UNIVERSIDAD DEL VALLE DE GUATEMALA

Facultad de Ingeniería

Evaluación de seguridad de la aplicación Web del Portal de Servicios Académicos en Línea

Eduardo Castellanos Nájera

Guatemala

2009

Evaluación de seguridad de la aplicación Web del Portal de Servicios Académicos en Línea

UNIVERSIDAD DEL VALLE DE GUATEMALA

Facultad de Ingeniería

Evaluación de seguridad de la aplicación Web del Portal de Servicios Académicos en Línea

Trabajo de investigación presentado por Eduardo Castellanos Nájera para optar al grado académico de Licenciado en Ingeniería en Ciencias de la Computación

Guatemala

2009

Vo.Bo.:

(f)_____

(Ing. Ana Miriam Aguilar)

Tribunal Examinador:

(f)_____

(Ing. Sergio Izquierdo)

(f)_____

(Ing. Ana Miriam Aguilar)

(f)_____

(Ing. Luis Furlán)

Fecha de aprobación: Guatemala (Pendiente)

PREFACIO

Este trabajo es el resultado de horas de arduo trabajo y la aplicación de conocimientos recién adquiridos mediante la investigación extensa de las vulnerabilidades que afectan a las aplicaciones web, la naturaleza de las mismas y las técnicas utilizadas para detectarlas y corregirlas. Aunque es imposible que se descubran todas las vulnerabilidades dentro del portal, se espera haber descubierto al menos las más importantes y que las soluciones que se proponen sean útiles al personal responsable de la aplicación.

Se aprovecha este espacio para agradecer a todas las personas que de alguna manera colaboraron en la realización de las pruebas y el desarrollo del trabajo escrito. Especialmente se le agradece a Marvin González y Harry Debroy por su amabilidad al contestar nuestras preguntas y facilitarnos el ambiente de "demo", a Ana Miriam Aguilar por su asesoría y apoyo, al Ing. Luis Furlán quién nos ayudó a definir el trabajo y a todos nuestros familiares y amigos que siempre estuvieron para apoyarnos.

Especialmente se le agradece a María Luisa Garzaro y a mis padres por su apoyo incondicional.

ÍNDICE

LISTA DE CUADROS

LISTA DE GRÁFICOS

RESUMEN

El trabajo descrito en este documento detalla el proceso de la evaluación de seguridad de la aplicación web del Portal de Servicios Académicos en Línea. Una evaluación de seguridad de una aplicación web se refiere a realizar un conjunto de pruebas sobre la aplicación web para identificar la mayor cantidad posible de debilidades o brechas de seguridad en la aplicación. Posteriormente se analizaron los resultados de cada prueba para así proponer soluciones a las vulnerabilidades y mejorar la seguridad de la aplicación. Aunque se examine minuciosamente la aplicación siempre cabe la posibilidad que existan vulnerabilidades muy particulares o sutiles que se puedan pasar por alto.

Para llevar a cabo tal evaluación se utilizó como referencia principal la *Guía de Pruebas de OWASP* desarrollada por el Proyecto Abierto de Seguridad en Aplicaciones Web (OWASP). Esta metodología define diez grupos de actividades en los cuales se enmarcan las 66 actividades que comprenden la evaluación completa. El primer grupo de actividades, "Recopilación de Información" es el único en el que no se realizan pruebas, ya que el objetivo de este grupo de actividades es recopilar toda la información necesaria para las actividades posteriores. Las demás actividades se dedican por completo a realizar pruebas para descubrir vulnerabilidades específicas de autenticación, autorización, manejo de sesiones, lógica del negocio, gestión de configuración, validación de datos, denegación de servicio, servicios web y AJAX.

La evaluación se desarrolló físicamente dentro del campus de la Universidad del Valle de Guatemala. Todas las pruebas se realizaron sobre una copia del portal de servicios académicos en línea en un servidor de aseguramiento de calidad (QA) para evitar que al hacer la evaluación se descubrieran y accidentalmente activara una vulnerabilidad que pudiera dañar, eliminar o corromper el sistema o los datos de producción del mismo.

Al concluir las pruebas se analizaron los resultados, se catalogaron las vulnerabilidades y se propusieron medidas para mitigarlas. Se discute brevemente el impacto de cada vulnerabilidad y se propone una estrategia para mitigarla.

I. INTRODUCCIÓN

Este trabajo de graduación tiene como objetivo principal identificar las vulnerabilidades presentes en la Aplicación Web del Portal de Servicios Académicos en línea y proponer recomendaciones para mitigar las vulnerabilidades encontradas.

La evaluación se limitó al análisis de la aplicación que se encuentra en https://www.uvg.edu.gt/portal/. Se analizó tanto la aplicación como la infraestructura sobre la cual se encuentra implementada. La evaluación se llevó a cabo en los meses de septiembre y octubre de 2009.

La evaluación se desarrolló siguiendo la metodología propuesta por la *Guía de Pruebas OWASP*. Se optó por seguir la metodología de la *Guía de Pruebas OWASP* ya que ha sido desarrollada abiertamente por un amplio grupo de profesionales en la seguridad de aplicaciones y ha sido mejorada y revisada a través de los años, habiéndose publicado la versión más reciente (3.0) en diciembre de 2008.

A lo largo de la evaluación se encontraron numerosas vulnerabilidades que dejan al sistema completamente abierto a modificaciones por parte de un tercero con la habilidad y el conocimiento necesarios por lo que se proponen recomendaciones para ayudar a mitigar las vulnerabilidades encontradas.

II. EVALUACIÓN DE VULNERABILIDADES

A. Vulnerabilidades

Una vulnerabilidad se define como un error o una mala configuración de software o hardware que un individuo malicioso puede utilizar para violar la política de seguridad del sistema. (Manuzuik, Gold, & Gatford, 2007)

El riesgo que una vulnerabilidad presenta para una organización depende de un número de factores:

- Evaluación de riesgo del fabricante
- Número de sistemas afectados dentro de la organización
- Número de sistemas críticos afectados dentro de la organización
- La exposición que representan los sistemas afectados a la organización (Manuzuik, Gold, & Gatford, 2007)

B. ¿Qué es una evaluación de vulnerabilidades?

Una evaluación de vulnerabilidades se podría comparar con una misión militar de reconocimiento. El propósito del reconocimiento es adentrarse en territorio enemigo y establecer las debilidades; las vulnerabilidades del oponente. Después del ejercicio, los comandantes militares deberían tener más información y comprensión del objetivo. De igual manera, las evaluaciones de vulnerabilidades son ejercicios de seguridad que ayudan a los profesionales de seguridad, líderes de negocios y hackers a identificar los riesgos de seguridad dentro de las redes, aplicaciones y sistemas. (Manuzuik, Gold, & Gatford, 2007)

1.Guía de pruebas de OWASP. La guía de pruebas OWASP

es un esfuerzo colaborativo de múltiples profesionales de seguridad de aplicaciones para desarrollar un marco metodológico que permita a cualquier experto en seguridad realizar pruebas siguiendo un proceso claro y objetivo.

El objetivo de la guía es recopilar todas las técnicas de comprobación más relevantes, explicarlas y mantener la guía actualizada. La metodología de pruebas de intrusión de aplicación web OWASP se basa en un enfoque / acercamiento de caja negra. La persona que realiza las pruebas tiene poca, o ninguna, información sobre la aplicación que va a ser comprobada. El modelo de pruebas consta de:

- *Auditor*: La persona que realiza las actividades de comprobación
- *Herramientas y metodología*: El núcleo de este proyecto de guía de pruebas
- *Aplicación*: La caja negra sobre la que realizar las pruebas

Las pruebas se dividen en dos fases:

- *Modo pasivo*: en el modo pasivo, la persona a cargo de la realización de las pruebas intenta comprender la lógica de la aplicación, juega con la aplicación; puede usarse una utilería para la recopilación de información, como un proxy HTTP, para observar todas las peticiones y respuestas HTTP. Al final de esta fase la persona debería comprender cuales son todos los puntos de acceso de la aplicación (cabeceras HTTP, parámetros, cookies).
- *Modo activo*: en esta fase la persona a cargo de la comprobación empieza a realizar las pruebas usando la metodología descrita en los siguientes apartados. (OWASP.org, 2008)

En las siguientes secciones se describirá brevemente las fases en las que se dividen los diferentes grupos de pruebas que especifica la Guía de Pruebas OWASP.

a. Administración de la configuración de la infraestructura. Las pruebas de administración de la configuración tienen como objetivo revelar información sensible o útil para un atacante respecto a la aplicación web y el ambiente en el que se encuentra, mediante el análisis de la infraestructura y topología. Se puede obtener información como código fuente, métodos HTTP permitidos, funcionalidades administrativas y métodos de autenticación. (OWASP.org, 2008)

Estas pruebas comprenden las pruebas del receptor de escucha de la base de datos, gestión de la configuración de la aplicación, búsqueda de archivos antiguos, copias de seguridad y archivos sin referencia y pruebas de métodos HTTP y XST.

b. Sistema de autenticación. La autenticación es la manera en que un usuario le demuestra al sistema que es quién dice ser. Existen tres mecanismos que se utilizan para realizar este proceso:

- *Algo que el usuario sabe*: El ejemplo más simple es una contraseña. En teoría si sabe el "secreto compartido", debe ser el usuario que dice ser.
- *Algo que el usuario tiene*: Esto podría ser una llave, un token, una insignia o una tarjeta inteligente. En teoría si se tiene la llave, debe ser el usuario que dice ser.
- *Algo que el usuario es*: Por ejemplo una característica fisiológica o conductual tal como una huella digital, el patrón de la retina o iris, la voz, una firma, etc. (Cannings, Dwivedi, & Lackey, 2008)

En esta sección se incluyen pruebas para enumeración de usuarios, pruebas de fuerza bruta, comprobación de los sistemas de recuperación de contraseñas y pruebas de CAPTCHA.

c. Administración de sesiones. En el núcleo de toda aplicación web se encuentra el sistema en que la aplicación mantiene los estados, y por lo tanto controla la interacción del usuario. La gestión de sesiones cubre ampliamente todos los controles que se realizan sobre el usuario, desde la autenticación hasta la salida de la aplicación.

HTTP es un protocolo sin estados, lo que significa que los servidores web responden a las peticiones de clientes sin relacionarlas entre sí. Por lo tanto es necesario implementar mecanismos para mantener los estados entre peticiones. Para ello se necesitan soluciones de terceros o bien de implementaciones a medida de mecanismos de gestión de sesiones. La mayoría de entornos de aplicación web, como ASP y PHP, proporcionan a los desarrolladores rutinas integradas para la gestión de sesiones. Por lo general, se emitirá algún tipo de identificación, que será referido como "Identificador de sesión" (en inglés, “Session ID”), o Cookie. (OWASP.org, 2008)

En esta sección se evalúan los atributos de las cookies y se comprueba si no existen variables de sesión expuestas.

d. Autorización. Una vez autenticado, el sistema puede utilizar el identificador del usuario (junto con la información asociada al mismo) para determinar qué acciones tiene permitidas realizar en la computadora, red o aplicación. El proceso de determinar los límites se llama autorización. (Lehtinen, Russel, & Ganimi Sr., 2006)

En esta sección se evalúan rutas transversales y escalamiento de privilegios dentro de la aplicación.

e. Validación de datos. Las rutinas de validación de datos son la primera defensa de una aplicación web. Estas rutinas son las encargadas de

asegurarse que cualquier tipo de entradas inesperadas a la aplicación no la afecten y los datos están en un formato útil. Cuando no se implementan estas rutinas, o no se hacen correctamente, la integridad de la aplicación y su información se puede comprometer. (Scambray, Sima, & Shema, 2006)

La validación de datos es la raíz de las vulnerabilidades más comunes dentro de las aplicaciones web. Los desarrolladores de aplicaciones tienen que tener como premisa que todos los datos externos a la aplicación son potenciales ataques y deben desconfiar totalmente de los mismos.

En esta sección se realizan pruebas para descubrir diferentes tipos de inyección: SQL, comandos del sistema, SSI, ORM y IMAP/SMTP.

f. Denegación de servicios. Un componente importante de la tríada de seguridad es la disponibilidad. ¿De qué nos sirve tener un sistema íntegro y confidencial si no podemos utilizarlo? Es necesario también poder asegurar que la aplicación estará disponible y pueda ser utilizada por los usuarios autorizados.

Existen múltiples maneras de crear un ataque de denegación de servicios. Cada una de estas maneras ataca un componente específico del servidor.

- Procesador: Se aprovechan cálculos matemáticos y operaciones de cifrado/descifrado para consumir todos los recursos del procesador.

- Memoria: Cuando se le envían datos a una aplicación, esta los tiene que almacenar temporalmente en la memoria. La memoria tiene un límite y cuando se envían muchos datos a la aplicación la podemos agotar si no se restringe el tamaño de los datos enviados.

- Conexiones a la base de datos: Muchas aplicaciones cuentan con conjuntos de *threads* compartidos para comunicarse con las bases de datos. Las

peticiones a las bases de datos utilizan estos hilos y cuando se ejecuta una transacción se reserva una tabla o se bloquea la base de datos mientras se actualiza.

Además existen otras maneras en las que los atacantes creativos llegan a consumir todos los recursos de la aplicación hasta provocar que ya no responda.

En esta sección se buscaron vulnerabilidades que podrían causar una denegación de servicio mediante Wildcards SQL, utilización de parámetros ingresados por el usuario como condiciones de un ciclo, desbordamiento de *búfer* y fallos en la liberación de recursos.

g. Comprobación de servicios web. Un servicio web es un componente de software auto contenido que realiza funciones específicas y publica información acerca de sus capacidades a otros componentes a través de una interfaz de red. Los servicios web están basados en estándares como Web Services Definition Language, un formato XML para describir los puntos de conexión que expone el servicio; Universal Description, Discovery, and Integration, que es un conjunto de protocolos y una infraestructura para la descripción y descubrimiento de servicios web. Por último tenemos Simple Object Access Protocol, un protocolo basado en XML utilizado para enviar mensajes y comunicaciones tipo RPC. (Scambray, Sima, & Shema, 2006)

En esta sección se obtuvo información de los servicios web, pruebas estructurales de XML, Comprobación de parámetros REST y pruebas de repetición.

C. Delimitación e impacto del tema

El Portal de Servicios Académicos en Línea se utiliza a diario por los estudiantes y el personal docente para la administración de notas, solicitud de asistencia financiera,

evaluación de docentes, consulta de saldos, asignación de cursos, solicitud de marbetes de parqueo e inscripción.

Este trabajo de graduación comprende la evaluación de seguridad del Portal de Servicios Académicos en Línea de la Universidad del Valle de Guatemala. La evaluación se centró en la aplicación que se encuentra en el URL https://www.uvg.edu.gt/portal/. Es importante aclarar que existen otros factores que podrían incidir en la aplicación, pero son ajenos a ella. Por ejemplo, si la aplicación de http://www.uvg.edu.gt/ tuviera vulnerabilidades, estas podrían afectar a la aplicación evaluada ya que comparten el mismo nombre de dominio (www.uvg.edu.gt).

Los servidores que se analizaron fueron:

- Servidor de "Demo", IP: 192.168.2.12
- Servidor de servicios web. IP: 192.168.2.10
- Servidor web UVG, www.uvg.edu.gt
- Servidor web del portal UVG, portal.uvg.edu.gt

Al llevar a cabo una evaluación de seguridad es altamente recomendado realizarla sobre sistemas de aseguramiento de calidad o copias lo más cercanas posible a los sistemas en producción. Se recomienda evitar evaluar la seguridad sobre sistemas en producción debido a que las evaluaciones tienden a llevar a los límites a las aplicaciones y en ocasiones se pueden corromper datos o afectar la disponibilidad del sistema.

Por lo tanto se requirió que la prueba de seguridad se llevara a cabo sobre el servidor de "Demo" utilizado internamente por el departamento de desarrollo de la aplicación. Al realizar la evaluación sobre el servidor de "Demo", se encontraron limitantes respecto a las funcionalidades a evaluar.

Una limitante es que no fue posible evaluar en su totalidad el sistema de autenticación ya que el módulo necesario para realizar la autenticación con el servidor

LDAP no estaba funcionando correctamente y tuvo que ser desactivado. Esta limitante afecto pruebas como la enumeración de usuarios (la cual se realizó sobre el servidor de producción), las pruebas de autenticación y las pruebas de fuerza bruta. Se optó por realizar las pruebas que no presentaban ningún riesgo a la aplicación en el servidor de producción, pero las pruebas que podían afectar la disponibilidad o integridad no se llevaron a cabo. La única prueba que no se realizó debido a esta limitante fue la de fuerza bruta.

La segunda limitación que se encontró es que existían funcionalidades que estaban incompletas o contenían errores ya que se encontraban en desarrollo por lo tanto no se pudieron evaluar y quedaron fuera del alcance de la evaluación. Las funcionalidades que no fueron evaluadas son:

- Evaluación de docentes
- Parqueos
- Configuración del sistema de ingreso de notas

D. Metodología

La metodología utilizada para realizar la evaluación fue la *Guía de Pruebas OWASP V3.0* publicada el 18 de diciembre de 2008. La guía presenta un marco de trabajo para pruebas de penetración, evaluaciones de seguridad en las cuales se busca obtener acceso o control sobre un sistema. A diferencia de una prueba de penetración, este trabajo presenta una evaluación de vulnerabilidades, lo que significa que no se efectuó el paso adicional de la prueba de penetración de explotar vulnerabilidades y obtener acceso o control del sistema.

La Guía de Pruebas OWASP detalla 66 pruebas divididas en 9 secciones. Por la extensión del trabajo y la cantidad de pruebas que se debían realizar, se evaluó la aplicación junto con José René Barnéond. Por lo tanto, en este informe únicamente se presentan los resultados de las siguientes pruebas:

- Pruebas del receptor de escucha de la base de datos

- Pruebas de gestión de configuración de la aplicación
- Archivos antiguos, copias de seguridad y sin referencias
- Métodos HTTP y XST
- Enumeración de usuarios
- Pruebas de fuerza bruta
- Comprobar Sistemas de recordatorio/restablecimiento de contraseñas
- Pruebas de CAPTCHA
- Pruebas para atributos de cookies
- Pruebas para variables de sesión expuestas
- Pruebas de ruta transversal
- Pruebas de escalamiento de privilegios
- Pruebas de cross-site scripting reflejado y basado en DOM
- Inyección SQL
- Inyección ORM
- Inyección SSI
- Inyección IMAP/SMTP
- Inyección de ordenes de sistema
- Denegación de servicio mediante ataques con comodines SQL
- Desbordamientos de búfer
- Entradas de usuario como bucle
- Fallar en la liberación de recursos
- Obtención de información en servicios web
- Pruebas estructurales de XML
- Comprobación de parámetros HTTP GET/REST
- Pruebas de repetición
- Pruebas de AJAX

Para todas las pruebas se siguió el procedimiento especificado por la Guía de Pruebas y se utilizaron las herramientas sugeridas sin cambios significativos.

E. Recopilación de información

1.Spiders, robots, y *crawlers*

a. Descripción. En esta prueba se busca el archivo robots.txt y se analizan sus contenidos si se encuentra. El archivo robots.txt contiene una lista de directorios que los motores de búsqueda como Google deberían ignorar. Lo importante es que un archivo robots.txt nos provee una excelente referencia de la estructura de los directorios y tal vez hasta nos pueda apuntar a fallas de configuración. (Scambray, Sima, & Shema, 2006)

b. Resultados. No se encontró un archivo robots.txt.

c. Recomendaciones. Para volver más eficiente la búsqueda dentro de la página de la Universidad del Valle de Guatemala es necesario crear y configurar apropiadamente un archivo de robots.txt. Robots legítimos como los de los buscadores web puedan indexar apropiadamente el dominio y no correr el riesgo de que información que no debería ser publicada sea publicada por error debido a la falta de configurar este archivo.

2.Reconocimiento mediante motores de búsqueda

a. Descripción. Los motores de búsqueda almacenan copias y la estructura de los sitios completos. Es posible encontrar páginas eliminadas o escondidas simplemente al hacer una búsqueda en Google o algún otro motor de búsqueda.

b. Resultados. Al realizar la búsqueda en Google este arrojó un total de 852 resultados en 0.30 segundos. Entre los resultados no se encontró ninguno de relevancia para el portal, aunque si vale la pena reportar que el buscador arrojó la página http://www.uvg.edu.gt/fuvg/?C=N;O=D. En esta página se encuentran listados los contenidos de la carpeta /fuvg. Se exploró el directorio y se

encontró que dentro de la carpeta "/_vti_pvt/" se encuentra un archivo llamado "service.pwd" el cual contiene la siguiente información:

```
# -FrontPage-
root:V30eUwjzHLCcg
webmaster:/xZBKAECqF7E6
fp_user:iA/6wmXvZi5qk
```

c. Recomendaciones. Se recomienda revisar el contenido de los archivos mencionados para verificar que no revelan información sensible y eliminarlos si fuera necesario.

3.Identificación de puntos de entrada de la aplicación

a. Descripción. La identificación de los puntos de entrada de una aplicación es probablemente la actividad más importante del reconocimiento ya que nos proporciona la superficie sobre la cual se realizarán las pruebas.

Los puntos de entrada relevantes son:

- Variables HTTP: cookies, peticiones al servidor POST y GET, variables HTTP_REFERER, etc.
- Mensajes SOAP: si la aplicación posee o utiliza SOAP pueden servir como punto de entrada.
- RSS y Servicios Atom: algunas aplicaciones web poseen procesadores de RSS y servicios Atom para alimentarse de estos y presentarlos de otra forma.
- Archivos XML: puede que la aplicación procese archivos XML de socios u otras aplicaciones de confianza en el internet.
- Sistema de Correo: en ocasiones las aplicaciones consumen correos de sistemas de correos y deben procesarlos. (Shah S. , 2006)

Antes de empezar las pruebas además, es importante familiarizarse con la aplicación y como ésta interactúa con el usuario/navegador web. (OWASP.org, 2008)

b. Resultados. Se realizó un inventario de los puntos de entrada de la aplicación para poder ser utilizados en las demás pruebas que se realizarán. La lista de puntos de entrada será expuesta en el apéndice del trabajo.

c. Comentarios. No se revelaron archivos anormales o inesperados. Se detectó que la aplicación se construyó principalmente utilizando AJAX.

4.Pruebas de firmas de aplicaciones web

a. Descripción. En esta prueba se busca identificar la firma digital del servidor. Identificar la firma del servidor permite saber el tipo exacto de servidor en el que se realizan las pruebas. De esta manera se puede buscar la versión del servidor y determinar si esta desactualizado o existen vulnerabilidades. (OWASP.org, 2008)

b. Resultados.

Servidor Web:

- Linux Fedora Core 10,
- Apache/2.2.11,
- mod_ssl/2.2.11,
- OpenSSL/0.9.8gm
- DAV/2
- PHP/5.2.8,
- mod_perl/2.0.4,
- Perl/v5.10.0

c. Recomendaciones. Mantener al día las actualizaciones de los diferentes componentes del servidor web para no permitir que vulnerabilidades conocidas y que ya han sido arregladas puedan ser explotadas.

5.Descubrimiento de aplicaciones

a. Descripción. En la actualidad se aprovecha al máximo cada servidor desplegado. Por esto es que es muy común encontrar más de una aplicación en un solo servidor. A veces las aplicaciones trabajan bajo el mismo dominio o comparten una dirección IP. El problema que esto ocasiona es que al compartir los mismos recursos, es posible que al comprometer una aplicación dentro del servidor también se comprometan las demás. Por lo cual es importante identificar las aplicaciones con las que se comparte el servidor.

b. Resultados.

1) uvg.edu.gt (192.168.10.3)

Tabla 1: Descubrimiento de aplicaciones en el servidor uvg.edu.gt

Números IP de la aplicación	200.35.167.66
Nombres de cliente compartiendo la dirección IP	kirika.uvg.edu.gt ns2.uvg.edu.gt www.uvg.edu.gt
Dominios utilizando el servidor de nombre bajo otro nombre	proesur.uvg.edu.gt uvg.edu.gt
Servidores de Nombre utilizados por este dominio	ns.uvg.edu.gt ns2.uvg.edu.gt
Dominios que comparte el servidor de nombres	cs.uvg.edu.gt gt proesur.uvg.edu.gt
Direcciones IP de los servidores de nombre	168.234.68.2 200.35.167.65 168.234.68.6 200.35.167.66 168.234.76.6 200.9.74.2

Continuación de la Tabla 1

Nombres reversos de los servidores de nombre	kirika.uvg.edu.gt mail.uvg.edu.gt
Otros nombres de los servidores de nombres	anarres.uvg.edu.gt kirika.uvg.edu.gt mail.gt mail.uvg.edu.gt ns.gt steve.uvg.edu.gt uvg.edu.gt www.gt www.uvg.edu.gt
Servidores de correo utilizados por este dominio	mail.uvg.edu.gt(primary)
Direcciones IP de los servidores de correo	168.234.76.6
Nombres reversos de los servidores de correo	kirika.uvg.edu.gt mail.uvg.edu.gt
Otros nombres de los servidores de correo	kirika.uvg.edu.gt ns2.uvg.edu.gt www.uvg.edu.gt

2) Sub-dominios

68.uvg.edu.gt
anarres.uvg.edu.gt
app.uvg.edu.gt
biblioteca.uvg.edu.gt
cie.uvg.edu.gt
cs.uvg.edu.gt
dti.uvg.edu.gt
kirika.uvg.edu.gt
kirk.uvg.edu.gt
mail.uvg.edu.gt
ns.uvg.edu.gt
ns2.uvg.edu.gt
proesur.uvg.edu.gt
sakai.uvg.edu.gt
steve.uvg.edu.gt
streaming.uvg.edu.gt
studyabroad.uvg.edu.gt
vcortez.uvg.edu.gt
www.uvg.edu.gt

Ilustración 1: Diagrama de nombres de dominio

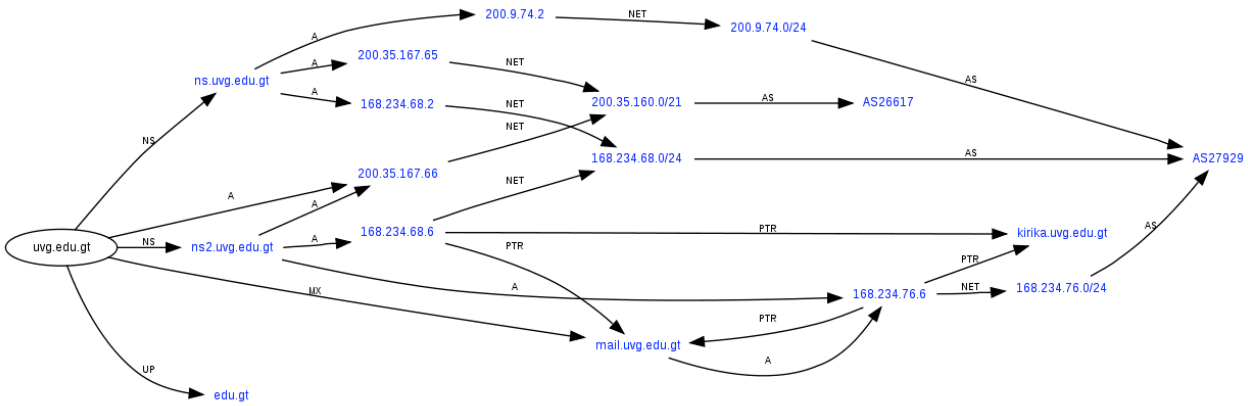


Tabla 2: Puertos abiertos en www.uvg.edu.gt

PORT	State	Service	Version
22/tcp	open	Ssh	OpenSSH 5.1 (protocol 1.99)
25/tcp	open	Smtp	
53/tcp	open	Domain	
80/tcp	open	http	Apache httpd 2.2.11 ((Unix) mod_ssl/2.2.11 OpenSSL/0.9.8g DAV/2 PHP/5.2.8 mod_perl/2.0.4 Perl/v5.10.0)
110/tcp	open	pop3	UW Imap pop3d 2007e.104
143/tcp	open	Imap	UW imapd 2007e.404
389/tcp	open	Ldap	(Anonymous bind OK)
443/tcp	open	ssl/http	Apache httpd 2.2.11 ((Unix) mod_ssl/2.2.11 OpenSSL/0.9.8g DAV/2 PHP/5.2.8 mod_perl/2.0.4 Perl/v5.10.0)
465/tcp	open	ssl/smtp	UW imapd 2007e.404
993/tcp	open	ssl/imap	UW Imap pop3d 2007e.104
995/tcp	open	ssl/pop3	
7777/tcp	open	Unknown	

3) portal.uvg.edu.gt (192.168.10.6)

Tabla 3: Puertos abiertos en portal.uvg.edu.gt

PORT	State	Service	Version
22/tcp	open	Ssh	OpenSSH 5.1 (protocol 2.0)
80/tcp	open	http	Apache httpd 2.2.9 ((Fedora))
995/tcp	open	ssl/pop3	UW Imap pop3d 2007e.104
7777/tcp	open	Unknown	

c. Recomendaciones.

Se recomienda validar que los puertos que se encontraron abiertos sean requeridos y estén permitidos por la política de seguridad. Si no son necesarios los servicios que escuchan en estos puertos, se recomienda desactivarlos.

6.Análisis de códigos de error

a. Descripción.

Al utilizar una aplicación y probar realizar acciones inusuales o ingresar datos inesperados se encuentran errores de diferentes tipos. Muchas veces, estos mensajes de error proporcionan información útil para que el desarrollador de la aplicación pueda ubicar exactamente donde se encuentra el error dentro de la aplicación.

El problema es que esta información también le resulta útil a un atacante debido a que se pueden exponer nombres de variables, tecnologías en uso, nombres de archivos, errores explotables e información general acerca del servidor.

Es por esto que es muy importante analizar los mensajes de error obtenidos y determinar si la información que presentan es inadecuada o podría ayudar a comprometer el sistema.

b. Resultados.

Tabla 4: Análisis de códigos de error

Error	Mensaje de error	Comentarios
Error 404 en portal.uvg.edu.gt	Apache/2.2.9 (Fedora) Server at portal Port 80	Se revela el sistema operativo y el web server en uso.
Error en /mod/common.php?calledFunction=listFiles	]	Sin comentarios

Continuación de la Tabla 5

Error	Mensaje de error	Comentarios
Error en /mod/teacher/registroNotas/rendimiento.php	JSONmsg{"type":"Error","reason":"[rnRendimiento] :: Ha fallado la petición hacia el servidor.", "responsetext":"[mod/teacher/registroNotas/configuracion.php::rendimiento] The name "asdfa" is not permitted in this context. Valid expressions are constants, constant expressions, and (in some contexts) variables. Column names are not permitted."}	Error de la base de datos.
Error 404 en www.uvg.edu.gt	Apache/2.2.11 (Unix) mod_ssl/2.2.11 OpenSSL/0.9.8g DAV/2 PHP/5.2.8 mod_perl/2.0.4 Perl/v5.10.0 Server at www.uvg.edu.gt Port 443	Se revela el sistema operativo y el web server en uso.
Error en /mod/student/consultaNotas/anios.php	JSONmsg{"type":"Error","reason":"Ha ocurrido una falla al intentar conectarse con el servicio web", "responsetext":"[mod/student/consultaNotas/procedures.php::anios] Caught exception: SOAP-ERROR: Parsing WSDL: Couldn\'t load from \'students/StudentsGeneral.asmx?wsdl\'" }	Revela la ubicación de los servicios web.
Error en /mod/student/consultaNotas/actividades.php	JSONmsg{"type":"Error","reason":"Ha ocurrido una falla al intentar conectarse con el servicio web", "responsetext":"[mod/student/consultaNotas/actividades.php::actividades] Caught exception: SOAP-ERROR: Parsing WSDL: Couldn\'t load from \'teachers/RegistroDeNotas.asmx?wsdl\'" }	Revela la ubicación de los servicios web.
Error en /mod/student/consultaNotas/cursos.php	JSONmsg{"type":"Error","reason":"Ha ocurrido una falla al intentar conectarse con el servicio web", "responsetext":"[mod/student/consultaNotas/procedures.php::cursos] Caught exception: SOAP-ERROR: Parsing WSDL: Couldn\'t load from \'students/StudentsGeneral.asmx?wsdl\'" }	Revela la ubicación de los servicios web.

Continuación de la Tabla 6

Error	Mensaje de error	Comentarios
Error en /mod/student/mapaCurricular/stu.mapaCurricular.php	JSONmsg{"type":"Error","reason":"Ha ocurrido una falla al intentar conectarse con el servicio web", "responsetext":"[mod/student/MapaCurricular/MapaCurricular.php::initialize] Caught exception: SOAP-ERROR: Parsing WSDL: Couldn\'t load from \'students/MapaCurricular.asmx?wsdl\'"}	Revela la ubicación de los servicios web.
Error en /mod/teacher/registroNotas/configuracionGeneral.php	JSONmsg{"type":"Error","reason":"[Graba-ConfiguraciónGeneral] :: Ha fallado al intentar almacenar los datos de la configuración general.", "responsetext":"[mod/registroNotas/configuracionGeneral.php::configuracionSet] Procedure or function 'spRnConfiguracionSet' expects parameter '@NotificaAuxiliar', which was not supplied."}	Revela información de procedimientos y variables de la base de datos SQL.

c. Recomendaciones. Personalizar todos los mensajes de error para que no revelen ninguna información de la arquitectura subyacente del sistema o infraestructura del mismo.

F. Pruebas de gestión de configuración de la infraestructura

1.Pruebas del receptor de escucha de la base de datos

a. Descripción. En esta prueba se busca algún escucha de una base de datos que esté disponible. Un receptor de escucha de base de datos no

debería estar disponible ya que podría revelar información acerca del servicio de base de datos utilizado. Adicionalmente, provee al atacante un posible punto de entrada si el receptor de escucha de la base de datos expusiera una vulnerabilidad. El receptor de escucha es específico para bases de datos Oracle. (OWASP.org, 2008)

b. Resultados. No se encontró un receptor de escucha activo de la Base de Datos.

c. Recomendaciones. Es importante tener presente configurar los servidores para que no expongan al público servicios innecesarios como los de las bases de datos.

2.Pruebas de gestión de configuración de la aplicación

a. Descripción. Las aplicaciones web esconden información que no es generalmente considerado durante el desarrollo o la configuración de la aplicación en sí.

Estos datos pueden ser descubiertos en el código fuente, en archivos de registro o en los códigos de error predeterminados de los servidores web. (OWASP.org, 2008)

b. Resultados

- Se utilizan páginas de error personalizadas, sin embargo, se filtran mensajes específicos que brindan información para el atacante
- Están habilitados módulos innecesarios dentro del servidor para el funcionamiento de la aplicación
    - mod_perl/2.0.4,

c. Recomendaciones

- Proporcionar mensajes de error genéricos que no revelen información acerca de la aplicación.
- Desactivar los módulos que no se utilicen.

3.Archivos antiguos, copias de seguridad y sin referencias

a. Descripción. Los archivos redundantes, legibles y descargables de un servidor web, como por ejemplo archivos antiguos, copias de seguridad y archivos renombrados, son una fuente importante de información. Es necesario verificar la existencia de estos archivos porque podrían contener partes importantes del código fuente, rutas de instalación así como contraseñas tanto de las aplicaciones como de las bases de datos. (OWASP.org, 2008)

b. Resultados. Se encontraron los siguientes archivos sin referencia y de respaldo.

- /mod/teacher/actualizarDatos/module.conf.bak
- /img/prog/Thumbs.db
- /img/default/form/date-trigger.psd
- /img/default/form/trigger.psd
- /img/default/form/search-trigger.psd
- /img/default/form/clear-trigger.psd
- /img/default/window/left-corners.psd
- /img/default/window/top-bottom.psd
- /img/default/window/left-right.psd
- /img/default/window/right-corners.psd

c. Recomendaciones. Los archivos psd y db contienen metadata

que puede ser extraída por un atacante. Eliminar los archivos del servidor de producción si no existe una razón para tenerlos en el servidor ya que proporcionan información a un posible atacante.

4.Métodos HTTP y XST

a. Descripción. Usualmente se utilizan los Métodos HTTP POST y

GET, pero existen ocasiones en las cuales los servidores web tienen habilitados otros métodos como PUT, DELETE, COPY, TRACE o TRACK que hacen a un sistema vulnerable.

Se evalúa la aplicación para verificar que no se permiten métodos HTTP potencialmente peligrosos y que no es posible realizar un ataque de Cross Site Tracing (XST). (OWASP.org, 2008)

b. Resultados

1) **http://www.uvg.edu.gt.** Se encuentra activado el método TRACE.

Se envió la petición siguiente:

```
------------------------------ snip ------------------------------
TRACE /Nessus2042783782.html HTTP/1.1
Connection: Close
Host: www.uvg.edu.gt
Pragma: no-cache
User-Agent: Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.0)
Accept: image/gif, image/x-xbitmap, image/jpeg, image/pjpeg, image/png, */*
Accept-Language: en
Accept-Charset: iso-8859-1,*,utf-8

------------------------------ snip ------------------------------

y se obtuvo esta respuesta:

------------------------------ snip ------------------------------
HTTP/1.0 200 OK
Date: Tue, 08 Sep 2009 15:55:02 GMT
Server: Apache/2.2.11 (Unix) mod_ssl/2.2.11 OpenSSL/0.9.8g DAV/2 PHP/5.2.8
mod_perl/2.0.4 Perl/v5.10.0
Connection: close
Content-Type: message/http
```

```
TRACE /Nessus2042783782.html HTTP/1.1
Connection: Close
Host: www.uvg.edu.gt
Pragma: no-cache
User-Agent: Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.0)
Accept: image/gif, image/x-xbitmap, image/jpeg, image/pjpeg, image/png, */*
Accept-Language: en
Accept-Charset: iso-8859-1,*,utf-8

------------------------------ snip ------------------------------
```

2) **http://portal.uvg.edu.gt.** Se encuentra activado el método TRACE.

Se envió la petición siguiente:

```
------------------------------ snip ------------------------------
TRACE /Nessus1549709331.html HTTP/1.1
Connection: Close
Host: portal.uvg.edu.gt
Pragma: no-cache
User-Agent: Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.0)
Accept: image/gif, image/x-xbitmap, image/jpeg, image/pjpeg, image/png, */*
Accept-Language: en
Accept-Charset: iso-8859-1,*,utf-8

------------------------------ snip ------------------------------

y se obtuvo esta respuesta:

------------------------------ snip ------------------------------
HTTP/1.1 200 OK
Date: Tue, 08 Sep 2009 15:57:36 GMT
Server: Apache/2.2.9 (Fedora)
Connection: close
Transfer-Encoding: chunked
Content-Type: message/http


TRACE /Nessus1549709331.html HTTP/1.1
Connection: Close
Host: portal.uvg.edu.gt
Pragma: no-cache
User-Agent: Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.0)
Accept: image/gif, image/x-xbitmap, image/jpeg, image/pjpeg, image/png, */*
Accept-Language: en
Accept-Charset: iso-8859-1,*,utf-8

------------------------------ snip ------------------------------
```

c. **Recomendaciones.** Desactivar los métodos TRACE en ambos servidores.

G. Pruebas de autenticación

1.Enumeración de usuarios

a. Descripción. Con esta prueba se verificará si es posible recopilar un conjunto válido de usuarios, interactuando con el mecanismo de autenticación de la aplicación. Esta prueba resultará útil para las pruebas mediante fuerza bruta, en las cuales se verifica si dado un usuario válido, es posible obtener la contraseña correspondiente. (OWASP.org, 2008)

b. Resultados. En el portal de "Demo" se encontró que se obtenían mensajes de error diferentes cuando se intentaba iniciar sesión con un usuario válido o un usuario inexistente. Sin embargo, en el servidor de producción no se observó este comportamiento.

c. Recomendaciones. Se recomienda mantener el comportamiento observado en el servidor de producción; mostrar mensajes de error genéricos que no revelan si un usuario existe o no dentro del sistema.

2.Pruebas de fuerza bruta

a. Descripción. En una prueba de fuerza bruta se intenta encontrar las credenciales para acceder al sistema utilizando todas las combinaciones posibles. Se utilizan los usuarios del sistema coleccionados durante la fase inicial de recopilación de información y los obtenidos mediante la enumeración de usuarios.

Las pruebas de fuerza bruta no son fáciles de llevar a cabo, debido al tiempo requerido y el posible bloqueo de la persona que esté realizando las pruebas. (OWASP.org, 2008)

b. Resultados. Esta prueba no se llevó a cabo ya que no se encontraba activado el módulo que procesa la autenticación en el portal de "Demo". Tampoco se intentó realizar sobre el portal de producción ya que se consideró que es muy probable que se produzca una condición de denegación de servicio al realizar esta prueba. Sin embargo, se observó que no existen mecanismos que prevengan un ataque de fuerza bruta, por lo que aun así se considera que la aplicación es vulnerable a un ataque de este tipo.

c. Recomendaciones. Se recomienda implementar un sistema que frustre los ataques de fuerza bruta. Estos ataques se podrían evitar de la siguiente manera:

- Implementar un CAPTCHA en el formulario de autenticación para evitar la automatización de intentos de inicio de sesión.
- Implementar una política de bloqueo de IP cuando se sobrepase un número definido de intentos fallidos.
- Implementar una política de bloqueo temporal de cuentas por un período de varios minutos u horas después de un número definido de intentos fallidos.

3.Comprobar sistemas de recordatorio de contraseñas vulnerables

a. Descripción. Se comprueba como la aplicación gestiona el proceso de "contraseña olvidada" y se revisa si existe un mecanismo que permita guardar la contraseña en el navegador ("recordar contraseña"). (OWASP.org, 2008)

b. Resultados. No se encontró un sistema de recordatorio/reset de contraseñas.

c. Recomendaciones. Es recomendable no implementar un mecanismo de este tipo. Estos mecanismos, por diseño vuelven vulnerables a las aplicaciones web ya que en la mayoría de los casos es posible recopilar suficiente información para recuperar o restablecer una contraseña.

4.Pruebas de CAPTCHA

a. Descripción. Un CAPTCHA se define como una prueba, cualquier prueba, que puede ser generada automáticamente y puede ser resuelta por la mayoría de humanos, pero que un programa de computación no puede resolver. (von Ahn, Blum, & Langford, 2004)

Usualmente los CAPTCHA se implementan como imágenes con texto o audio distorsionado. En ocasiones, la distorsión no es suficiente para producir una imagen que sólo puede ser interpretada por humanos, sino que también es posible que un programa de computadora la resuelva.

A menudo las implementaciones de CAPTCHA son vulnerables a diferentes tipos de ataques incluso si el CAPTCHA generado es irrompible. Esta sección ayudará a identificar este tipo de ataques. (OWASP.org, 2008)

b. Resultados. No se observó el uso de CAPTCHA dentro de la aplicación.

c. Recomendaciones. Se recomienda implementar CAPTCHA en formularios de autenticación para evitar la automatización de intentos de inicio de sesión.

Si se implementara un CAPTCHA, se recomendaría que no fuera desarrollado internamente, sino que se utilizara una alternativa como recaptcha.net. La ventaja de utilizar una alternativa externa es que éstas se han probado en ambientes de

producción y se ha demostrado su resistencia a ataques de OCR. Si se implementara una internamente es altamente probable que por inexperiencia o falta de conocimientos específicos del tema se diseñe e implemente un CAPTCHA incorrectamente o débil.

H. Gestión de sesiones

1.Pruebas para atributos de cookies

a. Descripción. Los valores de una cookie se establecen usando el encabezado HTTP Set-Cookie. Son el mecanismo de autorización/manejo de sesión más popular en las aplicaciones web. (Scambray, Sima, & Shema, 2006)

Los atributos que afectan la seguridad de las cookies son: HTTPOnly, Secure, Domain y Path.

El atributo Secure restringe al navegador a enviar las cookies únicamente sobre HTTPS. (Cannings, Dwivedi, & Lackey, 2008)

El atributo *HTTPOnly* se utiliza para prevenir ataques de robo de cookies por medio de cross-site scripting (XSS). Este atributo evita que se tenga acceso a la cookie desde scripts en la aplicación. (Cannings, Dwivedi, & Lackey, 2008)

El atributo *Domain* limita el alcance de los servidores que tienen permitido acceder a la cookie. (Cannings, Dwivedi, & Lackey, 2008)

El atributo *Path* limita aún más el alcance de qué aplicaciones en un servidor tienen acceso a una cookie. (Cannings, Dwivedi, & Lackey, 2008)

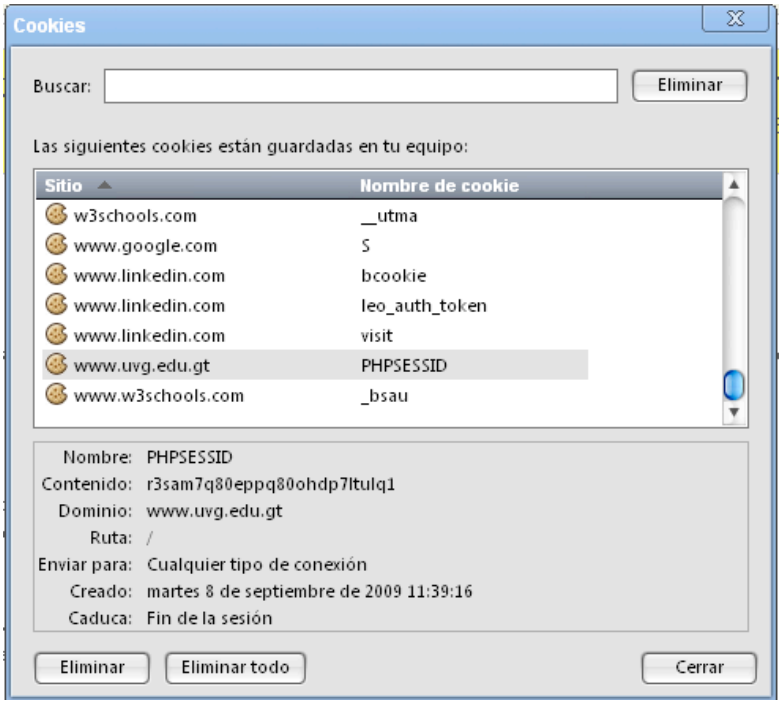
Las cookies son comúnmente un vector de ataque clave para usuarios maliciosos, por lo tanto la aplicación debe siempre tener la debida diligencia para proteger las

cookies. En esta prueba se verifica que los atributos de las cookies sean asignados correctamente. (OWASP.org, 2008)

b. Resultados. La aplicación no expira las cookies al salir de la sesión, simplemente las reestablece:

```
Set-Cookie: PHPSESSID=nfjv8ctod3j3n4r2ffuq3ms3n2; path=/
Set-Cookie: PHPSESSID=r3sam7q80eppq80ohdp7ltulq1; path=/
```

Ilustración 2: Cookie



La forma de crear cookies no es de seguridad óptima debido a los siguientes puntos:

- No se estableció de forma segura utilizando el parámetro HTTPOnly
- No se transmiten únicamente por canales cifrados, hace falta configurar la aplicación para habilitar el parámetro SECURE.

- No se limita la cookie a la aplicación. El parámetro PATH se especifica de tal manera que cualquier aplicación en www.uvg.edu.gt puede acceder a la cookie.
- No estableció una fecha de vencimiento

Independientemente de la forma en la que establece las cookies, la aplicación sí lleva un control interno de las cookies activas e inactivas ya que al realizar la prueba de restablecer la cookie al valor de inicio de sesión cuando ésta se cierra no permitió seguir autenticado.

1) **Cache.** Se determinó que la aplicación no almacena información sensible en la caché del navegador ya que la mayoría de la aplicación utiliza una aplicación desarrollada específica y no utiliza simplemente HTML para desplegar los datos sensibles.

c. **Recomendaciones.** Se recomienda configurar correctamente las cookies para que estas se limiten únicamente a la aplicación que las utilice. Esto se lograría configurando el valor de PATH de la siguiente manera:

path=/portal/

También re recomienda configurar la aplicación para que active las banderas HTTPOnly y SECURE de las cookies.

2.Pruebas para variables de sesión expuestas

a. **Descripción.** Los identificadores de sesión representan información confidencial porque ligan la identidad del usuario con su propia sesión. Es posible probar si el identificador de sesión está expuesto a esta

vulnerabilidad y tratar de crear un ataque de repetición de sesión. (OWASP.org, 2008)

b. Resultados. No se encontraron variables de sesión expuestas.

Todas las sesiones se manejan con la función interna de PHP para manejar las sesiones.

c. Recomendaciones. Sin recomendaciones.

I. Pruebas de autorización

1.Pruebas de ruta transversal

a. Descripción. Las vulnerabilidades de ruta transversal se utilizan para evadir permisos de directorios o archivos. Un ataque clásico es el "dot dot slash", el cuál utiliza la notación de los sistemas de archivos "../" que significa "moverse hacia el directorio padre" para acceder a recursos protegidos. (Scambray, Sima, & Shema, 2006)

b. Resultados. Se encontró una vulnerabilidad de ruta transversal en la función que lista los scripts de javascript que utiliza cada módulo (/mod/common.php).

Se puede utilizar esta función para listar los contenidos de cualquier carpeta del sistema operativo. Al ejecutar la función con los siguientes parámetros:

- calledFunction = listFiles
- dir = /etc
- ext = conf

Se obtuvo la siguiente respuesta:

["/etc/capi.conf","/etc/dhcp6c.conf","/etc/dnsmasq.conf","/etc/esd.conf","/etc/gpm-root.conf","/etc/gssapi_mech.conf","/etc/host.conf","/etc/idmapd.conf","/etc/initlog.conf","/etc/jwhois.conf","/etc/kerneloops.conf","/etc/krb5.conf","/etc/ld.so.conf","/etc/ldap.conf","/etc/lftp.conf","/etc/libuser.conf","/etc/logrotate.conf","/etc/ltrace.conf","/etc/mke2fs.conf","/etc/multipath.conf","/etc/nscd.conf","/etc/nsswitch.conf","/etc/ntp.conf","/etc/pam_smb.conf","/etc/pm-utils-hd-apm-restore.conf","/etc/prelink.conf","/etc/reader.conf","/etc/resolv.conf","/etc/rsyslog.conf","/etc/sensors3.conf","/etc/sestatus.conf","/etc/smartd.conf","/etc/sos.conf","/etc/sysctl.conf","/etc/updatedb.conf","/etc/urlview.conf","/etc/warnquota.conf","/etc/webalizer.conf","/etc/wvdial.conf","/etc/yp.conf","/etc/yum.conf"]

c. Recomendaciones. Aunque no es posible acceder a estos archivos mediante la función invocada, se puede obtener suficiente información acerca de un sistema con el simple hecho de analizar la estructura de directorios y la existencia de archivos.

Se recomienda validar los parámetros de esta función y restringir los tipos de archivo y carpetas que puede listar.

2.Pruebas de escalamiento de privilegios

a. Descripción. Durante esta fase, el evaluador debe verificar que no es posible para un usuario modificar sus privilegios/perfil dentro de la aplicación de manera que permitan ataques de escalada de privilegios. (OWASP.org, 2008)

Usualmente, se refiere a escalada vertical cuando es posible acceder recursos otorgados a cuentas con mayores privilegios, y escalada horizontal cuando es posible acceder recursos otorgados a cuentas similares a la del atacante. (OWASP.org, 2008)

b. Resultados. Se determinó que la aplicación es vulnerable a escalamiento de privilegios debido a que falta una validación apropiada en el lado del servidor para comprobar que un usuario tenga o no permisos de realizar ciertas operaciones sobre la aplicación. El escalamiento de privilegios detectado es horizontal y vertical. Es horizontal porque el sistema permite realizar acciones que solo deberían realizar usuarios con el mismo rol, pero sobre conjuntos de datos a los que no deberían tener acceso. Es vertical porque permite a un usuario con menor nivel de acceso realizar acciones de un usuario con mayor nivel de acceso. Actualmente este chequeo solo se realiza mediante no tener la opción u operación de forma explícita en un usuario sin permisos

Se diseñó una petición para el servidor para modificar la nota del alumno CASTELLANOS NAJERA, EDUARDO en la clase de Inteligencia Artificial.

```
POST http://192.168.2.12:80/demo/mod/teacher/registroNotas/actividadEstudianteSet.php
HTTP/1.1
Host: 192.168.2.12
Proxy-Connection: keep-alive
User-Agent: Mozilla/5.0 (Windows; U; Windows NT 5.1; en-US) AppleWebKit/530.5 (KHTML,
like Gecko) Chrome/2.0.172.43 Safari/530.5
Referer: http://192.168.2.12/demo/
Content-length: 699
Origin: http://192.168.2.12
X-Prototype-Version: 1.6.0_rc1
X-Requested-With: XMLHttpRequest
Content-type: application/x-www-form-urlencoded; charset=UTF-8
Accept: text/javascript, text/html, application/xml, text/xml, */*
Accept-Encoding: gzip,deflate,sdch
Cookie: ys-ext-comp-
1100=o%3Awidth%3Dn%253A720%5Eheight%3Dn%253A432%5Ex%3Dn%253A446%5Ey%3Dn%253A108;
PHPSESSID=qk7kdc2sstojgthhr404dlr231; ys-ext-comp-
1073=o%3Awidth%3Dn%253A550%5Eheight%3Dn%253A374%5Ex%3Dn%253A390%5Ey%3Dn%253A286
Accept-Language: es-419,es;q=0.8
Accept-Charset: ISO-8859-1,utf-8;q=0.7,*;q=0.3

jsonData=%7B%22ID_Actividad%22%3A33027%2C%22ID_Estado%22%3A3%2C%22NombreEstado%22%3A%
22Publicado%22%2C%22affected%22%3A%22%5B%7B%5C%22carnet%5C%22%3A+%5C%2205157%5C%22%2C
+%5C%22NombreEstudiante%5C%22%3A+%5C%22CASTELLANOS+NAJERA%2C+EDUARDO%5C%22%2C+%5C%22O
mitir%5C%22%3A+%5C%220%5C%22%2C+%5C%22Puntos%5C%22%3A+15%2C+%5C%22Porcentaje%5C%22%3A
+100%2C+%5C%22Comentarios%5C%22%3A+%5C%22%5C%22%2C+%5C%22Descripcion%5C%22%3A+%5C%22%
5C%22%2C+%5C%22Activo%5C%22%3A+%5C%221%5C%22%2C+%5C%22EstadoCurso%5C%22%3A+%5C%22ASIG
NADO%5C%22%2C+%5C%22ingFecha%5C%22%3A+%5C%22%5C%22%2C+%5C%22ingUsuario%5C%22%3A+%5C%2
2%5C%22%2C+%5C%22updFecha%5C%22%3A+%5C%22%5C%22%2C+%5C%22updUsuario%5C%22%3A+%5C%22%5
C%22%7D%5D%22%7D&_=
```

La parte importante de esta petición es la variable jsonData que de forma procesada contiene los siguientes datos:

```
{"ID_Actividad":33027,"ID_Estado":3,"NombreEstado":"Publicado","affected":"[{\"carnet
\": \"05157\", \"NombreEstudiante\": \"CASTELLANOS NAJERA, EDUARDO\", \"Omitir\":
\"0\", \"Puntos\": 15, \"Porcentaje\": 100, \"Comentarios\": \"\", \"Descripcion\":
\"\", \"Activo\": \"1\", \"EstadoCurso\": \"ASIGNADO\", \"ingFecha\": \"\",
\"ingUsuario\": \"\", \"updFecha\": \"\", \"updUsuario\": \"\"}]"}
```

Esta petición permite no solo modificar la nota de un usuario en específico, sino también publicarlas de tal forma que el atacante pueda comprobar el éxito de la operación inmediatamente. Al igual que este ataque hay muchos más que pueden ser realizados de manera similar, ya que la aplicación falla en validar apropiadamente las operaciones que un usuario puede o no realizar.

c. Recomendaciones. Se recomienda implementar mecanismos que validen los permisos de los usuarios antes de realizar una acción.

Se recomienda evitar utilizar el paradigma de esconder las opciones en lugar de validarlas para evitar que los usuarios tengan acceso a ellas.

J. Pruebas de validación de datos

1.Pruebas de cross-site scripting (XSS)

a. Descripción. Los ataques tipo cross-site scripting inyectan código malicioso en lugares en los cuales los usuarios lo pueden ver. El código malicioso comúnmente roba cookies, las cuales le permiten al atacante suplantar a la víctima o realizar ataques de ingeniería social que podrían engañar al usuario para obtener su contraseña u otra información. (Scambray, Sima, & Shema, 2006)

1) Reflejado. El Cross Site Scripting Reflejado es otro de los nombres para XSS no persistentes, donde el ataque no es almacenado en la aplicación vulnerable. (OWASP.org, 2008)

2) Basado en DOM. No todos los fallos de XSS requieren al atacante controlar el contenido devuelto por el servidor, pero se puede abusar

bastante de las malas prácticas de programación de JavaScript para obtener los mismos resultados. El resultado es el mismo que un fallo de XSS normal, la única diferencia es el medio usado.

En comparación con otros fallos de Cross-Site Scripting (Reflected y Stored XSS), donde un parámetro no controlado es pasado al servidor, devuelto al usuario y ejecutado en el contexto del navegador del usuario. Un fallo de DOM-Based Cross-Site Scripting controla el flujo del código usando elementos del DOM y del propio código del atacante. (OWASP.org, 2008)

b. Resultados.

Tabla 7: Páginas vulnerables a XSS

XSS	Método	URL	Parámetros
Reflected XSS	POST	/mod/student/asignacion/procedures.php	function
Reflected XSS	POST	/mod/student/consultaNotas/cursos.php	anio
Reflected XSS	POST	/mod/teacher/registroNotas/configuracion.php	nombreMateria, seccion, ciclo, anio, tipo, codmateria
Reflected XSS	POST	/mod/student/consultaNotas/actividades.php	codmateria, seccion, nombreMateria, nombreDocente
Reflected XSS	POST	/mod/teacher/registroNotas/actividadesSet.php	
Reflected XSS	POST	/teacher/registroNotas/actividadEstudianteGet.php	id_actividad, nombreActividad, puntos, descripción, id_estado, tipo

c. Recomendaciones. Se recomienda validar los parámetros

afectados utilizando un enfoque positivo, es decir con un "whitelist" que define que es lo que sí está permitido y además utilizar funciones que codifiquen correctamente los parámetros para que se desplieguen los datos correctamente y no permitan ejecutar código.

Se recomienda también seguir las recomendaciones de OWASP:

http://www.owasp.org/index.php/XSS_%28Cross_Site_Scripting%29_Prevention_Cheat_Sheet

2.Inyección SQL

a. Descripción. Un ataque de Inyección SQL consiste en la inserción o inyección de datos en una consulta SQL desde un cliente de la aplicación. Al explotar una vulnerabilidad de inyección SQL se pueden leer datos sensibles de la base de datos, modificar los datos (insertar/actualizar/borrar), realizar operaciones de administración sobre la base de datos (como reiniciar el DBMS), recuperar el contenido de un archivo del sistema de archivos del DBMS y, en algunos casos, ejecutar órdenes en el sistema operativo. (OWASP.org, 2008)

b. Resultados. Se encontró una instancia de inyección SQL.

POST /mod/student/asignacion/procedures.php

Se encontró la vulnerabilidad en todos los parámetros, excepto en el parámetro function.

c. Recomendaciones. Se recomienda validar los datos correctamente utilizando como referencia el artículo OWASP:

http://www.owasp.org/index.php/SQL_Injection_Prevention_Cheat_Sheet

3.Inyección ORM

a. Descripción. La inyección ORM es un ataque donde se utiliza inyección SQL contra un modelo de objeto de acceso a datos generado por ORM. Desde el punto de vista de quien está comprobando esta vulnerabilidad, este ataque es virtualmente idéntico a un ataque de inyección SQL. Sin embargo, la vulnerabilidad de inyección existe en el código generado por la herramienta ORM. (OWASP.org, 2008)

b. Resultados. No se encontraron instancias de inyección ORM.

c. Recomendaciones. No hay recomendaciones.

4.Inyección SSI

a. Descripción. Los Server-Side Includes son directivas evaluadas por el servidor web a la hora de procesar la página que se presentará al usuario. Representan una forma alternativa a los programas CGI o al código incrustado utilizando lenguajes interpretados del lado del servidor cuando sólo es necesario realizar tareas muy sencillas. Las implementaciones SSI más comunes proporcionan órdenes para incluir archivos externos, establecer y mostrar variables de entorno CGI del servidor web y ejecutar scripts CGI externos u órdenes del sistema operativo. (OWASP.org, 2008)

b. Resultados. No se encontraron vulnerabilidades de SSI.

c. Recomendaciones. No hay recomendaciones.

5.Inyección IMAP/SMTP

a. Descripción. IMAP y SMTP son protocolos utilizados para el envío y recepción de correos. Las aplicaciones web que envían correos o manejan casillas de correo usualmente se comunican con los servidores de correo por medio de estos protocolos. (Scambray, Sima, & Shema, 2006)

Todas las aplicaciones que de alguna manera envían correos deben revisarse para que no existan vulnerabilidades de tipo IMAP/SMTP. Generalmente estas vulnerabilidades afectan a aplicaciones tipo correo web.

El propósito de esta prueba es verificar la capacidad de inyectar instrucciones IMAP/SMTP arbitrarias en los servidores de correo, gracias a que los datos de entrada no son filtrados de forma adecuada.

Algunos ejemplos de ataques utilizando la técnica de inyección IMAP/SMTP son:

- Explotación de vulnerabilidades en el protocolo IMAP/SMTP
- Evasión de restricciones de la aplicación
- Anti-automatización para evasión de procesos
- Fugas de información
- Open Relay/SPAM

(OWASP.org, 2008)

b. Resultados. No se encontraron vulnerabilidades de inyección IMAP/SMTP.

c. Recomendaciones. Sin recomendaciones.

6.Inyección de ordenes de sistema

a. Descripción. La inyección de órdenes de sistema es una técnica que hace uso de una interfaz web para ejecutar instrucciones del sistema en el servidor.

Las vulnerabilidades de inyección de órdenes de sistema se dan cuando las aplicaciones necesitan recurrir a ejecutar programas externos para poder proporcionar alguna funcionalidad y los parámetros que se le envían a los programas no son validados correctamente. (OWASP.org, 2008)

Con la habilidad de ejecutar ordenes del sistema operativo, el usuario puede subir programas maliciosos o incluso obtener contraseñas. (OWASP.org, 2008)

b. Resultados. No se encontraron vulnerabilidades de inyección de órdenes del sistema.

c. Recomendaciones. Sin recomendaciones.

K. Pruebas de denegación de servicio

1.Denegación de servicio mediante ataques con comodines SQL

a. Descripción. Los ataques con comodines SQL fuerzan que una base de datos lleve a cabo peticiones que requieran un uso intensivo del procesador por utilizar varios *wildcards* o comodines. Esta vulnerabilidad se suele dar en funcionalidades de búsqueda de las aplicaciones web. (OWASP.org, 2008)

La explotación satisfactoria de este ataque causaría una denegación de servicio.

b. Resultados. Se probó dentro de la búsqueda de cursos la denegación de servicios por medio de un ataque a la base de datos SQL con las siguientes pruebas:

- bo
    - Tomó un total de 1.52 segundos
- ing
    - Tomó un total de 1.80 segundos
- %
    - Tomó un total de 19.10 segundos
- '%_[^!_%/%a?F%_D)_(F%)_%([)({}%){()}£$&N%_)$*£()$*R"_)][%](%[x])%a][$*"£$-9]_%'
    - Arrojó error: Incorrect syntax near '%'.
- '%64_[^!_%65/%aa?F%64_D)_(F%64)_%36([)({}%33){()}£$&N%55_)$*£()$*R"_)][%55](%66[x])%ba][$*"£$-9]_%54' bypasses modsecurity
    - Arrojó error: Incorrect syntax near '%'.
- _[r/a)_ _(r/b)_ _(r-d)_
    - Tomó un total de 6.07 segundos
- %n[^n]y[^j]l[^k]d[^l]h[^z]t[^k]b[^q]t[^q][^n]!%
    - Tomó un total de 0.25 segundos
- %_[aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa[! -z]@$!_%
    - Tomó un total de 0.34 segundos

Se concluye que la aplicación es vulnerable a ataques de denegación de servicios debido a la cantidad tan grande comparada con otras consultas normales que toman las consultas de '_[r/a)_ _(r/b)_ _(r-d)_' y especialmente '%'.

Página vulnerable: Búsqueda de cursos para asignación.

c. Recomendaciones. Se recomienda validar la entrada y utilizar secuencias de escape cuando sea necesario para evitar la inserción de comodines SQL.

2.Desbordamientos de *búfer*

a. Descripción. Las vulnerabilidades de desbordamiento de búfer se producen cuando se reserva un espacio de memoria de un tamaño fijo y el usuario provee datos que exceden el tamaño del búfer. Al excederse el tamaño del búfer, los datos que el usuario ingresa sobrescriben otros datos en la memoria y en ocasiones es posible ejecutar código remotamente al explotar esta vulnerabilidad. Comúnmente al desbordar un búfer se creará un error en la aplicación por lo que ésta podría cerrarse abruptamente o dejar de responder.

b. Resultados. Se intentó realizar desbordamiento de búfer en los siguientes lugares

- Página para inicio de sesión
    - Usuario
    - Password
- Asignación de cursos
    - Búsqueda de cursos
- Consulta de notas

- En los parámetros pasados por JSON

- Ayuda financiera
  - En los diferentes campos del formulario
- Auxiliaturas y Registro de notas
  - En los campos de notas y comentarios
  - En el programa y descripción del curso

No se encontró ninguna vulnerabilidad de desbordamiento de búfer.

c. Recomendaciones. Sin recomendaciones.

3.Entradas de usuario como parámetro de un ciclo

a. Descripción. Si los usuarios pueden proveer, directa o indirectamente, un valor que especifique el número de objetos a ser creados en el servidor de aplicaciones, y si el servidor no impone un límite superior en dicho valor, es posible causar que el entorno se quede sin memoria disponible.

El servidor empezará a asignar el número de objetos especificados por el usuario, pero ya que este número podría ser extremadamente grande podría llegar a agotar los recursos del servidor y efectivamente crear una condición de denegación de servicios. (OWASP.org, 2008)

b. Resultados. Haciendo uso de la prueba anterior (prueba para ingresar una gran cantidad de actividades en el programa de un curso) se

verificó que la aplicación no chequea por una cantidad de actividades lógicas antes de ingresarlas o desplegarlas por lo que el ataque anterior demuestra esta vulnerabilidad para la página.

c. Recomendaciones. Se recomienda implementar chequeos lógicos dentro de la aplicación para evitar estas vulnerabilidades.

4.Fallar en la liberación de recursos

a. Descripción. Comúnmente las aplicaciones reservan recursos, los utilizan y luego los liberan, sin embargo, existen ocasiones en las cuales, por diferentes razones, una aplicación no libera los recursos.

Las razones por las cuales una aplicación podría no liberar los recursos son principalmente dos: el desarrollador de la aplicación no libera los recursos explícitamente u ocurre una excepción o un error antes que el recurso pueda ser liberado.

b. Resultados. Se intentó con la misma petición de almacenar un comentario de 3 Megabytes introducir un error y enviar la petición varias veces y comprobar el desempeño de la aplicación, sin embargo luego de aproximadamente 30 veces de reenviar la petición no se observó degradación en la velocidad de respuesta de la aplicación.

c. Recomendaciones. Se recomienda siempre validar que los recursos se liberen aun cuando se produce un error.

L. Pruebas de servicios web

1.Obtención de información en servicios web

a. Descripción. Los servicios web cuentan con documentos públicos que los describen conocidos como WSDL. Un WSDL es un documento XML que contiene toda la información de un servicio web: todos los métodos y parámetros que expone. (OWASP.org, 2008)

La información recolectada por esta prueba es utilizada para analizar los servicios en las pruebas subsecuentes.

b. Resultados. Se interceptó el tráfico de salida del servidor web y mediante esto se determinó la dirección del servidor y la dirección dentro de él de los servicios web:

Servidor: 192.168.2.10

Servicios Web:

- http://192.168.2.10/aorozco/trunk/teachers/RegistroDeNotas.asmx?wsdl
- http://192.168.2.10/aorozco/trunk/teachers/TeachersGeneral.asmx?wsdl
- http://192.168.2.10/aorozco/trunk/students/StudentsGeneral.asmx?wsdl
- http://192.168.2.10/aorozco/trunk/students/AyudaFinanciera.asmx?wsdl
- http://192.168.2.10/aorozco/trunk/students/CuentaCorriente.asmx?wsdl
- http://192.168.2.10/aorozco/trunk/students/Inscripcion.asmx?wsdl
- http://192.168.2.10/aorozco/trunk/General.asmx?wsdl
- http://192.168.2.10/aorozco/trunk/students/EvaluacionDocente.asmx?wsdl
- http://192.168.2.10/aorozco/trunk/Seguridad.asmx?wsdl
- http://192.168.2.10/aorozco/trunk/students/Parqueos.asmx?wsdl

- http://192.168.2.10/aorozco/trunk/students/MapaCurricular.asmx?wsdl

Se debe notar que en la arquitectura del portal los servicios web se usan de forma interna por lo que no deberían estar publicados en internet. Se realizó una búsqueda de servicios web en la página de la universidad y no se descubrió ninguno.

c. Recomendaciones. Se deben validar y restringir los servicios web para que solo sean accesibles a las aplicaciones para las cuales están destinados ya que es posible utilizar un cliente genérico para interactuar con los servicios web y realizar prácticamente cualquier función (cambiar notas, asignar cursos, asignar auxiliares, etc.).

2.Pruebas estructurales de XML

a. Descripción. Un XML es un archivo estructurado, y para ser interpretado correctamente necesita estar bien formado. En algunas ocasiones, los analizadores de XML consumen muchos recursos cuando se encuentran XML mal formados o demasiado grandes. Por esto es necesario que el servidor este configurado para limitar los recursos asignados al análisis de XML. (OWASP.org, 2008)

b. Resultados. De las pruebas realizadas se obtuvieron los siguientes resultados:

- XML Mal formado
    - No es vulnerable, se probó a enviar una petición con XML mal formado al servicio web y éste responde con un error “HTTP/1.1 415 Unsupported Media Type”.
- XML con contenido exageradamente grande o inesperado

- Es vulnerable, se probó enviar un XML con cadenas de 1MB cada una y debajo de 5MB simplemente las ignora por lo que un ataque distribuido con peticiones de 4MB podría acabar con los recursos de memoria y/o procesador.

- Archivos adjuntos de tipo binario
  - No es vulnerable al envío de archivos adjuntos pues al enviar un archivo adjunto reporta un error "HTTP/1.1 415 Unsupported Media Type".
- Objetos BLOB de tamaño inesperado
  - La aplicación parece ignorar el adjunto y ser vulnerable únicamente de la misma forma que el ataque de XML con contenido exageradamente grande o inesperado.

c. Recomendaciones. Se recomienda configurar los servicios web para que ignoren los mensajes exageradamente grandes.

3.Comprobación de parámetros HTTP GET/REST

a. Descripción. Los servicios que no utilizan tecnologías SOAP para realizar consultas recurren a REST. REST (Representational State Transfer) son servicios web invocados con parámetros enviados por medio de consultas GET HTTP. Por lo tanto, los ataques que se pueden realizar son muy parecidos a los mencionados anteriormente en la guía. (OWASP.org, 2008)

b. Resultados. No aplica debido a que la aplicación utiliza métodos SOAP.

c. Recomendaciones. No hay recomendaciones.

4.Pruebas de repetición

a. Descripción. Un ataque de repetición es de tipo "man-in-the-middle" en el cual un mensaje es interceptado y sustituido por un atacante para suplantar el original.

En servicios web, como en cualquier otro tipo de tráfico HTTP, es posible capturar el tráfico enviado a un servicio web y reenviar el mensaje modificado al servicio web. Un atacante puede intentar reenviar el mensaje original o cambiar el mensaje para comprometer el servidor. (OWASP.org, 2008)

b. Resultados. Debido a que no existe ningún identificador de transacción ni de sesión es posible reenviar cualquier paquete capturado de SOAP que está siendo enviado al servidor y no solo reenviarlo sino modificarlo y luego enviarlo de nuevo.

Además se observó que los servicios web se invocan y ejecutan mediante canales sin cifrar, lo que permite que fácilmente se lleve a cabo un ataque "man-in-the-middle", se intercepten los paquetes y se envíe información arbitraria a los servicios web.

c. Recomendaciones. Se recomienda que los mensajes de los servicios web se transmitan sobre canales cifrados (SSL/TLS) y que además se utilicen identificadores de transacción para evitar ataques de repetición.

M. Pruebas AJAX

1. Pruebas de AJAX

a. Descripción. Las pruebas de AJAX involucran realizar las pruebas descritas anteriormente pero con los componentes AJAX de la aplicación.

b. Resultados. Todas las pruebas realizadas fueron pruebas AJAX ya que la aplicación se desarrolló apoyándose fuertemente sobre ésta tecnología.

Se encontró que al hacer peticiones AJAX, se almacenan las respuestas de las peticiones temporalmente en un elemento DIV. Esto hace más fácil los ataques de cross-site scripting ya que al almacenar la información en un elemento DIV, el navegador la interpreta y si encuentra código lo ejecuta.

c. Recomendaciones. Se recomienda almacenar las respuestas de las peticiones en variables en lugar de elementos HTML.

III. RESULTADOS

En la siguiente tabla se encuentra un resumen de los resultados de cada una de las pruebas efectuadas. A cada resultado se le asignó un valor de riesgo basado en cuán probable es que se explote la vulnerabilidad y que impacto tendría en el sistema.

Tabla 8: Resúmen de resultados

Categoría	Nombre	Conclusión	Comentarios/Solución	Riesgo
Pruebas de la gestión de configuración	Prueba del Listener de la Base de Datos	N/D	No se encontraron vulnerabilidades.	Ninguno
	Prueba de la gestión de la configuración de la aplicación	N/D	No se encontraron vulnerabilidades.	Ninguno
	Ficheros antiguos, de backup o no referenciados	Se encontraron archivos sin referencia y de respaldo.	No representan una amenaza muy grande, pero deberían ser eliminados.	Bajo
	Prueba de métodos de HTTP y XST	Se encontró activado el método TRACE.	Se recomienda desactivar el método TRACE en la configuración de los servidores.	Bajo
Pruebas de autentificación	Pruebas para la enumeración de usuarios	N/D	No se encontraron vulnerabilidades.	Ninguno
	Pruebas de Fuerza Bruta	No se encontraron mecanismos para prevenir este tipo de ataques.	Se recomienda implementar mecanismos para prevenir ataques de fuerza bruta.	Alto
	Pruebas para contraseñas recordadas vulnerables y su reinicio	N/D	No se encontraron vulnerabilidades.	Ninguno
	Pruebas de CAPTCHA	N/D	No se encontraron vulnerabilidades.	Ninguno
Gestión de sesión	Pruebas de los atributos de las Cookies	Ningún atributo de las cookies se encontró configurado correctamente.	Limitar las cookies a la aplicación utilizando el parámetro PATH y activar los parámetros HTTPOnly y SECURE en la configuración.	Alto
	Pruebas de las variables de sesión expuestas	N/D	No se encontraron vulnerabilidades.	Ninguno
Pruebas de autorización	Pruebas de ruta transversal	Se pudo utilizar una función para listar los contenidos de los directorios.	Se recomienda realizar una validación de los parámetros que se le suministran a la función afectada.	Medio
	Pruebas de escalada de privilegios	Se logró escalar privilegios.	Se recomienda implementar funciones o mecanismos que validen que el usuario tenga los permisos necesarios para realizar acciones.	Alto

Continuación de la Tabla 9

Categoría	Nombre	Conclusión	Comentarios/Solución	Riesgo
Pruebas de validación de datos	Prueba de Cross site scripting reflejado	Se encontraron múltiples instancias.	Se recomienda validar los datos correctamente siguiendo las recomendaciones de OWASP para evitar Cross-site scripting..	Alto
	Prueba de Cross site scripting basado en DOM	N/D	No se encontraron vulnerabilidades.	Ninguno
	Inyección SQL	Se encontró una instancia.	Se recomienda validar los datos correctamente siguiendo las recomendaciones de OWASP para evitar SQL injection.	Alto
	Inyección ORM	N/D	No se encontraron vulnerabilidades.	Ninguno
	Inyección SSI	N/D	No se encontraron vulnerabilidades.	Ninguno
	Inyección IMAP/SMTP	N/D	No se encontraron vulnerabilidades.	Ninguno
	Inyección de ordenes de sistema	N/D	No se encontraron vulnerabilidades.	Ninguno
Pruebas de denegación de servicios	Pruebas de ataques SQL con caracteres especiales	Se encontró una instancia.	Se recomienda validar correctamente las entradas.	Medio
	Prueba de desbordamiento de búfer por DoS	N/D	No se encontraron vulnerabilidades.	Bajo
	Entradas de usuario como bucle	Se encontró una instancia.	Se recomienda validar las entradas y limitar los valores proveídos por el usuario.	Alto
	Fallos en la liberación de recursos	Se encontró una instancia.	Se recomienda validar que siempre se liberen los recursos.	Bajo
Pruebas de servicios web	Recopilación de Información de servicios Web	Se encontró que los servicios web se comunican en texto claro.	Se recomienda cifrar el canal de comunicación con los servicios web mediante SSL/TLS.	Alto
	Pruebas estructurales de XML	Se encontraron multiples instancias.	Se recomienda validar las entradas para limitar el tamaño de los mensajes enviados a los servicios web.	Bajo
	Comprobación de parámetros HTTP GET/REST	N/D	No se encontraron vulnerabilidades.	Bajo
	Pruebas de repetición	Se determinó que no existe protección en los servicios web.	Se recomienda asegurar los servicios web mediante mecanismos de autenticación, autorización y prevención de repetición.	Alto
Pruebas de AJAX	Pruebas de AJAX	Múltiples vulnerabilidades	Todas las pruebas anteriores se realizaron sobre AJAX ya que la aplicación fue desarrollada con esta tecnología.	Alto

IV. CONCLUSIONES Y RECOMENDACIONES

- No se encontraron indicios de un ciclo de desarrollo de software seguro. Se recomienda que se implemente un proceso de el Ciclo de Desarrollo de Software Seguro al momento de desarrollar la aplicación para que se definan claramente los requerimientos de seguridad y los requerimientos funcionales.

- Se encontraron vulnerabilidades en la plataforma sobre la cual se ejecuta la aplicación web. Se recomienda que se implemente un mecanismo de control de actualizaciones de la plataforma sobre la cual se ejecuta la aplicación para evitar las ventanas de vulnerabilidad ocasionadas por la falta de parches.

- Se encontraron vulnerabilidades relacionadas con la falta de validación de los datos de entrada. Se recomienda que se implemente un mecanismo central de validación de datos y que se aplique consistentemente a todas las entradas de la aplicación.

V. BIBLIOGRAFÍA

Cannings, R., Dwivedi, H., & Lackey, Z. (2008). *Hacking Exposed Web 2.0: Web 2.0 Security Secrets and Solutions.* New York: McGraw-Hill.

Lehtinen, R., Russel, D., & Ganimi Sr., G. (2006). *Computer Security Basics, Second Edition.* Sebastopol, CA: O'Reilly Media, Inc.

Manuzuik, S., Gold, A., & Gatford, C. (2007). *Network Security Assessment: from Vulnerability to Patch.* Rockland, MA: Syngress Publishing Inc.

OWASP.org. (2008). The OWASP Testing Guide.

Scambray, J., Sima, C., & Shema, M. (2006). *Hacking Exposed Web Applications, Second Edition.* San Francisco, CA: McGraw-Hill.

von Ahn, L., Blum, M., & Langford, J. (2004). Telling Humans and Computers Apart Automatically. *Communications of the ACM , 47* (2), 57-60.

Wells, D. (1996, 04 22). *Authentication*. Retrieved 10 06, 2009, from objs.com: http://www.objs.com/survey/authent.htm

VI. APÉNDICE

A. Puntos de entrada

Tabla 10: Listado de códigos PHP como puntos de entrada

Listado de código PHP en el portal
./mod/font/timesi.php
./mod/font/timesbi.php
./mod/font/zapfdingbats.php
./mod/font/helveticai.php
./mod/font/helveticabi.php
./mod/font/times.php
./mod/font/timesb.php
./mod/font/helvetica.php
./mod/font/symbol.php
./mod/font/helveticab.php
./mod/font/courier.php
./mod/student/inscripcion/procesar.php
./mod/student/inscripcion/formulario.php
./mod/student/inscripcion/municipios.php
./mod/student/ctaCorriente/saldo.php
./mod/student/ayudaFinanciera/prueba.php
./mod/student/ayudaFinanciera/ayudaFinanciera.php
./mod/student/ayudaFinanciera/datosPersonales.php
./mod/student/consultaNotas/anios.php
./mod/student/consultaNotas/procedures.php
./mod/student/consultaNotas/horarioImprimir.php
./mod/student/consultaNotas/cursos.php
./mod/student/consultaNotas/actividades.php
./mod/student/consultaNotas/progCurso.php
./mod/student/consultaNotas/horarioInitialize.php
./mod/student/parqueos/cupos.php
./mod/student/parqueos/comprobanteInitialize.php
./mod/student/parqueos/formulario.php

./mod/student/parqueos/parqueos.php
./mod/student/parqueos/comprobanteImprimir.php
./mod/student/parqueos/enviaSolicitud.php
./mod/student/mapaCurricular/stu.mapaCurricular.php
./mod/student/asignacion/cupos.php
./mod/student/asignacion/asignaCursos.php
./mod/student/asignacion/secciones.php
./mod/student/asignacion/procedures.php
./mod/student/asignacion/asignacion.php
./mod/student/asignacion/buscar.php
./mod/student/asignacion/boletaImprimir.php
./mod/student/asignacion/boletaInitialize.php
./mod/student/evDocente/procedures.php
./mod/student/evDocente/cursos.php
./mod/student/evDocente/grabar.php
./mod/student/evDocente/formulario.php
./mod/student/student.php
./mod/home/home.php
./mod/teacher/registroNotas/actividadesSet.php
./mod/teacher/registroNotas/plantillaSave.php
./mod/teacher/registroNotas/actaFinalAsistenciaSet.php
./mod/teacher/registroNotas/registroNotas.php
./mod/teacher/registroNotas/auxiliaturas.php
./mod/teacher/registroNotas/actividadesSinNota.php
./mod/teacher/registroNotas/actividadEstudianteGet.php
./mod/teacher/registroNotas/configuracionGeneral.php
./mod/teacher/registroNotas/configuracion.php
./mod/teacher/registroNotas/actaFinalAsistencia.php
./mod/teacher/registroNotas/actividadEstudianteSet.php
./mod/teacher/registroNotas/actaFinalImprimir.php
./mod/teacher/registroNotas/actaFinalInitialize.php
./mod/teacher/registroNotas/auxiliaresGet.php
./mod/teacher/registroNotas/nombramientos.php
./mod/teacher/registroNotas/rendimiento.php
./mod/teacher/registroNotas/actividadesGet.php
./mod/teacher/registroNotas/plantillaLoad.php
./mod/teacher/registroNotas/auxiliaresSet.php
./mod/teacher/teacher.php

./mod/general/general.datosPersonales.php
./mod/general.php
./mod/mailSoporte.php
./mod/fpdf.php
./mod/mail.php
./mod/academico.php
./mod/common.php
./index.php

B. Glosario

AJAX: AJAX es un acrónimo para Asynchrnonous Javascript and XML.

ASP: es un marco de trabajo desarrollado por Microsoft para construir sitios, aplicaciones y servicios web.

Caja negra: se refiere a un sistema del cual no se conoce su funcionamiento o características internas.

CGI: es un acrónimo para Common Gateway Interface, un estándar que define como un servidor web genera las páginas web.

COPY: es un método HTTP que permite realizar la copia de archivos.

Crawler: es un programa de computadora que navega sitios web de una manera automatizada.

DELETE: método HTTP utilizado para eliminar archivos.

DOM: Document Object Model por sus siglas en inglés, se refiere a la estructura de objetos del documento. Es una interfaz de programación utilizada para interactuar con los elementos que componen un documento XML o HTML.

GET: método HTTP utilizado para eliminar recuperar archivos.

Hacker: término utilizado para referirse a criminales informáticos, aunque hay algunos que no son criminales, sino que únicamente se apasionan por la seguridad informática.

HTTP: Hyper Text Transport Protocol por sus siglas en inglés, es el protocolo utilizado para transmitir páginas web.

HTTP Headers: encabezados HTTP o cabeceras HTTP, son la parte de una petición HTTP que especifica la operación a realizar, los parámetros, cookies, etc.

HTTP_REFERER: página que hace inició, o desde la cual se inició, la solicitud que se lleva a cabo.

IMAP: Internet Message Access Protocol. Es un protocolo para la comunicación con casillas de correo.

JSON: JavaScript Object Notation. Es un lenguaje utilizado para representar objetos.

LDAP: Lightweight Directory Access Protocol. Es un protocolo utilizado para acceder a directorios de usuarios.

man-in-the-middle: Ataque en el que se interceptan y modifican las comunicaciones entre dos o más partes sin que alguno se dé cuenta.

Metadata: Son datos que describen los datos.

Open Relay: Se refiere a un sistema de correo que está configurado de tal manera que permite que cualquier persona envíe correos a través de él sin autenticación.

ORM: Object Relational Mapping por sus siglas en inglés. Consiste en convertir los objetos de una base de datos a objetos en un lenguaje de programación.

PHP: Lenguaje de programación interpretado diseñado para páginas web.

POST: método HTTP utilizado para enviar información.

Proxy HTTP: Aplicación que funciona de intermediario en conexiones HTTP entre clientes y servidores.

PUT: Método HTTP utilizado para enviar archivos al servidor.

REST: Transferencia de estado representacional.

RPC: Llamada a procedimiento remoto es un protocolo que permite ejecutar código en otra máquina remota.

RSS: Formato XML para compartir o sindicar contenido en la web.

Rutas transversales: Debilidad informática que ocurre cuando no se validan correctamente los permisos de un usuario y se le permite acceder directorios superiores.

Servidor de nombres: Programa o servidor que implementa un servicio de resolución de nombres.

SMTP: Simple Mail Transfer Protocol. Protocolo utilizado para enviar y recibir correos.

SOAP: Simple Object Access Protocol. Protocolo estándar que define como objetos pueden comunicarse mediante XML.

SPAM: Correo masivo no deseado.

SQL: Structured Query Language. Lenguaje utilizado para interactuar con las bases de datos.

SSI: Server Side Includes. Lenguaje interpretado utilizado en paginas web.

SSL: Secure Sockets Layer. Protocolo criptográfico que proporciona comunicaciones seguras en Internet.

Threads: Hilos de procesamiento.

TRACE: Metodo HTTP que responde con la petición HTTP que le fue hecha.

TRACK: Metodo HTTP que responde con la petición HTTP que le fue hecha.

URL: Universal resource locator. Secuencia de caracteres utilizada para nombrar recursos en Internet.

XML: eXtensible Markup Language es un metalenguaje de etiquetas desarrollado por W3C.