UNIVERSIDAD DEL VALLE DE GUATEMALA

Facultad de Ingeniería

Evaluación de seguridad del portal de estudiantes de la Universidad del Valle de Guatemala

José René Barnéond

Guatemala

2009

Evaluación de seguridad del portal de estudiantes de la Universidad del Valle de Guatemala

UNIVERSIDAD DEL VALLE DE GUATEMALA

Facultad de Ingeniería

Evaluación de seguridad del portal de estudiantes de la Universidad del Valle de Guatemala

Trabajo de investigación presentado por José René Barnéond para optar al grado académico de Licenciado en Ingeniería en Ciencias de la Computación

Guatemala

2009

Vo. Bo.

(f) _____

(Ing. Luis Furlan)

Tribunal Examinador

(f) _____

(Ing. Sergio Izquierdo)

(f) _____

(Ing. Ana Miriam Aguilar)

Fecha de aprobación del exámen: 9 de Diciembre de 2009

PREFACIO

Este trabajo fue el resultado de un esfuerzo en equipo que inició con la decisión de los encargados de los diferentes departamentos de informática en la Universidad del Valle de Guatemala de evaluar el estado actual de la seguridad del portal de estudiantes de la misma.

Este trabajo, *Evaluación de Seguridad del Portal de Estudiantes de la Universidad del Valle de Guatemala*, fue posible gracias a la ayuda, principalmente, del encargado de desarrollo de la aplicación Marvin Raúl González (MaGo), a mi camarada y aliado Eduardo Castellanos y a mi ilustre asesora Ana Miriam Aguilar.

Se agradece también a todo el equipo de cómputo de la Universidad del Valle por facilitarnos sus instalaciones e infraestructura para realizar las pruebas necesarias para el desarrollo del trabajo.

Por último, gracias a Dios.

ÍNDICE

LISTA DE CUADROS

LISTA DE GRÁFICOS

RESUMEN

Una evaluación de seguridad, o evaluación de vulnerabilidades, consiste en calificar el desempeño de un sistema o dispositivo en la presencia de amenazas de seguridad mediante la simulación de ataques que se podrían dar en la realidad. Las vulnerabilidades son debilidades o errores en el sistema, programación, o arquitectura subyacente que permiten que mediante un ataque se viole o amenace la seguridad de los bienes del sistema analizado.

La seguridad informática se basa en proteger tres pilares principales: integridad, disponibilidad, y confidencialidad. Con las pruebas realizadas en este trabajo se pretende evaluar las capacidades que posee el portal de estudiantes de la Universidad del Valle de Guatemala para proteger estos tres pilares frente a la posibilidad de atacantes internos o externos.

Finalmente, se presenta un cuadro de resultados donde se pueden apreciar fácilmente los puntos críticos en los que falla la aplicación y que deben ser atendidos con premura, así como las demás secciones de su funcionamiento que deben ser revisadas.

I. INTRODUCCIÓN

En las siguientes páginas de este trabajo se presenta un análisis de vulnerabilidades de la aplicación de estudiantes de la Universidad del Valle de Guatemala. En él se pretende evaluar de acuerdo a un estándar predefinido y popular, el nivel de seguridad general de la aplicación así como los errores de lógica, programación, y seguridad que ésta padece actualmente.

Para poder medir la seguridad de la aplicación de una forma estándar, se realizó una serie de pruebas definidas por una metodología (Guía de pruebas de OWASP) para determinar cuántas de las pruebas pueden ser superadas por la aplicación. Estas evaluaciones pueden ser de varios tipos en función de la cantidad de información que se comparte entre las partes (los examinadores y los examinados).

La evaluación de seguridad se llevó a cabo de forma transparente ya que tanto el examinador de la seguridad, como el equipo de informática fueron advertidos de la existencia de las pruebas. La pruebas individuales fueron realizadas en ocasiones utilizando un enfoque de caja negra (dónde no se tenía ningún conocimiento interno de la parte evaluada) y en ocasiones utilizando un enfoque de caja blanca (se solicitó al equipo de informática que brindara la información necesaria).

Al concluir las pruebas se encontraron un total de 16 vulnerabilidades de las cuales cuatro son de prioridad baja, tres son prioridad media, siete de prioridad alta, y dos son de nivel crítico que deben ser atendidas urgentemente.

II. EVALUACIÓN DE VULNERABILIDADES

A. ¿Qué es una evaluación de vulnerabilidades?

Una evaluación de vulnerabilidades es el proceso por el cual se prueba un dispositivo o sistema en la presencia de amenazas de seguridad mediante la simulación de métodos de ataque que utilizaría un atacante real. La mayoría de las evaluaciones ejecutan muchas pruebas individuales de seguridad en cada sistema objetivo. Estas pruebas se desarrollan por profesionales de seguridad participantes en la industria que son altamente conocidos y expertos en el tema. Los resultados de las pruebas son procesados y filtrados en un reporte de la evaluación que provee detalles de qué tipo de riesgo fue descubierto y qué se puede hacer para remediarlo. (Moses, 2003)

Muchas veces se identifica un riesgo que es irrelevante o inexistente. Estos riesgos no exactos son clasificados como falsos positivos, y deberían ser descartados. Los falsos positivos ocurren durante la fase de pruebas, cuando en una prueba en particular se falla en interpretar los resultados. (Moses, 2003)

Los métodos de ataques y herramientas de evaluación de vulnerabilidades varían significativamente en la industria. Es importante notar que casi todos los métodos de ataque brindan información acerca de las vulnerabilidades de un sistema, y los dos más comunes son:

- Tipo A: el atacante elige deliberadamente una organización y algo dentro de ella que le interesa por lo que ataca todas las capas de seguridad continuamente hasta lograr un nivel de privilegios satisfactorio.
- Tipo B: el atacante elige una vulnerabilidad o vulnerabilidades específicas y un conjunto de herramientas con las que se siente cómodo y esto determina quién será su víctima. (Moses, 2003)

El ataque de tipo A es el que brinda información más completa al hacer una evaluación de seguridad, pues cada capa de seguridad debe ser analizada e investigada a fondo para entender completamente su interacción con las demás capas y su efecto general sobre la infraestructura de la aplicación. Cada capa debería ser luego probada contra una serie de pruebas para descubrir vulnerabilidades específicas o generales. (Moses, 2003)

Una vulnerabilidad es una imperfección o debilidad en el diseño, implementación, u operación y manejo de un sistema que puede ser explotado para violar la política de seguridad del sistema. Una amenaza es un ataque potencial, que al explotar una vulnerabilidad puede causar daño a los bienes o activos de una aplicación (como información en una base de datos o en el sistema de archivos). (OWASP.org, 2008)

Dentro de la metodología de pruebas que se hacen contra una aplicación, cada prueba de forma individual se puede realizar de tres maneras:

- Caja negra: el personal que realiza la prueba lleva a cabo los ataques sin ningún conocimiento previo de la infraestructura, mecanismos de defensa y canales de comunicación del sistema objetivo.
- Caja gris: el personal que realiza la prueba la hace con conocimiento limitado de la infraestructura del sistema objetivo, mecanismos de defensa y canales de comunicación.
- Caja blanca: el personal realiza la prueba conociendo perfectamente la infraestructura de la aplicación, los mecanismos de defensa, y los canales que utiliza el sistema objetivo. (Bozidar, 2008)

Asimismo, el enfoque general de la evaluación puede seguir uno o más de cinco estrategias:

- Prueba externa: se realiza desde afuera de la infraestructura del sistema u organización objetivo, es decir desde Internet.

- Prueba interna: se realiza desde la infraestructura y ambiente de la aplicación u organización objetivo.
- Prueba ciegas: al equipo que prueba la aplicación no le proveen ninguna información específica, pero el equipo de IT está al tanto de la realización de las pruebas.
- Prueba doblemente ciega: es una extensión de la prueba a ciegas en la que el equipo de informática no es informado de la realización de las pruebas. (Mehta, 2005)

B. Guía de pruebas de OWASP

OWASP ("Open Web Application Security Project" por sus siglas en inglés) o Proyecto Abierto de Seguridad de Aplicaciones Web, es una comunidad mundial abierta enfocada en mejorar la seguridad de aplicaciones de software. (OWASP.org, 2009)

La guía de pruebas OWASP es una guía que comprende los procedimientos y herramientas de seguridad de aplicaciones. La mejor forma de emplear la guía es utilizarla como parte de una verificación de seguridad completa. (OWASP.org, 2008)

La guía OWASP está dividida en dos fases: pasiva y activa. En la fase pasiva el personal de prueba trata de entender la lógica de la aplicación y "juega" con la aplicación para conocerla mejor. En esta fase se pueden utilizar herramientas para recolectar información como un proxy que observe todas las peticiones y respuestas HTTP de y hacia el servidor web. Al final de esta etapa el personal que realice la prueba debe entender todos los puntos de acceso de la aplicación. En la parte activa el investigador empieza a probar el sistema utilizando la guía de OWASP como metodología. Esta etapa está dividida en ocho fases:

- Administración de la configuración
- Sistema de autenticación
- Administración de sesiones
- Autorización

- Lógica del negocio
- Validación de datos
- Denegación de servicios (DoS)
- Servicios Web

Cada uno de estos temas es desarrollado brevemente a continuación e incluye una lista de las pruebas que se realizaran en cada una de las fases. Las pruebas serán descritas con mayor profundidad en las páginas subsiguientes.

1. **Administración de la configuración.** Las pruebas de administración de la configuración son muy importantes ya que en muchas ocasiones revelan, mediante el análisis de la infraestructura y topología, información acerca de la aplicación web. Se puede obtener información como código fuente, métodos HTTP permitidos, funcionalidades administrativas, y métodos de autenticación. (OWASP.org, 2008)

Estas pruebas comprenden las pruebas de SSL/TLS, gestión de la configuración de la infraestructura, gestión de extensiones de archivo, e interfaces administrativas.

2. **Sistema de autenticación.** Un sistema de autenticación determina la identidad de una entidad para asegurar que un mensaje procede de quién dice venir. Todos los esquemas de autenticación están basados en la posesión de algún tipo de información secreta conocida únicamente por el usuario y posiblemente (pero no necesariamente) por el sistema de autenticación mismo. (Wells, 1996)

Se pueden autenticar objetos (determinar su procedencia) o personas (su identidad). La autenticación depende en ocasiones de más de un factor. (OWASP.org, 2008)

En esta división se incluyen pruebas para comprobar que las credenciales sean transmitidas a través de un canal seguro, determinar si existen cuentas de usuario predeterminadas o adivinables, intentar saltar el sistema de autenticación, auditar la salida de sesión y caché de navegación, y el estado del sistema, si hubiese uno, de autenticación de factores múltiples.

3. **Administración de sesiones.** En el centro de toda aplicación basada en tecnologías web se encuentra la forma en la que ésta mantiene un estado y por consiguiente controla toda interacción con el usuario y la aplicación. Las pruebas de administración y manejo de sesiones cubren ampliamente todos los controles de un usuario, desde el inicio de sesión, hasta el cierre de la misma. El protocolo HTTP tiene una orientación sin estado, por lo que los servidores web responden al cliente peticiones sin vincular una petición con otra subsecuente. Incluso operaciones simples de la lógica del negocio requieren que múltiples peticiones de un usuario sean asociadas las unas con las otras a través de una "sesión". Esto requiere de soluciones de un tercero, ya sea mediante un software comercial o implementaciones del equipo de desarrollo. Las aplicaciones web más populares como ASP y PHP proveen a los desarrolladores con rutinas de manejo de sesión integradas que proveen funciones típicas como una llave de identificación que se les provee a los usuarios, generalmente llamada identificador de sesión o "cookie". (OWASP.org, 2008)

Esta sección incluye pruebas para probar el esquema general de manejo de sesiones, determinar si se pueden fijar, y también si es posible realizar un ataque de CSRF ("Cross-site Request Forgery" por sus siglas en inglés).

4. **Autorización.** Autorización es el concepto de permitir acceso a recursos solamente a aquellos que tiene permisos a accederlos. Probar el esquema de autorización significa entender cómo se da el proceso de autorización, y usar esta información para burlar este mecanismo. El proceso de autorización viene luego de una autenticación exitosa, así que el personal que realiza la prueba debe primero conseguir credenciales para poder iniciar sesión en el sistema. (OWASP.org, 2008)

En esta unidad se realizarán pruebas para determinar si es posible burlar o evitar el sistema de autenticación y así logra realizar operaciones para las que no se tiene permiso con un usuario sin privilegios elevados.

5. **Lógica del negocio.** La comprobación de errores en la lógica del negocio en una aplicación web dinámica y multifuncional requiere pensar en formas poco convencionales. Si el mecanismo de autenticación de una aplicación está diseñado para que se ejecuten los pasos uno, dos y tres en ese orden para autenticarse, ¿qué pasa si se va del paso uno directo al paso tres? Como este ejemplo hay muchos otros que se pueden formular pero lo que se mantiene constante es que se debe "pensar en formas poco convencionales". Este tipo de vulnerabilidad no puede ser detectada por un escáner de vulnerabilidades y se basa en las aptitudes y capacidades creativas de la persona que ejecuta la prueba. Además, este tipo de vulnerabilidad es una de las más difícil de probar y detectar, pero al mismo tiempo usualmente resulta ser uno de las más perjudiciales para la aplicación en caso de ser explotada. (OWASP.org, 2008)

En esta sección se comprobará de forma general la lógica del negocio de las diferentes funcionalidades de la aplicación.

6. **Validación de datos.** La debilidad más común en la seguridad de aplicaciones web es la de fallar en validar apropiadamente las entradas que provienen del cliente, usuario, o ambiente, antes de utilizarlas. Esta debilidad lleva a la fuente de las mayores vulnerabilidades en aplicaciones web, como XSS, inyección SQL, inyección de código, ataques de Unicode, ataques al sistema de archivos, y sobrecargas de búfer. Datos que provienen del exterior del sistema no deben ser utilizados sin validación nunca, ya que pueden ser arbitrariamente modificados por un atacante para lanzar un ataque. (OWASP.org, 2008)

Se realizarán pruebas para comprobar la existencia de XSS almacenado o basado en flash, inyecciones de tipo LDAP, XML, XPath, y código. También se realizarán pruebas para determinar si existe un posible desbordamiento de búfer o contrabando y división de peticiones HTTP.

7. **Denegación de servicios.** El tipo más común de denegación de servicios es en el que se utiliza una red para hacer que un servidor sea

inaccesible por usuarios válidos. El concepto fundamental de un ataque de denegación de servicios basado en la red es un usuario malicioso que inunda con suficiente tráfico al servidor objetivo para que éste sea incapaz de responder el número de peticiones que está recibiendo, incluyendo las de usuarios legítimos. Cuando el usuario malicioso utiliza un gran número de máquinas para inundar el servicio de tráfico generalmente se conoce como un ataque de denegación de servicios distribuido. Estos tipos de ataque son generalmente impredecibles por el desarrollador de la aplicación y son prevenidos a un nivel de arquitectura de red. (OWASP.org, 2008)

En esta sección se realizarán pruebas para determinar si es posible bloquear cuentas de usuarios legítimos, reservar objetos en el servidor y agotar sus recursos, escribir datos a disco datos con el propósito de saturarlo, o almacenar demasiada información en el objeto de sesión del usuario que es almacenado en la memoria del servidor.

8. **Comprobación de servicios web.** Los sistemas SOA (Arquitectura Orientada a Servicios de sus siglas en inglés) o aplicaciones de servicios web son una tecnología nueva que está permitiendo a los negocios interoperabilidad y está creciendo rápidamente. Los clientes de servicios web son generalmente otros servidores y no usuarios finales. Estos servicios web están expuestos a la red igual que cualquier otro servicio. (OWASP.org, 2008)

Aquí se probarán los servicios web, ataques de XML a nivel de contenido, y adjuntos maliciosos (con virus, o malware).

C.Recopilación de información

1. Spiders, Robots, y Crawlers

a. Descripción. Las arañas (conocidos también como robots o rastreadores) son programas que recorren el Internet de forma automática para recopilar

información acerca de sitios utilizando el recurso “robots.txt”. Algunos sitios como Google y otros buscadores web utilizan programas de este tipo para indexar contenido web, los spammers lo usan para escanear y buscar direcciones de correo, y de igual manera pueden tener muchos otros usos. (Robotstxt.org, 2009)

El archivo robots.txt se encuentra generalmente en la raíz del dominio y éste contiene instrucciones para los robots sobre cómo deben navegar su sitio y obtener información. En este archivo se encuentra lo que se conoce como Protocolo de Exclusión de Robots. Un robot tiene, de cualquier manera, la opción de ignorar completamente las instrucciones dadas por el archivo robots.txt y explorar el sitio sin que el dueño del dominio lo desee por lo que no se recomienda “esconder” contenido de esta forma. (Robotstxt.org, 2009)

b. Procedimiento. Primero se debe determinar la existencia del archivo robots.txt que debe estar ubicado en el directorio raíz del servidor web. Una vez se determina la existencia se pueden utilizar las herramientas para administrador de sitios web de Google® para poder analizar el archivo, o bien se puede realizar de forma manual una revisión del contenido del mismo. (OWASP.org, 2008)

c. Resultados. Se determinó que el dominio donde se encuentra el portal no cuenta con un archivo robots.txt en la raíz del directorio, por lo que no son necesarias pruebas adicionales.

d. 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. De esta forma Robots legítimos como los de los buscadores web pueden indexar apropiadamente el dominio y no correr el riesgo de que información que no debería ser publicada este disponible por error debido a la mala configurar este archivo.

2. Reconocimiento mediante motores de búsqueda

a. Descripción. Una vez el GoogleBot ha terminado de recorrer, éste comienza a indexar la página web basado en las etiquetas y atributos asociados como <TITLE> (el título de la página), para poder devolver resultados de búsqueda relevantes. (OWASP.org, 2008)

Si el archivo robots.txt no se actualiza durante la vida de la página de Internet, es posible que el contenido que no se desee retornar en las búsquedas de los motores de búsqueda sea devuelto. (OWASP.org, 2008)

b. Procedimiento. Utilizando la opción avanzada de búsqueda en el motor de búsqueda de Google: "site:https://www.uvg.edu.gt" (o su equivalente en otro motor de búsqueda) se deben analizar todos los resultados que arroje el buscador para encontrar información acerca de la página que no debería estar publicada. (OWASP.org, 2008)

c. Resultados. Al realizar la búsqueda en Google éste arrojó un total de 852 resultados en 0.30 segundos. Entre los resultados no se encontró ninguno de relevancia para el portal, aunque sí 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 notó 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
```

d. Recomendaciones. Aunque la información encontrada no pertenece al portal específicamente, puede que sí pertenezca a algún servicio o aplicación en el mismo servidor, que podría llegar a comprometer el sistema completo. Es recomendable configurar apropiadamente los accesos a este tipo de páginas con

información sensible y asimismo el archivo robots.txt para que los buscadores sepan qué indexar y qué no indexar.

3. Identificación de puntos de entrada de la aplicación

a. Descripción. Los puntos de entrada proveen información de una aplicación. Los valores que se introducen en estos son llevados a la base de datos, servidor LDAP, motores de procesamiento, y otros componentes de la aplicación. Si estos puntos de entrada y los valores que reciben no son protegidos pueden abrir vulnerabilidades en potencia en la aplicación. 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)

Enumerar los puntos de entrada de la aplicación y en consecuencia su superficie de ataque es un requisito clave para poder realizar pruebas a fondo. Antes de empezar las pruebas es importante familiarizarse con la aplicación y cómo ésta interactúa con el usuario/explorador web. (OWASP.org, 2008)

b. Procedimiento. Utilizar un proxy que permite registrar todas las acciones realizadas en un navegador web. Se debe navegar por la aplicación

buscando todos los puntos de entrada, como formas y acciones, y posteriormente archivar adecuadamente los puntos de entrada encontrados para analizarlos y determinar su potencial como vector de ataque. (OWASP.org, 2008)

c. 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.

d. Recomendaciones. No hay recomendaciones

4. Pruebas de firmas de aplicaciones web

a. Descripción. Una huella digital sirve para identificar de forma única, de acuerdo a sus características, a un ser humano. De la misma forma un sistema operativo o servidor web tiene características únicas en su implementación de comunicación que sirvan para identificarlo en la red. Mediante el análisis de banderas del protocolo de red, opciones, y data enviada en los paquetes que un dispositivo envía a la red, se puede identificar de forma relativamente puntual el sistema operativo o servidor web al cual pertenecen estos datos. (Allen, 2007) (Shah S. , 2004)

b. Procedimiento. Esta prueba se hizo de tipo caja blanca, de tal forma que simplemente se le solicitó al equipo de informática de la aplicación que nos brindaran la información acerca del servidor web y su infraestructura. (OWASP.org, 2008)

c. 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

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

5. Descubrimiento de aplicaciones

a. Descripción. Con la proliferación de servidores web, la relación tradicional de 1:1 entre direcciones IP y servidores web ha perdido mucho de su significado original. No es poco común encontrar múltiples páginas o aplicaciones web cuyo nombre simbólico se resuelve a la misma dirección IP. (OWASP.org, 2008)

Muchas aplicaciones tienen vulnerabilidades y estrategias de ataque conocidas que pueden ser explotadas para lograr obtener control remoto o alterar datos dentro de un servidor. Si el servidor sobre el cual corre la aplicación objetivo cuenta con más aplicaciones, las vulnerabilidades de estas afectan también a la aplicación analizada. (OWASP.org, 2008)

b. Procedimiento. Se utilizará un conjunto de herramientas automatizadas para descubrir las aplicaciones que se encuentran dentro del servidor uvg.edu.gt.

c. 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	Nombres de cliente compartiendo dirección IP con registros-a	Dominios utilizando el servidor de nombre bajo otro nombre
200.35.167.66	kirika.uvg.edu.gt ns2.uvg.edu.gt www.uvg.edu.gt	proesur.uvg.edu.gt uvg.edu.gt
Servidores de Nombre utilizados por este dominio	**Dominios que comparte el servidor de nombres**	**Direcciones IP de los servidores de nombre**
ns.uvg.edu.gt ns2.uvg.edu.gt	cs.uvg.edu.gt gt proesur.uvg.edu.gt	168.234.68.2 168.234.68.6 168.234.76.6 200.35.167.65 200.35.167.66 200.9.74.2
Nombres reversos de los servidores de nombre	**Otros nombres de los servidores de nombres**	**Servidores de correo utilizados por este dominio**
kirika.uvg.edu.gt mail.uvg.edu.gt	anarres.uvg.edu.gt kirika.uvg.edu.gt mail.gt mail.uvg.edu.gt ns.gt steve.uvg.edu.gt uvg.edu.gt	mail.uvg.edu.gt(primary)

	www.gt www.uvg.edu.gt	
Direcciones IP de los servidores de correo	Nombres reversos de los servidores de correo	Otros nombres de los servidores de correo
168.234.76.6	kirika.uvg.edu.gt mail.uvg.edu.gt	kirika.uvg.edu.gt ns2.uvg.edu.gt www.uvg.edu.gt

2) Sub-dominios

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

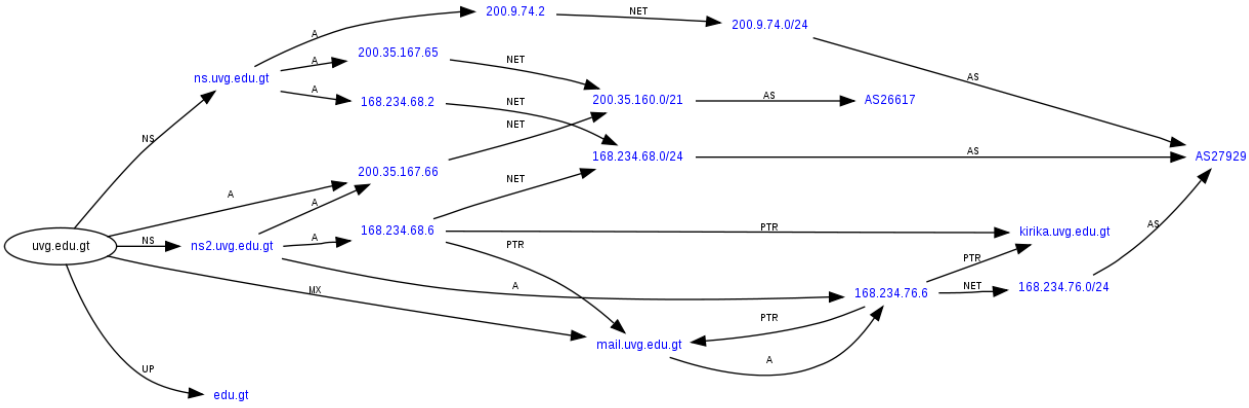


Ilustración 1: Diagrama de nombres de dominio

www.uvg.edu.gt es un alias de kirika.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
993/tcp  open  ssl/imap UW imapd 2007e.404
995/tcp  open  ssl/pop3 UW Imap pop3d 2007e.104
7777/tcp open  unknown
```

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

```
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
```

d. Recomendaciones. No hay recomendaciones.

6. Análisis de códigos de error

a. Descripción. A menudo durante el análisis de vulnerabilidades de una aplicación web se encuentran muchos códigos de error generados por aplicaciones o servidores web. Es posible hacer que estos errores se desplieguen fabricando una petición particular con herramientas o de forma manual. Estos códigos son muy útiles para la recopilación de información ya que normalmente revelan información importante acerca de las bases de datos, errores en el código, y otros componentes tecnológicos que estén vinculados directamente con la aplicación web. (OWASP.org, 2008)

Un error muy común que se suele encontrar es el error “HTTP 404 Not Found” o página no encontrada. Este error comúnmente contiene detalles muy útiles acerca de

la infraestructura interna de la aplicación web y sus componentes asociados. (OWASP.org, 2008)

b. Procedimiento. Se navegará a través de la aplicación web en busca de lugares donde se pueden producir errores. Se intentará acceder a direcciones inexistentes y determinar la arquitectura de la aplicación.

c. Resultados.

Tabla 2: 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 Tabla 2

Error	Mensaje de Error	Comentario
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. Revela información como cuales son las expresiones validas y campos.

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.

Continuación Tabla 2

Error	Mensaje de Error	Comentario
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	Revela la ubicación de los servicios web.

	\'students/StudentsGeneral.asmx?wsdl\'" }	
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.

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

D. Pruebas de gestión de configuración de la infraestructura

1. Pruebas de SSL/TLS

a. Descripción. SSL (*Secure Sockets Layer* en inglés) es un protocolo de red de computadoras que provee autenticación, confidencialidad e integridad

de información intercambiada a través de una red de computadoras. Fue diseñado en 1944 por la compañía Netscape Communications cuando se dieron cuenta que los usuarios de su navegador necesitaban de una forma de comunicación segura. (Boncella, 2004)

SSL está compuesto de dos capas, en la parte de más bajo nivel apoyado sobre un protocolo de transporte fiable se encuentra el protocolo SSL de registro. Este protocolo se usa para la encapsulación de otros protocolos de más alto nivel, siendo uno de estos el protocolo de "handshake" de SSL, que le permite al cliente y servidor que se autentiquen mutuamente y negocien el algoritmo y llaves de cifrado antes de que el protocolo de aplicación transmita cualquier tipo de información. Una ventaja de este protocolo es su independencia de la capa de aplicación por lo que un protocolo de más alto nivel puede montarse sobre éste de forma transparente. El protocolo SSL tiene tres propiedades básicas:

- Privacidad de conexión, pues se utiliza un algoritmo de cifrado después del saludo inicial
- Capacidad de autenticar al equipo remoto utilizando cifrado de llave pública.
- La conexión es fiable pues se usan chequeos de integridad para el mensaje. (Freier, Karlton, & Kocher, 1996)

TLS son las siglas de Transport Layer Security y fue desarrollado por la IETF como sucesor de SSL. Las diferencias entre los dos protocolos son muy pequeñas y mayormente técnicas. TLS utiliza cifrado más fuerte y puede trabajar en diferentes puertos, por esto se usa generalmente en correo electrónico, aunque también tiene la capacidad de actuar como asegurador de comunicaciones cliente-servidor. (Indiana University, 2009)

Debido a las estrictas leyes de exportación de armas de los Estados Unidos, la exportación de algoritmos o herramientas de cifrado de alto nivel estaba prohibido, por lo que algunos servidores pueden estar utilizando un soporte de cifrado débil o nulo. (OWASP.org, 2008)

Aún cuando se utiliza un algoritmo y herramientas de cifrado fuertes y probadas, una mala configuración del servidor podría dejar expuesta la aplicación para el uso de un cifrado más débil para obtener acceso al canal de comunicación que se supone seguro. (OWASP.org, 2008)

El protocolo HTTP viaja en la red en forma de texto claro, por lo que generalmente se asegura haciéndolo pasar por un túnel SSL o TLS. (OWASP.org, 2008)

b. Procedimiento. Se realizarán las pruebas SSL/TLS usando la herramienta de software Nessus versión 3 que analiza automáticamente los niveles de cifrado que soporta la aplicación.

c. Resultados. Se detectó el uso de algoritmos de cifrado o configuraciones de cifrado débiles, las cuales podrían permitir a un atacante realizar un ataque man-in-the-middle y descifrar la información cifrada. Los certificados utilizados son válidos hasta el 28 de agosto de 2010.

El servidor soporta el uso de cifrados SSL anónimos. Esto permite al administrador configurar servicios que cifren el tráfico sin la necesidad de generar y configurar certificados SSL, pero no ofrece ningún mecanismo para verificar la identidad del servidor, lo cual deja el servicio vulnerable a un ataque man-in-the-middle. También se encontró que el servidor está configurado para permitir la utilización de cifrados débiles. Un atacante podría obligar a que la comunicación se transmitiera mediante un cifrado débil para después descifrarlo.

Algoritmos de cifrado débiles (< 56-bit key)

SSLv2

EXP-RC2-CBC-MD5 Kx=RSA(512) Au=RSA Enc=RC2(40) Mac=MD5 export

EXP-RC4-MD5 Kx=RSA(512) Au=RSA Enc=RC4(40) Mac=MD5 export

SSLv3

EXP-EDH-RSA-DES-CBC-SHA Kx=DH(512) Au=RSA Enc=DES(40) Mac=SHA1 export

EXP-DES-CBC-SHA Kx=RSA(512) Au=RSA Enc=DES(40) Mac=SHA1 export
EXP-RC2-CBC-MD5 Kx=RSA(512) Au=RSA Enc=RC2(40) Mac=MD5 export
EXP-RC4-MD5 Kx=RSA(512) Au=RSA Enc=RC4(40) Mac=MD5 export
TLSv1
EXP-EDH-RSA-DES-CBC-SHA Kx=DH(512) Au=RSA Enc=DES(40) Mac=SHA1 export
EXP-DES-CBC-SHA Kx=RSA(512) Au=RSA Enc=DES(40) Mac=SHA1 export
EXP-RC2-CBC-MD5 Kx=RSA(512) Au=RSA Enc=RC2(40) Mac=MD5 export
EXP-RC4-MD5 Kx=RSA(512) Au=RSA Enc=RC4(40) Mac=MD5 export

Leyenda:
{OpenSSL algoritmo de cifrado}
Kx={Método de intercambio de llaves}
Au={Autenticación}
Enc={Método de cifrado simétrico}
Mac={message authentication code}
{Bandera de export}

d. Recomendaciones. Activar y soportar únicamente algoritmos de cifrado seguros con una llave de más de 64 bits en la aplicación.

2. Pruebas de gestión de configuración de la infraestructura

a. Descripción. La complejidad que se crea al combinar las diferentes tecnologías de servidores web con los cientos de aplicaciones que pueden estar montados sobre ellos hace que la revisión de las configuraciones de la infraestructura como un todo sea de suma importancia. En el caso de las aplicaciones la regla de "el eslabón más débil" también aplica, ya que una sola vulnerabilidad explotable puede comprometer al sistema completo, incluso aplicaciones que solo

tenían en común estar en el mismo servidor. Una configuración errónea inofensiva para una aplicación puede causar que otra quede totalmente expuesta, por lo que es primordial realizar un análisis profundo de la configuración y vulnerabilidades conocidas. (OWASP.org, 2008)

b. Procedimiento. La prueba será realizada baje el enfoque de caja blanca. Luego de obtener la información de la infraestructura interna de la aplicación web se buscará literatura e información sobre las vulnerabilidades conocidas de la infraestructura y en caso de que existan, los parches o actualizaciones que las arreglen.

c. Resultados. Infraestructura de la aplicación según reporte del cliente:

- 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
- Servicios Web:
  - Microsoft Windows 2003 Server
  - IIS V6.0
  - C#.NET 2005
- Servidor de Base de Datos:

- Microsoft Windows 2003 Server
- Microsoft SQL Server 2005

Vulnerabilidades encontradas en los servidores y aplicaciones[1]:

- Servidor Web
  - Apache
    - Prioridad baja:
      - Denegación de servicios por módulo mod_deflate
      - Salto de autenticación utilizando opciones y AllowOverride
    - Prioridad moderada:
      - Underwrite de la memoria HEAP en la librería APR-util
      - DoS por el traductor XML de la librería APR-util
      - Sobrecarga producida por el problema falla-por-uno (off-by-one) en la librería APR-util
    - Prioridad importante:
      - DoS en el módulo mod_proxy al utilizarlo como proxy reverso.

[1] Las severidades de las vulnerabilidades fueron tomadas como las reportó el fabricante o autor de la vulnerabilidad

- Filtrado de información en el módulo mod_proxy_ajp mediante la creación de una petición HTTP específica. (The Apache Software Foundation, 2009)

- PHP
  - PHP 'exif_read_data()' Vulnerabilidad de DoS por procesamiento de JPG
    - http://www.securityfocus.com/bid/35440
  - PHP Vulnerabilidades generales que permiten DoS por diferentes razones.
    - http://www.securityfocus.com/bid/33927
  - PHP Vulnerabilidad de ejecución de código arbitrario por interrupciones.
    - http://www.securityfocus.com/bid/35867
  - PHP 'imageRotate()' Memoria no inicializada produce filtrado de información privada
    - http://www.securityfocus.com/bid/33002
  - PHP Múltiples funciones en modo seguro violan restricciones.
    - http://www.securityfocus.com/bid/35435
  - PHP 'mb_ereg_replace()' Vulnerabilidad en la evaluación de la cadena
    - http://www.securityfocus.com/bid/34873
  - PHP 'popen()' Función produce un desbordamiento del búfer.
    - http://www.securityfocus.com/bid/33216

        - PHP 'proc_open()' Parámetros de entorno en modo seguro violan restricciones
            - http://www.securityfocus.com/bid/32717
    - OpenSSL
        - 'ChangeCipherSpec' Denegación de servicio por paquete DTLS
            - http://www.securityfocus.com/bid/35174
        - 'dtls1_retrieve_buffered_fragment()' Denegación de servicio por paquete DTLS
            - http://www.securityfocus.com/bid/35417
        - Múltiples vulnerabilidades
            - http://www.securityfocus.com/bid/34256
        - Múltiples vulnerabilidades de denegación de servicio por paquetes DTLS
            - http://www.securityfocus.com/bid/35001
        - 'zlib' Denegación de servicio remota por compresión de memoria
            - http://www.securityfocus.com/bid/31692
        - 'EVP_VerifyFinal' Verificación de firma de la función
            - http://www.securityfocus.com/bid/33150

            (SecurityFocus, 2009)

- Servicios Web
    - IIS

- Vulnerabilidades en el IIS podrían permitir escalamiento de privilegios
    - http://www.microsoft.com/technet/security/Bulletin/MS09-020.mspx
- Microsoft IIS FTPd NLST: sobrecarga de búfer remota.
    - http://www.securityfocus.com/bid/36189
- Microsoft IIS FTPd Globbing Functionality: Denegación de servicios remota
    - http://www.securityfocus.com/bid/36273
- Microsoft Collaboration Data Objects: Sobrecarga de búfer remota
    - http://www.securityfocus.com/bid/15067
- Microsoft XML Parser: Denegación de servicios remota
    - http://www.securityfocus.com/bid/11384
- Microsoft ASN.1: Mala administración de largo de enteros lleva a corrupción de memoria
    - http://www.securityfocus.com/bid/9633
- Peticiones Unicode al WebDAV producen vulnerabilidades de salto de autenticación
    - http://www.securityfocus.com/bid/34993
- Microsoft IIS: Notificación de cambio de archivo produce escalamiento de privilegios.
    - http://www.securityfocus.com/bid/27101

        - Microsoft IIS ASP: Ejecución remota de código
            - http://www.securityfocus.com/bid/27676
        - Microsoft IIS 6.0: Múltiples vulnerabilidades de la administración web.
            - http://www.securityfocus.com/bid/8244
        - MS IIS Revelación de IP interna o nombre de red interna.
            - http://www.securityfocus.com/bid/3159 (SecurityFocus, 2009)

- Servidor de bases de datos
    - Microsoft SQL 2005
        - Vulnerabilidad en MSSQL 2005 podría permitir ejecución remota de código
            - http://www.microsoft.com/technet/security/Bulletin/MS09-004.mspx
        - Vulnerabilidades en GDI+ podrían permitir ejecución remota de código
        - http://www.microsoft.com/technet/security/Bulletin/MS08-052.mspx (Microsoft, 2009)

d. Recomendaciones. Existe una versión más reciente del servidor web APACHE que corrige las vulnerabilidades en la versión actualmente en producción. De igual forma la versión de PHP utilizada y los diferentes componentes pueden ser actualizados. Respecto a los productos de Microsoft existen recomendaciones o incluso parches para remediar las vulnerabilidades en sus productos.

3. Pruebas de gestión de extensiones de archivo

a. Descripción. Las extensiones de los archivos a menudo determinan qué tipo de acción o tecnología utilizarán los servidores para responder a las peticiones que se les hacen. A pesar de que esto se hace de acuerdo al estándar respectivo RFC y estándares web, el uso de estas extensiones revela mucha información acerca de las tecnologías y aplicaciones utilizadas en el servidor y simplifica la determinación del escenario de ataque más indicado para la plataforma específica. (OWASP.org, 2008)

Es de mucha utilidad esta prueba ya que revela como maneja el servidor las acciones de diferentes extensiones lo que nos permitiría por ejemplo, descargar un archivo con código fuente si lográramos renombrarlo de .php a .txt. (OWASP.org, 2008)

b. Procedimiento. Esta prueba se realizará de tipo caja blanca, se solicitará al equipo de IT que listen todos los archivos y carpetas contenidas dentro del ambiente de la aplicación web y se revisarán para determinar si deberían estar ahí o si están de alguna manera liberando información confidencial o importante.

c. Resultados. Se revisaron un total de 844 archivos/carpetas y se encontró únicamente un archivo de tipo copia de respaldo "./mod/teacher/actualizarDatos/module.conf.bak" y no contiene mayor información que pueda ser utilizada para atacar la aplicación.

Los archivos con extensión .res y .conf contienen información de configuración de acciones y códigos PHP pero es probable que sea necesario que estén accesibles de forma pública. Además la aplicación utiliza extensiones html, js, php y css.

d. Recomendaciones. Eliminar el archivo ./mod/teacher/actualizarDatos/module.conf.bak del servidor web y si es posible restringir el acceso a los archivos con extensión .ref y .conf.

4. Interfaces administrativas de aplicación e infraestructura

a. Descripción. Las interfaces administrativas son comúnmente montadas en la aplicación o en el servidor. Esto ocurre para permitir realizar las tareas relacionadas con la administración por ciertos usuarios desde el mismo lugar. Las pruebas de esta etapa deben llevarse a cabo para determinar si estas aplicaciones existen, y como podrían ser accedidas por un usuario sin credenciales. (OWASP.org, 2008)

Estas interfaces son en muchas ocasiones implementadas sin establecer si es necesario montarlas; si lo es generalmente no se consideran las posibles consecuencias de hacerlo, e incluso no se adoptan medidas para asegurar la interfaz de forma adecuada. (OWASP.org, 2008)

b. Procedimiento. La metodología de esta prueba será de caja blanca. Se pedirá al encargado de la aplicación que enumere las interfaces administrativas de la aplicación.

c. Resultados. El encargado de IT reportó que la aplicación no cuenta con ninguna interfaz administrativa.

d. Recomendaciones. No hay recomendaciones.

E. Comprobación del sistema de autenticación

1. Transmisión de credenciales a través de un canal cifrado

a. Descripción. Al ingresar nuestras credenciales en un portal estamos enviando información sensible a través de Internet que se desea proteger

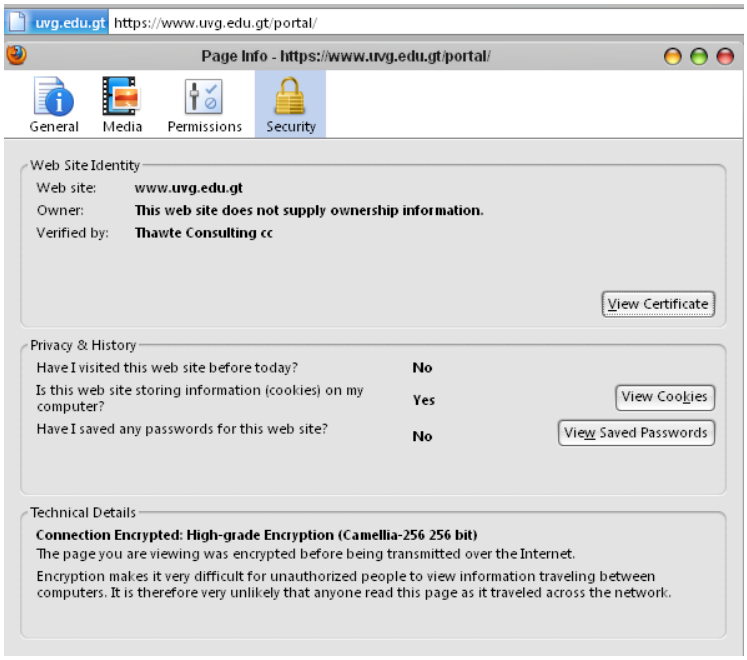
de atacantes malintencionados que podrían instalar un analizador de paquetes en medio. Para prevenir esto se utilizan tecnologías de cifrado como SSL o TLS. (Gribov, 2000)

HTTPS es una versión segura de HTTP, lo que hace únicamente es enviar el tráfico HTTP por un canal cifrado de SSL. Al conectarse por HTTPS a una página la persona se asegura que: al acceder a la página que se solicita, se está haciendo a través de un canal cifrado, y que la página a la que se accese tiene un certificado extendido por una autoridad certificadora conocida. (Gribov, 2000)

b. Procedimiento. Se revisará si la forma única para el acceso a la aplicación utiliza cifrado, y si hay alguna forma de acceder a ella sin utilizar cifrado.

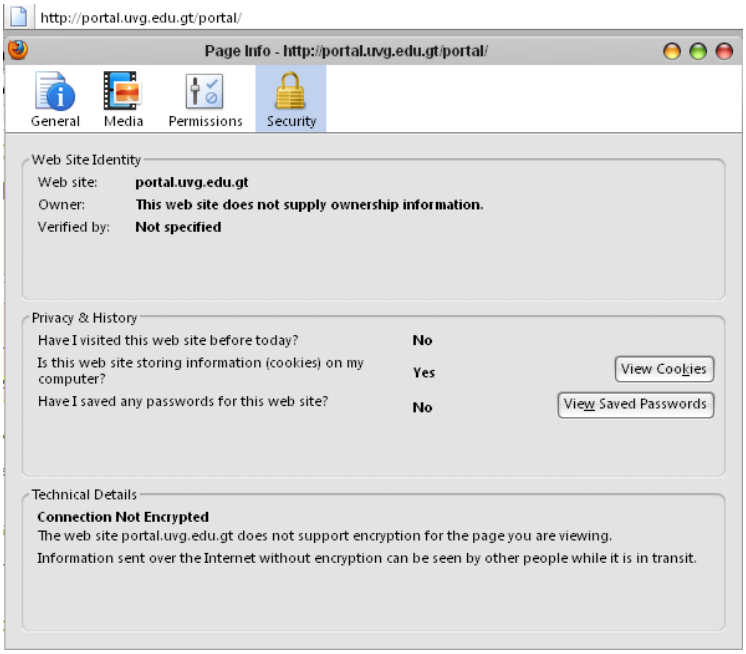
c. Resultados. La conexión a la página del portal (https://www.uvg.edu.gt/portal) es cifrada con un nivel de cifrado de 256 bits. La Universidad además utiliza un certificado firmado por Thawte Consulting cc.

Ilustración 2: Seguridad del portal en https://www.uvg.edu.gt/portal/



Sin embargo, es también posible acceder al portal a través de la dirección http://portal.uvg.edu.gt/portal, que es la antigua dirección del portal, dirección que muchos usuarios puede que aún tengan almacenada como favoritos. Al entrar por esta dirección la conexión a la página no es cifrada por lo que cualquier persona conectada a la misma subred, con las herramientas y el conocimiento necesario, podría "escuchar" los paquetes que pasan entre la página y el usuario.

Ilustración 3: Seguridad del portal en http://portal.uvg.edu.gt/portal/



d. Recomendaciones. Deshabilitar el uso al público de la página http://portal.uvg.edu.gt/portal.

2. Cuentas de usuario predeterminadas o adivinables (diccionario)

a. Descripción. Las aplicaciones web de hoy típicamente corren sobre código de fuente abierta o comercial y están instaladas en servidores que simplemente requieren configuración o personalización del administrador del servidor. Además la mayoría de dispositivos de hardware como ruteadores de red y servidores de bases de datos tienen interfaces administrativas web. (OWASP.org, 2008)

Estas aplicaciones están comúnmente mal configuradas y las credenciales que traen por defecto para la autenticación inicial no son actualizadas o cambiadas. Además es muy frecuente encontrar cuentas genéricas que quedan de un proceso de administración o pruebas que utilizan combinaciones de usuarios y contraseñas comunes que se dejan habilitadas en el sistema. (OWASP.org, 2008)

b. Procedimiento. La metodología de esta prueba será de caja blanca por lo que se preguntará al personal de IT si utilizan alguna combinación de usuario y contraseña común o adivinable. También se pedirá que brinden información sobre si tienen alguna política para la administración de usuarios y contraseñas.

c. Resultados. El equipo de informática se rige por un estándar de definición de usuarios y contraseñas por lo que no debería haber ninguna cuenta predeterminada ni contraseña fácil de adivinar.

d. Recomendaciones. No hay recomendaciones.

3. Saltarse el sistema de autenticación

a. Descripción. Muchas veces el personal de IT y desarrolladores de la aplicación están conscientes de lo importante que es el papel de la autenticación dentro de la arquitectura de la aplicación por lo que implementan protocolos muy seguros. En estas ocasiones tratar de atacar el sistema de autenticación de forma directa puede no ser la forma más fácil o probable de acceder a la aplicación. Atacar otros componentes de la aplicación como secuestrar o falsificar una sesión autenticada o atacar el subsistema de manejo de identidades puede resultar más productivo para saltarse el sistema de autenticación. (Scambray, Shema, & Sima, 2006)

Negligencia, ignorancia o simplemente no darle la importancia correcta a este tipo de amenazas de seguridad a menudo resulta en esquemas de autenticación que pueden ser saltados al simplemente saltar la página de autenticación y llamar directamente a la página "privada" interna que se supone solo debería poder ser ingresada una vez estando autenticado. (OWASP.org, 2008)

b. Procedimiento. Para intentar eludir el esquema de autenticación existen básicamente cuatro ataques que se pueden utilizar:

- Petición de página directa: solicitar de forma directa una página que se conoce debería estar protegida de acceso sin estar autenticado.

- Modificación de parámetros: en ocasiones la página identifica el estado de autenticación de una petición únicamente por un parámetro en la misma.
- Predicción de identificador de sesión: en ocasiones la aplicación web utiliza esquemas de sesión conocidos como SESSION ID o PHPSESSID. Si el esquema que utiliza es predecible se puede generar un identificador de sesión y utilizar la sesión autenticada antigua de otro usuario.

c. Resultados.

- Petición directa de páginas (navegación forzada)
  - No es vulnerable
- Modificación de parámetros
  - No es vulnerable
- Predicción de identificadores de sesión

  No es posible, los identificadores de sesión son generados por PHP.

- Inyección SQL

  No es vulnerable a inyección SQL en la pantalla de autorización.

d. Recomendaciones. Utilizar el registro de sesiones para realizar un control de operaciones realizadas sobre el sistema de tal forma que aún si el sistema de autenticación es burlado, una operación legítima puede ser identificada.

4. Pruebas de gestión del caché de navegación y de salida de sesión

a. Descripción. Cada vez que se visita una página de Internet utilizando un navegador, éste guarda la data de la página en disco duro (incluyendo las imágenes que la página pueda contener) para que no sea necesario reenviar la petición de la página al servidor y esperar la respuesta de nuevo cuando se utilizan las opciones

de regresar y adelantar en el historial. Esta data que el navegador guarda en memoria local para ahorrar tiempo se denomina Cache. (Indiana University, 2009)

El manejo de sesión Web es un término que describe como una aplicación Web coordina de forma transparente la autorización de un usuario para cada petición HTTP sin tener que autenticarse cada vez. La carga del manejo de sesión debe recaer únicamente sobre la aplicación Web debido a la orientación sin estado del protocolo HTTP. El sistema de manejo de sesión debe enviar al usuario a través de su agente de envío (como el navegador Web) una llave de sesión luego de que el usuario se haya autenticado exitosamente. En la mayoría de los casos esta llave se traslada al navegador utilizando el comando Set-Cookie de HTTP y es almacenada en el cliente. Esta llave de sesión debe ser enviada por el cliente en cada petición HTTP al servidor para poder identificarse ante la aplicación Web. Luego es tarea de la aplicación determinar si el cliente está autorizado para acceder a la página que realiza la petición o no. (Belani, 2004)

Una sesión Web es terminada normalmente cuando se dispara uno de dos eventos:

- El usuario termina la sesión utilizando un botón u opción de cerrar la sesión
- El usuario se mantiene sin actividad por un período de tiempo y la aplicación automáticamente termina la sesión.

Ambos casos deben ser implementados de forma cautelosa para evitar introducir debilidades o vulnerabilidades que un atacante podría utilizar para explotar y obtener acceso sin autorización a la aplicación. (OWASP.org, 2008)

b. Procedimiento. El primer paso es determinar la existencia del botón u opción para cierre de sesión. Si éste existe se debe comprobar qué es lo que sucede con las cookies una vez se ejecute esta opción. Lo esperado es que modifique la cookie y coloque un valor inválido, o que modifique la fecha de expiración a una que esté en el pasado.

Luego se debe probar a presionar regresar en el navegador luego de haber terminado la sesión y verificar que no se puedan acceder a las opciones permitidas una vez autenticado.

Se procede a verificar si existe un mecanismo interno de control de sesiones. Para esto se guarda el valor de la cookie y si se restablece luego de ejecutar la opción de cierre de sesión. Una vez más se prueba acceder a las funciones de la aplicación sin autenticación.

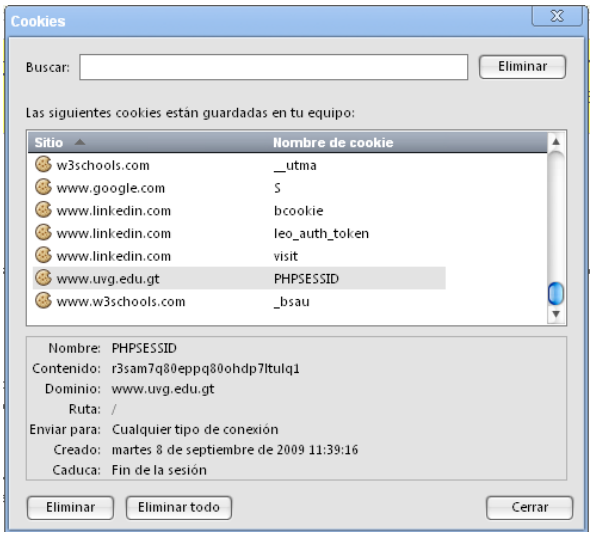
El último factor que se debe revisar es el tiempo de cerrado de sesión automático. Si existe, se debe revisar que este se esté cumpliendo y de igual manera la manera en la que se realiza.

c. Resultados.

1) Cookies. La aplicación no expira las cookies al salir de la sesión, simplemente las restablece:

```
Set-Cookie: PHPSESSID=nfjv8ctod3j3n4r2ffuq3ms3n2; path=/
Set-Cookie: PHPSESSID=r3sam7q80eppq8Oohdp7ltulq1; path=/
```

Ilustración 4: Cookies del portal de estudiantes



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

- No la estableció de forma segura utilizando el parámetro “Secure”
- 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. Al realizar la prueba de restablecer la cookie al valor de inicio de sesión cuando ésta se cierra, no permitió seguir autenticado.

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

3) Cerrado de sesión automático. Actualmente no existe ningún mecanismo de cerrado de sesión automático por tiempo, pero la sesión sí termina cuando se cierra el navegador, según la información provista por el entrevistado de informática.

d. Recomendaciones. Establecer una fecha de expiración de la cookie apropiada para la aplicación. Eliminar o alterar la cookie correctamente al terminar una sesión autenticada. Es recomendable, y una buen práctica, implementar un tiempo máximo válido de una sesión inactiva para que ésta no pueda ser aprovechada por algún usuario malicioso.

5. Pruebas para autenticación de factores múltiples

a. Descripción. Autenticación es la acción o proceso de verificar que una persona o usuario es quien dice ser. Esto se puede lograr mediante la verificación de cualesquiera de los siguientes factores:

- Algo que el usuario sabe – como una contraseña o PIN,
- Algo que el usuario tiene – como una llave o tarjeta inteligente,
- Algo que el usuario es – como características biométricas, huella digital, cara, ojos, etc. (SafeNet, 2008)

En la época actual los robos de identidad son comunes y fáciles de perpetrar por lo que es vital que la identidad digital de una persona pueda ser confiable siempre. Para lograr esto se debe aumentar la dificultad de autenticación para que el usuario pruebe que es quien dice ser más allá de cualquier duda. Entre más factores se utilicen para autenticar a una persona, más será la confianza que se tendrá en ella y en el esquema de autenticación. (SafeNet, 2008)

b. Procedimiento. El objetivo de esta prueba es comprobar la fortaleza del esquema de autenticación múltiple frente amenazas complejas como:
   1) Robo de credenciales (mediante Phishing, escuchas en la red, etc.),
   2) Uso de credenciales débiles (adivinado de contraseña por fuerza bruta),
   3) Ataques a la sesión (robo o fijación de sesión),
   4) Ataques por troyanos o malware,
   5) Reutilización de contraseñas.

c. Resultados. La aplicación no cuenta con un sistema de autenticación de factores múltiples por lo que la realización de esta prueba no aplica.

d. Recomendaciones. No hay recomendaciones

F. Pruebas de gestión de sesiones

1. Pruebas para el esquema de gestión de sesiones

a. Descripción. Las cookies normalmente son utilizadas para implementar un esquema de manejo de sesiones. Cuando un usuario accede a una aplicación que necesita mantener un registro de las acciones y la identidad de un usuario a través de múltiples peticiones, una cookie (o en ocasiones más de una) es generada por el servidor y enviada al cliente. El cliente luego enviará la cookie al servidor en todas las conexiones siguientes hasta que la cookie expire o sea destruida. La data guardada en la cookie puede proveer al servidor con una gran cantidad de información sobre quién es el usuario, qué acciones ha hecho hasta ahora, y cuáles son sus preferencias, etc. Logrando de esta manera crear un esquema orientado a estados a partir del protocolo HTTP. (OWASP.org, 2008)

Las cookies HTTP, en ocasiones conocidas también como cookies web o simplemente cookies, son porciones de texto enviadas por el servidor a un navegador web y luego intercambiadas entre el servidor y el navegador. Son utilizadas para autenticar, rastrear y mantener información específica sobre los usuarios. (Yeary, 2007)

b. Procedimiento. Esta prueba se realizará de caja blanca, se entrevistará al equipo de informática para determinar el estado y robustez del esquema de gestión de sesiones.

c. Resultados. Se entrevistó a un representante de IT encargado del desarrollo de la aplicación para obtener información sobre las cookies y el manejo de sesiones:

- Las cookies son generadas aleatoriamente por la aplicación comercial sobre la que se basa el sistema.
- El identificador de sesión tiene 31 caracteres de longitud.
- Tiempo de expiración de la sesión actualmente no está implementado.
- Configuración de la cookie
    - No persistente: Si. Vence cuando se cierra la sesión del explorador.
    - Segura: No. Es enviada sobre un canal seguro, pero no tiene el atributo "secure".
    - HTTPOnly: No. Es posible leerla mediante un script.

d. Recomendaciones. Modificar el manejo de cookies para que contengan los atributos de "secure" y HTTPOnly. Implementar un esquema de expiración de las cookies.

2. Pruebas para fijación de sesión

a. Descripción. Cuando una aplicación no renueva la cookie luego de un inicio de sesión exitoso, es posible que se encuentre una vulnerabilidad de fijación de sesión y se fuerce a un usuario a utilizar una cookie conocida por el atacante. Las vulnerabilidades de fijación de sesión ocurren cuando:

- Una aplicación web autentica a un usuario sin antes invalidar el identificador de sesión existente.
- Un atacante es capaz de forzar un identificador de sesión conocido a un usuario de tal forma que una vez se autentique, el atacante puede tener acceso a la sesión autenticada.

b. Procedimiento. Se realizará una prueba en la que se fuerce a utilizar un identificador de sesión a un usuario que se desea autenticar y se determinará si utilizando el mismo identificador se tiene acceso autenticado a la aplicación.

c. Resultados. Se determinó mediante las pruebas realizadas que la aplicación es vulnerable a la fijación de sesiones. Se realizó una prueba de contexto en la que se envió una sesión específica a la aplicación, perteneciente a otro usuario, y luego de que se recibió la respuesta positiva de la aplicación ambos usuarios se encontraban autenticados con el mismo usuario. Esto significa que un atacante podría por medio de ingeniería social u otros ataques, obligar a un usuario a utilizar una sesión y tener acceso total a sus privilegios.

d. Recomendaciones. Implementar un mejor esquema de control de sesiones de tal forma que se invaliden las sesiones al salir de la sesión y al iniciar una sesión autenticada. Además revisar los atributos de las cookies para que éstas sean seguras y puedan ser leídas únicamente por el navegador y no mediante scripts.

3. Pruebas para CSRF

a. Descripción. CSRF (*"Cross Site Request Forgery"* en inglés) es un vector de ataque que le permite a un atacante enviar peticiones HTTP arbitrarias desde un usuario víctima. El escenario típico involucra a una víctima que ha establecido un cierto nivel de privilegios con un sitio objetivo, y esto le permite al atacante iniciar acciones para las que no tiene autorización. (Shiflett, 2004)

La falsificación de peticiones cruzadas a un sitio explota la confianza que existe entre un sitio y un usuario en particular. El objetivo del ataque es el sitio, y el usuario es tanto la víctima como el cómplice sin saberlo. Debido a que la petición es enviada por el usuario víctima y no el atacante, es muy difícil determinar si una petición en particular representa un ataque CSRF o no. (Shiflett, 2004)

b. Procedimiento. La prueba consiste en encontrar un sitio que contenga una acción que requiera de privilegios para utilizarla. Luego de esto se debe crear una página HTML que tenga la petición HTTP al sitio con una acción restringida. Luego acceder a la aplicación para tener una sesión válida y autenticada.

c. Resultados. No se encontró ningún mecanismo en la aplicación diseñado para evitar la falsificación cruzada de petición a un sitio (CSRF).

d. Recomendaciones. Implementar algún mecanismo para evitar el ataque de CSRF como la utilización de peticiones POST en lugar de GET en el HTML o asegurar que las peticiones provengan de una forma que el usuario ya solicitó y que no simplemente realizó la petición.

G. Pruebas de autorización

1. Pruebas para saltarse el esquema de autorización

a. Descripción. Autenticación es el mecanismo por el cual los sistemas identifican de forma segura a sus usuarios. En contraste, autorización es el mecanismo por el cual un sistema determina qué nivel de acceso tiene un usuario autenticado en particular. Ambos sistemas dependen íntimamente uno del otro ya que la autenticación asegura que el usuario es quién dice ser, para que posteriormente la autorización dicte los permisos correctamente a los recursos que el sistema debe asegurar. (Thompson & Gross, 1997)

b. Procedimiento. Probar las diferentes funciones que requieren privilegios en la aplicación tratando de responder a las siguientes preguntas:

- ¿Es posible acceder aún si el usuario no está autenticado?
- ¿Es posible acceder después de cerrar la sesión?

- ¿Es posible acceder a funciones que deberían ser accesibles solo para un usuario diferente?

c. Resultados. Se determinó que es posible mediante la utilización de una llamada a una función que simplemente está escondida, alterar datos para los que se debería autorizar los cambios. Esto se logró enviando una petición de la forma que lo hacen los maestros, pero con una sesión de alumno normal iniciada. Al enviar la petición la aplicación responde de forma positiva y efectivamente los cambios en las notas fueron realizados.

La petición utilizada únicamente fue alterada para contener más datos en la cookie:

```
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
```

y lo que se envió a la aplicación fue:

```
{"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\": \"\"}]"}
```

d. Recomendaciones. El esquema de la aplicación debería modificarse para que las opciones, como la de cambio de nota, no estén simplemente ocultas o no disponibles para un alumno, sino que también (y aparte de estar escondidas) estén validadas para que una persona sin privilegios (aún estando autenticada como usuario válido) no tenga autorización para realizar estos cambios.

H. Pruebas de la lógica del negocio

1. Comprobación de la lógica de negocio

a. Descripción. La comprobación de errores en la lógica del negocio en una aplicación Web dinámica y multifuncional requiere pensar en formas poco convencionales. Si el mecanismo de autenticación de una aplicación está diseñado para que se ejecuten los pasos uno, dos y tres en ese orden para autenticarse, ¿qué pasa si del paso uno directo ejecuta el paso 3? Como este ejemplo hay muchos otros que se pueden formular pero lo que se mantiene constante es que uno debe "pensar en formas poco convencionales". Este tipo de vulnerabilidad no puede ser detectado por un escáner de vulnerabilidades y se basa en las aptitudes y capacidades creativas del que ejecuta la prueba. Además, este tipo de vulnerabilidad es el más difícil de probar y detectar, pero al mismo tiempo, usualmente resulta ser muy perjudicial para la aplicación en caso de ser explotado. (OWASP.org, 2008)

b. Procedimiento. Se recorrerán las funcionalidades principales del portal dónde se manejan datos y se intentará realizar acciones incoherentes, sin sentido, o que vayan en contra de lo que el sistema espera que el usuario realice.

c. Resultados.

- Estudiantes
  - Auxiliaturas
    - Permite editar ciertas opciones de los cursos de auxiliaturas antiguas.
  - Ayuda financiera
    - No se encontraron errores en la lógica del negocio
  - Mapa curricular
    - No se encontraron errores en la lógica del negocio
  - Consulta de notas
    - No se encontraron errores en la lógica del negocio

    - Consulta de saldo
        - No se encontraron errores en la lógica del negocio
- Docentes
    - Registro de notas
        - No se encontraron errores en la lógica del negocio

d. Recomendaciones. Corregir los permisos que tiene un auxiliar y terminar el proceso de asignación de una auxiliatura de forma que el auxiliar ya no pueda realizar ningún cambio.

I. Pruebas de validación de datos

1. Pruebas de XSS almacenado

a. Descripción. XSS (*"Cross Site Scripting"* en inglés) es una técnica de ataque que involucra devolver código sometido por el atacante al navegador web de un usuario. El código que el atacante envía está escrito generalmente en HTML/JavaScript, pero también podría extenderse a VBScript, ActiveX, Java, Flash, y cualquier otra tecnología soportada por el navegador. (Auger, 2009)

El ataque de XSS almacenado es cuando la data en la que se planta el ataque es almacenada en el servidor para luego ser desplegada a otros usuarios. Este tipo de ataque puede volver de víctima a los usuarios de la aplicación web. (Perrin, 2008)

Una aplicación tiene XSS almacenado cuando:

- Datos que pudieran provenir de un atacante es guardado en una base de datos o algún otro lugar en el servidor.
- La información es recuperada de de la base de datos u otro lugar y desplegada en el navegador del usuario.
- La data no es filtrada ni validada adecuadamente. (Stolk & Benson, 2009)

b. Procedimiento. Primero se deben identificar las formas en las que el usuario realiza una entrada de texto. Luego es necesario realizar un análisis del código HTML. El objetivo final es determinar los métodos de validación de datos que utilizan y si es posible de alguna forma almacenar un script en un campo.

c. Resultados. Se encontraron varias instancias de cross site scripting almacenado.

Tabla 3: XSS Almacenado

Método	Url	Parametros vulnerables	Modulo
POST	http://192.168.2.12:80/demo/mod/teacher/registroNotas/actividadEstudianteSet.php	JSON [Comentarios]	Notas
POST	http://192.168.2.12:80/demo/mod/student/ayudaFinanciera/ayudaFinanciera.php	JSON [todos los parámetros]	Ayuda Financiera

d. Recomendaciones. Realizar una validación apropiada de los datos de entrada de las diferentes formas en donde el usuario o atacante puede ingresar texto con código malicioso.

2. Pruebas de XSS basado en flash

a. Descripción. Macromedia flash tiene su propio lenguaje para desarrollo llamado ActionScript. Es muy simple y similar a lenguajes como JavaScript, C, y PERL. Este lenguaje entre otras cosas es capaz de realizar animaciones complejas, simulaciones, y creación de juegos. Lo importante para XSS de este lenguaje es que posee la función getURL(). Esta función nos permite redirigir al usuario final del script a otra página web. Aunque el parámetro normalmente sea un URL como http://www.ejemplo.com, podríamos especificar un URL JavaScript en vez de éste como javascript:alert(document.cookie). (eyeonsecurity.net, 2002)

En el ejemplo anterior se ejecuta un comando que despliega una ventana de alerta con la cookie perteneciente al dominio donde está siendo almacenada la página con el documento flash. (eyeonsecurity.net, 2002)

b. Procedimiento. Determinar dónde existe algún objeto de flash dentro de la aplicación. Si se encuentra se deben revisar las funciones comunes como getURL() que permiten realizar XSS basado en flash.

c. Resultados. No existe ningún objeto flash dentro de la aplicación.

d. Recomendaciones. No hay recomendaciones.

3. Inyección LDAP

a. Descripción. LDAP (Lightweight Directory Access Protocol) es un estándar que las computadoras y dispositivos de red utilizan para acceder a información común sobre la red. La habilidad para proveer acceso sobre la red para datos no hace que LDAP se diferencie de otras docenas de protocolos estándares conocidos, pero hay un número de características que hacen que LDAP sea muy apropiado para acceder y actualizar muchos tipos de información común. (Donely, 2003)

Por ejemplo información sobre empleados puede ser guardada en un directorio de tal forma que la gente y las aplicaciones puedan localizar la información de sus contactos. Esta información puede contener correo electrónico, direcciones y números telefónicos o incluso data adicional que identifica a los empleados sin ambigüedad. (Donely, 2003)

La inyección LDAP es un ataque del lado del servidor, el cual puede permitir que información sensible sobre los usuarios y objetos representados en la estructura del LDAP sean revelados, modificados, o insertados. Esto se logra mediante la

manipulación de parámetros de entrada que luego son pasados a las funciones internas de búsqueda, adición o modificación. (OWASP.org, 2008)

b. Procedimiento. Determinar los puntos dónde es posible que se realice una petición que llegue hasta el directorio de LDAP y realizar ataques conocidos para poder insertar, borrar, o modificar registros, o saltarse el esquema de autenticación/autorización.

c. Resultados. No se encontró ninguna inyección de LDAP.

d. Recomendaciones. No hay recomendaciones.

4. Inyección XML

a. Descripción. XML (eXtensible Markup Language) es un lenguaje para documentos que contienen información estructurada. La información estructurada contiene tanto contenido (palabras, imágenes, etc.) como alguna información acerca de que rol juega ese contenido (por ejemplo contenido en la sección de título tiene un significado distinto que contenido en la sección de pie de página). Casi todos los documentos tienen alguna estructura. (Walsh, 1998)

Un lenguaje de marca (*"markup")* es un mecanismo para identificar las estructuras dentro de un documento. La especificación de XML define un estándar sobre como agregar marcas de estructura a los documentos. (Walsh, 1998)

La inyección XML es el ataque que consiste en inyectar un documento XML a una aplicación, si el analizador XML falla en hacer una validación apropiada del documento entonces los resultados de la prueba serán exitosos. (OWASP.org, 2008)

b. Procedimiento. Se debe buscar alguna forma donde se puede enviar un documento XML a la aplicación realizado de forma maliciosa para poder inyectar XML exitosamente.

c. Resultados. No se encontraron inyecciones XML.

d. Recomendaciones. No hay recomendaciones.

5. Inyección XPATH

a. Descripción. XPath es un lenguaje de expresiones que permite procesar valores conforme al modelo de data XDM (XPath Data Model). Éste modelo provee una representación de árbol de los documentos XML así como valores atómicos como números enteros, cadenas, valores booleanos, y secuencias que contengan tanto referencias a nodos en un documento XML como a elementos atómicos. (W3C, 2007)

El resultado de una expresión XPATH puede ser una selección de nodos de los documentos de entrada, o un valor atómico, o cualquier secuencia permitida por el modelo de datos. El nombre del lenguaje se deriva de su más distintiva cualidad, la expresión de rutas de directorio, que le provee medios jerárquicos para seleccionar nodos del árbol en un XML. (W3C, 2007)

La inyección XPath es una técnica de ataque usada para explotar sitios web que construyen consultas XPath a partir de entradas provistas por el usuario. (OWASP.org, 2008)

b. Procedimiento. Determinar los sitios y formas donde utilicen XML y XPath para realizar consultas mediante él y realizar una serie de ataques conocidos de inyección de XPath.

c. Resultados. No se encontraron inyecciones XPath.

d. Recomendaciones. No hay recomendaciones.

6. Inyección de código

a. Descripción. Inyección de código es el nombre general que se le da a varios tipos de ataque que dependen de la inserción de código que es interpretado por la aplicación. Este tipo de ataque puede ser realizado al agregar cadenas de caracteres en una cookie o valores en los argumentos del URI. Este ataque se aprovecha de una mala validación de entradas/salidas como por ejemplo:

- Clase de caracteres permitida.
- Formato de la data.
- Tamaño de la data esperado.
- Para inyecciones numéricas, su valor. (OWASP.org, 2009)

b. Procedimiento. En una cadena de consulta o petición a la aplicación se debe intentar incluir comandos para que sean procesados como parte de la petición o un archivo que se esté subiendo.

c. Resultados. No se encontraron inyecciones de código.

d. Recomendaciones. No hay recomendaciones.

7. Pruebas de desbordamiento de búfer

a. Descripción. Un búfer es un pedazo de memoria direccionado continuo, como un vector o un puntero en el lenguaje C. En C y C++ (y algunos otros lenguajes) no existe una verificación de límites automática lo que significa que un usuario puede escribir más allá del fin del búfer. (Grover, 2003)

En general, un desborde de búfer ocurre cuando un programa escribe más información en el búfer de la que tiene reservada en memoria. Esto le permite al atacante sobrescribir datos que controlan la ejecución del programa y ejecuta su propio código en lugar del código del proceso. (Ogorkiewicz & Frej, 2004)

b. Procedimiento. El procedimiento es probar para determinar si existen vulnerabilidades respecto a las diferentes sobrecargas (de búfer, de memoria heap, o de pila). La regla básica en una sobrecarga es que se le provee a la aplicación una entrada mucho más grande de la que espera y se observa cómo responde la aplicación a esta petición.

c. Resultados. No se encontraron desbordamientos de ningún tipo.

d. Recomendaciones. No hay recomendaciones.

8. Pruebas de HTTP Splitting/Smuggling

a. Descripción. La esencia del ataque de HTTP Splitting consiste en la capacidad de un atacante para enviar una única petición HTTP que fuerce al servidor web a formar una cadena de salida, que es interpretada por la víctima objetivo como dos respuestas HTTP en lugar de una sola respuesta. Este tipo de vulnerabilidad puede ser explotada para realizar muchos ataques basados en la aplicación web. (SecuriTeam, 2005)

Esta vulnerabilidad es relativamente nueva y puede ser utilizada para lanzar diferentes ataques como:

- XSS – Script de sitio cruzado
- Desfase cruzado de usuario – para robar usuarios y contraseña a un usuario específico
- Envenenamiento del cache web
- Secuestro de página
- Envenenamiento del cache del navegador (SecuriTeam, 2005)

Este tipo de ataques son generalmente realizados en aplicaciones web al inyectar caracteres inesperados o de forma maliciosa en las entradas de usuario que luego son generalmente usadas para una redirección 302 en el encabezado de “Location” o el de

“Set-Cookie”. Este ataque generalmente introduce caracteres de cambio de línea en sus muchas formas para poder alterar el encabezado de la página. (SecuriTeam, 2005)

El contrabando de peticiones HTTP (HRS: HTTP Request Smuggling en inglés) es una nueva técnica de hackers que elige como blanco a los dispositivos HTTP. Cada vez que una petición HTTP que se origina en un cliente pasa a través de más de una entidad que la procesa, hay una alta probabilidad de que estas entidades sean vulnerables a HRS. (Linhart, Klein, & Orrin, 2005)

Un ataque HRS envía múltiples peticiones HTTP especialmente diseñadas para causar que los dos dispositivos atacados vean diferentes grupos de peticiones, permitiendo al atacante contrabandear una petición a un dispositivo, sin que el otro esté al tanto de ello. (Linhart, Klein, & Orrin, 2005)

b. Procedimiento. Para la división de peticiones HTTP el primer paso es identificar las formas donde la entrada del usuario afecta de alguna forma el encabezado de la respuesta HTTP. Luego el que realiza la prueba debe preparar la petición HTTP correcta con caracteres como cambio de línea y regreso a inicio (CR/LF en sus múltiples formas) para dividir la respuesta.

c. Resultados. No se encontraron vulnerabilidades de división de peticiones o contrabando de peticiones.

d. Recomendaciones. No hay recomendaciones.

J. Pruebas de denegación de servicio

1. Bloqueando cuentas de usuario

a. Descripción. Este caso de denegación de servicio requiere que se revise si es posible que un atacante pueda bloquear una cuenta de usuario válida al tratar de iniciar sesión sin éxito repetidas veces. (OWASP.org, 2008)

b. Procedimiento. El primer paso es conseguir una cuenta de usuario válida. Luego se debe verificar si existe algún mecanismo de bloqueo de cuentas por un determinado número de intentos fallidos. Para hacer esto se debe probar un mínimo de quince veces seguidas iniciar sesión con credenciales falsas (contraseña incorrecta pero usuario válido) y comprobar si la aplicación indica que la cuenta ha sido bloqueada.

c. Resultados. La aplicación no cuenta con ningún método para el bloqueo de cuentas que se sospechen estén siendo atacadas. Como no posee ningún mecanismo para recuperación o recordatorio de contraseña tampoco se cuenta aquí con oportunidad para bloquear cuentas.

d. Recomendaciones. Es importante encontrar un balance entre seguridad y funcionalidad. Actualmente la aplicación se encuentra totalmente orientada a funcionalidad en este sentido ya que no existe ningún mecanismo para prevenir o detener un ataque de fuerza bruta. La mejor opción a seguir quizás es la que utiliza Google, pues combina lo mejor de la funcionalidad y la seguridad. Utiliza un CAPTCHA pero únicamente cuando el usuario ha tratado de iniciar sesión sin éxito un número de veces. Si el CAPTCHA es suficientemente bueno debería detener los ataques de fuerza bruta. Si no lo es quizás se podría implementar una política balanceada que permita que un usuario intente iniciar sesión un número lógico de veces.

2. Reserva de objetos especificada por usuario

a. Descripción. Si un usuario ingresa, ya sea de forma directa o indirecta, un valor que especificará cuantas instancias de un objeto crear en el servidor de la aplicación, y si el servidor no tiene políticas para establecer un límite superior aceptable, puede que sea posible dejar sin memoria al ambiente del servidor. (OWASP.org, 2008)

b. Procedimiento. Se deben buscar situaciones o formas web en las que se solicite un número que sea utilizado posteriormente para crear objetos. También pueden ser situaciones en las que no se pida el número de forma directa sino que vaya atado a una cualidad o incluso un simple botón de agregar por ejemplo podría utilizarse para agotar los recursos del servidor.

c. Resultados. Se determinó que la parte en la que probablemente se reserve memoria para objetos dentro del servidor es el ingreso de notas. La edición del curso permite crear actividades dentro de un curso de forma unitaria, o múltiple, sin embargo está validado para que sólo se puedan ingresar de 25 en 25 actividades.

Creando una petición POST específica en la que se agregaron aproximadamente 1200 actividades nuevas y enviando la petición al servidor se observó que mientras el servidor procesa la petición, la aplicación deja de responder. Responder esta petición toma alrededor de un minuto por lo que un ataque en el que se realiza esta petición de forma masiva o repetida podría resultar en la denegación de los servicios de la aplicación. Además, la petición aunque no haya sido realizada con éxito totalmente, sí agregó más de 1000 actividades al curso, por lo que luego de agregarlas el simple hecho de pedir que refresque los datos del curso representa una gran carga.

Se probó a realizar un proceso análogo enviando un detalle de nota con un comentario que mide aproximadamente 670 páginas en Microsoft Word pero no tomó mucho tiempo por lo que probablemente este texto es simplemente cortado antes de entrar a la base de datos o guardado como tal, aunque si es cortado este límite no se descubrió pues las aplicaciones utilizadas para las pruebas no soportaban peticiones ni respuestas más grandes.

d. Recomendaciones. Establecer un límite máximo en la cantidad de actividades que puede tener un curso y en general, cualquier campo u objeto debería estar limitado a parámetros que sean aceptables y brinden un balance entre seguridad y funcionalidad.

3. Escritura a disco de datos suministrados por usuario

a. Descripción. Con esta prueba se pretender probar si es posible causar una denegación de servicios mediante llenar el disco duro de la aplicación. Esto se logra mediante el envío de peticiones extremadamente grandes que no sean chequeadas y simplemente son procesadas, o que sí sean chequeadas pero de todas formas guarda la petición inválida en sus registros. (OWASP.org, 2008)

b. Procedimiento. Esta prueba será realizada en dos partes, una de caja negra, y otra de caja blanca. En la primera, se intentará determinar dónde se le envía información a la aplicación y si esta la guarda sin establecer límites máximos aceptables. En la segunda se pedirá información a los encargados de la aplicación para revisar su proceso de manejo de registros de la aplicación para determinar si es posible acabarse de esta manera los recursos.

c. Resultados. Según la información brindada por los representantes de IT de la Universidad del Valle de Guatemala, la aplicación no almacena datos a disco duro de forma directa sino a través de una base de datos (Microsoft SQL 2005). Sin embargo, mediante la prueba de suministrar un campo de comentarios de un tamaño verdaderamente grande sabemos que la aplicación no valida el tamaño del comentario por lo que creando una gran cantidad de actividades y llenándolas con comentarios de aproximadamente 3mb podría llenarse un disco duro de 500GB creando 1000 actividades de 20 alumnos en 5 cursos.

También indicaron que sí tienen registro activado, pero éste se almacena en una partición distinta a la de la aplicación y el sistema operativo del servidor por lo que aún si se logra llenar esa partición de registros no debería representar un problema grave para la arquitectura del sistema en general.

d. Recomendaciones. Establecer límites máximos para todos los campos dentro de la aplicación para evitar que ataques de este tipo, aunque tardados, puedan representar una amenaza para la aplicación.

4. Almacenamiento excesivo en la sesión

a. Descripción. Esta prueba pretende determinar si es posible almacenar demasiada información en el objeto de sesión de un usuario para así acabar con los recursos del servidor web. Se deben tomar precauciones al almacenar demasiados datos en una sesión de usuario, como respuestas recibidas de la base de datos. Este problema es potenciado aún más si la sesión también guarda información incluso antes de iniciar sesión, pues un usuario puede atacar esta vulnerabilidad sin siquiera estar autenticado. (OWASP.org, 2008)

b. Procedimiento. Ésta es una prueba sumamente difícil de realizar de forma caja negra por lo que se realizará de caja blanca. Se preguntará al equipo de desarrollo e informática qué es lo que almacenan en los objetos de sesión de los usuarios y se analizará si es un manejo adecuado de información o podrían ser vulnerables a este ataque.

c. Resultados. El entrevistado de informática reportó que dependiendo del rol del usuario se guarda diferente información como nombres, contraseñas (cifradas), dirección IP del usuario, e identificadores. En general ninguno de los datos que se almacenan puede crecer dinámicamente por lo que no es posible agotar los recursos de memoria del servidor mediante la explotación de la sesión.

d. Recomendaciones. No hay recomendaciones.

K. Comprobación de servicios web

1. Probando WSDL

a. Descripción. WSDL ("*Web Services Description Language"* por sus siglas en inglés) es un lenguaje que sirve para describir las capacidades de un

servicio web. Fue propuesto por IBM y Microsoft y combina lo mejor de NASSL (*"Network Accesible Services Languange"* por sus siglas en inglés) de IBM y SOAP de Microsoft. WSDL está basado en XML y ayuda a mejorar interoperabilidad entre aplicaciones, sin importar el protocolo o esquema de codificado que utilicen. Esencialmente, un documento WSDL describe como invocar a un servicio, y provee información sobre la data que se intercambia, la secuencia de mensajes para una operación, restricciones del protocolo, y la dirección del servicio. Un WSDL define servicios como una colección de nodos finales, pero separa la definición abstracta de la implementación concreta. (Srinivas, 2001)

Servicios web se refieren a una manera de acceder a servicios sobre la web. Son una nueva forma de aplicación web. Son aplicaciones auto-contenidas, auto-descriptivas, modulares que pueden ser publicadas, localizadas, e invocadas a través de la web. Realizan funciones que pueden ir de peticiones sencillas hasta procesos complejos de negocios. (Gardner, 2001)

La revolución de servicios computarizados en las compañías dió auge a sistemas de cómputo aislados. Estas compañías tenían software desarrollado y personalizado para sus necesidades específicas. Sin embargo, las fusiones, compras y crecimiento de estas mismas compañías notaron la creciente necesidad de compartir información contenida en estos sistemas aislados de forma eficiente. Y así nació la necesidad de conectar estos sistemas de forma directa mediante el uso de servicios web. (Ananthamurthy, 2002)

b. Procedimiento. Luego de tener una lista de los servicios web que están corriendo en el servidor, se debe probar en ellos si es posible invocar a funciones no visibles desde el portal. También se debe intentar saltar el esquema de autenticación si hubiese alguno y determinar que es posible hacer sin tener credenciales, o credenciales apropiadas para cada operación.

c. Resultados. Se realizaron peticiones a los diferentes servicios web con sus diferentes métodos de invocación y se observó que no hay ninguna forma

de autenticación. Se puede realizar una operación sobre el servicio web "StudentsGeneral.asmx?wsdl" utilizando el método wsAcEstudianteInfo y consecuentemente enumerar todos los usuarios del sistema. Bastó con probar por prueba y error para determinar que los identificadores de usuario inician en el 100,000 y están ordenados alfabéticamente. De la misma manera para los demás servicios web es posible solicitar las notas de los estudiantes, el detalle de un curso, e incluso alterar las notas utilizando un objeto JSON de la forma


```
{ "ID_Actividad":"31379", "ID_Estado":"3", "NombreEstado":"Publicado", "affected":"[
{\"Activo\": \"1\", \"carnet\":\"05229\", \"EstadoCurso\":
\"ASIGNADO\",\"ID_ActividadEstudiante\": \"445032\", \"ingConsola\":\"192.168.10.3\",
\"ingFecha\": \"2009-08-21T18:24:00\",\"ingUsuario\":
\"wdelaroca@uvg.edu.gt\",\"NombreEstudiante\": \"JUAREZ BARRIOS, MARIA PRISCILA\",
\"Omitir\": \"0\", \"Porcentaje\": \"84.00000\",\"updConsola\": \"192.168.10.3\",
\"updFecha\":\"2009-08-26T13:39:00\", \"updUsuario\":\"wdelaroca@uvg.edu.gt\" } ]" }
```


d. Recomendaciones. Implementar algún mecanismo que permite autenticar en los servicios web que las peticiones vengan únicamente del sitio confiado y además, para evitar un ataque del tipo "hombre en el medio" enlazarlo de alguna manera con las sesiones activas del servidor web.

2. Comprobación de XML a nivel de contenido

a. Descripción. Los ataques a nivel de contenido tienen como objetivo el servidor que ejecuta el servicio web, y cualquier aplicación que sea utilizada por el servicio, como servidores web, bases de datos, servidores de aplicación, sistemas operativos, etc. Los vectores de este ataque incluyen inyección SQL, inyección XPath, desbordamiento de búfer e inyección de comandos. (OWASP.org, 2008)

b. Procedimiento. Se utilizará la herramienta wsdl dig de McAfee para comprobar las vulnerabilidades del servicio web en XML a nivel del contenido. Se harán pruebas para determinar si es posible realizar XSS, inyección de XPath, e inyección de SQL.

De forma manual, se enviará un mensaje extremadamente grande al servicio web para determinar si es susceptible a algún desbordamiento de búfer.

c. Resultados.

- Inyección de SQL. Los siguientes servicios son vulnerables en uno o más métodos a inyección de SQL
  - Teachers/RegistroDeNotas
  - Teachers/TeachersGeneral
  - Students/StudentsGeneral
  - Students/AyudaFinanciera
  - Students/CuentaCorriente
  - Students/Inscripción
  - Students/EvaluaciónDocente
  - Students/Parqueos
  - General
  - Seguridad

4) Desbordamiento de Búfer. No hay ninguna vulnerabilidad reportada de desbordamiento de búfer de la que sufra la arquitectura de la aplicación (Microsoft IIS versión 6.0). También se determino que la capacidad máxima que procesa el servidor es aproximadamente 5 megabytes.

d. Recomendaciones. Limitar aún más las peticiones al servidor de servicios web ya que 5MB sigue siendo muy holgado. Además se deben validar los valores de entrada del servicio web para que no sea susceptible a la inyección de código de SQL.

3. Adjuntos SOAP maliciosos

a. Descripción. SOAP (*"Simple Object Access Protocol"* por sus siglas en inglés) fue creado en 1998 por Dave Winer, Don Box, Bob Atkinson, y

Mohsen Al-Ghosein con soporte de Microsoft. Es una forma de comunicar dos programas corriendo en diferentes sistemas operativos a través de internet, el protocolo HTTP y XML. (SearchSOA.com, 2008)

SOAP consiste de tres partes:

- Sobre (*"Envelope"* en inglés): construye y define un sistema estructural para expresar qué está en el mensaje, quién debe manejarlo, y si es opcional o mandatorio.
- Reglas de codificación: define un mecanismo de serialización que puede ser utilizado para intercambiar instancias de tipos de datos específicos a cada aplicación.
- Representación RPC que define una convención que puede ser utilizada para representar llamadas a procesos remotas y sus respuestas. (Box, et al., 2000)

Los archivos de tipo binario, incluyendo ejecutables y documentos, pueden contener malware, que puede ser enviado al servidor de muchas maneras utilizando un servicio web; puede ser enviado como un adjunto, o utilizando DIME ("Direct Internet Message Encapsulation" por sus siglas en inglés). Un atacante puede crear un documento XML (mensaje SOAP) que se mande al servicio web que contiene malware como adjunto. (OWASP.org, 2008)

b. Procedimiento. Determinar si existe algún servicio web que permita enviar archivos adjuntos. Una vez se encuentre un servicio que acepte adjuntos se debe intentar enviar un archivo adjunto que tenga una firma de virus conocida (pero que no lo sea) como los archivos de EICAR.

c. Resultados. No se encontró ningún servicio web que acepte archivos adjuntos.

d. Recomendaciones. No hay recomendaciones.

III. RESULTADOS

A. Valoración del riesgo

A continuación se presentan los criterios utilizados para valorar los niveles de riesgo de una vulnerabilidad específica.. Los niveles de severidad del riesgo fueron determinados con base en la metodología de valoración de riesgos OWASP.

$$Riesgo = Probabilidad\ de\ ocurrencia * Impacto$$

Ecuación 1: Fórmula para determinación de riesgo (OWASP.org, 2008)

1. **Valoración de la probabilidad de ocurrencia.** Para valorar la probabilidad de ocurrencia se utilizaron factores en una escala de uno a nueve:

- Nivel de conocimientos necesarios para realizar el ataque
    - Sin conocimientos técnicos (1)
    - Con algunos conocimientos técnicos(3)
    - Usuario avanzado (4)
    - Con conocimientos de redes y programación (6)
    - Con conocimientos de intrusiones de seguridad (9)
- Motivación para realizar el ataque
    - Poca motivación, ninguna recompensa (1)
    - Posible recompensa (4)
    - Recompensa alta (9)
- Oportunidad de encontrar y explotar la vulnerabilidad
    - Ningún acceso (1)
    - Acceso limitado (4)
    - Acceso total (9)
- Número de posibles atacantes

  - Desarrolladores y administradores del sistema (2)
  - Usuarios de la intranet (4)
  - Socios (5)
  - Usuarios autenticados (6)
  - Usuarios anónimos de Internet (9)
- Facilidad de descubrimiento de la vulnerabilidad
  - Imposible (1)
  - Difícil (3)
  - Fácil (5)
  - Existen herramientas automatizadas (9)
- Facilidad de explotación
  - En teoría es posible explotarla, pero muy difícil (1)
  - Es difícil (3)
  - Es fácil (5)
  - Existen herramientas automatizadas (9)
- Conocimiento de la vulnerabilidad
  - Desconocida (1)
  - Oculta (4)
  - Obvia (6)
  - Pública (9)
- Detección de la intrusión
  - Se detecta activamente en la aplicación (1)
  - Se registra y se revisa (3)
  - Se registra pero no se revisa (8)
  - No se registra (9) (OWASP.org, 2008)

2. **Valoración del impacto.** Para valorar el impacto de la vulnerabilidad se utilizaron factores en una escala de uno a nueve:

- Pérdida de confidencialidad
  - Revelación mínima de datos no sensibles (2)

  - Revelación mínima de datos críticos o amplia de datos no sensibles (6)
  - Amplia revelación de datos críticos o todos los datos revelados (9)
- Pérdida de integridad
  - Mínima alteración, datos ligeramente corruptos (1)
  - Mínimos datos seriamente corruptos (3)
  - Gran volumen de datos ligeramente alterados (5)
  - Gran cantidad de datos seriamente dañados o todos los datos totalmente corruptos (9)
- Pérdida de disponibilidad
  - Mínimo número de servicios secundarios interrumpidos (1)
  - Mínimo número de servicios primarios interrumpidos (5)
  - Gran número de servicios secundarios interrumpidos (5)
  - Gran número de servicios primarios interrumpidos (7)
  - Todos los servicios interrumpidos (9)
- Control de responsabilidad
  - Totalmente trazable (1)
  - Es posible que se pueda trazar (7)
  - Completamente anónimo (9)
- Daño financiero
  - Menor al coste de arreglar la vulnerabilidad (1)
  - Leve efecto en el beneficio anual (3)
  - Efecto significativo en el beneficio anual (7)
  - Bancarrota (9)
- Daño a la reputación
  - Daño mínimo (1)
  - Pérdida de las cuentas principales (4)
  - Pérdida del buen nombre (5)
  - Daño sobre la marca (9)
- Cumplimiento de leyes
  - Violación leve (2)

- Clara violación (5)
- Violación prominente (7)

- Violación de la privacidad
  - Un individuo (3)
  - Cientos de personas (5)
  - Miles de personas (7)
  - Millones de personas (9)

3. Determinación del riesgo. Luego de evaluar los factores anteriormente mencionados se realizó un promedio del puntaje por variable (impacto y probabilidad de ocurrencia) y se utilizó la siguiente escala para la determinación de la severidad:

- Bajo: de 0 a 3
- Medio: de 3 a 6
- Alto: de 6 a 9

Al tener la severidad de cada variable se utilizó la siguiente tabla para determinar la valoración final del riesgo:

Tabla 4: Valoración del riesgo

Impacto	ALTO	Medio	Alto	Crítico
	MEDIO	Bajo	Medio	Alto
	BAJO	< Bajo	Bajo	Medio
		BAJO	MEDIO	ALTO
	Probabilidad de ocurrencia			

4. Tabla de resultados

Tabla 5: Resultados gestión de la configuración

	Identificador	Nombre	Vulnerabilidades	Riesgo	Recomendación
Pruebas de la Gestión de Configuración	OWASP-CM-001	Pruebas de SSL/TLS	Acepta niveles de cifrado inseguros	bajo	Configurar el servidor para que no soporte niveles de cifrado inseguros (< 56bits)
	OWASP-CM-003	Prueba de la Gestión de la configuración de la infraestructura	Vulnerabilidades en las versiones de software actuales (PHP y Apache)	Medio	Mantener al día las actualizaciones y parches de los sistemas de la infraestructura del portal
	OWASP-CM-005	Prueba del manejo de extensiones de archivos	Un archivo de tipo .bak encontrado	< bajo	Eliminar el archivo.
	OWASP-CM-007	Interfaces de administración de aplicación e infraestructura	--	N/A	--

Tabla 6: Resultados autenticación

	Identificador	Nombre	Vulnerabilidades	Riesgo	Recomendación
Pruebas de Autentificación	OWASP-AT-001	Transporte de credenciales sobre un canal cifrado	Se encontró una forma de entrar al portal sin utilizar un canal cifrado	Alto	Eliminar el acceso mediante portal.uvg.edu.gt o advertir sobre su uso
	OWASP-AT-003	Pruebas para cuentas de usuario adivinables (Diccionario)	--	N/A	--
	OWASP-AT-005	Pruebas para saltarse el esquema de autentificación	--	N/A	--
	OWASP-AT-007	Pruebas para el cierre de sesión y la gestión del cache del explorador	No hay expiración de cookies ni se restablece el valor inicial.	Medio	Establecer expiración de las cookies y modificarlas correctamente al cierre de sesión
	OWASP-AT-009	Pruebas sobre autentificación de múltiples factores	--	N/A	--

Tabla 7: Resultados gestión de sesión

	Identificador	Nombre	Vulnerabilidades	Riesgo	Recomendación
Gestión de sesión	OWASP-SM-001	Prueba de Gestión del esquema de sesión	Atributos de las cookies no seguros y no existe expiración automática.	Alto	Implementar un mecanismo de expiración automático y establecer los valores adecuados para las cookies
	OWASP-SM-003	Pruebas de la Fijación de la sesión	Es posible fijar sesión	Alto	Se deben revisar el esquema de sesión y restablecer la sesión al iniciar sesión exitosamente
	OWASP-SM-005	Pruebas de CSRF	No hay mecanismos para evitar los ataques de CSRF	Alto	Implementar un mecanismo que prevenga o aminore este riesgo.

Tabla 8: Resultados autorización

	Identificador	Nombre	Vulnerabilidades	Riesgo	Recomendación
Pruebas de Autorización	OWASP-AZ-002	Prueba para saltarse el esquema de autorización	Es posible burlar el esquema de autorización	Crítico	Es posible realizar funciones a las que no se tiene autorización mediante el envío de peticiones directas al servidor

Tabla 9: Resultados lógica del negocio

	Identificador	Nombre	Vulnerabilidades	Riesgo	Recomendación
Comprobación de la lógica de negocio	OWASP-BL-001	Comprobación de la lógica de negocio	Es posible modificar ciertos valores en auxiliaturas antiguas	< bajo	Revisar la forma de cerrar apropiadamente un curso

Tabla 10: Resultados validación de datos

	Identificador	Nombre	Vulnerabilidades	Riesgo	Recomendación
Pruebas de validación de datos	OWASP-DV-002	Prueba de XSS almacenado	Existen vulnerabilidades de XSS	Alto	Validar las entradas para prevenir este tipo de ataques
	OWASP-DV-004	Prueba de Cross site scripting basado en Flash	--	N/A	--
	OWASP-DV-006	Inyección LDAP	--	N/A	--
	OWASP-DV-008	Inyección XML	--	N/A	--
	OWASP-DV-010	Inyección XPATH	--	N/A	--
	OWASP-DV-012	Inyección de código	--	N/A	--
	OWASP-DV-014	Desbordamiento de búfer	--	N/A	--
	OWASP-DV-016	Pruebas de división de cabeceras HTTP	--	N/A	--

Tabla 11: Resultados denegación de servicios

	Identificador	Nombre	Vulnerabilidades	Riesgo	Recomendación
Pruebas de Denegación de Servicios	OWASP-DS-002	Bloqueo de cuentas de usuario	No hay mecanismos de bloqueo de cuentas	bajo	Implementar algún sistema que evite un ataque de fuerza bruta sin bloquear a un usuario legítimo
	OWASP-DS-004	Reserva de objetos especificada por usuario	Es posible denegar acceso al servidor temporalmente	Alto	Establecer límites máximos de actividades en los cursos y cualquier campo que determine una cantidad
	OWASP-DS-006	Escritura a disco de datos suministrados por usuario	Es posible agotar los recursos de disco de la base de datos	Medio	Establecer un límite máximo para tamaño de comentarios u otros campos
	OWASP-DS-008	Almacenamiento excesivo en la sesión	--	N/A	--

Tabla 12: Resultados servicios web

	Identificador	Nombre	Vulnerabilidades	Riesgo	Recomendación
Pruebas de Servicios Web	OWASP-WS-002	Pruebas de WSDL	No hay autenticación	Crítico	Implementar autenticación autorización en los servicios web.
	OWASP-WS-004	Comprobación de XML a nivel de contenido	Los servicios son vulnerables a inyección SQL	Alto	Validar las entradas en los servicios así como en el portal.
	OWASP-WS-006	Adjuntos SOAP maliciosos	--	N/A	--

IV. CONCLUSIONES Y RECOMENDACIONES

- La evaluación reveló que existen defectos de nivel crítico y alto en el portal. Existen un total de:
    - 2 vulnerabilidades con severidad crítica.
    - 7 vulnerabilidades con severidad alta.
    - 3 vulnerabilidades con severidad media.
    - 4 vulnerabilidades con severidad baja.
- Deben realizarse acciones inmediatas respecto a los elementos marcados como “alto” y “crítico” dentro de los resultados.
- Los elementos con un riesgo medio deben tratarse a corto plazo (1-2 meses).
- Los elementos con riesgo bajo deben tratarse a largo plazo (de dos meses a un año).
- Una vez remediados los defectos críticos, altos y medios debe realizarse otra prueba para determinar que hayan sido reparados en su totalidad.
- La seguridad es un proceso. Luego de la segunda evaluación de seguridad deben realizar evaluaciones periódicas para asegurar el funcionamiento seguro del portal en el tiempo.
- Se recomienda eliminar todas las funcionalidades, archivos de respaldo, aplicaciones, y elementos del portal que no sean necesarios. En ocasiones una brecha de seguridad puede ocurrir a través de una aplicación/herramienta que no esté en uso.
- Deben establecer ventanas de mantenimiento en las que actualicen el software y aplicaciones utilizadas para el portal.
- Validar todas las entradas y salidas de la aplicación para evitar ataques de inyección.

V. BIBLIOGRAFÍA

- Allen, J. M. (22 de 09 de 2007). OS and Application Fingerprinting Techniques.

- Ananthamurthy, L. (22 de 10 de 2002). *Introduction to Web Services*. Recuperado el 02 de 10 de 2009, de developer.com: http://www.developer.com/services/article.php/1485821/Introduction-to-Web-Services.htm

- Auger, R. (27 de 09 de 2009). *Cross Site Scripting*. Recuperado el 29 de 09 de 2009, de Webappsec: http://projects.webappsec.org/Cross-Site-Scripting

- Belani, R. (14 de 04 de 2004). *Basic Web Session Impersonation*. Recuperado el 22 de 09 de 2009, de SecurityFocus: http://www.securityfocus.com/infocus/1774

- Boncella, R. (02 de 2004). Secure Sockets Layer (SSL). Topeka, KS, United States.

- Box, D., Ehnebuske, D., Kakivaya, G., Layman, A., Mendelsohn, N., Nielsen, H. F., y otros. (08 de 05 de 2000). *Simple Object Access Protocol (SOAP) 1.1* . Recuperado el 02 de 10 de 2009, de World Wide Web Consortium: http://www.w3.org/TR/2000/NOTE-SOAP-20000508/

- Bozidar, S. (06 de 06 de 2008). *Understanding Penetration Testing Methodology*. Recuperado el 05 de 10 de 2009, de Articlesbase.com: http://www.articlesbase.com/security-articles/understanding-penetration-testing-methodology-440101.html

- Donely, C. (2003). *Introduction to LDAP*. Manning Publications Co.

- eyeonsecurity.net. (25 de 08 de 2002). *Bypassing JavaScript Filters, the Flash! Attack*. Recuperado el 29 de 09 de 2009, de eyeonsecurity.net: http://www.cgisecurity.com/lib/flash-xss.htm#_Toc18055086

- Freier, A. O., Karlton, P., & Kocher, P. C. (18 de 11 de 1996). The SSL Protocol Version 3.0.

- Gardner, T. (02 de 10 de 2001). *An Introduction to Web Services*. Recuperado el 02 de 10 de 2009, de ariadne.ac.uk: http://www.ariadne.ac.uk/issue29/gardner

- Gribov, S. (2000). *JumpStart to the Web technologies tutorial: Web security*. Recuperado el 2009, de Sergey: http://www.sergey.com/web_course/part_12.html

- Grover, S. (10 de 03 de 2003). *Buffer Overflow Attacks and Their Countermeassures*. Recuperado el 29 de 09 de 2009, de linuxjournal.com: http://www.linuxjournal.com/article/6701

- Indiana University. (24 de 08 de 2009). *University Information Technology Services*. Recuperado el 22 de 09 de 2009, de Indiana University: http://kb.iu.edu/data/ahic.html

- Indiana University. (13 de 05 de 2009). What is the difference between SSL and TLS. Indiana.

- Linhart, C., Klein, A., & Orrin, S. (2005). HTTP REQUEST SMUGGLING.

- Mehta, P. (27 de 04 de 2005). *Guide to penetration testing, Part 3: Penetration testing strategies*. Recuperado el 05 de 10 de 2009, de TechTarget.com: http://searchnetworking.techtarget.com/generic/0,295582,sid7_gci1083715,00.html

- Microsoft. (2009). *Microsoft Security Bulletin*. Recuperado el 2009, de Microsoft TechNet: http://www.microsoft.com/technet/security/current.aspx

- Moses, L. A. (18 de 07 de 2003). Basic_Vulnerability_Assessment. Chicago, IL, USA.

- Ogorkiewicz, M., & Frej, P. (23 de 07 de 2004). *Analysis of Buffer Overflow Attacks*. Recuperado el 29 de 09 de 2009, de windowsecurity.com: http://www.windowsecurity.com/articles/Analysis_of_Buffer_Overflow_Attacks.html

- OWASP.org. (13 de 09 de 2009). *Code Injection*. Recuperado el 29 de 09 de 2009, de OWASP.org: http://www.owasp.org/index.php/Code_Injection

- OWASP.org. (16 de 09 de 2009). *Main Page*. Recuperado el 05 de 10 de 2009, de owasp.org: http://www.owasp.org/index.php/Main_Page

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

- Perrin, C. (18 de 03 de 2008). *What is cross-site scripting*. Recuperado el 29 de 09 de 2009, de techrepublic: http://blogs.techrepublic.com.com/security/?p=426

- Robotstxt.org. (2009, 7 16). *The Web Robots Pages*. Retrieved 9 4, 2009, from http://www.robotstxt.org/

- Robotstxt.org. (16 de 7 de 2009). *The Web Robots Pages*. Recuperado el 4 de 9 de 2009, de http://www.robotstxt.org/robotstxt.html

- SafeNet. (2008). Multi-Factor Authentication Protecting Applications and Critical Data against Unauthorized Access. Belcamp, Maryland, USA.

- Scambray, J., Shema, M., & Sima, C. (2006). *Web Application Hacking Exposed.*

- SearchSOA.com. (10 de 07 de 2008). *Simple Object Access Protocol (SOAP) Tutorial*. Recuperado el 02 de 10 de 2009, de techtarget.com: http://searchsoa.techtarget.com/generic/0,295582,sid26_gci1049566,00.html

- SecuriTeam. (19 de 04 de 2005). *[REVS] Introduction to HTTP Response Splitting*. Recuperado el 29 de 09 de 2009, de derkeiler.com: http://www.derkeiler.com/Mailing-Lists/Securiteam/2005-04/0104.html

- SecurityFocus. (2009). *Vulnerabilities*. Recuperado el 2009, de SecurityFocus: http://www.securityfocus.com/bid

- Shah, S. (19 de 05 de 2004). *An Introduction to HTTP fingerprinting* . Recuperado el 23 de 09 de 2009, de net-square.com: http://net-square.com/httprint/httprint_paper.html#fpttheory

- Shah, S. (2006, 02 11). *Detecting Web Application Security Vulnerabilities*. Retrieved 09 22, 2009, from http://onlamp.com/pub/a/sysadmin/2006/11/02/webapp_security_scans.html

- Shiflett, C. (13 de 12 de 2004). *Security Corner: Corss-Site Request Forgeries*. Recuperado el 23 de 09 de 2009, de PHP AND WEB APPLICATION SECURITY: http://shiflett.org/articles/cross-site-request-forgeries

- Srinivas, D. (24 de 11 de 2001). *What is WSDL?* Recuperado el 02 de 10 de 2009, de jguru.com: http://www.jguru.com/faq/view.jsp?EID=559994

- Stolk, J., & Benson, P. (12 de 08 de 2009). Stored Cross Site Scripting. Auckland, New Zealand.

- The Apache Software Foundation. (2009). *Apache httpd 2.2 vulnerabilites*. Recuperado el 2009, de The Apache HTTP Server Project: http://httpd.apache.org/security/vulnerabilities_22.html

- Thompson, J., & Gross, J. (1997). *Authentication and Authorization*. Recuperado el 2009, de ACM: http://www.acm.uiuc.edu/workshops/security/auth.html

- W3C. (23 de 01 de 2007). *XML Path Language (XPATH) 2.0*. Recuperado el 29 de 09 de 2009, de w3.org: http://www.w3.org/TR/xpath20/

- Walsh, N. (03 de 10 de 1998). *A Technical Introduction to XML*. Recuperado el 29 de 09 de 2009, de XML.com: http://www.xml.com/pub/a/98/10/guide0.html?page=1

- Wells, D. (22 de 04 de 1996). *Authentication*. Recuperado el 06 de 10 de 2009, de objs.com: http://www.objs.com/survey/authent.htm

- Yeary, D. (2007). *What is Http Cookie*. Recuperado el 2009, de streetdirectory: http://www.streetdirectory.com/travel_guide/62823/computers_and_the_internet/what_is_http_cookie.html

VI. APÉNDICE

A.Puntos de entrada

Tabla 13: Lisatdo 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

- Algoritmo: Un proceso definido sin ambigüedades o un conjunto de reglas para solucionar un problema en un número finito de pasos
- Aplicación: Es un programa de software desarrollado para una función concreta que una vez ejecutado, permite trabajar con el ordenador, como puede ser una hoja de cálculo. Puede ser sinónimo de programa o software .
- Aplicación web: Sitio Web que contiene páginas con contenido sin determinar, parcialmente o en su totalidad. El contenido final de estas páginas se determina sólo cuando un visitante solicita una página del servidor Web. Dado que el contenido final de la página varía de una petición a otra en función de las acciones del visitante, este tipo de página se denomina página dinámica.
- ASP: Active Server Pages (ASP) es una tecnología del lado servidor de Microsoft para páginas web generadas dinámicamente, que ha sido comercializada como un anexo a Internet Information Server (IIS).
- Atom: Protocolo de sindicación, sucesor de RSS.
- Autenticación: La identificación positiva de una entidad de red tal como un servidor, un cliente, o un usuario.
- Autorización: La autorización es el proceso que determina si se permite o no a una identidad autenticada tener acceso a un recurso solicitado o ejecutar una operación solicitada.

- Base de datos: Una base de datos es un formato estructurado para organizar y mantener informaciones que pueden ser fácilmente recuperadas. Un ejemplo simple de base de datos es una hoja de cálculo.
- Búfer: Es una cantidad de bytes guardados en una "memoria virtual" con el fin de evitar posibles insuficiencias de entrega de datos en trasmisiones continuas. Muy usado en grabadoras de discos.
- Cache: Memoria volátil que permite mantener información que puede ser requerida en otro momento, evitando la repetición del proceso
- Caja blanca: el personal realiza la prueba conociendo perfectamente la infraestructura de la aplicación, los mecanismos de defensa, y los canales que utiliza el sistema objetivo
- Caja gris: el personal que realiza la prueba la hace con conocimiento limitado de la infraestructura del sistema objetivo, mecanismos de defensa y canales de comunicación.
- Caja negra: el personal que realiza la prueba lleva a cabo los ataques sin ningún conocimiento previo de la infraestructura, mecanismos de defensa y canales de comunicación del sistema objetivo.
- Canal seguro: La instalación por la que se transmiten datos de forma segura entre localidades de una red de computación.
- Cifrar: Cifrar es convertir datos (texto sin formato) en un valor aparentemente aleatorio y sin sentido (texto cifrado), que es difícil de descifrar sin una clave secreta. Se utiliza para proporcionar confidencialidad a los mensajes.
- Cliente: Normalmente, en términos informáticos, un cliente se refiere a aquella máquina o aplicación que recibe la información solicitada de otra máquina o aplicación por el principio del retroalimentación. Se encuentra dentro del marco de la arquitectura "Cliente-Servidor"
- Código fuente: Es el conjunto de instrucciones compuestas mediante un lenguaje de programación, confrontando una aplicación o programa todavía no

compilado. Cuando estamos hablando de código abierto, es el código que se puede modificar.

- Confidencialidad: La privacidad o confidencialidad consiste en garantizar la confidencialidad de los datos para que no puedan ser visualizados por usuarios malintencionados con software de supervisión de la red. La privacidad suele proporcionarse mediante el cifrado.
- Cookie: Bloques de texto que se guardan en archivos en disco que permiten identificar al usuario cuando acceda nuevamente un sitio Web.
- Crawlers: Componente del buscador que recoge listas mediante "rastreo" automático en Internet. El rastreador de un buscador (también denominado “araña” o “robot”) sigue los vínculos a las páginas web. Realiza copias de las páginas que encuentra y las coloca en un almacén de información para procesar los datos y posteriormente incluirlos en el índice del buscador.
- Credenciales: Las credenciales son el conjunto de elementos que utiliza un objeto principal para probar su identidad. Un ejemplo habitual de conjunto de credenciales es el nombre de usuario y la contraseña.
- CSRF: Un ataque CSRF fuerza al navegador web validado de una víctima a enviar una petición a una aplicación web vulnerable, la cual entonces realiza la acción elegida a través de la víctima. Al contrario que en los ataques XSS, los cuales explotan la confianza que un usuario tiene en un sitio en particular, el cross site request forgery explota la confianza que un sitio tiene en un usuario en particular.
- Denegación de Servicios: Ataque de negación de servicio. Lograr que el usuario no pueda acceder a un determinado servicio o que no pueda proveer a otros. Estos ataques intentan corromper o saturar los recursos de la víctima por medio de peticiones de conexión para lograr desactivar su sistema o haciendo que deniegue el acceso a otros usuarios.
- Dirección IP: Identificador para un equipo o dispositivo en una red TCP/IP. Las redes que utilizan el protocolo TCP/IP envían mensajes basándose en la

dirección IP del destino. Presenta un formato de dirección numérica de cuatro números separados por puntos, y cada número oscila entre 0 y 255.

- Directorio Raíz: Es el punto inicial de los sistemas jerárquicos de archivos. El directorio raíz constituye el origen a partir del cual se despliega la estructura jerárquica de directorios, subdirectorios y archivos, a manera de un árbol y sus ramas.
- DNS: Acrónimo de Domain Name System (Sistema de Nombres de Dominio). Es un sistema que traduce un nombre de dominio a un número IP y de esta forma, se localiza el servidor donde reside un sitio Web.
- Dominio: Es un nombre base que agrupa a un conjunto de equipos conectados entre sí y que contienen información accesible desde una red interna o externa.
- Escáner de vulnerabilidades: Aplicación que analiza y determina las vulnerabilidades que posee un objetivo como una aplicación web o un conjunto de equipos de escritorio.
- Falso negativo: Evento que se da como inexistente cuando realmente si existe, por ejemplo, decir que un sistema está limpio de virus cuando realmente está infectado.
- Falso positivo: Evento que se da como existente cuando realmente no existe, por ejemplo, decir que un sistema está infectado de virus cuando realmente está limpio.
- Handshake: Proceso de negociación de una conexión entre dos computadoras que usan TCP/IP.
- Hash: Un hash es un valor numérico de longitud fija que identifica datos de forma unívoca. Los valores hash se utilizan para comprobar la integridad de los datos que se envían a través de canales no seguros. Puede compararse el valor hash de los datos recibidos con el valor hash de los datos que se enviaron para determinar si se alteraron los datos. Los valores hash también se utilizan en las firmas digitales. Dado que se pueden utilizar valores hash pequeños para

representar cantidades de datos de mayor tamaño, sólo es necesario firmar el hash de un mensaje, en lugar de todos los datos del mismo.

- HTML: (Hyper Text Markup Language) Es el lenguaje de marcado usado como el estándar para especificar el formato y delimitar el contenido que permite la visualización de páginas Web, desde un navegador. Se basa en etiquetas (instrucciones que le dicen al texto como deben mostrarse) y atributos (parámetro que dan valor a la etiqueta). La versión más avanzada se conoce como XHTML (Extensible Hypertext Markup Language).
- HTTP: HyperText Transfer Protocol. Protocolo de Transferencia de Hipertexto. Es el protocolo de transferencia de archivos de Hipertexto, para ser mostrados mediante un navegador.
- HTTPS: HyperText Transfer Protocol Secured. Protocolo de Transferencia de Hipertexto Asegurado. Es el protocolo de transferencia de archivos de Hipertexto de manera segura, para ser mostrados mediante un navegador y garantizarnos que el sitio dispone de certificado oficial y por lo tanto, la información que se muestra en pantalla es verídica y segura.
- Integridad: La garantía de que los datos enviados no han sido alterados.
- Interface Administrativa: Aplicación que se utiliza para gestionar un sistema.
- LDAP: Protocolo para el acceso a a directorios jerárquicos de información. Basado en el estándar X.500, pero significativamente más simple por lo que también se le denomina x.500-lite, se diferencia de éste porque soporta TCP/IP, necesario para cualquier tipo de acceso a Internet. Aunque no está ampliamente extendido, debería poderse implementar en la práctica mayoría de aplicaciones que se ejecutan virtualmente sobre plataformas informáticas para obtener información de directorios tales como direcciones de correo y llaves públicas. Ya que es un protocolo abierto, no afecta el tipo de servidor en el que se aloje el directorio.
- Llave de cifrado: Información que permite obtener texto claro o cifrado utilizando un algoritmo de cifrado.

- Malware: Programa o parte de un programa que tiene un efecto malicioso en la seguridad de los sistemas. Este término engloba muchas definiciones como virus, gusanos, troyanos, spyware, etc.
- Navegador: Un navegador (también llamado navegador Web o de Internet) es el programa que permite visualizar los contenidos de las páginas Web en Internet. También se conoce con el nombre de browser. Algunos ejemplos de navegadores Web son: Internet Explorer, Netscape Navigator, Opera, Mozilla, etc.
- Paquete: Cantidad mínima de datos que se transmite a través de una red o entre dispositivos. Tiene una estructura y longitud distinta según el protocolo al que pertenezca.
- Phishing: Técnica de ingeniería social usada para engañar a los usuarios con el fin de obtener sus contraseñas, nombres de usuario y otro tipo de información personal.
- PHP: (Hyper Text Pre-Prosessor / Personal Home Pages), Lenguaje de programación de licencia libre, embebido dentro del HTML y ejecutado en el servidor antes de ser enviado al navegador, usado para crear paginas dinámicas (datos dinámicos)
- Política: Regla que indica que hacer en una situación concreta.
- Privacidad: Ver Confidencialidad.
- Privilegios: Un privilegio es el derecho de un usuario de ejecutar diversas operaciones relacionadas con el sistema como, por ejemplo, cerrar el sistema, cargar controladores de dispositivos o cambiar la hora del sistema. El testigo de acceso de un usuario contiene una lista de privilegios del usuario o del grupo de usuarios.
- Protocolo: Conjunto de estándares, normas y formatos para el intercambio de datos que asegura la uniformidad entre computadores y aplicaciones.

- Proxy: Un servidor proxy actúa como un intermediario entre una red interna (por ejemplo, una intranet) y una conexión externa a Internet. De esta forma, se puede compartir una conexión para recibir ficheros desde servidores Web.
- Puntos de entrada: Lugar lógico dentro de una aplicación dónde esta espera una entrada de datos.
- RSS: Really Simple Syndication), es un formato XML formulado para canalizar las noticias de un sitio web especializado en noticias, las cuales pueden cambiar muy frecuentemente.
- Ruteador: Elemento que determinan la trayectoria más eficiente de datos entre dos segmentos de red. Operan en la capa superior del modelo OSI a la de los puentes -la capa de red- no están limitado por protocolos de acceso o medio. También se denomina ENCAMINADOR
- Servicios Web: Sistema de software diseñado para permitir interoperabilidad máquina a máquina en una red. En general, los servicios web son sólo APIs Web que pueden ser accedidas en una red, como internet, y ejecutadas en un sistema de hosting remoto.
- Servidor: Máquina que responde a las peticiones de los clientes con el fin de servirles los servicios o recursos que están solicitando. Sigue la arquitectura Cliente-Servidor.
- Servidor web: Servidor con un servicio para HTTP habilitado.
- Sesión: Son los lapsos que tiene un usuario navegando sobre el sistema, puede ser uno o varios. Una sesión tiene hora de inicio, hora de finalización y fecha de realización. Las fechas de las sesiones dependen de las fechas de inicio y final de la aplicación.
- Sistema Operativo: Programa principal de control que maneja la computadora, y cumple el papel de planificador y agente del tránsito de datos, además de administrar las rutinas para encender la PC, abrir programas y apagar el equipo. El sistema operativo es el primer programa que se carga en la memoria de la PC después del encendido. Realiza tareas de administración de trabajos, de

tareas, de datos, de dispositivos y de seguridad. El sistema operativo más popular es Windows.

- Sitio web: Lugar en la World Wide Web representado por una dirección electrónica, en el que se encuentra ubicada toda la información relacionada con una institución gubernamental, educativa o comercial.
- SMTP: También llamado protocolo simple de transferencia de correo electrónico. Protocolo de red basado en texto utilizado para el intercambio de mensajes de correo electrónico entre computadoras y/o distintos dispositivos (PDA's, Celulares, etc).
- SOAP: SOAP es un protocolo ligero basado en XML para intercambiar información en un entorno distribuido. Lo utilizan los servicios Web.
- Software: Componentes intangibles de una computadora, al conjunto de programas y procedimientos necesarios para hacer posible la realización de una tarea específica.
- SPAM: (Stupid Pointless Annoying Messages), Mensajes de correo electrónico no deseados, mensajes publicitarios enviados sin ningún destino específico.
- Spammer: Persona que envía SPAM
- Spiders: Ver Crawlers.
- SSL: Secure Sockets Layer. [Capa de Conexiones Seguras]. Protocolo diseñado por la empresa Netscape para ofrecer comunicaciones encriptadas en Internet.
- TLS: Protocolo del tipo EAP que garantiza la privacidad y la seguridad de datos entre aplicaciones cliente/servidor que se comunican vía Internet. Trabaja en dos niveles: El protocolo de registro TLS - situado en el nivel superior de un protocolo de transporte seguro como TCP asegura que la conexión es privada empleado encriptación simétrica de datos y asegura que la conexión es fiable. También se utiliza para la encapsulación de protocolos de nivel superior, tales como el TLS handshake Protocol. Y, el protocolo de handshake TLS - permite la autenticación entre el servidor y el cliente y la negociación de un algoritmo

de encriptación y claves criptográficas antes de que el protocolo de la aplicación transmita o reciba cualquier dato. TLS es un protocolo independiente que permite que protocolos de niveles superiores se sitúen por encima de él de manera transparente. Basado en SSL de Netscape 3.0, TLS supercede y es una extensión de SSL, si bien no son interoperables.

- Topología: Arquitectura o diseño de una red de computadoras.
- Troyano: Programa informático cuya ejecución tiene unos efectos imprevistos y, generalmente, insospechados para el usuario infectado. No se les puede denominar virus porque no se replican.
- URI: (Uniform Resource Identifier), Igual que el anterior, pero referido a identificadores específicos, puede constar de nombres, localizadores, o ambos.
- URL: (Uniform Resource Locator), Es la dirección global de cualquier documento o recurso en la Web, visto como localizador. Consta de dos partes, la primera identifica el protocolo usado y la segunda, la dirección IP o el nombre de dominio, separados por barras dobles "//", por ejemplo: http://www.icad.com.ve.
- Vulnerabilidad: Se trata de fallos o huecos de seguridad detectados en algún programa o sistema informático, que los virus utilizan para propagarse e infectar. Estos errores de programación y/o diseño permiten que un tercero se aproveche de ellos para realizar acciones tales como ataques, intrusiones o cualquier otro uso indebido.
- Web: Sistema distribuido con mecanismos de hipertexto. Que permiten mezclar texto, gráficos y archivos de sonido juntos.
- XML: Sencillo formato de texto, también derivado del SGML, y diseñado especialmente para documentos web. Permite la creación de etiquetas propias.
- Xpath: XPath es un lenguaje para direccionar partes de un documento XML, diseñado para ser utilizado tanto por XSL como por XPointer.
- XSS: (Cross-site Scripting) Es una brecha de seguridad que se produce en páginas Web generadas dinámicamente. En un ataque por XSS, una aplicación

Web se envía con un script que se activa cuando lo lee el navegador de un usuario o una aplicación vulnerable. Dado que los sitios dinámicos dependen de la interacción del usuario, es posible ingresar un script malicioso en la página, ocultándolo entre solicitudes legítimas. Los puntos de entrada comunes incluyen buscadores, foros, blogs y todo tipo de formularios online en general. Una vez iniciado el XSS, el atacante puede cambiar configuraciones de usuarios, secuestrar cuentas, envenenar cookies, exponer conexiones SSL, acceder sitios restringidos y hasta instalar publicidad en el sitio víctima.