundamentales de cualquier centro de estudios, pues su misión no es otra que la de servir de hilo conductor de la gestión administrativa del alumno a lo largo de su vida académica. Hoy en día muchos centros escolares poseen herramientas que ayudan al control de esta actividad y agilizan sus procesos. La Universidad de las Ciencias Informáticas (UCI) también cuenta con un sistema de este tipo.

El Sistema de Gestión Académica, Akademos, surge para automatizar el proceso docente de la UCI. Cuenta con siete módulos que permiten satisfacer las demandas de los clientes. Es utilizado por más de diez mil usuarios, estudiantes, profesores, directivos y el personal de secretaría, entre los cuales es ampliamente aceptado. Ha sido desplegado fuera del centro con los módulos de matrícula y reportes, exactamente en las Facultades Regionales manteniendo sus buenos resultados.

Actualmente el país viendo las potencialidades del uso de software libre ha decidido explotar las ventajas que esto proporciona y la universidad no ajena a ello, ha trazado una política de migración a software libre, y al estar en este marco, de forma novedosa, Akademos esta siendo reestructurado contemplando los requerimientos de la migración y analizando profundamente todos los cambios a que estará sujeto.

La migración trae como consecuencia cambios en el lenguaje de programación, el servidor Web y el gestor de base de datos, lo que conlleva a cambios potenciales en lo que a seguridad refiere. Hoy en día la seguridad del sistema esta basada fundamentalmente en la que brinda implícitamente el gestor de base de datos SQL Server 2000 y el servidor Web Internet Information Service (IIS), los cuales tienen la característica de ser software propietario, lo que provoca que para la migración deban de analizarse todos los posibles candidatos para reemplazar las actuales herramientas con la menor pérdida de confidencialidad, integridad y

disponibilidad de los datos, además de asegurar que todos los implicados en el sistema (estudiantes, directivos, profesores y personal de secretaria) tengan un papel activo en todos los procesos del mismo teniendo siempre en cuenta que el dinamismo del proceso de gestión académica constituye la principal fuente de riesgos para un sistema que intente automatizarlo.

Encontrar una forma eficiente de controlar la seguridad a partir de las nuevas necesidades del sistema y contemplando las políticas de migración es la premisa de esta investigación.

Situación Problémica

En la actualidad el sistema ha sido desplegado en la sede central y facultades regionales de la UCI, así como en el Instituto Superior del MININT Eliseo Reyes Rodríguez "Capitán San Luis". Aunque Akademos gestiona de manera eficiente la información generada, resulta compleja su instalación y montaje en centros de estudios con particularidades diferentes a las que existen en la UCI, dada las características de la arquitectura que tiene definida y su dependencia aunque sea mínima a otros sistemas de la universidad.

El uso por parte de Akademos de software propietario es otro de los inconvenientes, dadas las restricciones que tiene el país con las compañías norteamericanas y las muy caras licencias que estos imponen por la utilización de sus productos. Además de los servicios de soporte que estas herramientas necesitan.

Como resultado de los cambios en la arquitectura no concebidos en la propuesta inicial del sistema, la seguridad se ha visto afectada. Las malas prácticas de programación dadas por el desconocimiento de los programadores de cómo estaba concebida la gestión de la seguridad y las necesidades de la nueva versión, hacen necesario redefinir la seguridad en Akademos 2.0.

Problema Científico

¿Cómo gestionar la seguridad en Akademos 2.0 teniendo en cuenta las nuevas funcionalidades del sistema y las políticas de migración a software libre?

Objetivo General

Proponer un método eficiente de gestionar la seguridad en Akademos 2.0 que se rija por las exigencias de la migración a software libre y añada nuevas funcionalidades al método de gestión de seguridad actual.

Objetivos Específicos

- Redefinir la forma de registrar las incidencias de todos los usuarios en el sistema.
- Mejorar el método usado para la asignación de permisos en el sistema.
- Definir un estándar para brindar Servicios Web seguros que sean utilizados por otras aplicaciones que interactúen con Akademos.

Objeto de Estudio

Sistema Automatizado para la Gestión Académica , Akademos 2.0.

Campo de Acción

Módulo de Seguridad de Akademos 2.0.

Tareas

- Criticar el diseño original del Módulo de Seguridad y analizar su estado actual.
- Analizar los componentes de seguridad, para aplicaciones Web, más usados actualmente a nivel global.
- Brindar la propuesta de seguridad de Akademos 2.0.
- Estandarizar los servicios Web que brindará el sistema a otras aplicaciones.

Pregunta Científica

¿Qué estándares y protocolos de seguridad utilizar en Akademos 2.0 y cómo integrar el sistema a estos?

Ideas a defender

- Estudio de la estructura actual del módulo de Seguridad adquiriendo conocimientos sobre lo positivo y experiencias sobre lo mal concebido.
- Propuesta de un estándar para el desarrollo de Servicios Web.
- Propuesta de seguridad de Akademos 2.0.

Métodos Científicos de Investigación

Por método o proceso científico se entiende aquellas prácticas utilizadas y ratificadas por la comunidad científica como válidas a la hora de proceder con el fin de exponer y confirmar sus teorías. Las teorías científicas, destinadas a explicar de alguna manera los fenómenos que observamos, pueden apoyarse o no en experimentos que certifiquen su validez. Como métodos

científicos a usar en esta investigación tenemos los métodos teóricos y empíricos los cuales aportaran al desarrollo del trabajo calidad y organización.

- **Métodos Teóricos**:

- Analítico – sintético
- Inductivo – deductivo
- Hipotético – deductivo
- Análisis histórico – lógico
- Genético
- Modelación

De estos métodos se usan:

- Analítico – sintético, ya que dada una gran cantidad de información sobre los anteriores sistemas, se puede establecer puntos de comparación y canalizar lo que se estime conveniente para mejorar el desarrollo del sistema actual.
- Inductivo – deductivo, permitirá el diseño del sistema a partir de un sistema previo y de los requisitos funcionales que se tengan.
- Análisis histórico – lógico, en el que se estudiará como se gestiona en la actualidad la seguridad en aplicaciones Web.

- **Métodos Empíricos**

- Observación
- Entrevista
- Encuesta
- Experimento

De los cuales se utlizan:

- Entrevista, dado que se hace necesario saber de cada módulo del sistema donde hayan posibles brechas de seguridad y cuales son los usuarios que deben acceder a cada uno de ellos.

Capítulo 1: Fundamentación Teórica

1.1 Introducción

La seguridad se hace cada vez más una premisa para el desarrollo del internet, el crecimiento de la gran red de redes ha puesto en marcha mecanismos de control y vigilancia, los cuales se innovan cada día dado la gran versatilidad de la información en la red y los cambios que esta última sufre. Muchos son los usuarios que hoy en día acceden a los diferentes servicios que se brindan y muchas son las prestaciones que estos realizan. Las amenazas que genera esta gran concentración de información no tienen un mecanismo de prevención eficiente, ya que cada día son más diversos los ataques que se llevan a cabo y en mayor número. La Web no esta exenta de todos los peligros que la conectividad trae consigo, un sistema de seguridad eficaz es la premisa de todo desarrollador Web.

Akademos en su nueva versión, al cual accederán muchos usuarios será proclive a ataques informáticos de diferentes índoles. Revisar y estudiar como se encuentra gestionada la seguridad en general y en específico en aplicaciones Web, es el objetivo de este capítulo.

1.2 Gestión académica

El proceso de gestión académica no es más que el control, organización y dirección de todas las actividades del proceso docente. Este proceso se encuentra regido por la secretaría docente, profesores y directivos del sistema. [1]

Sistemas de Gestión Académica

Muchos son los sistemas que actualmente funcionan tanto nacional como internacionalmente, un breve repaso de algunos de estos se hará a continuación.

Internacionales:

S.I.G.A: Sistema Integrado de Gestión Académica

Permite como herramienta confiable, la constatación y medición de procesos y acciones que realiza la Universidad.

SIGA permite recopilar y tratar la información de los distintos niveles institucionales, lo que relaciona la labor docente, investigativa, de creación artística y extensión, y permite conocer cuantitativa y cualitativamente las iniciativas desarrolladas por los distintos organismos de la Institución.

Desde los datos, e información ingresada por las diversas Unidades de la Institución, es posible generar Indicadores y Reportes de análisis en Docencia de Pregrado; Postgrado, de Actividad Académica, de Investigación y Extensión.

SIGA está abierto permanentemente para que cada facultad complete manualmente su información y realice la actualización continua. También existe la posibilidad de efectuar cargas masivas de información para agilizar el proceso y actualizar los antecedentes de cada Facultad o Instituto.

Análisis Institucional y Proyectos genera periódicamente Indicadores de Gestión, en los ámbitos descritos, y los presenta en línea a través del Sitio de la Dirección a cargo, además se efectúan y entregan reportes de ingreso de información por Facultad.

XesCampus

XesCampus está construida en torno a una base de datos relacional a la que se accede mediante tecnologías estándar de Internet (HTML, XML, ASP), tanto desde la propia interfaz de usuario del sistema, como desde las herramientas ofimáticas y de análisis de datos. La aplicación está dividida en varios módulos que abordan procesos característicos de la gestión académica (planificación, matrícula, expedientes, bolsa de empleo, títulos).

El enfoque del sistema es integral e integrado, pretende ser una solución de gestión y de análisis que agrupe toda la información relacionada con la labor académica y que, por tanto, incluya todos los procesos de la institución. Al mismo tiempo, todos estos procesos están integrados con el resto de sistemas de información de la organización.

XesCampus permite a múltiples usuarios interaccionar directamente con la información a través de tecnologías ofimáticas y de Internet.

UNAN-León

El objetivo principal del sistema es proveer a la universidad de mecanismos automatizados que faciliten la planificación, organización, gestión y control académico.

Entre las principales funciones del sistema tenemos:

- Simplificar y organizar los trámites y procesos académicos.
- Apoyar a las secretarias académicas en los procesos académicos.
- Registrar y controlar las prematrículas.
- Registrar y controlar las matrículas.
- Registrar y controlar el pago de aranceles de los estudiantes.
- Registrar y controlar las becas.
- Registrar y controlar las estadísticas.

- Registrar y controlar los convenios de la universidad.
- Emitir resultados del proceso de admisión.

Nacionales:

GESTACAD: **Sistema de Gestión Académica**

Este sistema surgió con la idea de desarrollar un software que permitiera automatizar la gestión académica de las Universidad de Matanzas, el mismo gestiona parte de la información académica de los estudiantes universitarios y la información de los profesores que forman parte del proceso docente educativo.

El sistema está concebido por módulos, entre los que se diferencian, los módulos de actualización de datos y el sitio Web, a través del cual se muestran las diversas salidas de la aplicación.

UCIMAT

Este sistema fue implementado en el curso 2002/2003 con vistas a agilizar el censo de estudiantes. Permite búsquedas de estudiantes por determinados criterios, incluye la matrícula y modificación de los datos de los estudiantes, así como reportes generales de los datos que están matriculados en el sistema, impidiendo que se puedan realizar otras operaciones importantes entre los que se puede destacar el reingreso, registro de traslado u otros datos requeridos.

El sistema fue desarrollado por profesores de nuestro centro, gestiona la información académica de los estudiantes universitarios y la información de los profesores que forman parte del proceso docente educativo. El sistema está concebido por módulos, entre los que se diferencian, los de actualización de datos y el sitio Web, a través del cual se muestran las diversas salidas de la aplicación.

Akademos

Akademos es el actual Sistema de Gestión Académica que se usa en la UCI. Está dividido por 7 módulos y está centrado en una aplicación Web. Usa tecnologías ASP.NET y Servicios Web XML que sirven para la interoperabilidad con otros sistemas. El sistema permite la interacción de todos sus usuarios con el fin de gestionar el proceso docente de la universidad.

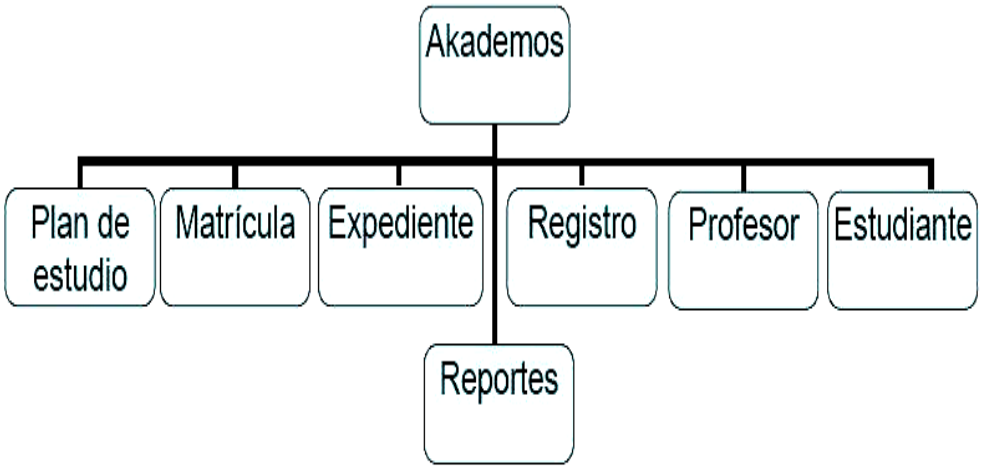


Figura 1.1: Estructura de los módulos de Akademos

1.3 Normas básicas de seguridad

Existen principios básicos de seguridad que cualquier aplicación o servicio Web debe cumplir para un buen funcionamiento de la gestión de la seguridad, los cuales son:

- **Validación de la entrada y salida de información:** La entrada y salida de información es el principal mecanismo que dispone un atacante para enviar o recibir código malicioso contra el sistema. Por tanto, siempre debe verificarse que cualquier dato entrante o saliente es apropiado y en el formato que se espera. Las características de estos datos deben estar predefinidas y debe verificarse en todas las ocasiones.

- **Diseños simples:** Los mecanismos de seguridad deben diseñarse para que sean los más sencillos posibles, huyendo de sofisticaciones que compliquen excesivamente la vida a los usuarios. Si los pasos necesarios para proteger de forma adecuada una función o módulo son muy complejos, la probabilidad de que estos pasos no se ejecuten de forma adecuada es muy elevada.
- **Utilización y reutilización de componentes de confianza:** Debe evitarse reinventar la rueda constantemente. Por tanto, cuando exista un componente que resuelva un problema de forma correcta, lo más inteligente es utilizarlo.
- **Defensa en profundidad:** Nunca confiar en que un componente realizará su función de forma permanente y ante cualquier situación. Se ha de disponer de los mecanismos de seguridad suficientes para que cuando un componente del sistema falle ante un determinado evento, otros sean capaces de detectarlo.
- **Tan seguros como el eslabón más débil:** La frase "se garantiza la seguridad, ya que se utiliza SSL" es realmente muy popular, pero también es muy inexacta. La utilización de SSL garantiza que el tráfico en tránsito entre el servidor y el cliente se encuentra cifrado, pero no garantiza nada acerca de los mecanismos de seguridad existentes.
  Por tanto, no se debe fiar únicamente de los mecanismos de seguridad "exteriores", sino que es preciso identificar cuales son los puntos precisos en los que deben establecerse las medidas de seguridad. Si no se hace este trabajo, seguro que los atacantes si lo harán.
- **La seguridad gracias al desconocimiento no funciona:** El simple hecho de ocultar algo no impide que, a medio o largo plazo, llegue a ser descubierto. Tampoco es ninguna garantía de que tampoco será descubierto a corto plazo.
- **Verificación de privilegios:** Los sistemas deben diseñarse para que funcionen con los menos privilegios posibles. Igualmente, es importante que los procesos únicamente dispongan de los privilegios necesarios para desarrollar su función, de forma que queden compartimentados.

- **Ofrecer la mínima información:** Ante una situación de error o una validación negativa, los mecanismos de seguridad deben diseñarse para que faciliten la mínima información posible. De la misma forma, estos mecanismos deben estar diseñados para que una vez denegada una operación, cualquier operación posterior sea igualmente denegada.

Otros aspectos son: Consideraciones de arquitectura, mecanismos de autenticación, gestión de sesiones de usuario, control de acceso, registro de actividad, prevención de problemas comunes, consideraciones de privacidad y criptografía.[2]

1.4 Seguridad para aplicaciones Web

La seguridad en los sitios Web debe tener una atención especial por parte de sus desarrolladores, dado su nivel de complejidad y la diversidad de amenazas que se pueden presentar. El grado de confianza que pueda generar una aplicación se ve afectado por la calidad de la seguridad en esta. Como medidas generales de seguridad se deben tener en cuenta las siguientes:

- **Ejecutar aplicaciones con privilegios mínimos**

  Cuando la aplicación se ejecuta, lo hace en un contexto que tiene privilegios específicos en el equipo local y posiblemente en equipos remotos.

- **Conocer a los usuarios**

  En muchas aplicaciones los usuarios pueden tener acceso al sitio sin necesidad de proporcionar credenciales. Si es el caso, la aplicación obtiene acceso a recursos al ejecutarse en el contexto de un usuario predefinido.

- **Protegerse contra entradas malintencionadas**

Como regla general, nunca se debe dar por sentado que la entrada proveniente de los usuarios es segura. A los usuarios malintencionados les resulta fácil enviar información potencialmente peligrosa desde el cliente a la aplicación.

- **Tener acceso seguro a bases de datos**

Normalmente, las bases de datos tienen sus propios sistemas de seguridad. Un aspecto importante de una aplicación Web protegida es diseñar un modo de que ésta pueda tener acceso a la base de datos de forma segura.

- **Crear mensajes de error seguros**

Si no se es cuidadoso, un usuario malintencionado puede deducir información importante sobre la aplicación a partir de los mensajes de error que ésta muestra.

- **Mantener secretos de forma segura**

Secretos son cualquier información que no se desea que conozcan otras personas. Un secreto típico es una contraseña o una clave cifrada. Por supuesto, si un usuario malintencionado puede desvelar el secreto, los datos protegidos por dicho secreto se verán expuestos.

- **Usar cookies de forma segura**

Las cookies constituyen un modo fácil y útil de almacenar la información específica disponible sobre los usuarios. Sin embargo, como se envían al explorador del equipo, son vulnerables a la suplantación u otros usos malintencionados.

- **Protegerse contra amenazas de denegación de servicio**

Un modo indirecto en el que un usuario malintencionado puede comprometer una aplicación es haciendo que ésta no esté disponible. El usuario malintencionado puede mantener la aplicación demasiado ocupada como para que pueda servir a otros usuarios, o puede simplemente bloquearla.[3]

1.5 Sistemas Gestores de Bases de Datos(SGBD)

1.5.1 PostgreSQL

PostgreSQL es un servidor de base de datos objeto relacional libre, liberado bajo la licencia BSD. Como muchos otros proyectos open source, el desarrollo de PostgreSQL no es manejado por una sola compañía sino que es dirigido por una comunidad de desarrolladores y organizaciones comerciales las cuales trabajan en su desarrollo, dicha comunidad es denominada el PGDG (PostgreSQL Global Development Group).[4]

Principales características:

- ✓ Alta concurrencia.
- ✓ Amplia variedad de tipos nativos.
- ✓ Claves ajenas también denominadas Llaves ajenas o Llaves Foráneas (foreign keys).
- ✓ Disparadores (triggers).
- ✓ Vistas.
- ✓ Integridad transaccional.
- ✓ Herencia de tablas.
- ✓ Tipos de datos y operaciones geométricas.

1.5.2 MySQL

MySQL es un sistema de gestión de base de datos relacional, multihilo y multiusuario con más de seis millones de instalaciones. MySQL AB desarrolla MySQL como software libre en un esquema de licenciamiento dual. MySQL AB pertenece a Sun Microsystems desde enero de 2008.

Por un lado se ofrece bajo la GNU GPL para cualquier uso compatible con esta licencia, pero las empresas que quieran incorporarlo en productos privativos pueden comprar a la empresa una licencia específica que les permita este uso. Está desarrollado en su mayor parte en ANSI C.

Al contrario que proyectos como Apache, donde el software es desarrollado por una comunidad pública y el copyright del código está en poder del autor individual, MySQL es propiedad y está patrocinado por una empresa privada, que posee el copyright de la mayor parte del código.

Esto es lo que posibilita el esquema de licenciamiento anteriormente mencionado. Además de la venta de licencias privativas, la compañía ofrece soporte y servicios. Para sus operaciones contratan trabajadores alrededor del mundo que colaboran vía Internet. MySQL AB fue fundado por David Axmark, Allan Larsson, y Michael Widenius.

Principales características:

- Usa GNU Automake, Autoconf, y Libtool para portabilidad.
- Uso de multihilos mediante hilos del kernel.
- Usa tablas en disco b-tree para búsquedas rápidas con compresión de índice
- Tablas hash en memoria temporales.
- El código MySQL se prueba con Purify (un detector de memoria perdida comercial) así como con Valgrind, una herramienta GPL.
- Completo soporte para operadores y funciones en cláusulas select y where.
- Completo soporte para cláusulas group by y order by, soporte de funciones de agrupación.
- Seguridad: ofrece un sistema de contraseñas y privilegios seguro mediante verificación basada en el host y el tráfico de contraseñas está cifrado al conectarse a un servidor.
- Soporta gran cantidad de datos. MySQL Server tiene bases de datos de hasta 50 millones de registros.
- Se permiten hasta 64 índices por tabla (32 antes de MySQL 4.1.2). Cada índice puede consistir desde 1 hasta 16 columnas o partes de columnas. El máximo ancho de límite son 1000 bytes (500 antes de MySQL 4.1.2).
- Los clientes se conectan al servidor MySQL usando sockets TCP/IP en cualquier plataforma. En sistemas Windows se pueden conectar usando named pipes y en sistemas Unix usando ficheros socket Unix.

- En MySQL 5.0, los clientes y servidores Windows se pueden conectar usando memoria compartida.
- MySQL contiene su propio paquete de pruebas de rendimiento proporcionado con el código fuente de la distribución de MySQL.

1.6 Lenguajes de programación

1.6.1 PHP

PHP es un lenguaje de programación usado normalmente para la creación de páginas Web dinámicas. PHP es un acrónimo recursivo que significa "PHP Hypertext Pre-processor" (inicialmente PHP Tools, o, Personal Home Page Tools), y se trata de un lenguaje interpretado. Últimamente también puede ser utilizado para la creación de otros tipos de programas incluyendo aplicaciones con interfaz gráfica usando las librerías Qt o GTK+.

Ventajas

- Es un lenguaje multiplataforma.
- Capacidad de conexión con la mayoría de los manejadores de base de datos que se utilizan en la actualidad, destaca su conectividad con MySQL (no funciona muy bien con tecnologías Microsoft)
- Capacidad de expandir su potencial utilizando la enorme cantidad de módulos (llamados ext's o extensiones).
- Posee una amplia documentación en su página oficial, entre la cual se destaca que todas las funciones del sistema están explicadas y ejemplificadas en un único archivo de ayuda.
- Es libre, por lo que se presenta como una alternativa de fácil acceso para todos.

- Permite las técnicas de Programación Orientada a Objetos.
- Biblioteca nativa de funciones sumamente amplia e incluida.
- No requiere definición de tipos de variables.
- Tiene manejo de excepciones.

Desventajas

- No posee una abstracción de base de datos estándar, sino bibliotecas especializadas para cada motor (a veces más de una para el mismo motor).
- No posee adecuado manejo de internacionalización, Unicode, etc.
- Por su diseño dinámico no es pasible de ser compilado.
- Por sus características promueve la creación de código desordenado y complejo de mantener.

1.6.2 JAVA

Java es un lenguaje de programación orientado a objetos desarrollado por Sun Microsystems a principios de los años 90. El lenguaje en sí mismo toma mucha de su sintaxis de C y C++, pero tiene un modelo de objetos más simple y elimina herramientas de bajo nivel, que suelen inducir a muchos errores, como la manipulación directa de punteros o memoria.

Las aplicaciones Java están típicamente compiladas en un bytecode, aunque la compilación en código máquina nativo también es posible. En el tiempo de ejecución, el bytecode es normalmente interpretado o compilado a código nativo para la ejecución, aunque la ejecución directa por hardware del bytecode por un procesador Java también es posible.

La implementación original y de referencia del compilador, la máquina virtual y las librerías de clases de Java fueron desarrolladas por Sun Microsystems en 1995. Desde entonces, Sun ha controlado las especificaciones, el desarrollo y evolución del lenguaje a través del Java Community Process, si bien otros han desarrollado también implementaciones alternativas de estas tecnologías de Sun, algunas incluso bajo licencias de software libre.

Entre noviembre de 2006 y mayo de 2007, Sun Microsystems liberó la mayor parte de sus tecnologías Java bajo la licencia GNU GPL, de acuerdo con las especificaciones del Java Community Process, de tal forma que prácticamente todo el Java de Sun es ahora software libre (aunque la biblioteca de clases de Sun que se requiere para ejecutar los programas Java todavía no es Open Source).

1.7 Frameworks

En el desarrollo de software, un framework es una estructura de soporte definida en la cual otro proyecto de software puede ser organizado y desarrollado. Típicamente, un framework puede incluir soporte de programas, bibliotecas y un lenguaje interpretado entre otros software para ayudar a desarrollar y unir los diferentes componentes de un proyecto.

Un framework representa una arquitectura de software que modela las relaciones generales de las entidades del dominio. Provee una estructura y una metodología de trabajo la cual extiende o utiliza las aplicaciones del dominio.

Los Frameworks son diseñados con el intento de facilitar el desarrollo de software, permitiendo a los diseñadores y programadores pasar más tiempo identificando requerimientos de software que tratando con los tediosos detalles de bajo nivel de proveer un sistema funcional. Sin embargo, hay quejas comunes acerca de que el uso de frameworks añade código innecesario y que la

preponderancia de frameworks competitivos y complementarios significa que el tiempo que se pasaba programando y diseñando ahora se gasta en aprender a usar frameworks.

Fuera de las aplicaciones en la informática, un framework puede ser considerado como el conjunto de procesos y tecnologías usados para resolver un problema complejo. Es el esqueleto sobre el cual varios objetos son integrados para una solución dada.

Un framework simplifica el desarrollo de una aplicación mediante la automatización de algunos de los patrones utilizados para resolver las tareas comunes. Además, un framework proporciona estructura al código fuente, forzando al desarrollador a crear código más legible y más fácil de mantener. Por último, un framework facilita la programación de aplicaciones, ya que encapsula operaciones complejas en instrucciones sencillas.

A continuación se describen algunos de lo frameworks mas usados para el desarrollo de aplicaciones Web.

1.7.1 Symfony

Symfony es un completo framework diseñado para optimizar, gracias a sus características, el desarrollo de las aplicaciones Web. Para empezar, separa la lógica de negocio, la lógica de servidor y la presentación de la aplicación Web. Proporciona varias herramientas y clases encaminadas a reducir el tiempo de desarrollo de una aplicación Web compleja. Además, automatiza las tareas más comunes, permitiendo al desarrollador dedicarse por completo a los aspectos específicos de cada aplicación. El resultado de todas estas ventajas es que no se debe reinventar la rueda cada vez que se crea una nueva aplicación Web.

Symfony está desarrollado completamente con PHP 5. Ha sido probado en numerosos proyectos reales y se utiliza en sitios Web de comercio electrónico de primer nivel. Symfony es compatible con la mayoría de gestores de bases de datos, como MySQL, PostgreSQL, Oracle y SQL Server

de Microsoft. Se puede ejecutar tanto en plataformas *nix (Unix, Linux, etc.) como en plataformas Windows. A continuación se muestran algunas de sus características.

Características de Symfony

Symfony se diseñó para que se ajustara a los siguientes requisitos:

- Fácil de instalar y configurar en la mayoría de plataformas (y con la garantía de que funciona correctamente en los sistemas Windows y *nix estándares) .
- Independiente del sistema gestor de bases de datos.
- Sencillo de usar en la mayoría de casos, pero lo suficientemente flexible como para adaptarse a los casos más complejos.
- Basado en la premisa de *"*convenir en vez de configurar*"*, en la que el desarrollador solo debe configurar aquello que no es convencional.
- Sigue la mayoría de mejores prácticas y patrones de diseño para la Web.
- Preparado para aplicaciones empresariales y adaptable a las políticas y arquitecturas propias de cada empresa, además de ser lo suficientemente estable como para desarrollar aplicaciones a largo plazo.
- Código fácil de leer que incluye comentarios de PHPDocumentor y que permite un mantenimiento muy sencillo.
- Fácil de extender, lo que permite su integración con librerías desarrolladas por terceros.

Automatización de características de proyectos Web

Symfony automatiza la mayoría de elementos comunes de los proyectos Web, como por ejemplo:

- La capa de internacionalización que incluye Symfony permite la traducción de los datos y de la interfaz, así como la adaptación local de los contenidos.
- La capa de presentación utiliza plantillas y layouts que pueden ser creados por diseñadores HTML sin ningún tipo de conocimiento del framework. Los helpers incluidos

permiten minimizar el código utilizado en la presentación, ya que encapsulan grandes bloques de código en llamadas simples a funciones.

- Los formularios incluyen validación automatizada y relleno automático de datos (*"repopulation"*), lo que asegura la obtención de datos correctos y mejora la experiencia de usuario.
- Los datos incluyen mecanismos de escape que permiten una mejor protección contra los ataques producidos por datos corruptos.
- La gestión de la caché reduce el ancho de banda utilizado y la carga del servidor.
- La autenticación y la gestión de credenciales simplifican la creación de secciones restringidas y la gestión de la seguridad de usuario.
- El sistema de enrutamiento y las URL limpias permiten considerar a las direcciones de las páginas como parte de la interfaz, además de estar optimizadas para los buscadores.
- El soporte de e-mail incluido y la gestión de APIs permiten a las aplicaciones Web interactuar más allá de los navegadores.
- Los listados son más fáciles de utilizar debido a la paginación automatizada, el filtrado y la ordenación de datos.
- Los plugins, las factorías (patrón de diseño *"*Factory*"*) y los *"*mixin*"* permiten realizar extensiones a medida de Symfony.
- Las interacciones con Ajax son muy fáciles de implementar mediante los helpers que permiten encapsular los efectos JavaScript compatibles con todos los navegadores en una única línea de código.

Entorno de desarrollo y herramientas

Symfony puede ser completamente personalizado para cumplir con los requisitos de las empresas que disponen de sus propias políticas y reglas para la gestión de proyectos y la programación de aplicaciones. Por defecto incorpora varios entornos de desarrollo diferentes e incluye varias herramientas que permiten automatizar las tareas más comunes de la ingeniería del software:

- Las herramientas que generan automáticamente código han sido diseñadas para hacer prototipos de aplicaciones y para crear fácilmente la parte de gestión de las aplicaciones.
- El framework de desarrollo de pruebas unitarias y funcionales proporciona las herramientas ideales para el desarrollo basado en pruebas (*"*test-driven development*"*).
- La barra de depuración Web simplifica la depuración de las aplicaciones, ya que muestra toda la información que los programadores necesitan sobre la página en la que están trabajando.
- La interfaz de línea de comandos automatiza la instalación de las aplicaciones entre servidores.
- Es posible realizar cambios *"*en caliente*"* de la configuración (sin necesidad de reiniciar el servidor).
- El completo sistema de log permite a los administradores acceder hasta el último detalle de las actividades que realiza la aplicación. [5]

1.7.2 CodeIgniter

Es un framework para desarrollo de aplicaciones en PHP. Es Open Source, tiene una interfaz simple y un acceso a sus librerías bien estructurado. Cumple perfectamente el fin de cualquier framework, una estructura definida que de soporte a un proyecto Web y ayude a que este proyecto sea organizado y desarrollado. CodeIgniter es adecuado para desarrollos que no requieran un framework que marque mucho la aplicación, a parte para cuando sea necesario mucho rendimiento, pensado para aquellas aplicaciones que se ejecutan en hosting compartido que ejecutan muchas versiones de PHP con diferentes configuraciones.

También puede ser útil si no se quiere usar un framework con configuraciones iniciales, o que sea necesaria la línea de comandos. Dado esto podemos poner como sus principales características que posee un bajo uso de recursos, tiene un rendimiento excepcional y es altamente compatible con gran variedad de versiones y configuraciones de PHP.

1.8 Servicios Web

Un servicio Web es una colección de protocolos y estándares que sirven para intercambiar datos entre aplicaciones. Distintas aplicaciones de software desarrolladas en lenguajes de programación diferentes, y ejecutadas sobre cualquier plataforma, pueden utilizar los servicios Web para intercambiar datos en redes de ordenadores como Internet. La interoperabilidad se consigue mediante la adopción de estándares abiertos.[6]

Ventajas de los servicios Web

- Aportan interoperabilidad entre aplicaciones de software independientemente de sus propiedades o de las plataformas sobre las que se instalen.
- Los servicios Web fomentan los estándares y protocolos basados en texto, que hacen más fácil acceder a su contenido y entender su funcionamiento.
- Al apoyarse en HTTP, los servicios Web pueden aprovecharse de los sistemas de seguridad firewall sin necesidad de cambiar las reglas de filtrado.
- Permiten que servicios y software de diferentes compañías ubicadas en diferentes lugares geográficos puedan ser combinados fácilmente para proveer servicios integrados.
- Permiten la interoperabilidad entre plataformas de distintos fabricantes por medio de protocolos estándar.

Inconvenientes de los servicios Web

- Para realizar transacciones no pueden compararse en su grado de desarrollo con los estándares abiertos de computación distribuida como CORBA (Common Object Request Broker Architecture).
- Su rendimiento es bajo si se compara con otros modelos de computación distribuida, tales como RMI (Remote Method Invocation), CORBA, o DCOM (Distributed Component Object Model). Es uno de los inconvenientes derivados de adoptar un formato basado en texto. Y

es que entre los objetivos de XML no se encuentra la concisión ni la eficacia de procesamiento.

- Al apoyarse en HTTP, pueden esquivar medidas de seguridad basadas en firewall, cuyas reglas tratan de bloquear o auditar la comunicación entre programas a ambos lados de la barrera.
- Existe poca información de servicios Web para algunos lenguajes de programación

Estándares empleados por los servicios Web

- **Web Services Protocol Stack**: Así se denomina al conjunto de servicios y protocolos de los servicios Web.
- **XML (Extensible Markup Language):** Es el formato estándar para los datos que se vayan a intercambiar.
- **SOAP (Simple Object Access Protocol) o XML-RPC (XML Remote Producer Call):** Protocolos sobre los que se establece el intercambio.
- **Otros protocolos:** los datos en XML también pueden enviarse de una aplicación a otra mediante protocolos normales como HTTP (Hypertext Transfer Protocol), FTP (File Transfer Protocol), o SMTP (Simple Mail Transfer Protocol).
- **WSDL (Web Services Description Languages):** Es el lenguaje de la interfaz pública para los servicios Web. Es una descripción basada en XML de los requisitos funcionales necesarios para establecer una comunicación con los servicios Web.
- **UDDI (Universal Description, Discovery and Integration):** Protocolo para publicar la información de los servicios Web. Permite comprobar qué servicios Web están disponibles.
- **WS-Security (Web Service Security):** Protocolo de seguridad aceptado como estándar por OASIS (Organization for the Advancement of Structured Information Standards). Garantiza la autenticación de los actores y la confidencialidad de los mensajes enviados.

1.9 Métodos de encriptación

La encriptación es la única forma eficiente de transmitir información confidencial por Internet. El objetivo de la encriptación es garantizar la confidencialidad, integridad e irrefutabilidad de la información. El objetivo es desarrollar y aplicar mecanismos de encriptación que no puedan detectarse ni piratearse teóricamente.

La operación de cifrado consiste en algoritmos matemáticos complejos en los que la clave es una cifra larga. La fuerza de la encriptación depende de la longitud de la clave, es decir, el número de bits que tiene el número. Es imposible piratear un código mediante métodos técnicos si la clave utilizada es lo suficientemente larga.

A continuación se describen los principales métodos de encriptación usados en la Web y en general en cualquier aplicación informática.

1.9.1 MD5

En computación un hash se refiere a una función o método para generar claves o llaves que representen de manera casi unívoca a un documento, registro, archivo, etc. Una función hash puede generar claves iguales para objetos diferentes, ya que el rango de posibles claves es mucho menor que el de posibles objetos a resumir (las claves suelen tener en torno al centenar de bits, pero los ficheros no tienen un tamaño límite). Un algoritmo de hash es **MD5** (acrónimo de Message-Digest Algorithm *5*, Algoritmo de Resumen del Mensaje 5) el cual es un algoritmo de reducción criptográfico de 128 bits ampliamente usado. La codificación del MD5 de 128 bits es representada típicamente como un número de 32 dígitos hexadecimal. Es muy utilizado para autenticación cifrada.

1.9.2 SHA

La familia SHA (Secure Hash Algorithm, Algoritmo de Hash Seguro) es un sistema de funciones hash criptográficas relacionadas de la Agencia de Seguridad Nacional de los Estados Unidos y publicadas por el National Institute of Standards and Technology (NIST). El primer miembro de la

familia fue publicado en 1993 es oficialmente llamado SHA. Sin embargo, hoy día, no oficialmente se le llama SHA-0 para evitar confusiones con sus sucesores. Dos años más tarde el primer sucesor de SHA fue publicado con el nombre de SHA-1. Existen cuatro variantes más que se han publicado desde entonces cuyas diferencias se basan en un diseño algo modificado y rangos de salida incrementados: SHA-224, SHA-256, SHA-384, y SHA-512 (llamándose SHA-2 a todos ellos).

SHA-1 ha sido examinado muy de cerca por la comunidad criptográfica pública, y no se ha encontrado ningún ataque efectivo. No obstante, en el año 2004, un número de ataques significativos fueron divulgados sobre funciones criptográficas de hash con una estructura similar a SHA-1; lo que ha planteado dudas sobre la seguridad a largo plazo de SHA-1.

SHA-0 y SHA-1 producen una salida resumen de 160 bits de un mensaje que puede tener un tamaño máximo de 2^{64} bits, y se basa en principios similares a los usados por el profesor Ronald L. Rivest del MIT en el diseño de los algoritmos de resumen de mensaje MD4 y MD5.

1.10 Protocolos de seguridad

1.10.1 Secure Socket Layer (SSL)

El protocolo SSL es un sistema diseñado y propuesto por Netscape Communications Corporation. Se encuentra en la pila OSI entre los niveles de TCP/IP y de los protocolos HTTP, FTP, SMTP, etc. Proporciona sus servicios de seguridad cifrando los datos intercambiados entre el servidor y el cliente con un algoritmo de cifrado simétrico, típicamente el RC4 o IDEA, y cifrando la clave de sesión de RC4 o IDEA mediante un algoritmo de cifrado de clave pública, típicamente el RSA. La clave de sesión es la que se utiliza para cifrar los datos que vienen del cliente y van al servidor seguro. Se genera una clave de sesión distinta para cada transacción, lo cual permite que aunque sea reventada por un atacante en una transacción dada, no sirva para

descifrar futuras transacciones. MD5 se usa como algoritmo de hash. Proporciona cifrado de datos, autenticación de servidores, integridad de mensajes y, opcionalmente, autenticación de cliente para conexiones TCP/IP.

1.10.2 HTTPS

El protocolo HTTPS es la versión segura del protocolo HTTP. El sistema HTTPS utiliza un cifrado basado en las Secure Socket Layer (SSL) para crear un canal cifrado (cuyo nivel de cifrado depende del servidor remoto y del navegador utilizado por el cliente) más apropiado para el tráfico de información sensible que el protocolo HTTP. Cabe mencionar que el uso del protocolo HTTPS no impide que se pueda utilizar HTTP. Es aquí, cuando el navegador advertirá sobre la carga de elementos no seguros (HTTP), estando conectados a un entorno seguro (HTTPS).

Los protocolos https son utilizados por navegadores como: Safari, Internet Explorer, Mozilla Firefox, Opera,... entre otros.

Es utilizado principalmente por entidades bancarias, tiendas en línea, y cualquier tipo de servicio que requiera el envío de datos personales o contraseñas.

El puerto estándar para este protocolo es el 443.

1.10.3 OpenSSL

OpenSSL es un proyecto de software desarrollado por los miembros de la comunidad Open Source para libre descarga y está basado en SSLeay desarrollado por Eric Young y Tim Hudson. Consiste en un robusto paquete de herramientas de administración y librerías relacionadas con la criptografía, que suministran funciones criptográficas a otros paquetes como OpenSSH y navegadores Web (para acceso seguro a sitios HTTPS). Estas herramientas ayudan al sistema a implementar el Secure Sockets Layer (SSL), así como otros protocolos relacionados con la seguridad, como el Transport Layer Security (TLS). Este paquete de software es importante para

cualquiera que esté planeando usar cierto nivel de seguridad en su máquina con un sistema operativo libre basado en GNU/Linux. OpenSSL también permite crear certificados digitales que se pueden aplicar a un servidor, por ejemplo Apache.

1.11 Estimación de costos

Para el desarrollo de esta investigación se usan los siguientes recursos materiales:

- 1 PC estimada en 500 CUC
- 1 Mesa de PC 50 CUC
- 1 Silla 15 CUC

También se usan materiales de oficina como libretas, lápices y hojas con un costo total de 20 CUC.

Además se consumirá también alrededor de 50 CUC en energía eléctrica y conexión a Internet para un total de 635 CUC.

1.12 Conclusiones

En este capítulo se ha hecho un estudio del estado del arte donde se analizaron las herramientas, sistemas y protocolos que pudieran ser propuestas para su previa utilización en el nuevo sistema Akademos 2.0. Además de haberse realizado un análisis de las principales normas de seguridad que se deben tener en cuenta cuando se vaya a desarrollar un software.

Capítulo 2: Análisis y estudio de la propuesta

2.1 Introducción

En este capítulo se analizará como se encuentra gestionada la seguridad en Akademos hoy en día, los errores e inconvenientes que presenta y se estudiarán los posibles estándares y herramientas a utilizar como propuesta para la gestión de la seguridad en Akademos 2.0.

2.2 Seguridad actual de Akademos

La seguridad hoy día en Akademos se encuentra centrada en un módulo que integra todos los restantes. Este módulo es el encargado de gestionar la seguridad en tres niveles: datos, recursos lógicos y elementos específicos del sistema.

2.2.1 Seguridad en los datos

Para gestionar la seguridad en los datos el sistema utiliza la seguridad que trae implícita el gestor de bases de datos SqlServer 2000, el cual es capaz de ocuparse de la seguridad tanto lógica como de integridad de los datos almacenados.

El gestor tiene definidos usuarios con permisos sobre la base de datos y solo los administradores del sistema son los encargados de definir estos usuarios y sus permisos. Se utilizan procedimientos almacenados para la inserción, modificación y eliminación en la base de datos, previniendo así los ataques de Sql Injection. Además como medida para la preservación de los datos, se generan copias de seguridad de la base de datos (backups), todos los días a una hora específica.

2.2.2 Seguridad en los recursos lógicos

En cuanto a la seguridad de los recursos lógicos se encarga el sistema operativo, Windows en este caso, el cual a través del IIS (Internet Information Service) restringe el acceso a componentes del sistema ya sean las páginas generadas por la aplicación, los servicios que este brinda o los archivos de cualquier índole que se generan.

2.2.3 Seguridad en elementos específicos del sistema

Por último con respecto a la seguridad de los elementos específicos del sistema, Akademos tiene implementados varios niveles de acceso con lo cual se restringen los permisos de un usuario. También se controlan las incidencias de todos estos usuarios en su decursar por la aplicación con el fin de controlar su paso por el sistema y las acciones que estos realizan.

2.3 .NET Framework

Akademos al usar tecnologías .Net para su desarrollo, utiliza también el .NET framework 1.1.

.NET Framework es un componente integral de Windows que admite la creación y la ejecución de la siguiente generación de aplicaciones y servicios Web XML. El diseño de .NET Framework está enfocado a cumplir los objetivos siguientes:

- Proporcionar un entorno coherente de programación orientado a objetos, en el que el código de estos se pueda almacenar y ejecutar de forma local, pero distribuida en Internet, o de forma remota.
- Proporcionar un entorno de ejecución de código que reduzca lo máximo posible la implementación de software y los conflictos de versiones.
- Ofrecer un entorno de ejecución de código que fomente la ejecución segura del mismo, incluso del creado por terceras personas desconocidas o que no son de plena confianza.

- Proporcionar un entorno de ejecución de código que elimine los problemas de rendimiento de los mismos que utilicen secuencias de comandos o intérpretes de comandos.
- Ofrecer al programador una experiencia coherente entre tipos de aplicaciones muy diferentes, como las basadas en Windows o en el Web.
- Basar toda la comunicación en estándares del sector para asegurar que el código de .NET Framework se puede integrar con otros tipos de código.

.NET Framework contiene dos componentes principales: Common Language Runtime y la biblioteca de clases de .NET Framework. Common Language Runtime es el fundamento de la tecnología. El motor de tiempo de ejecución se puede considerar como un agente que administra el código en tiempo de ejecución y proporciona servicios centrales, como la administración de memoria, la administración de subprocesos y la interacción remota, al tiempo que aplica una seguridad estricta a los tipos y otras formas de especificación del código que fomentan su seguridad y solidez. De hecho, el concepto de administración de código es un principio básico del motor de tiempo de ejecución. El código destinado al motor de tiempo de ejecución se denomina código administrado, a diferencia del resto, que se conoce como código no administrado. La biblioteca de clases, el otro componente principal de .NET Framework, es una completa colección orientada a objetos de tipos reutilizables que se pueden emplear para desarrollar aplicaciones que abarcan desde las tradicionales herramientas de interfaz gráfica de usuario (GUI) o de línea de comandos hasta las aplicaciones basadas en las innovaciones más recientes proporcionadas por ASP.NET, como los formularios Web Forms y los servicios Web XML.

.NET Framework puede alojarse en componentes no administrados que cargan Common Language Runtime en sus procesos e inician la ejecución de código administrado, con lo que se crea un entorno de software en el que se pueden utilizar características administradas y no administradas. En .NET Framework no sólo se ofrecen varios hosts de motor de tiempo de ejecución, sino que también se admite el desarrollo de estos hosts por parte de terceros.

2.4 IIS (Internet Information Services)

Internet Information Services (IIS) ofrece los servicios de soporte de un servidor Web seguro, disponible y escalable en el que ejecutar sitios y aplicaciones Web. IIS es una herramienta de creación de una plataforma de comunicación sólida de aplicaciones de red dinámicas. Akademos hoy en día usa como servidor Web IIS, el cual tiene perfecto acoplamiento con el framework de .Net que usa el sistema ya que los dos pertenecen a tecnologías Microsoft. IIS tiene muchas características que lo hacen un potente servidor Web entre las que se encuentran:

- Ayuda a limitar el acceso de los usuarios anónimos a la aplicación.
- Tiene una configuración para los permisos muy versátil y efectiva.
- Permite restringir el acceso al sistema por direcciones IP.
- Puede utilizar el cifrado para proteger la información privada que los clientes intercambian con el servidor Web. (SSL)
- Permite el control de sesiones.

IIS como toda herramienta propietaria tiene sus inconvenientes entre los cuales se destaca que requiere de pagos de licencias muy costosas para su uso y explotación, IIS solo funciona en sistemas operativos Windows y por último, se ha visto superado por Apache en casi todos los aspectos.

2.5 Autenticación

La autenticación es el proceso por el que se comprueba la identidad de alguien o algo, para ver si es lo que dice ser. Ese "alguien" o "algo" se denomina principal. La autenticación requiere pruebas de identidad, denominadas credenciales. Por ejemplo, una aplicación cliente puede presentar una contraseña como sus credenciales. Si la misma presenta las credenciales correctas, se asume que es quien dice ser. [7]

IIS provee de varias formas de autenticación las cuales se mencionan a continuación:

- Anónima
- Básica
- Básica sobre SSL
- Implícita
- Autenticación de Windows integrada
- Certificados de cliente a través de SSL

El actual sistema utiliza la autenticación integrada a Windows la cual resulta útil sobre todo en entornos en Intranet. Utiliza NTLM o Kerberos. El cliente debe pertenecer al mismo dominio que el servidor o a alguno que posea relaciones de confianza con el del servidor. IIS autorizará el acceso a la aplicación si las credenciales coinciden con las de una cuenta de usuario válida. El uso de este tipo de autenticación hace fácil el trabajo si se encuentran bajo un mismo dominio el servidor y el cliente, creando dependencia la aplicación a un directorio activo.

2.6 Gestión de usuarios

La gestión de los usuarios es, hoy en día, un área de entidad propia, en la medida en que la relación de éstos con los sistemas es crucial a la hora de asegurar un correcto funcionamiento de los servicios puestos a su disposición.

El sistema actualmente no gestiona sus propios usuarios ya que usa los usuarios y grupos de usuarios definidos en el directorio activo, de ahí que se deriven muchos problemas a la hora de la asignación de permisos y el control de roles.

2.7 Permisos y roles

Se pudiera decir que la raíz de los problemas que hoy presenta la seguridad en Akademos es que el sistema no gestiona sus usuarios, lo cual trae como consecuencia dependencia al directorio activo y entrada directa de información a las tablas de la base de datos.

Actualmente el sistema no gestiona sus propios roles, simplemente al estar bajo un directorio activo se alimenta de la información que este almacena, como son sus usuarios y grupos de dominio. Esta información se guarda parcialmente en la base de datos (solo los grupos de dominios), y es usada para la asignación de permisos. Esto trae como consecuencia que al no estar la aplicación conectada a un directorio activo, los roles se tengan que poner manualmente en la base de datos, ya que no existe una interfaz capaz de gestionarlos.

En cuanto a los permisos, el sistema cuenta con una pequeña interfaz donde se pueden dar permisos para matricular, pero dado el problema de los roles, esos permisos no son más que adicionar, en el grupo de dominio que ya tiene aprobación, el usuario deseado; generando una traba ya que no solo existe este permiso sino que existen muchos más definidos y no hay manera de gestionarlos. Otro inconveniente es la inexistencia de un formulario para la autentificación dada la sujeción de Akademos a un directorio activo y la dificultad de no gestionar sus propios roles y usuarios.

2.8 Incidencias

Las incidencias de los usuarios en su decursar en el sistema son registradas en la base de datos, dando como resultado un sistema de log que podrá ser usado para posteriores revisiones al sistema. Por otra parte, no obstante la valiosa información recogida en las incidencias, no existe una aplicación o interfaz capaz de generar reportes de incidencias, teniendo que recurrir en caso de ser necesario directamente a las tablas de la base de datos que recogen esta información.

2.9 Encriptación y validaciones

Uno de los mayores errores que presenta el sistema actualmente, en cuanto a seguridad concierne, es el de la encriptación, solo dos de sus módulos de manera novedosa usan MD5 como método de encriptación, pero en general no existe un estándar ni se usa un método para cifrar adecuadamente las variables que se pasan por URL, dándole información al usuario que no debería saber y que podría en un futuro usar en contra de la aplicación.

Todos los datos que son introducidos al sistema son validados a través de expresiones regulares y java script, tratando de tener en cuenta la defensa en profundidad y que no se puede confiar nunca en las entradas de datos de los usuarios.

2.10 Control de errores

El control de errores es fundamental si queremos garantizar la robustez de una aplicación. Akademos apegado a esta idea, controla los errores en la capa de presentación, en la de negocio y en la base de datos.

En la presentación cuenta con una página que es la manejadora de errores, ocultando los mensajes que el sistema emite y dándole a usuario la mínima información sobre estos. En el negocio tiene definidas clases que manejan los errores que el usuario visualizará y en los datos cuenta con una tabla que, siendo una especialización de las incidencias del sistema, almacena los errores que se producen en la aplicación, con el fin garantizar una guía para el mantenimiento o el desarrollo de nuevas versiones y también para revisiones de calidad.

Todas estas medidas están muy bien concebidas pero mal implementadas, por ejemplo: la página que se encarga de manejar los errores en la interfaz emite algunas veces mensajes de error que se pudieran usar en contra del sistema, hay clases, de las que se encuentran definidas, que no cumplen con su objetivo, aparte de algunas repetir código innecesariamente (copian

código de otras) y por último la tabla que almacena los errores tiene campos que dependen mucho de los objetos que trae consigo el compilador VS-2003 para aplicaciones .NET creando dependencia del lenguaje.

2.11 Servicios Web en Akademos

Las prestaciones de servicios en Akademos es una de sus mayores funcionalidades, ya que casi todos los sistemas del centro se retroalimentan de los datos que almacena este. Para el desarrollo de estos servicios se utilizan XML-WebServices, los cuales con todos sus estándares posibilitan la comunicación entre aplicaciones sin depender del lenguaje en que estén programadas. La seguridad de estos servicios la maneja principalmente el IIS, el cual controla el acceso de los usuarios, aunque en la actualidad estos servicios sean de total acceso.

2.12 Estado de las tablas de la base de datos pertenecientes al subsistema de seguridad

Para la buena gestión de la seguridad en el sistema existen diversas tablas en la base de datos que posibilitan almacenar información referente a los permisos existentes, roles y control de incidencias. Existen ocho tablas que permiten gestionar la seguridad del sistema:

- tblRoles: se guardan los roles que tendrán permisos asignados en el sistema, nótese que estos son grupos de dominio derivados del directorio activo, haciéndose esto una traba ya que se crea dependencia de este último.

- tblRolMiembro: supuestamente en esta tabla se deberían almacenar los usuarios que pertenecen a un rol, pero en la actualidad se encuentra vacía, ya que los roles al ser grupos de dominio tienen definidos cuales son sus usuarios.
- tblAccion: se encarga de almacenar las acciones predefinidas en el sistema.
- tblTipoObjeto: se almacenan los tipos de objetos previamente definidos en el sistema.
- tblTipoObjetoAccion: tiene como propósito tratar de crear reglas o conceptos en cuanto las acciones que se puedan realizar sobre los distintos tipos de objetos. En la actualidad la tabla no se usa, ya que en los permisos se definen esas reglas.
- tblPermiso: tabla encargada de almacenar los permisos que tendrán los usuarios que interactúen con el sistema.
- tblIncidencia: se encarga de almacenar las incidencias generadas por el paso de los usuarios por el sistema.
- tblError: tabla que tiene como funcionalidad almacenar los errores que genere el sistema.

2.13 Propuestas para la seguridad de Akademos 2.0

2.13.1 Método de encriptación

Como método de encriptación tenemos MD5, cuyo concepto y características se encuentran abordadas en el capitulo 1 y solo nos queda preguntarnos: ¿Porqué MD5 y no cualquier otro encriptador? Las respuestas son las siguientes:

1- Uno de los protocolos de seguridad propuesto para el nuevo sistema trabaja con MD5 implícitamente.

2- PHP, que es el lenguaje propuesto para Akademos 2.0, tiene funciones definidas para el uso de MD5.
3- Ha resultado ser un algoritmo fiable y muy usado en la actualidad, ejemplo de esto es que usando un ataque de fuerza bruta contra él, se tendrián que hacer 340282366920938463463374607431768211456 intentos para poder descifrarlo y por otra parte, casi todos los lenguajes tienen definidos tanto clases como funciones que permiten el trabajo con MD5.

2.13.2 Protocolos de seguridad

SSL y HTTPS son los protocolos de seguridad propuestos para el sistema dado que, por una parte SSL tiene características que lo hacen uno de los protocolos de seguridad más usados en la actualidad y se ha convertido en el estándar internacional de comunicaciones seguras, además este protocolo tiene como característica principal que trata de resolver tres problemas fundamentales:

1- Seguridad de la información

SSL garantiza que terceros no tengan acceso a la información mientras viaja por internet al encriptarla.

2- Integridad de los datos

La información recibida desde un servidor por SSL puede ser "validada" para comprobar que no ha sido alterada en la trayectoria.

3- Autenticidad de los datos

Mediante los algoritmos de encriptación, es posible comprobar que los datos realmente han llegado del servidor que el cliente espera. Esto evita que alguien se haga pasar por un sitio para

cometer fraudes (evitando ataques como Phishing o Man in the Middle). Mediante el uso de certificados digitales avalados por autoridades certificadoras, SSL incorpora un eslabón más en la cadena de confianza. Lo que un certificado digital hace es agregar el endoso de un tercero que garantiza la integridad y existencia de la organización que envía los datos.

Durante el protocolo SSL Handshake, el cliente y el servidor intercambian una serie de mensajes para negociar las mejoras de seguridad (ver Figura 2.1).

Por otra parte, en referencia a HTTPS, se puede decir que al ser la versión segura de HTTP, garantiza la seguridad en la transmisión de datos en la Web usando SSL como protocolo de seguridad. Es también soportado por PHP y garantiza la transmisión de datos principalmente de formularios.

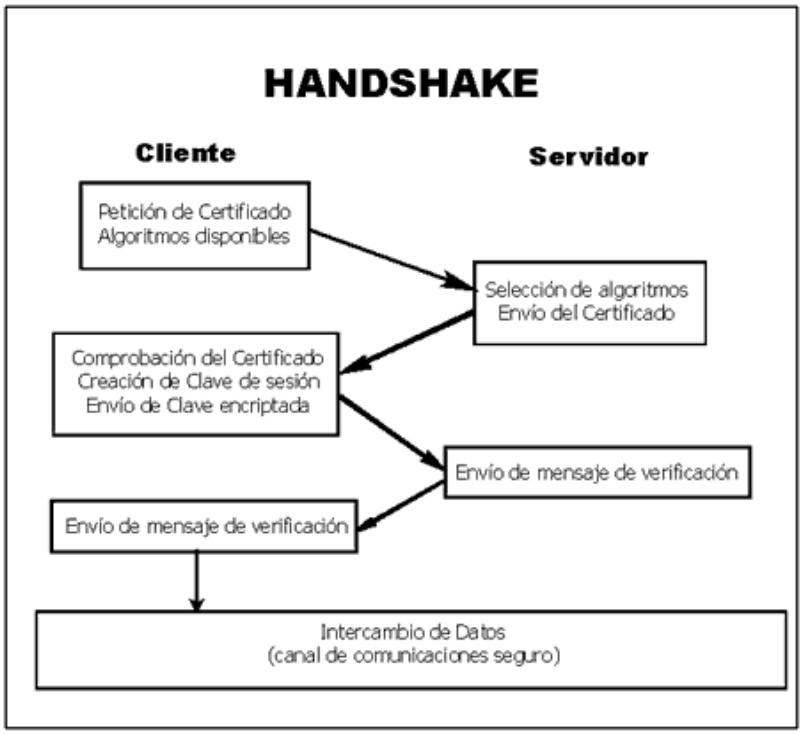


Figura 2.1: Proceso de Handshake del SSL.

2.13.3 Apache

Al ser Apache la propuesta de arquitectura de servidor Web para el sistema, se usarán las características de seguridad que este trae consigo, como son:

- Se puede restringir el acceso por direcciones IP.
- Se pueden utilizar muchos módulos entre los que se encuentra mod_security.
- En Apache la autorización a recursos es gestionada o bien mediante la directiva en el fichero principal de configuración, o bien mediante la configuración de la carpeta a través de ficheros .htaccess.
- El control de acceso se puede llevar a cabo mediante las directivas y , o a través del fichero de configuración .htaccess para controlar una carpeta específica.
- En Apache la autentificación puede estar gestionada por distintos módulos, dependiendo de la forma de implementación.

Apache es el servidor Web mas usado en la actualidad ver Figura 2.2.

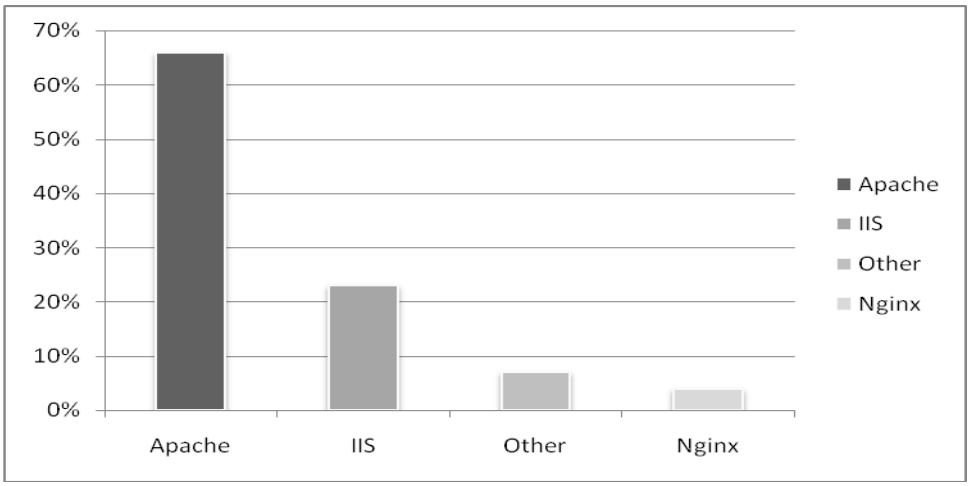


Figura 2.2: Uso de los principales servidores Web a nivel global.

2.13.4 PostgreSQL

PostgreSQL será el gestor de base de datos a usarse en el nuevo sistema Akademos 2.0. Este SGBD tiene muchas potencialidades en cuanto a seguridad refiere entre las cuales se pueden destacar que a seguridad de la base de datos está implementada en varios niveles:

- Protección de los ficheros de la base de datos. Todos los ficheros almacenados en la base de datos están protegidos contra escritura por cualquier cuenta que no sea la del superusuario de PostgreSQL.
- Las conexiones de los clientes al servidor de la base de datos están permitidas, por defecto, únicamente mediante sockets Unix locales y no mediante sockets TCP/IP. Ha de arrancarse el demonio con la opción -i para permitir la conexión de clientes no locales.
- Las conexiones de los clientes se pueden restringir por dirección IP y/o por nombre de usuario mediante el fichero pg_hba.conf situado en PG_DATA.
- Las conexiones de los clientes pueden ser autentificadas mediante otros paquetes externos.
- A cada usuario de PostgreSQL se le asigna un nombre de usuario y (opcionalmente) una contraseña. Por defecto, los usuarios no tienen permiso de escritura a bases de datos que no hayan creado.
- Los usuarios pueden ser incluidos en grupos, y el acceso a las tablas puede restringirse en base a esos grupos.
- PostgreSQL proporciona mecanismos para permitir a los usuarios limitar el acceso que otros usuarios tendrán a sus datos.
- Permite la creación de funciones y reglas para realizar determinadas acciones sobre la base de datos.
- PostgreSQL proporciona utilidades para realizar las copias de seguridad de su sistema.

2.13.5 Symfony

El framework Symfony será el utilizado en la nueva versión de Akademos y este potente framework proporciona muchas ventajas en lo que a seguridad concierne, entre sus principales funcionalidades en cuanto a seguridad tenemos:

- Seguridad de la Acción: La posibilidad de ejecutar una acción puede ser restringida a usuarios con ciertos privilegios. Las herramientas proporcionadas por Symfony para este propósito permiten la creación de aplicaciones seguras, en las que los usuarios necesitan estar autenticados antes de acceder a alguna característica o a partes de la aplicación. Añadir esta seguridad a una aplicación requiere dos pasos: declarar los requerimientos de seguridad para cada acción y autenticar a los usuarios con privilegios para que puedan acceder estas acciones seguras.
- Restricción de Acceso: Antes de ser ejecutada, cada acción pasa por un filtro especial que verifica si el usuario actual tiene privilegios de acceder a la acción requerida. En Symfony, los privilegios están compuestos por dos partes: las acciones seguras requieren que los usuarios estén autenticados y las credenciales son privilegios de seguridad agrupados bajo un nombre y que permiten organizar la seguridad en grupos. Si el usuario no posee permisos suficientes se le redirecciona a la pagina “secure”, ver Figura 2.3.

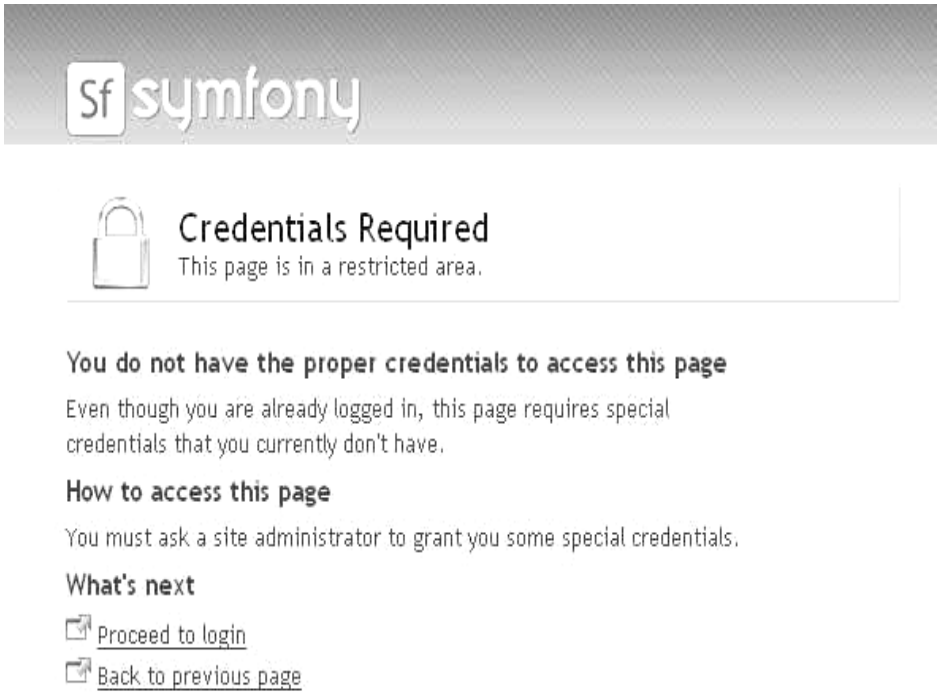


Figura 2.3: La página por defecto de la acción "secure".

- Otorgando acceso: Para obtener acceso a áreas restringidas, los usuarios necesitan estar autenticados y/o poseer ciertas credenciales.
- Métodos de Validación y Manejo de Errores: La validación de los datos de la acción –normalmente los parámetros de la petición– es una tarea repetitiva y tediosa. Symfony incluye un sistema de validación, utilizando métodos de la clase acción.
- Filtros: El mecanismo de seguridad puede ser entendido como un filtro, por el que debe pasar cada petición antes de ejecutar la acción. Según las comprobaciones realizadas en el filtro, se puede modificar el procesamiento de la petición. Symfony extiende esta idea a clases de filtros. Se puede especificar cualquier número de clases de filtros a ser ejecutadas antes de que se procese la respuesta, y además hacerlo de forma sistemática para todas las peticiones.

- Sesiones de usuarios: Symfony maneja automáticamente las sesiones del usuario y es capaz de almacenar datos de forma persistente entre peticiones. Utiliza el mecanismo de manejo de sesiones incluido en PHP y lo mejora para hacerlo mas configurable y más fácil de usar.
- Manejo de sesiones: El manejo de sesiones de Symfony se encarga de gestionar automáticamente el almacenamiento de los IDs de sesión tanto en el cliente como en el servidor. Sin embargo, si se necesita modificar este comportamiento por defecto, es posible hacerlo. Se trata de algo que solamente lo necesitan los usuarios más avanzados. En el lado del cliente, las sesiones son manejadas por cookies. El manejo de sesiones de Symfony esta basado en las sesiones de PHP.

Por último en cuanto al tratado de los Logs, única forma de comprender lo sucedido cuando falla la ejecución de una petición, consiste en echar un vistazo a la traza generada por el proceso que se ejecuta. Afortunadamente, tanto PHP como Symfony guardan mucha información de este tipo en archivos de log.

Además de los archivos de log creados por PHP, Symfony también guarda mucha información de sus propios eventos en otros archivos de log. Si se dispone de una aplicación Symfony ejecutándose, se puede observar que la sintaxis de los archivos de log generados es muy sencilla. Cada evento resulta en una nueva línea en el archivo de log de la aplicación. Cada línea incluye la fecha y hora a la que se ha producido, el tipo de evento, el objeto que ha sido procesado y otros detalles relevantes que dependen de cada tipo de evento y/o objeto procesado. Estos archivos de log contienen mucha información, como por ejemplo las consultas SQL enviadas a la base de datos, las plantillas que se han procesado, las llamadas realizadas entre objetos, etc; ver figura 2.4. [5]

```
Nov 15 16:30:25 symfony [info ] {sfAction} call "barActions->executemessages()"
Nov 15 16:30:25 symfony [debug] SELECT bd_message.ID, bd_message.SENDER_ID,
bd_...
Nov 15 16:30:25 symfony [info ] {sfCreole} executeQuery(): SELECT bd_message.ID...
Nov 15 16:30:25 symfony [info ] {sfView} set slot "leftbar" (bar/index)
Nov 15 16:30:25 symfony [info ] {sfView} set slot "messageblock" (bar/mes...
Nov 15 16:30:25 symfony [info ] {sfView} execute view for template "messa...
Nov 15 16:30:25 symfony [info ] {sfView} render "/home/production/miproyecto/...
Nov 15 16:30:25 symfony [info ] {sfView} render to client
```

Figura 2.4: Contenido de un archivo de log de Symfony.

2.13.6 Validaciones

Todos los formularios de entrada de datos al sistema estarán validados con javascript apoyado en expresiones regulares, no solo se validará en la capa de presentación sino que también se tendrá en cuenta a la hora de programar las clases de negocio haciendo uso de las funcionalidades que trae consigo PHP, el cual tiene muchas formas de verificar la integridad de los datos entrados por el usuario.

2.13.7 Control de errores

Se contará con una única página de control de errores, es decir esta página será la única encargada de mostrar los errores que se presenten en el sistema, brindado la más mínima información al usuario, solo mensajes que no comprometan la integridad de la aplicación. Se utilizarán clases definidas para cada excepción las cuales heredaran de la clase Exception que trae consigo PHP y en cuanto a la tabla que almacenará los errores ocurridos en la base de datos seguirá siendo una especialización de las incidencias del sistema y almacenará la traza del error y el mensaje que este emita.

2.13.8 Seguridad en la base de datos

PostgreSQL como gestor de base de datos seleccionado para el nuevo sistema tiene características con respecto a la seguridad que lo hacen un potente gestor. Entre todas estas características conforman un sistema de seguridad que cumple con las principales normativas de seguridad: integridad, disponibilidad y confidencialidad. Para lograr esto PostgreSQL tiene definidas las siguientes políticas en cuanto a seguridad respecta:

1- Los objetos cuando se crean son asignados a un propietario que usualmente es el usuario que lo creo aunque el superusuario tenga acceso a este, para que otros usuario puedan realizar alguna acción sobre el objeto se le deben asignar privilegios.
2- Tiene definidos distintos tipos de permisos abreviándolos con una letra simbólica.
3- Se pueden agregar o quitar permisos a través de comandos especiales y también verlos mediante tablas.
4- Existen tablas del sistema que permiten al gestor realizar funciones de mantenimiento y control de eventos.
5- Posee una eficiente configuración para la gestión de copias de seguridad (backup) utilizando comandos específicos para la reproducción parcial o total de los datos almacenados.
6- Al utilizar funciones para realizar las distintas operaciones garantiza la protección contra ataques como SQL Injection.
7- Al poder agrupar los usuarios en subconjuntos es más fácil la asignación de permisos.
8- Establece máximo de conexiones evitando una posible denegación de servicios y limita las conexiones por direcciones IP permitiendo controlar las conexiones al sistema.

2.13.9 Servicios Web

Como propuesta de Akademos 2.0 en cuanto a servicios Web, se tiene un conjunto de funcionalidades propuestas por cada módulo existente en el sistema y el acceso a estos será anónimo es decir: cualquier usuario podrá, dado la utilidad de los datos almacenados, acceder a

todos los servicios brindados, aunque se registrarán las incidencias que esto genere. Se tendrá un estrecho vínculo con la UDDI del centro donde se describirán las funcionalidades y características de los servicios. La utilización de funciones predefinidas por PHP, el lenguaje a usar, para la creación de XML-WebServices facilitará el trabajo con estos.

2.13.10 Seguridad en los elementos específicos del sistema

La seguridad en los elementos específicos del sistema se puede desglosar en cuatro aspectos fundamentales: usuarios, roles, permisos e incidencias.

El sistema podrá gestionar sus propios usuarios. Los roles se podrán crear en el sistema a través de una interfaz que permitirá asignar un usuario o grupos de usuarios previamente insertados en el sistema o tomar grupos de dominio si el sistema se encuentra en uno. Después de creados estos roles se les asignaran los permisos deseados, que no son más que las acciones que podrán realizar sobre determinado objeto en determinada estructura.

Por otra parte se encuentran las incidencias las cuales se registrarán cada vez que un usuario realice una acción en el sistema y se podrán realizar auditorias a estas acciones que realizan los usuarios a través de reportes que facilitará un monitor de incidencias, el cual tendrá varios criterios de búsqueda y solo el administrador del sistema tendrá acceso a este.

2.13.11 Tablas de la base de datos

Como propuesta de tablas para almacenar los datos que tienen que ver con la seguridad del sistema tenemos cinco tablas que estarán relacionadas entre sí y con las tablas principales de la aplicación. Estas tablas son:

1- Tbl_Usuario: almacena los datos de los usuarios.
2- Tbl_Rol: está encargada de almacenar los roles creados en el sistema, ya sean estos creados por un usuario o derivados de un directorio activo.

3- Tbl_UsuarioRol: tabla que guarda los roles que tiene un usuario.
4- Tbl_Accion: guarda las acciones predefinidas en el sistema.
5- Tbl_TipoObjeto: en esta tabla se almacenan los tipos de objetos previamente definidos en el sistema.
6- Tbl_Permiso: se encarga de los permisos que no son más que las acciones que un rol puede realizar sobre determinado objeto en determinada estructura.
7- Tbl_Incidencia: se encarga de guardar las incidencias del paso de los usuarios por el sistema.
8- Tbl_Error: tabla que tiene como funcionalidad almacenar los errores que genere el sistema.

2.14 Conclusiones

En este capítulo se ha descrito como se encuentra actualmente la seguridad en Akademos, aprendiendo de lo que se pueda reutilizar y criticando lo mal concebido. Además se ha hecho la propuesta para la gestión de la seguridad en el nuevo sistema Akademos 2.0 partiendo del previo estudio de las herramientas y protocolos competentes y más usados a nivel global para el desarrollo de la seguridad.

Capítulo 3: Descripción de la propuesta

3.1 Introducción

En este último capítulo se profundizará la propuesta ofrecida en el capítulo 2. Además se brindará una propuesta de prototipo de interfaz para la asignación de permisos.

3.2 MD5 y PHP

PHP tiene definida una función para el uso de MD5 la cual es:

string md5 (string cadena).

Un ejemplo del uso de esta función es el siguiente fragmento de código:

```
<?php
$cadena = 'akademos';
if (md5 ($cadena) == ‘e7c382047a0be716f47c15f6d09b7e8f)
{
   echo "Ha acertado escogiendo la palabra  akademos";
   exit();
}
?>
```

El uso de MD5 permitirá encriptar también las variables que se pasen por URL, ejemplo:

```
<? php
```

```
$iduser = “5614-89jk-632h”;

$idusuario = md5 ($iduser);

Header (“Location: estudiantes.PHP?id=’$idusuario’”);

?>
```

3.3 Protocolos de seguridad

Al ser SSL y HTTPS los protocolos de seguridad a utilizar en el sistema, se empleará para el desarrollo de los mismos las funcionalidades que trae consigo Apache, específicamente mod_ssl, el cual es un módulo que provee una fuerte criptografía para Apache usando SSL y apoyándose en OpenSSL como toolkit para desarrollar aplicaciones seguras.

3.4 Servicios Web en PHP

PHP tienes varios kit de herramientas(toolkit) para tratar con XML-WebServices entre los que se encuentra NuSOAP, existen otros, pero este es uno de los que están en una fase de desarrollo mucho más avanzada. Sin ir más lejos, PHP a partir de su versión 5.0 comienza a dar soporte para SOAP, pero aún está en fase experimental.

¿Que es NuSOAP?

NuSOAP es un kit de herramientas (ToolKit) para desarrollar Web Services bajo el lenguaje PHP. Está compuesto por una serie de clases que harán mucho más fácil el desarrollo de Web Services. Provee soporte para el desarrollo de clientes (aquellos que consumen los Web

Services) y de servidores (aquellos que los proveen). NuSOAP está basado en SOAP 1.1, WSDL 1.1 y HTTP 1.0/1.1

¿Por qué NuSOAP y no otro?

1. Está en una fase madura de desarrollo.
2. No necesita módulos adicionales.
3. Es muy fácil su instalación y uso.

Un ejemplo de trabajo con los servicios Web utilizando NuSOAP es el siguiente:

```
<? php

require_once (“lib/nusoap.PHP”);

$oSoapClient=newsoapclient(' http://akademos.uci.cu/servicios/servicios.asmx’, ‘wsdl’);

echo $oSoapClient->call ('ObtenerMatriculados', array ());

?>
```

3.5 Apache

Apache posee varias características que hacen posible una excelente gestión de la seguridad, dada las particularidades antes expuestas de este servidor Web, solo queda profundizar en la manera de configurar y adaptar estas funcionalidades al nuevo sistema. La primera propiedad a describir seria de la restringir el acceso por IP, por ejemplo si deseas restringir el acceso a tu Intranet para permitir solamente la red 176.16:

```
Order Deny, Allow
Deny from all
Allow from 176.16.0.0/16
```

o por IP:

Order Deny, Allow
Deny from all
Allow from 127.0.0.1

Las distintas acciones que lleva a cabo para verificar la validez de la aplicación, se pueden agrupar en tres tipos: autentificación, autorización y control de acceso.

La autentificación es el proceso por el cual se verifica la identidad de una persona. De una forma simple, este proceso se puede llevar a cabo mediante un nombre de usuario y una contraseña, pero se pueden llegar a utilizar otros métodos para validar la identidad de una persona, como mediante el uso de certificados, tarjetas etc…

En Apache la autentificación puede estar gestionada por distintos módulos, dependiendo de la forma de implementación. Si decide llevarla a cabo gestionando ficheros con listas de usuarios y contraseñas (encriptadas), deberá utilizar el módulo mod_auth. Sin embargo, si decide llevarla a cabo mediante base de datos, deberá utilizar los módulos mod_auth_dbm.

La autorización es el proceso por el cual se verifica que un usuario con una identidad conocida, tiene acceso al recurso solicitado. Para llevar a cabo esta acción, se suelen utilizar listas de permisos en las cuales se enumeran cada una de las acciones que puede realizar un usuario, o las que no puede hacer. Normalmente, para simplificar la gestión de estos ficheros, los usuarios se suelen unir en grupos proporcionando los permisos al grupo.

En Apache la autorización a recursos es gestionada o bien mediante la directiva <directory> en el fichero principal de configuración, o bien mediante la configuración de la carpeta a través de ficheros .htaccess.

El control de acceso es el proceso por el cual se verifica que la máquina desde la que se ha hecho la petición, tiene acceso al recurso. Los controles de acceso se utilizan para limitar y controlar las máquinas que tienen acceso a un recurso independientemente del usuario que accede, ya que estos controles se llevan a cabo antes de que se realice el proceso de autentificación.

En Apache, el control de acceso se puede llevar a cabo mediante las directivas <directory><files>y <location>, o a través del fichero de configuración .htaccess para controlar una carpeta específica.

En todo caso y para poder llevar a cabo la configuración de las tres características aquí enumeradas (autentificación, autorización y control de acceso), es necesario tener la directiva AllowOverride con el valor AuthConfig, para así permitir el uso de las distintas directivas de autentificación.

Por otra parte tenemos mod_ssl que es un módulo que usa SSL para el envío de datos en la aplicación. Esta es una lista de cosas que se puede hacer con mod_security:

- Filtración simple
- Filtración basada en expresiones regulares
- Validación de codificación de la URL
- Validación de codificación Unicode
- Auditing
- Prevención del ataque NULL Byte
- Limitar la memoria de subida
- Enmascarar la identidad del servidor

3.6 PostgreSQL

El SGBD PostgreSQL posee muchas herramientas para un buen manejo de la seguridad de los datos que almacena, en primer plano tenemos la autenticación, que no es más que el proceso mediante el cual el servidor de la base de datos y el postmaster se aseguran de que el usuario que está solicitando acceso a la base de datos es en realidad quien dice ser. Todos los usuarios que quieren utilizar PostgreSQL se comprueban en la tabla pg_user para asegurarse que están autorizados a hacerlo. Actualmente, la verificación de la identidad del usuario se realiza de distintas formas:

Desde la shell del usuario

Un demonio que se lanza desde la shell del usuario, anota el id original de este antes de realizar un setuid al id del usuario PostgreSQL. El id original se emplea como base para todo tipo de comprobaciones.

Desde la red

Si PostgreSQL se instala como distribuido, el acceso al puerto TCP del postmaster está disponible para todo el mundo. El ABD configura el fichero pg_hba.conf situado en el directorio PG_DATA, especificando el sistema de autentificación a utilizar en base al equipo que realiza la conexión y la base de datos a la que se conecta. Ver pg_hba.conf para obtener una descripción de los sistemas de autentificación disponibles. Por supuesto la autentificación basada en equipos no es perfecta incluso en los sistemas Unix. Es posible, para determinados intrusos, enmascarar el equipo de origen. Estos temas de seguridad están fuera del alcance de PostgreSQL.

Además el SGBD también ofrece la posibilidad para crear usuarios y grupos de usuarios, para el poder realizar esta acción se debe ejecutar el programa createuser. Para añadir un usuario o un

grupo de usuarios a un nuevo grupo uno de los usuarios se debe crear el grupo y añadir al resto a el. En PostgreSQL estos pasos no pueden realizarse actualmente mediante el comando create group. Los grupos se definen añadiendo los valores a la tabla pg_group, y usando el comando grant para asignar privilegios al mismo. Actualmente no hay una forma fácil de crear grupos de usuarios. Hay que añadirlos/actualizarlos uno a uno en la tabla pg_group_table.

Para la asignación y denegación de privilegios a los usuarios y grupos de usuarios se usan los comandos grant y revoke. Usuarios distintos al creador pueden no tener permisos de acceso a menos que el creador se los conceda, una vez que el objeto ha sido creado. Una vez que un usuario tiene privilegios sobre un objeto, tiene posibilidad de ejecutar ese privilegio. No hay necesidad de conceder privilegios al creador de un objeto; el creador obtiene automáticamente todos los privilegios, y puede también eliminar el objeto.

Los posibles privilegios son:

SELECT

Acceso a todas las columnas de una tabla/vista específica.

INSERT

Inserta datos en todas las columnas de una tabla específica.

UPDATE

Actualiza todas las columnas de una tabla específica.

DELETE

Elimina filas de una tabla específica.

RULE

Define las reglas de la tabla/vista.

ALL

Otorga todos los privilegios.

Por último cabe decir que deben realizarse copias de seguridad de las bases de datos regularmente. Dado que PostgreSQL gestiona sus propios ficheros en el sistema, no se recomienda confiar en los sistemas de copia de seguridad del sistema para las copias de respaldo de las bases de datos; no hay garantía de que los ficheros estén en un estado consistente que permita su uso después de la restauración.

PostgreSQL proporciona dos utilidades para realizar las copias de seguridad de su sistema: pg_dump para copias de seguridad de bases de datos individuales y pg_dumpall para realizar copias de seguridad de toda la instalación de una sola vez.

3.7 Symfony

A partir de las características expuestas en el capítulo anterior sobre las potencialidades que ofrece Symfony para la gestión de la seguridad, a continuación describiremos algunas con una mayor profundidad, exponiendo ejemplos de cómo configurar algunas funcionalidades del framework.

Para la restricción de acceso ya se había expuesto que Symfony filtra las credenciales del usuario activo y verifica las acciones que este pueda realizar, para restringir el acceso a una acción se crea y se edita un archivo de configuración YAML llamado ‘security.yml’ en el directorio config/ del módulo. En este archivo, se pueden especificar los requerimientos de seguridad que los usuarios deberán satisfacer para cada acción o para todas (all) las acciones. Un ejemplo de security.yml seria el siguiente:

Ver:
is_secure: off # *Todos los usuarios pueden ejecutar la acción "ver"*
Modificar:
is_secure: on # *La acción "modificar" es sólo para usuarios autenticados*

Borrar:

is_secure: on # *Sólo para usuarios autenticados*

credentials: admin # *Con credencial "admin"*

All:

is_secure: off # *off es el valor por defecto*

Las acciones no incluyen restricciones de seguridad por defecto, así que cuando no existe el archivo security.yml o no se indica ninguna acción en ese archivo, todas las acciones son accesibles por todos los usuarios. Si existe un archivo security.yml, Symfony busca por el nombre de la acción y si existe, verifica que se satisfagan los requerimientos de seguridad. Lo que sucede cuando un usuario trata de acceder una acción restringida depende de sus credenciales:

- Si el usuario está autenticado y tiene las credenciales apropiadas, entonces la acción se ejecuta.
- Si el usuario no está autenticado, es redireccionado a la acción de login.
- Si el usuario está autenticado, pero no posee las credenciales apropiadas, será redirigido a la acción segura por defecto.

Las páginas login y secure son bastante simples, por lo que seguramente será necesario personalizarlas. Se puede configurar que acciones se ejecutan en caso de no disponer de suficientes privilegios en el archivo settings.yml de la aplicación cambiando el valor de las propiedades mostradas en el listado siguiente:

```
All:
    .actions:
        login_module: default
        login_action: login
        secure_module: default
        secure_action: secure
```

Por otra parte, se encuentran los logs que genera el framework, Symfony define 8 niveles diferentes para los mensajes de log: emerg, alert, crit, err, warning, notice, info y debug, que son los mismos niveles que define el paquete PEAR::Log (http://pear.PHP.net/package/Log/). El archivo de configuración logging.yml de cada aplicación permite definir el nivel de los mensajes que se guardan en el archivo de log, como se muestra a continuación:

```
prod:
        enabled: off
        level: err
        rotate: on
        purge: off
dev:
test:
#all:
        # enabled: on
        # level: debug
        # rotate: off
        # period: 7
        # history: 10
        # purge: on
```

Por defecto, en todos los entornos salvo en el de producción, se guardan en los archivos de log todos los mensajes (hasta el nivel menos importante, el nivel debug). En el entorno de producción, no se utilizan por defecto los archivos de log. Además, en este mismo entorno, si se activan los logs asignando el valor on a la opción enabled, solamente se guardan los mensajes más importantes (de crit a emerg). En el archivo logging.yml se puede modificar el nivel de los mensajes guardados para cada entorno de ejecución, de forma que se limite el tipo de mensajes que se guardan en el archivo de log.

Por último,se tienen las sesiones que como está reflejado en el capítulo anterior son manejadas por cookies, la cookie de Symfony se llama Symfony, pero se puede cambiar su nombre editando el archivo de configuración factories.yml.

El manejo de sesiones de Symfony está basado en las sesiones de PHP. Por tanto, si la gestión de la sesión en la parte del cliente se quiere realizar mediante parámetros en la URL en lugar de cookies, se debe modificar el valor de la directiva use_trans_sid en el archivo de configuración PHP.ini.

En el lado del servidor, Symfony guarda por defecto las sesiones de usuario en archivos. Se pueden almacenar en la base de datos cambiando el valor del parámetro class en factories.yml.

Las clases de almacenamiento de sesiones disponibles son sfMySQLSessionStorage, sfPostgreSQLSessionStorage y sfPDOSessionStorage; la última es la preferida. La opción database define la conexión a utilizar; Symfony luego utiliza databases.yml para determinar los parámetros de la conexión (host, nombre de la base de datos, usuario, y password) para realizar la conexión.
La expiración de la sesión se produce automáticamente después de sf_timeout segundos. El valor de esta constante es 30 minutos por defecto y puede ser modificado para cada entorno en el archivo de configuración settings.yml.[5]

3.8 Descripción de las tablas

Como se había visto anteriormente en el capítulo 2, las tablas propuestas para gestionar la información correspondiente a la seguridad del sistema son: Tbl_Usuario, Tbl_Rol, Tbl_UsuarioRol, Tbl_Accion, Tbl_TipoObjeto, Tbl_Permiso, Tbl_Incidencia y Tbl_Error. Dada la

descripción de las tablas ya expuesta en el capítulo anterior solo queda definir los atributos y relaciones de estas.

Tbl_Usuario: idusuario, contraseña y descripción.

Tbl_Rol: idrol, nombre y descripción.

Tbl_UsuarioRol: idusuario e idrol.

Tbl_Accion: idacción y nombre.

Tbl_TipoObjeto: idtipoobjeto y nombre.

Tbl_Permiso: idpermiso**,** idrol, idacción, idobjeto, idtipoobjeto, idestructura, inhabilitado.

Tbl_Incidencia: idincidencia, idobjeto, idtipoobjeto, idacción, fecha, IP, idusuario, valorprevio, valoractual.

Tbl_Error: idincidencia, mensaje, traza.

Las relaciones entre estas tablas quedan reflejadas en la siguiente figura:

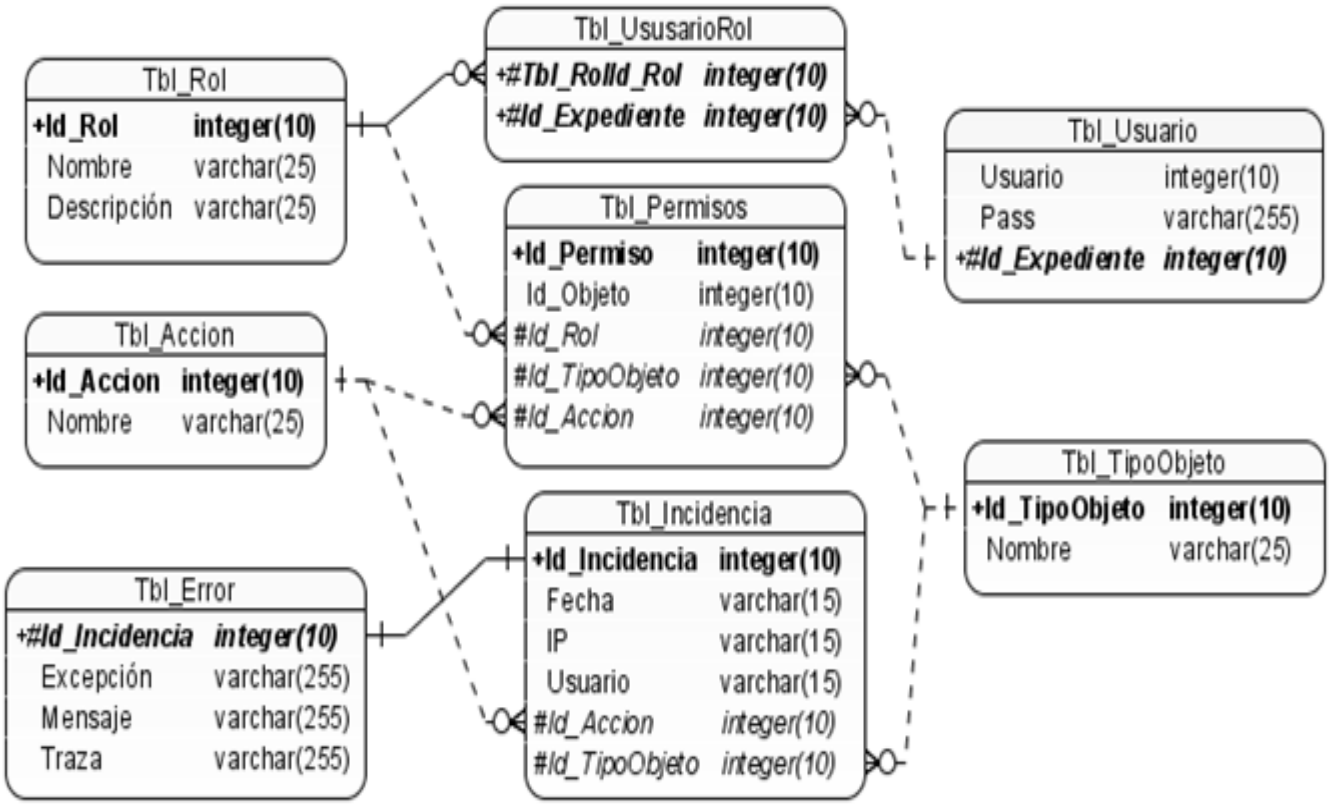


Figura 3.1: Relaciones entre las tablas de seguridad.

3.9 Propuesta de interfaz para la asignación de permisos

Para la asignación de permisos en el sistema, se contará con una interfaz que permitirá asignar a un rol o a un conjunto de roles las acciones que podrán realizar, dichas acciones se encuentran predefinidas en el sistema y solo habrá que escoger la o las acciones que se desea que realicen estos roles y sobre que tipo de objeto además de especificar la estructura donde tendrá acceso. Cuando se cree el permiso se tendrá que especificar si se quiere habilitar el mismo.

Figura 3.2: Interfaz para la asignación de permisos.

3.10 Incidencias

Las incidencias que se registraran cada vez que un usuario realice una acción en el sistema, tendrán además de todas las características especificadas en el epígrafe 3.8,un monitor de incidencias, el cual permitirá hacer reportes para analizar el decursar de los usuarios en el sistema. Solo tendrá acceso a este monitor de incidencias el administrador del sistema, este monitor contará con un buscador que tendrá varios parámetros de búsqueda y se podrán ver también las acciones que estén realizando los usuarios en el momento.

3.11 PHP y el control de errores

PHP 5 tiene un modelo de excepciones similar al de otros lenguajes de programación. Una excepción puede ser lanzada, intentada o capturada en PHP. Un bloque de intento (try) debe incluir por lo menos un bloque de captura (catch). Los bloques de captura múltiples pueden ser usados para capturar diferentes tipos de clases; la ejecución continuará después del último bloque de captura definido. Las excepciones pueden ser lanzadas dentro de bloques de captura.

Cuando es lanzada una excepción, la siguiente línea de código no será ejecutada y PHP intentará encontrar el primer bloque de captura de excepciones. Si una excepción no es capturada, se despliega un error fatal de PHP con un mensaje de que la excepción no fue capturada, a menos que exista un manejador de errores definido como set_exeception_handler (). El lenguaje cuenta con una clase predefinida Exception, que es la encargada por defecto del manejo de excepciones, aunque también permite heredar de esta para personalizar los tipos de errores.

3.12 Conclusiones

En este capítulo se ha profundizado en el estudio de la propuesta brindada en el capítulo anterior, analizando las características y configuraciones especiales de cada herramienta y protocolos propuestos. También se ha brindado un prototipo de interfaz para la asignación de permisos y definido el toolkit a utilizar para desarrollar los servicios Web que brindará el sistema.

Conclusiones generales

A partir de la investigación realizada se arriba a las siguientes conclusiones:

- La seguridad del actual Sistema de Gestión Académica (Akademos) al tener errores tanto de concepto como de implementación no resuelve las necesidades del nuevo sistema.
- Para la gestión de la seguridad se propone utilizar como protocolos de seguridad HTTPS y SSL y como método de encriptación MD5.
- Para el control de la seguridad en los elementos específicos del sistema se propone comprobar permisos al realizarse cualquier acción en el sistema y registrar las incidencias que se generen.
- Se propone que el sistema gestione sus propios usuarios y roles para que pueda lograr una independencia total y llegar a ser un producto.

Recomendaciones

- Utilizar la propuesta brindada para el desarrollo del módulo de seguridad en el nuevo sistema Akademos 2.0.

- Continuar realizando estudios sobre la seguridad en general teniendo en cuenta el constante cambio de protocolos, herramientas y tecnologías usadas para el desarrollo de aplicaciones informáticas y el surgimiento de nuevas amenazas.

Bibliografía citada

[1] FRANK BENAVIDES DALMENDRAY, D. C. R. Estudio de Alternativas para la Migración del Sistema Automatizado para la Gestión Académica ¨Akademos¨ a software libre. Mayo 2007.

[2] OWASP .A Guide to Building Secure Web Applications.Enero 2008, Disponible en: http://www.owasp.org/.

[3] MSDN. Procedimientos de seguridad básicos para aplicaciones Web. Enero 2008, Disponible en: http://msdn.microsoft.com/es-es/library/t4ahd590(VS.80).aspx.

[4] WIKIPEDIA. PostgreSQL. Febrero 2008, Disponible en: http://es.wikipedia.org/wiki/PostgreSQL.

[5] FABIEN POTENCIER, F. Z. Symfony la guía definitiva. Octubre 2007.

[6] W3C.ES. Guía Breve de Servicios Web. Febrero 2008, Disponible en: http://www.w3c.es/Divulgacion/GuiasBreves/ServiciosWeb.

[7] GONZÁLEZ, B. Seguridad en servicios Web. Autenticación y autorización. Interoperabilidad. Febrero 2008, Disponible n: http://www.desarrolloWeb.com/articulos/1640.PHP.

Bibliografía consultada

1. BARNETT, R. C. Preventing Web Attacks with Apache. 2006.

2. BORGHELLO, C. Seguridad Lógica. Enero 2008, Disponible en: http://www.segu-info.com.ar/logica/seguridadlogica.htm.

3. CUENCA, C. L. Seguridad en Apache. Febrero 2008 Disponible en: http://www.desarrolloWeb.com/articulos/2499.PHP.

4. FABIEN POTENCIER, F. Z. Symfony la guía definitiva.Octubre 2007.

5. FRANK BENAVIDES DALMENDRAY, D. C. R. Estudio de Alternativas para la Migración del Sistema Automatizado para la Gestión Académica ¨Akademos¨ a software libre. Mayo 2007.

6. FREITAG, P. 20 ways to Secure your Apache Configuration. Diciembre 2005, Disponible en: http://www.petefreitag.com/item/505.cfm.

7. GARCÍA, A. P. Como configurar la seguridad del servidor Web

Apache en GNU / Linux (Debian). Abril 2004, Disponible en: http://www.adictosaltrabajo.com/tutoriales/tutoriales.PHP?pagina=apache_secure_debian.

8. KEN COAR, R. B. Apache práctico. 2004,

9. MOBILY, T. Hardening Apache. 2004,

10. PARETS, I. G. G. NORMAS DE SEGURIDAD DE LAS TECNOLOGIAS PARA LA GESTION DEL CONOCIMIENTO SERIE DE NORMAS ISO-IEC 27000. 2005, nº

11. RISTIC, I. Apache Security. Mayo 2006, Disponible en: http://www.apachesecurity.net.

12. RODICIO, C. G. Ingeniería en Sistemas de Información con sentido común. Mayo 2008, Disponible en: http://www.cesareox.com/docencia/sxbd/postgresql.html.

13. TRAVIESO, I. Y. M. La Criptografía como elemento de la seguridad informática. Septiembre 2003, Disponible en: http://bvs.sld.cu/revistas/aci/vol11_6_03/aci11603.htm#cargo.

14. UBUNTU, G. PostgreSQL. Enero 2008, Disponible en: http://www.guia-ubuntu.org/index.PHP?title=PostgreSQL

Glosario de términos

- Software Libre: Es la denominación del software que brinda libertad a los usuarios sobre su producto adquirido y por tanto, una vez obtenido, puede ser usado, copiado, estudiado, modificado y redistribuido libremente.
- Cookies: Es un fragmento de información que se almacena en el disco duro del visitante de una página Web a través de su navegador, a petición del servidor de la página.
- Framework: Es una estructura de soporte definida en la cual otro proyecto de software puede ser organizado y desarrollado.
- SGBD: Son un tipo de software muy específico, dedicado a servir de interfaz entre la base de datos, el usuario y las aplicaciones que la utilizan.
- URL: significa Uniform Resource Locator, es decir, localizador uniforme de recurso. Es una secuencia de caracteres, de acuerdo a un formato estándar, que se usa para nombrar recursos, como documentos e imágenes en Internet, por su localización.
- Servidor Web: Es un programa que implementa el protocolo HTTP.
- HTTP: El protocolo de transferencia de hipertexto (HTTP, Hypertext Transfer Protocol) es el protocolo usado en cada transacción de la Web.
- Open Source: Es el término con el que se conoce al software distribuido y desarrollado libremente.
- Firewall: Es un elemento de hardware o software utilizado en una red de computadoras para controlar las comunicaciones, permitiéndolas o prohibiéndolas según las políticas de red que haya definido la organización responsable de la red.
- Logs: Un registro oficial de eventos durante un periodo de tiempo en particular.
- Demonio: Es un tipo especial de proceso informático que se ejecuta en segundo plano en vez de ser controlado directamente por el usuario.

Anexos

Anexo 1: Mensaje de error generado en Akademos.

Anexo 2: Servicios Web prestados actualmente por el sistema.

servicios

The following operations are supported. For a formal definition, please review the Service Description.

- ObtenerGrupos
- ObtenerAsignaturasDadoIdPersona
- ObtenerEgresados
- ObtenerEgresadosDadoCurso
- ObtenerEstudiantesDadoAFG
- VerificarEgresado
- ObtenerFacultades
- ObtenerGruposDadoFacultad
- ObtenerEstudianteDadoIdPersona
- ObtenerMatriculados
- ObtenerCantidadPersonasDadoGrupo

Anexo 3: Descripción de un servicio Web en la uddi de la universidad.

- **ObtenerCantidadPersonasDadoGrupo**

Ver WSDL

Estilo: document

Tipo de operacion: *Request-response*. El servicio recibe un mensaje, y envia la respuesta

Input: ObtenerCantidadPersonasDadoGrupoSoapIn (soap:body, use = literal) *Ver WSDL*

parameters tipo *ObtenerCantidadPersonasDadoGrupo*

- idGrupo - optional; tipo *string*

Output: ObtenerCantidadPersonasDadoGrupoSoapOut (soap:body, use = literal) *Ver WSDL*

parameters tipo *ObtenerCantidadPersonasDadoGrupoResponse*

- ObtenerCantidadPersonasDadoGrupoResult tipo *int*

Anexo 4: Arquitectura de seguridad con tecnologías Microsoft