UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

FACULTAD DE INGENIERÍA

TESINA

AUDITORÍA DE SISTEMAS Y CÓDIGO

QUE PARA OBTENER EL TÍTULO DE
INGENIERO EN COMPUTACIÓN

PRESENTA:

JULIO JIMÉNEZ UNZUETA

AVAL DE TESINA: ING. ALEJANDRO NÚÑEZ SANDOVAL

CIUDAD UNIVERSITARIA, NOVIEMBRE DE 2011

GRACIAS

A mi familia, como una muestra del cariño y eterno aprecio por el apoyo que siempre me han brindado para llevar a cabo todo aquello que he decidido emprender. Porque el camino que hemos recorrido juntos y lo que hemos alcanzado es resultado de un esfuerzo conjunto, y porque la conclusión satisfactoria de este proyecto tan importante para mí se debe en gran medida a la guía que me dieron a lo largo de mi trayectoria escolar.

A ti papá, por tu ejemplo de superación incansable, por no darte por vencido hasta que todos tus hijos seamos profesionistas. Por la confianza que siempre me has tenido, porque crees en mí y me tienes paciencia, por tu infinito apoyo y porque a través de todas tus enseñanzas has forjado la persona que hoy puedo presumir ser. No hay manera de agradecerte.

A ti mamá, porque siempre tienes las palabras para alentarme a continuar sin importar qué tan grandes parezcan los obstáculos. Porque siempre me demuestras tu amor incondicional y porque siempre que lo necesito estás ahí para ayudarme a levantar cuando me caigo. Por todo el esfuerzo que junto a mi papá has hecho para que salgamos adelante, ¡gracias!

A Bali, "porque sólo nos espera el éxito". Me has dado el ejemplo más grande de que cuando se trabaja y se lucha por lo que uno quiere, se cosechan grandes alegrías. Por todas aquellas veces en que sin ser tu obligación te encargaste de sacarnos adelante y porque siempre me recuerdas lo importante que es estar bien como familia.

A Marcela, porque me enseñaste que soñar es el primer paso para poder establecer metas y que no descuidar el espíritu es fundamental para ser una persona íntegra. Porque aún nos cuidas a Javo y a mí, y porque a través de ti he aprendido a ver la parte humana de las cosas, a no tomarme los problemas tan en serio y a alivianarme cuando las presiones son muchas.

A Javo, auténtico compañero de vida. Porque las travesuras, los deportes y la escuela no hubieran sido lo mismo nomás por mi cuenta. A través de todo el tiempo que hemos pasado juntos me has motivado a fijarme nuevas metas y me has dado la confianza y motivación para obtener un sinnúmero de logros. Gracias porque a pesar de las peleas y los desacuerdos siempre estás ahí para apoyarme.

A mis sobrinos "Punksbwan", Larissita, "Cachetes" y Lizzy porque siempre me recuerdan lo importante que es no dejar de asombrarse de la vida. Porque hacen que trabaje mi paciencia todos los días, por hacerme reír y porque me han hecho redescubrir la magia detrás de las cosas sencillas y los sentimientos auténticos.

A mi madrina Hortensia Sierra, por todas esas pláticas que me ayudan a ponerme en sintonía entre los muchos giros y vueltas de la vida. Porque siempre tienes la manera de sacarme una sonrisa y hacerme contactar con mi yo interior; por el ejemplo de vida que compartes con todas las personas que te conocemos.

A mis amigos, mi segunda familia, porque siempre estuvieron ahí para hacer completamente disfrutable la facultad y la vida estudiantil, porque juntos vivimos (y a veces también sobrevivimos) alegrías, tristezas, penas, logros, dificultades y satisfacciones que nos hicieron crecer y madurar.

En la Facultad: especialmente a Asminda García Gaytán, porque me brindaste tu amistad y tu ayuda, y continúas demostrándome que la grandeza de las personas se mide por la forma en que ayudan a las demás. Gracias a “Ayan”, “Serg”, Vic, Alondra, Arlette y a tod@s quienes me dieron la oportunidad de conocerlos y me regalaron excelentes momentos.

En la Subdirección de Servicios Web (DGTIC): a Alfredo Peña, Nancy Escorcia, Jessica Cruz, Jorge Gómez y Diana Gutiérrez por haber hecho de la subdirección un lugar agradable, por compartir conmigo su conocimiento y darme la oportunidad de desarrollar mis capacidades.

En UNAM-CERT: a Paco, Miriam, Isra, Mayra, Alex y Ceci por acompañarme más allá del trabajo, las clases y los proyectos. A Jessica, Dante, Edgar, Jair, Paulo, Perla, David Campos, David Hernández, Abraham y Miguel porque sin ustedes no habría sido posible terminar satisfactoriamente el plan de becarios; su deseo para que las cosas funcionaran realmente marcó diferencia. Al Ing. Rubén Aquino Luna por todas las facilidades brindadas.

Al Ing. Alejandro Núñez Sandoval, por la confianza mostrada durante la realización de este proyecto, así como por el apoyo otorgado y la disposición para proporcionarme la asesoría requerida para su culminación satisfactoria.

A la Universidad Nacional Autónoma de México, a la Facultad de Ingeniería, la Dirección General de Tecnologías de la Información y Comunicación (DGTIC) y a todos los profesores que a través de sus enseñanzas transmitieron su conocimiento y me brindaron las bases para iniciar mi trayectoria profesional.

“Por mi raza hablará el espíritu”
Julio Jiménez Unzueta

Índice

INTRODUCCIÓN **6**

CAPÍTULO 1. ORGANIGRAMA **10**

CAPÍTULO 2. DESCRIPCIÓN DE PROYECTOS **11**

CAPÍTULO 3. PROYECTO PRINCIPAL **12**

3.1 OBJETIVOS DEL PROYECTO 12
3.2 DEFINICIÓN DE LA PROBLEMÁTICA 12
3.3 JUSTIFICACIÓN DE LA PROPUESTA 13
3.4 METODOLOGÍA 13
3.4.1 Análisis Dinámico de la Aplicación *16*
3.4.2 Análisis estático de la aplicación *17*
3.4.3 Evaluación de impacto *17*
3.4.3.1 Métricas del CVSS 18
3.4.3.2 Métricas del grupo Base 18
3.4.3.3 Ecuación base 19
3.5 GENERACIÓN DEL BOLETÍN TÉCNICO 20
3.6 REMEDIACIÓN DE LOS PROBLEMAS IDENTIFICADOS 21

CAPÍTULO 4. RESULTADOS **22**

4.1 TRABAJO FUTURO 27

CONCLUSIONES **28**

REFERENCIAS **29**

ANEXO A. GUÍA DE MEJORES PRÁCTICAS PARA LA FORTALECER LA SEGURIDAD DE LA INFORMACIÓN EN LOS SISTEMAS INSTITUCIONALES **31**

PRESENTACIÓN 31
BASES DE DATOS 31
Base de datos MySQL *31*
SQL Server *36*
SERVIDORES WEB 47
Servidor web Apache 2.x *47*
Internet Information Services (IIS) *54*
DESARROLLO 57
Tecnología .NET *57*
IMPLEMENTACIÓN DE SERVICIOS EN SERVIDORES UNIX/LINUX 60

ANEXO B. INSTALACIÓN SUGERIDA DE MODSECURITY SOBRE SERVIDOR WEB APACHE 2.X **62**

Instalación del módulo *62*
Reglas *65*

ANEXO C. CHECKLIST MÍNIMO PARA EL USO DE SQL SERVER, IIS, MYSQL Y APACHE 2.X **67**

REFERENCIAS DE LOS ANEXOS **70**

Índice de gráficos

GRÁFICO I_1 – VENTANA DE EXPOSICIÓN (DÍAS) POR INDUSTRIA EN 2010 8
GRÁFICO 1_1. ESTRUCTURA GENERAL DE LA DIRECCIÓN GENERAL DE SISTEMAS DEL TRIBUNAL ELECTORAL 10
GRÁFICO 3_1 – PROCESO GENERAL DE LA AUDITORÍA 16
GRÁFICO 4_1 – CONTEO DE RIESGOS POR APLICACIÓN ANALIZADA 22
GRÁFICO 4_2 – PORCENTAJES DE RIESGOS DE IMPACTO SEVERO DE ACUERDO CON OWASP 23
GRÁFICO 4_3 – DISTRIBUCIÓN POR TIPO DE RIESGO OWASP 24
GRÁFICO 4_4 – RIESGOS POR REVISIÓN SOBRE EL SISTEMA F 26

Introducción

El crecimiento acelerado de las redes de computadoras y la Internet modificaron drásticamente la forma en que se accede y se interconectan los sistemas de información/comunicación.

La interconexión de redes y la capacidad de comunicación remota trajeron grandes avances en las sociedades, el manejo de la información, así como nuevos retos en prácticamente todos los ámbitos de la tecnología. A partir de esto, las aplicaciones web proliferaron a la par que Internet se volvió indispensable para las comunicaciones globales y se ha llegado al punto en el que más del 90% de la interacción con los sistemas de información (ya sean educativos, financieros, sociales, etc.) se realiza mediante una aplicación y/o interfaz web.

Como sucede con todos los avances tecnológicos, el verdadero potencial, las capacidades y usos de las aplicaciones web se fueron descubriendo conforme su popularidad aumentó, y no fue sino hasta pasado ese período que fue posible darse cuenta de los riesgos implicados. El boom del desarrollo de aplicaciones web trajo grandes consecuencias en el ámbito de seguridad, pues también proveyó a los atacantes nuevos puntos de acceso a los sistemas al aumentar la superficie de ataque sobre los mismos. La no percepción de lo anterior creó un punto ciego en la postura general de seguridad, llevando a pensar que el aseguramiento de la red perimetral es suficiente para proteger los sistemas.

La realidad es que una vez que las defensas perimetrales son superadas o burladas mediante los puentes que crean las aplicaciones web entre un usuario (normalmente remoto) y los componentes que le dan funcionalidad, los elementos internos de la infraestructura y el software de los sistemas que no se encuentran asegurados de forma adecuada no son capaces de defenderse por sí mismos.

El *2010 Verizon Data Breach Investigation Report* confirma que “la mayor parte de los incidentes de seguridad y casi todos los datos robados (95%) durante 2009 fueron perpetrados de forma remota por grupos delictivos capaces de explotar las vulnerabilidades presentes en servidores y aplicaciones”.

Cuando las organizaciones no cuentan con el aseguramiento y la protección adecuada para sus sitios web, las consecuencias son claras: robo de información, infección por malware, pérdida de confianza por parte de los usuarios, así como fallos al tratar de cumplir ciertos requerimientos y/o regulaciones de la industria o el ramo de la organización. Además, el impacto que puede provocar el mal funcionamiento de las aplicaciones dentro de organizaciones que ofrecen servicios como los antes mencionados puede llegar a ser muy alto. Tal fue el caso de la aerolínea Aeroméxico, que el 24 de mayo de este año tuvo que hacer frente a un caos total en el aeropuerto internacional de la ciudad de México derivado de un fallo en la migración de su servicio de *check-in* hacia una nueva plataforma tecnológica. Como resultado, la salida de los vuelos se vio atrasada y también se vieron afectados los aeropuertos de Guadalajara y Cancún, provocando la molestia de los usuarios y dejando a la empresa con una labor titánica para poder limpiar la imagen que debe

mantener dentro del ramo en que se desempeña, así como para asumir la responsabilidad de indemnizar a los afectados. [1]

A decir de Jeremiah Grossman, fundador y CTO de Whitehat Security, los sitios web son hoy en día el objetivo por elección de los ataques realizados en internet. Dichos ataques se han ido moviendo de la capa perimetral bien asegurada hacia la capa de las aplicaciones web, que es más accesible y la cuál es usada por muchas personas día a día para llevar a cabo sus actividades [2]. Los sitios donde se compra, se realizan transacciones bancarias, se realizan pagos, se agendan viajes y se registra información se encuentran bajo un bombardeo de ataques que buscan robar números de tarjetas de crédito y otro tipo de información personal y/o privada.

El documento publicado por WhiteHat Security [5] presenta una visión estadística de la seguridad recopilada a lo largo de 5 años de los resultados provistos por su herramienta de administración de vulnerabilidades llamada WhiteHat Sentinel, que actualmente administra más de 3000 sitios web pertenecientes a unas 400 organizaciones de diversas partes del mundo.

El camino recorrido en materia de aseguramiento de aplicaciones web no ha sido sencillo, y está muy lejos de ser el definitivo; de acuerdo con el reporte estadístico de Whitehat Security hacia el invierno de 2011, la mayor parte de los sitios web estuvieron expuestos al menos a una vulnerabilidad seria (aquellas vulnerabilidades catalogadas como HIGH, CRITICAL o URGENT, de acuerdo con la nomenclatura convenida en el Payment Card Industry Data Security Standard o PCI-DSS) durante un período de entre 9 y 12 meses del año. Únicamente el 16% de los sitios estuvieron expuestos durante menos de 30 días a lo largo de todo el año.

Las estadísticas más importantes del documento reportan que el 71% de los sitios educativos, 58% de las redes sociales y el 51% de los sitios de venta al menudeo estuvieron expuestos al menos a una vulnerabilidad seria durante cada uno de los días del 2010. En contraste, el 51% de los sitios bancarios, 22% de los sitios de servicios financieros y 19% de los sitios del sector médico/salud estuvieron expuestos durante menos de 30 días a lo largo de todo el año pasado. Así, WhiteHat reporta que un sitio web promedio estuvo expuesto a 230 vulnerabilidades serias durante el año 2010, como se ilustra en la figura *I1*.

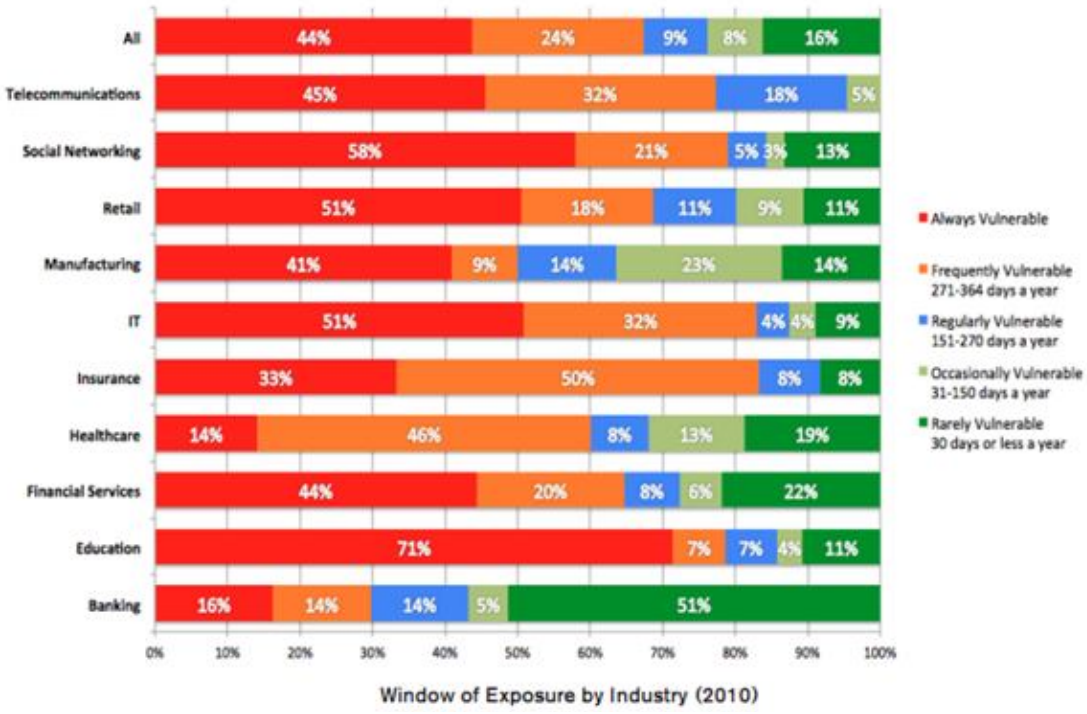


Gráfico I_1 – Ventana de exposición (días) por Industria en 2010

Existen una serie de factores que incrementan los riesgos de la seguridad de las aplicaciones web, siendo algunos de los más importantes:

1. *Ubicuidad de las aplicaciones web***.** Como se ha mencionado anteriormente, las aplicaciones web se encuentran en todos los ámbitos de la vida diaria y su esparcimiento y popularidad continúa creciendo rápidamente tanto en redes privadas como públicas. Debido a lo anterior, un atacante encontrará un número importante de aplicaciones web que podrían ser objeto de un ataque.

2. *Calidad del código en los sistemas.* Con la proliferación de plataformas de desarrollo fácilmente accesibles y modulares (como Drupal, WordPress, etc), las sofisticadas interfaces de programación de aplicaciones (APIs) ofrecidas por tecnologías como .NET y JAVA, así como la dependencia sobre combinaciones de tecnologías para ofrecer funcionalidad como los "sistemas LAMP" (Linux, Apache, MySQL, PHP), es posible integrar/desarrollar sistemas complejos en poco tiempo. Sin embargo, al utilizar código proveniente de desarrolladores cuyos conocimientos y capacidades suelen variar considerablemente entre uno y otro, la calidad general del código de la aplicación no es homogénea. Esto vuelve necesario permanecer al pendiente de las publicaciones de seguridad publicadas para cada una de las tecnologías sobre la que se basa el sistema desarrollado, pues con que una de éstas

presente problemas, el sistema completo puede verse comprometido o expuesto a riesgos que pongan en peligro la integridad o disponibilidad del mismo.

3. *Madurez de la seguridad en las tecnologías de la información y comunicación.* A pesar de que el desarrollo de aplicaciones web ha recorrido un camino largo, la implementación de la seguridad no ha ido a la par y aún tiene mucho camino por recorrer. Por ejemplo, HTTP ni siquiera implementa sesiones para separar/identificar usuarios, y los mecanismos de autenticación y autorización básicos fueron agregados años después de que la tecnología se volviera popular. Esto se debe, en parte, a la constante evolución y crecimiento de las tecnologías de la información.

4. *Cambio constante en las aplicaciones web.* Normalmente, hay un gran número de personas trabajando a la par sobre una aplicación web: desarrolladores, administradores de sistemas, responsables de contenido, diseñadores, etc. De ellas, sólo un número muy bajo cuenta con entrenamiento de seguridad adecuado a pesar de que son los responsables de realizar cambios de manera constante a aplicaciones complejas que ofrecen sus servicios a través de Internet. Con este nivel de dinamismo, es difícil adherirse a un proceso de administración de cambios simples (o incluso garantizar que se hacen cumplir de forma consistente las políticas de seguridad).

Dadas las circunstancias anteriormente descritas, no resulta sorprendente que el aseguramiento de aplicaciones web haya adquirido relevancia a lo largo de los últimos años. La difusión cada vez más común de noticias referentes a incidentes de seguridad informática derivados de deficiencias en las aplicaciones web ha ayudado a incrementar la toma de conciencia en organización e instituciones a nivel mundial, particularmente aquellas de gran relevancia como las del gobierno y el sector público.

El siguiente trabajo describe una metodología que se propuso e implementó dentro de la Dirección de Seguridad Informática del Tribunal Electoral para cubrir las necesidades de la institución y para hacer frente a la situación de seguridad que enfrentan la gran mayoría de los sistemas web actuales. Presenta también los resultados estadísticos de una muestra significativa de aplicaciones analizadas dentro de la primera etapa del proyecto de auditoría de sistemas y código.

Capítulo 1. Organigrama

La Dirección General de Sistemas es una unidad de la estructura orgánica del Tribunal Electoral del Poder Judicial de la Federación cuya misión es planear, dirigir y controlar las acciones en materia informática para proporcionar servicios de cómputo, telecomunicaciones, información, sistemas digitales y documentales a las diversas áreas del Tribunal Electoral.

La estructura general de esta unidad está compuesta por dos Jefaturas de Unidad con carácter técnico y de enlace: la Jefatura de Unidad de Desarrollo de Sistemas y la de Soporte Técnico y Telecomunicaciones; así como por Direcciones de Área y Subdirecciones y/o Departamentos derivados como puede observarse en el gráfico *1_1*.

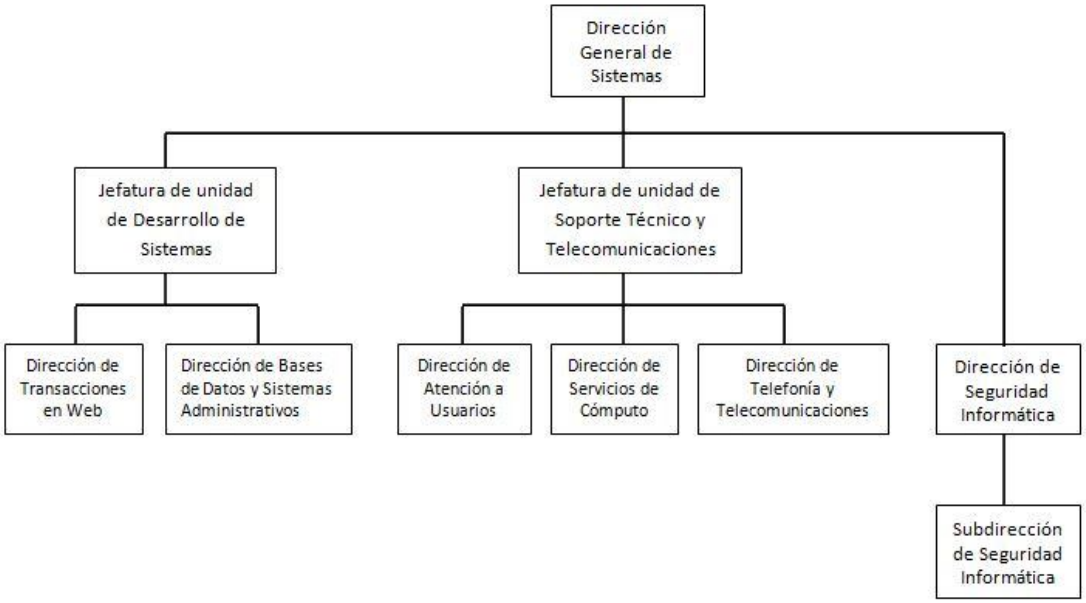


Gráfico 1_1. Estructura general de la Dirección General de Sistemas del Tribunal Electoral

De manera particular, la Dirección de Seguridad Informática del Tribunal Electoral no depende de una Jefatura de Unidad, y tiene por objetivo organizar, supervisar, controlar, así como evaluar técnica y administrativamente las actividades relacionadas con la seguridad informática a fin de que se implanten procesos tecnológicos seguros en la operación de la red de datos del Tribunal Electoral.

Capítulo 2. Descripción de Proyectos

Dentro del área de Seguridad Informática se llevan a cabo una diversidad de proyectos para garantizar la seguridad de la información y verificar que la institución cuenta con los niveles de seguridad requeridos para el cumplimiento de sus funciones. Entre los principales proyectos que se encuentran en operación están:

- *Monitoreo y administración de dispositivos de infraestructura de seguridad perimetral*: Con las actividades desarrolladas en este proyecto se instrumentan las políticas de seguridad para los sistemas informáticos, se implanta y verifica el cumplimiento de los criterios para el acceso controlado y seguro a los recursos de la red interna y externa de la institución.

- *Atención a incidentes de seguridad informática*: Se lleva a cabo la atención a eventos que son identificados como posibles riesgos a los activos informáticos que una vez valuados en este sentido se han declarado como incidentes de seguridad, para que de esta manera se lleve a cabo el proceso de análisis de cómputo forense y se identifiquen las causas y los efectos que este evento tiene sobre los activos informáticos.

- *Administración de la Unidad de Certificación Electrónica y actividades derivadas*: A través de este proyecto se implanta y opera la infraestructura de llave pública del Tribunal Electoral, al tiempo que se administra el uso de los certificados digitales en los procesos jurisdiccionales, administrativos y tecnológicos.

- *Investigación sobre tecnologías nuevas y actuales en materia de seguridad de la información*: Con las actividades pertenecientes a este proyecto se analiza y estudia el impacto que tiene sobre la seguridad del Tribunal Electoral la forma en que se utilizan las tecnologías de la información. Posteriormente se elaboran dictámenes, opiniones, informes y propuestas para mejorar y/o corregir su forma de uso, o para implantar nuevas tecnologías que mejoren el estado de seguridad de la información y recursos de la institución.

- *Auditoría de sistemas institucionales:* A través de este proyecto se supervisa el cumplimiento de políticas y lineamientos de auditoría y monitoreo de las redes, equipos de cómputo y sistemas informáticos para garantizar que la seguridad de la información de la institución no pueda verse afectada.

Capítulo 3. Proyecto Principal

La propuesta del proyecto de Auditoría de Sistemas y Código surgió a partir de la necesidad de desarrollar un procedimiento formal de pruebas para la liberación de sistemas informáticos, así como contar con normatividad en materia de desarrollo de software de calidad.

3.1 Objetivos del proyecto

Tomando como fundamento las necesidades anteriores, los objetivos que definí para este proyecto fueron:

- Llevar a cabo un proceso de análisis de aplicaciones web y auditoría de código sobre los sistemas institucionales para identificar las deficiencias que presentan y determinar el estado de seguridad de la información y la infraestructura.

- Generar los boletines técnicos correspondientes para dar a conocer al equipo de desarrolladores los problemas o deficiencias que presente el código de los sistemas, así como proporcionarles las recomendaciones y/o mecanismos de solución para la disminución y/o mitigación de los mismos.

- Evaluar los desarrollos de aplicaciones previos a su liberación para garantizar que éstos cumplen con los niveles de protección a la información e infraestructura requeridos por la institución.

- Elaborar una base de conocimiento que recopile las mejores prácticas para el uso de las tecnologías empleadas y las configuraciones recomendadas para la infraestructura utilizada por el área de desarrollo de sistemas

3.2 Definición de la problemática

Aún resulta común que en el proceso de desarrollo de software arquitectos y/o programadores asuman que cierto tipo de combinaciones en las entradas de datos e información que recibe una aplicación son poco probables de presentarse, resultando en la implementación frágil de una aplicación que fallará o se verá comprometida cuando se den estas circunstancias.

“La noción de que una vasta mayoría de las vulnerabilidades de seguridad en los sistemas computarizados modernos se deben a una calidad pobre del software es bien conocida”[3]. El software se puede ver afectado por una cantidad de factores que van desde un diseño hasta una implementación deficiente, donde la realización de suposiciones no válidas es uno de los errores principales y más comunes [4]. Identificar la fuente de los problemas de seguridad mejora la calidad del software, y en consecuencia permite reducir el riesgo de

introducir elementos o puntos débiles a un sistema. Sin embargo, los requisitos para alcanzar dicho objetivo no se limitan a una sola actividad.

Por otro lado, debido a que las deficiencias de una aplicación web son específicas al sistema y/o sitio web de una organización, no existe una receta exacta para mitigar las deficiencias de seguridad de un sistema. Lo anterior significa que a pesar de que las aplicaciones web pudieran parecerse, las tecnologías utilizadas y la forma de implementación serán diferentes de organización a organización: pudiera ser que las herramientas para desarrollar la aplicación estén abiertas a la elección de los programadores, o que estén determinadas por algún contrato o licencia que se haya adquirido previamente.

3.3 Justificación de la propuesta

Instituciones del sector público con gran relevancia como el Poder Judicial de la Federación requieren contar con procedimientos que permitan garantizar que los sistemas informáticos desarrollados para automatizar los procedimientos jurisdiccionales y administrativos de la organización cuenten con niveles de calidad suficientes para que la información contenida y procesada en éstos se mantenga íntegra, de manera que no sufra alteraciones no autorizadas.

Por otro lado, también debe garantizar que los servicios ofrecidos por dichos sistemas se mantengan disponibles durante los períodos de operación, de manera para que no se afecte la productividad de los procesos que se automatizan, aún cuando se integren nuevos sistemas de cómputo.

El desarrollo de un procedimiento formal para la prueba de los sistemas que serán liberados fortalece el proceso de desarrollo de aplicaciones informáticas que cumplen con los estándares de calidad que garantizan la confidencialidad, integridad y disponibilidad de los recursos de información requeridos por la institución.

3.4 Metodología

Actualmente, existen múltiples esfuerzos para proporcionar un marco de referencia para el desarrollo de software de calidad, desarrollo de código seguro, así como metodologías para evaluar la la seguridad de las aplicaciones. Proyectos como OWASP y organismos como WASC (Web Application Security Consortium) ofrecen un marco para la identificación y clasificación de vulnerabilidades, problemas y/o defectos que se pueden encontrar en las aplicaciones web.

Puesto que no existe una metodología específica para la mitigación de vulnerabilidades web, dentro de este proyecto usé una técnica ecléctica que tuvo como bases principales la metodología para la prueba de aplicaciones web del proyecto OWASP (OWASP testing guide), el análisis estático de aplicaciones y el uso de herramientas automatizadas para la prueba de diversos aspectos de la aplicación web.

Una de las principales razones que me llevaron a tomar esa decisión fue que el enfoque de la guía OWASP para la prueba de aplicaciones web se fundamenta en el análisis dinámico de la aplicación y va enfocado a mejorar la seguridad al permitir identificar las principales fallas en las aplicaciones que ofrecen sus servicios hacia Internet.

Por otro lado, el proyecto también publica de forma periódica un top 10 con los principales riesgos en las aplicaciones web que ponen en riesgo la seguridad de la información, por lo que me pareció adecuado que éstos constituyeran el criterio de clasificación de los riesgos en el proyecto.

En su última versión publicada (2010), los principales riesgos del top 10 son los siguientes:

- *A1: Injection* – La inyección de código (como la de SQL o LDAP) ocurre cuando se envían datos no confiables hacia un intérprete del lenguaje en cuestión como parte de un comando o consulta. Dichos datos no confiables pueden provocar que el intérprete ejecute acciones distintas a las programadas, así como proveer el acceso a información no autorizada.

- *A2: Cross-Site Scripting (XSS)* - Las fallas Cross-Site scripting ocurren cuando la aplicación toma datos no confiables y los envía al navegador web sin realizar la validación y/o "escapado" de caracteres correspondiente. El XSS permite a un atacante ejecutar scripts en el navegador del usuario para robar sesiones, defasar sitios web o redirigir al usuario a sitios maliciosos.

- *A3: Broken Authentication and Session Management (BA & SM)* – Las funciones de las aplicaciones relacionadas con la autenticación y administración de la session suelen no estar implementadas de forma correcta, permitiendo así a un atacante comprometer contraseñas, llaves, tokens de session o explotar otras fallas de implementación para suplantar la identidad de los usuarios.

- *A4: Insecure Direct Object References (IDOR)* – Una referencia directa a un objeto ocurre cuando un desarrollador expone una referencia hacia un objeto de implementación interno, como un archivo, directorio o llave de bases de datos. Sin el control de acceso u otro mecanismo de protección correspondiente, un atacante puede manipular estas referencias para acceder a datos no autorizados.

- *A5: Cross-Site Request Forgery (CSRF)* – Un ataque de CSRF obliga al navegador de un usuario autenticado enviar una solicitud HTTP manipulada que incluye la cookie de sesión de la victima junto con otra información de autenticación hacia una aplicación web vulnerable, haciéndole crer que dicha solicitud fue generada por un usuario válido.

- *A6: Security Misconfiguration (SM)* – Una buena seguridad requiere que se hayan definido e implementado una configuración segura para la aplicación, plataformas de desarrollo, servidores de aplicación, web, de bases de datos y de sistema

operativo. Todas estas configuraciones deberían estar definidas, implementadas y revisadas puesto que la mayoría no cuenta con valores seguros por defecto. Aquí se incluye también mantener el software al día, incluyendo las bibliotecas de código usadas por la aplicación.

- *A7: Insecure Cryptographic Storage (ICS)* – Muchas aplicaciones web no protegen de forma adecuada información sensible como números de tarjetas de crédito, números de seguro social, credenciales de autenticación, etc. mediante el uso de herramientas de cifrado o hashing. Un atacante podría robar o modificar la información débilmente protegida para realizar un robo de identidad, fraude con tarjetas de crédito u otros crímenes.

- *A8: Failure To Restrict URL Access (FTRUA)* – Algunas aplicaciones web revisan los derechos de acceso a una URL antes de mostrar vínculos y/o botones protegidos. Sin embargo, las aplicaciones deben ejecutar ese tipo de control de acceso cada vez que se accede a una página para evitar que un atacante pueda modificar una URL para accede a lugares no autorizados.

- *A9: Insufficient Transport Layer Protection (ITLP)* – Frecuentemente las aplicaciones fallan al autenticar, cifrar y proteger la confidencialidad e integridad del tráfico de red sensible (que contiene información confidencial). Cuando llegan a hacerlo, normalmente lo hacen a través de algoritmos débiles, usan certificados no válidos o expirados, o no los usan de forma adecuada.

- *A10: Unvalidated Redirects and Forwards (UR&F)* – Las aplicaciones web redireccionan al usuario de forma frecuente hacia otras páginas y sitios web, y usan datos no confiables para determinar el destino de dichas redirecciones. Sin una validación adecuada, un atacante puede redirigir a un usuario hacia sitios de phishing o que contienen malware, así como usar la redirección para acceder a páginas no autorizadas.

El proceso para la realización de la auditoría de los sistemas que analicé lo inicié siempre con la preparación del ambiente controlado para la ejecución de pruebas estáticas y dinámicas. Para esta actividad eché mano de mecanismos de virtualización para obtener una reproducción exacta o lo más parecida posible de los componentes de la aplicación que se encontraban en producción, como los servidores web y los servidores de bases de datos.

Una vez que el ambiente controlado se encontraba funcional, llevé a cabo las pruebas dinámicas y estáticas correspondientes con el propósito de identificar los problemas presentes en el sistema que se estaba revisando, así como priorizar la implementación de las soluciones requeridas.

En el siguiente paso interpreté los resultados obtenidos y generé el boletín técnico correspondiente para que las personas responsables del desarrollo y/o administración del sistema evaluado conocieran los riesgos que identifiqué, y así nos pusiéramos de acuerdo para calendarizar la implementación de las soluciones.

El gráfico *3_1* muestra un esquema general del procedimiento:

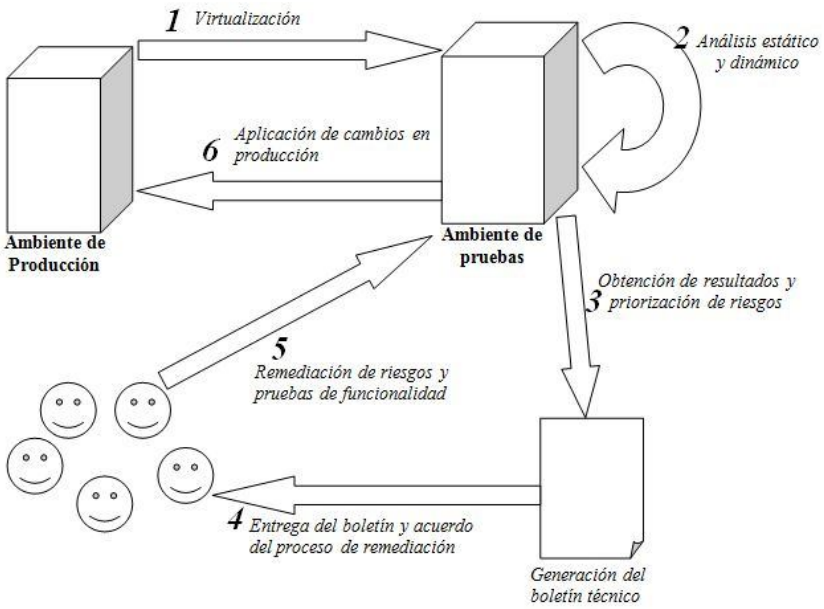


Gráfico 3_1 – Proceso general de la auditoría

Por último volví a evaluar el sistema para verificar que se habían solucionado los problemas reportados, y repetí los pasos de la metodología cuando fue necesario.

3.4.1 Análisis Dinámico de la Aplicación

El análisis dinámico de las aplicaciones comprendió una serie de acciones en las que me familiaricé con la interfaz de la aplicación, entendí la lógica de las interacciones que ofrecía, y realicé supuestos sobre lo que debía ser el funcionamiento "normal" o esperado del mismo para intentar realizar operaciones que generaran errores. Todo lo anterior con la finalidad de identificar claramente todos los puntos de interacción con el usuario (los cuales en la mayoría de los casos ofrecen mecanismos para que éste introduzca datos al sistema y lleve a cabo operaciones dentro del mismo) e incluso para detectar los defectos y/o problemas más evidentes de la aplicación.

Posteriormente, usé diversas herramientas automatizadas como Nessus, QualysGuard e IBM Rational AppScan para obtener las características del sistema: sistema operativo, puertos de comunicación abiertos, servicios que ofrecía el equipo, versiones del servidor, software de desarrollo y otras tecnologías empleadas por la aplicación. Con esta información pude llevar a cabo una investigación de las diferentes vulnerabilidades publicadas para cada uno de estos componentes, y determiné las herramientas que usaría para las pruebas posteriores.

Finalmente, con ayuda de la información obtenida y las características identificadas, ejecuté las pruebas de la configuración de la aplicación, de los controles de autenticación, autorización y sesión, así como de la validación de datos dentro del sistema.

3.4.2 Análisis estático de la aplicación

Siempre que me fue posible obtener el código fuente de la aplicación que estaba analizando, llevé a cabo un análisis estático del código mediante la ayuda de herramientas automatizadas como Microsoft FxCop o Gendarme para el análisis de código .NET, y RATS (Rough Auditing Tool for Security) para el análisis de código Perl y PHP entre otros.

Con este tipo de pruebas localicé las secciones de código más propensas a sufrir errores, analicé de forma detallada funcionalidades o secciones de interés particular dentro de la aplicación (por ejemplo, el código que implementaba la funcionalidad para subir archivos al servidor), identifiqué posibles puntos muertos o inalcanzables dentro del código, examiné los códigos de error que pude generar a partir de las pruebas dinámicas de la aplicación, y también verifiqué que se llevaran a cabo los procedimientos pertinentes para la validación de los datos que son proporcionados por el usuario. Por otro lado, al llevar a cabo este tipo de análisis identifiqué malas prácticas de programación y algunos otros malos hábitos que estaban siendo repetidos en más de un sistema.

3.4.3 Evaluación de impacto

Uno de los aspectos más importantes dentro de la administración de los recursos de TI de cualquier organización es la toma de decisiones para la remediación de los problemas que éstos pudieran presentar. De manera general las instituciones suelen contar con equipos de características de hardware diferentes, sistemas operativos diferentes e incluso plataformas de software distintas; por ello, la identificación y valoración de la importancia de los problemas de seguridad que encontré se volvió fundamental para determinar cuáles eran aquellos que presentaban el riesgo más grande para la información y/o los recursos de la organización y debían ser remediados primero.

Para llevar a cabo el proceso de evaluación de impacto, utilicé la calificación CVSS de cada problema reportado por las herramientas automatizadas de escaneo. El Common Vulnerability Scoring System (CVSS) es un estándar de la industria para la valoración de vulnerabilidades de TI que ayuda a priorizar y coordinar una respuesta al problema de seguridad identificado al otorgarle una calificación de impacto que se deriva de las características de sus propiedades.

La principal ventaja que obtuve al usar como base la calificación del Common Vulnerability Scoring System fue que pude unificar los resultados arrojados por las herramientas automatizadas, pues en más de una ocasión el impacto del mismo problema sobre una aplicación era valorado de forma diferente según la herramienta empleada.

Para aquellos problemas que no fueron detectados mediante el uso de una herramienta automatizada, propuse la utilización de la fórmula de las métricas base provista por el CVSS para estimar el impacto que representaba sobre el sistema analizado y llevar a cabo el proceso de priorización para la implementación de las soluciones.

Me decidí por calcular el valor usando sólo las métricas base debido principalmente al nivel de profundización que se requiere para poder analizar y determinar los componentes de las otras dos métricas opcionales que considera el CVSS (métricas temporales, y métricas de entorno).

3.4.3.1 Métricas del CVSS

Los grupos de métricas que considera este sistema de calificación son tres:

1. Base: *Representa las características intrínsecas de una vulnerabilidad, y es el único obligatorio para la generación de una calificación CVSS.*
2. Temporal: *Representa las características de una vulnerabilidad que cambian a lo largo del tiempo. (Es un elemento opcional)*
3. Entorno: *Representa las características de una vulnerabilidad que son particulares al entorno del usuario. (Es un elemento opcional)*

Cada uno de estos grupos produce un valor numérico entre 1 y 10 así como un vector (una representación textual abreviada) que refleja los valores usados para determinar la calificación final.

3.4.3.2 Métricas del grupo Base

Las métricas del grupo base se conforman por aquellas cualidades que determinan las características que permiten identificar cómo se puede tener acceso para aprovecharse de la vulnerabilidad y si existen condiciones adicionales necesarias para poder explotarla. Adicionalmente se incluyen tres métricas de impacto que miden cómo una vulnerabilidad afecta al activo de TI en caso de ser explotada; cada uno de estos impactos está definido de forma independiente como el grado de pérdida de confidencialidad, integridad y disponibilidad del recurso en cuestión.

1) Vector de acceso (AV): Aquí se define si para poder aprovecharse un problema de seguridad se requiere de *acceso local (L)*, *acceso a redes adyacentes (A)* o *acceso a la red (N)*. Este último suele referirse como “acceso remoto”, y mientras más remoto sea el acceso, mayor es la calificación de vulnerabilidad.

2) Complejidad de Acceso (AC): Define qué tan sencillo es llevar aprovecharse de un problema de seguridad. Los posibles valores son *alto (H)* cuando se requiere de

alguna técnica muy especializada; *medio (M)* cuando existen condiciones particulares o no tan especializadas que deben cumplirse, y *bajo (L)* cuando no existen condiciones extenuantes o especializadas para poder aprovecharse del problema. Mientras más baja es la complejidad, mayor es el valor para este punto.

3) Autenticación (Au): Identifica el número de veces que es necesario autenticarse al sistema para poder aprovecharse del problema de seguridad (sin importar la complejidad de dicha autenticación). Los valores posibles son: *múltiple (M)*, *única (S)*, *ninguna (N)*. Mientras menos autenticaciones sean requeridas, mayor es el valor para este punto.

4) Impacto en la Confidencialidad (C): El impacto se define como *ninguno (N)*, cuando la confidencialidad no se ve en riesgo; *parcial (P)* cuando es posible que se revele algún tipo de información; y *completo (C)* cuando es posible acceder a todos los recursos del sistema.

5) Impacto en la Integridad (I): El impacto se define como *ninguno (N)*, cuando no hay modificación sobre el sistema; *parcial (P)* cuando es posible modificar algunos datos pero no se tiene el control completo de ello; y *completo (C)* cuando es posible modificar cualquier tipo de archivo del sistema.

6) Impacto en la Disponibilidad (A): El impacto se define como *ninguno (N)*, cuando la disponibilidad del sistema no se ve afectado; *parcial (P)* cuando el rendimiento se ve afectado o existen interrupciones en el servicio; y *completo (C)* cuando es posible tirar el sistema o apagar el equipo y/o servicio.

3.4.3.3 Ecuación base

La ecuación base es la fundamentación del valor CVSS y se puede definir como:

$$BaseScore = redondeo\_a\_1\_decimal(((0.6 * impacto) + (0.4 * Explotabilidad) - 1.5) * f(impacto))$$

Donde :

$$\text{Impacto} = (10.41 * (1 - \text{ConfImpact}) * (1 - \text{IntegImpact}) * (1 - \text{AvailImpact}))$$

$$\text{Explotabilidad} = 20 * \text{VectorAcceso} * \text{ComplejidadAcceso} * \text{Autenticacion}$$

$$f(\textit{impacto}) = \begin{cases} 0 & \text{si Impacto es 0,} \\ 1.176 & \text{en cualquier otro caso} \end{cases}$$

$$\text{VectorAcceso} = \begin{cases} 0.395 & \text{si requiere acceso local,} \\ 0.646 & \text{si requiere acceso de red adyacente,} \\ 1.0 & \text{si es accesible desde la red} \end{cases}$$

$$\text{ComplejidadAcceso} = \begin{cases} 0.35 & \text{si es Alta,} \\ 0.61 & \text{si es Media,} \\ 0.71 & \text{si es Baja} \end{cases}$$

$$\text{Autenticacion} = \begin{cases} 0.45 & \text{si es múltiple,} \\ 0.56 & \text{si es única,} \\ 0.704 & \text{si no se requiere} \end{cases}$$

$$\text{ConfImpact} = \begin{cases} 0.0 & \text{si es Ninguno,} \\ 0.275 & \text{si es Parcial,} \\ 0.660 & \text{si es Completo} \end{cases}$$

$$\text{IntegImpact} = \begin{cases} 0.0 & \text{si es Ninguno,} \\ 0.275 & \text{si es Parcial,} \\ 0.660 & \text{si es Completo} \end{cases}$$

$$\text{AvailImpact} = \begin{cases} 0.0 & \text{si es Ninguno,} \\ 0.275 & \text{si es Parcial,} \\ 0.660 & \text{si es Completo} \end{cases}$$

La guía del CVSS ofrece varios ejemplos que consideran también las métricas temporales y de entorno para el cálculo de la calificación, por lo que de ser necesario se puede consultar dicha documentación.

3.5 Generación del boletín técnico

Una vez que conocía o pude determinar el impacto de los problemas encontrados, generé el boletín de seguridad con el reporte de las fallas encontradas y las posibles acciones y/o medidas que podían realizarse/adoptarse para solucionar o mitigar el riesgo en cuestión. Con la entrega de estos documentos involucré a los administradores y desarrolladores en el proceso de toma de conciencia de la seguridad de la información y el desarrollo seguro de aplicaciones web.

3.6 Remediación de los problemas identificados

El proceso de remediación de los riesgos presentes en los sistemas corrió a cargo de los administradores/desarrolladores responsables. Después de que les hice llegar los boletines técnicos correspondientes, llevé a cabo una reunión en la que por un lado presenté la valoración de seguridad del sistema que analicé y por otro resolví las dudas que tuvieron al respecto.

Posteriormente a ellos les asigné la planeación del tiempo de resolución de los riesgos presentados, así como la toma de decisión para asumir el riesgo que representaba algún problema para la aplicación web (previa justificación para asumir dicho riesgo). En este punto encontré que los factores que afectaron las decisiones que tomaron administradores y desarrolladores fueron la criticidad del sistema web, la facilidad para implementar las medidas de solución sugeridas, la factibilidad de implementar el cambio sin afectar el funcionamiento del sistema, y el calendario de actividades al que debían ajustarse.

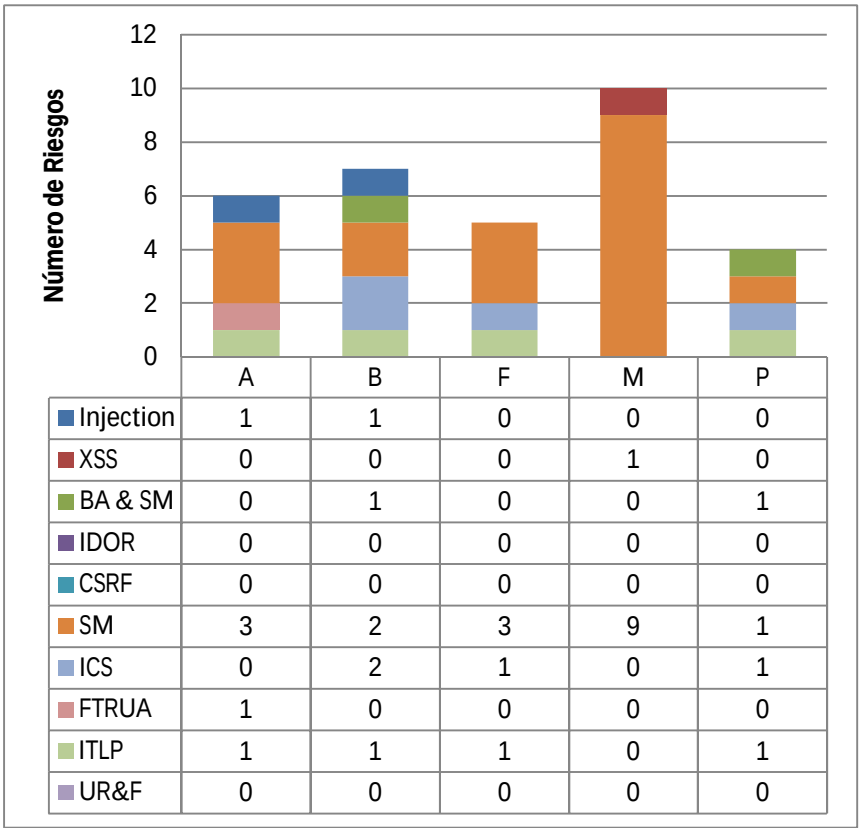
Por último, una vez que se cumplió el plazo acordado (o a petición del administrador correspondiente) nuevamente llevé a cabo los procesos de análisis de la aplicación y corroboré el estado de seguridad del sistema y me aseguré de que se remediaron los riesgos que reporté con anterioridad. Así, me cercioré de que el sistema se encontrara en el nivel de seguridad aceptable que determinamos para el mismo.

Capítulo 4. Resultados

En esta primera etapa del proyecto se auditaron y analizaron los sistemas que presentaban el riesgo más alto de ser comprometidos al ofrecer sus servicios a través de Internet y/o la red interna. Con relación a éstos se generó un total de 34 boletines técnicos para reportar los problemas y/o deficiencias encontradas durante la auditoría y el análisis de las aplicaciones.

En cumplimiento con las normativas y los requerimientos de confidencialidad de la institución y del proyecto en general, en este reporte se protege la identidad de los sistemas analizados y las deficiencias han sido agrupadas en categorías de acuerdo con la clasificación de los riesgos del proyecto OWASP. Adicionalmente, se presentan únicamente los resultados de una muestra representativa de los sistemas auditados y analizados, y se toma uno de los ejemplos más representativos para ilustrar el proceso de revisión y dar una idea general del proceso completo.

El estado de seguridad en que se encontraban originalmente los sistemas web seleccionados para la muestra, y analizados bajo el marco del top 10 de riesgos publicado por el Open Web Application Security Project (2010) se presenta en el gráfico 4_1 a continuación:



	A	B	F	M	P
Injection	1	1	0	0	0
XSS	0	0	0	1	0
BA & SM	0	1	0	0	1
IDOR	0	0	0	0	0
CSRF	0	0	0	0	0
SM	3	2	3	9	1
ICS	0	2	1	0	1
FTRUA	1	0	0	0	0
ITLP	1	1	1	0	1
UR&F	0	0	0	0	0

Gráfico 4_1 – Conteo de riesgos por aplicación analizada

Dentro de las aplicaciones web que aquí se reportan, se identificaron un total de 32 riesgos de seguridad, donde las aplicaciones F y P fueron las que presentaron el menor número de riesgos (5 y 4, respectivamente) mientras que la M fue la que presentó el mayor número de problemas, con 10 riesgos.

El promedio de problemas de seguridad, sin tomar en cuenta los extremos es de 6 riesgos por aplicación; sin embargo, de la tabla se puede observar que el 40% de las aplicaciones analizadas presentaron menos de 6 vulnerabilidades.

De acuerdo con el top 10 de OWASP, los riesgos que tienen un impacto severo para las organizaciones son la inyección de código (A1), la mala administración de sesión y autenticación (A3), y el almacenamiento criptográfico inseguro (A7); todos los demás riesgos son considerados de impacto moderado. Analizando el gráfico anterior (Gráfico 4_1), se puede apreciar que la mitad de las aplicaciones auditadas presentó riesgos de impacto severo: la aplicación “A” tuvo 1/6; la “B” 4/7; la “F” 1/5 y la “P” 2/4.

Lo anterior significa que del total de riesgos identificados (32) en los cinco sistemas analizados, sólo el 25% (8) son considerados de impacto severo. El gráfico 4_2 muestra el porcentaje de riesgos considerados de impacto severo para cada aplicación analizada:

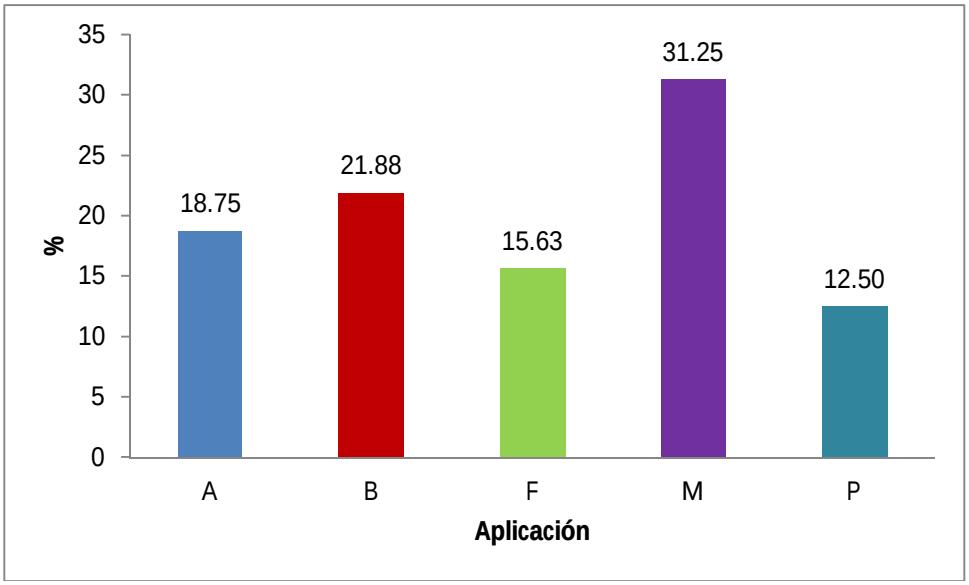


Gráfico 4_2 – Porcentajes de riesgos de impacto severo de acuerdo con OWASP

El principal problema de seguridad que presentaron los sistemas analizados fue la mala configuración (56%). Esto no resultó sorpresivo puesto que es una categoría que comprende una amplia gama de aspectos a considerar, que van desde el correcto aseguramiento del equipo en que se ejecuta la aplicación hasta la puesta a punto de las opciones de configuración propias del sistema web o la tecnología empleada para su desarrollo y funcionamiento.

A pesar de que los riesgos de Cross-Site Scripting (XSS) se colocaron en la parte baja de la tabla con un 6%, se debe recordar que aún cuando el proyecto de OWASP considera su impacto como moderado, la frecuencia con la que se presenta este riesgo (es el único riesgo que tiene el valor "very widespread" o "muy frecuente" para la frecuencia del riesgo dentro del top 10) provoca que sea considerado como uno de los principales riesgos por prácticamente todas las herramientas automatizadas para escaneo de vulnerabilidades.

El gráfico 4_3 muestra la forma en que estuvo dada la distribución completa de los riesgos encontrados:

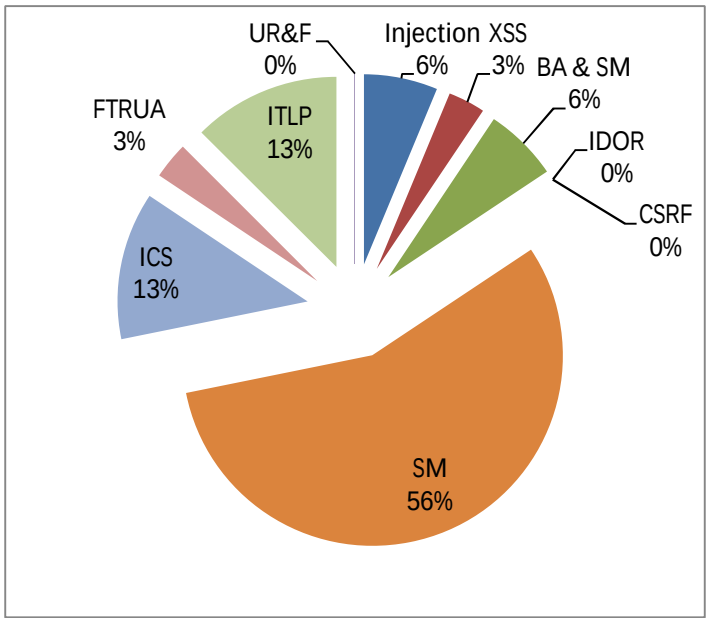


Gráfico 4_3 – Distribución por tipo de riesgo OWASP

Por su parte, el análisis estático de código de las aplicaciones que pudieron ser analizadas permitió identificar algunas malas prácticas de programación, entre las que destacaron:

- Falta de implementación de los mecanismos de validación de datos proporcionados al sistema por el usuario.
- Almacenamiento en claro de información sensible, particularmente cadenas de conexión a otros sistemas o componentes que complementan la funcionalidad de las aplicaciones.

Este análisis permitió determinar además que otra de las fuentes que originan riesgos de gran impacto en las aplicaciones se deben principalmente a la falta de mecanismos de validación de datos provenientes de fuentes externas al sistema, de manera que se vuelve necesario adoptar una postura en la que todos los datos son considerados como "no confiables" y deben ser revisados para cumplir con formatos, tipos, rangos, longitudes, etc.

Resulta importante mencionar también que para algunos sistemas fue particularmente evidente la reutilización de código en múltiples secciones de la aplicación. Sin embargo, la ventaja que se supone debe ofrecer esta práctica se vio opacada por la inmadurez del código dentro del contexto de la seguridad, pues al repetir código inseguro en varias secciones, aumentó el riesgo de la aplicación de manera considerable.

Por otro lado, al ser la mala configuración el principal problema encontrado en los sistemas, el proceso de implementación de las sugerencias de remediación para la mayoría de los riesgos se presentó como una actividad sencilla; sin embargo lo anterior no significó que el proceso estuviera exento de limitantes provocadas por factores como la calendarización de actividades y/o la criticidad del sistema. A pesar de que varias de las acciones podían implementarse en un período corto de tiempo, en varias ocasiones hubo que esperar para evitar que se afectaran los procesos de negocio de la institución o para ajustarse al calendario de mantenimiento existente para el equipo que almacena dicha aplicación.

Como ejemplo se presentan las revisiones realizadas al sistema F:

En el primer análisis realizado a la aplicación, se encontró que el sistema presentaba 3 malas configuraciones de seguridad, un problema de almacenamiento criptográfico inseguro y otro más de protección insuficiente de la capa de transporte.

Después del primer boletín técnico generado, se solucionaron los dos riesgos de mayor prioridad, y se volvió a analizar el sistema. Sin embargo, a pesar de que efectivamente se habían remediado dichos riesgos, la forma de implementación de la solución provocó la aparición de un nuevo riesgo en la categoría de malas configuraciones.

Después de comentada la situación con los desarrolladores, se dio solución al problema que apareció de mala configuración de seguridad. Dicha modificación al sistema dejó al descubierto un problema de Cross-Site Scripting, por lo que se trabajó para dar solución a dicho riesgo.

Finalmente, en la última revisión hecha al sistema, se habían solucionado los problemas anteriores y el estado de seguridad en que se encontraba el sistema presentaba sólo dos riesgos derivados de malas configuraciones de seguridad.

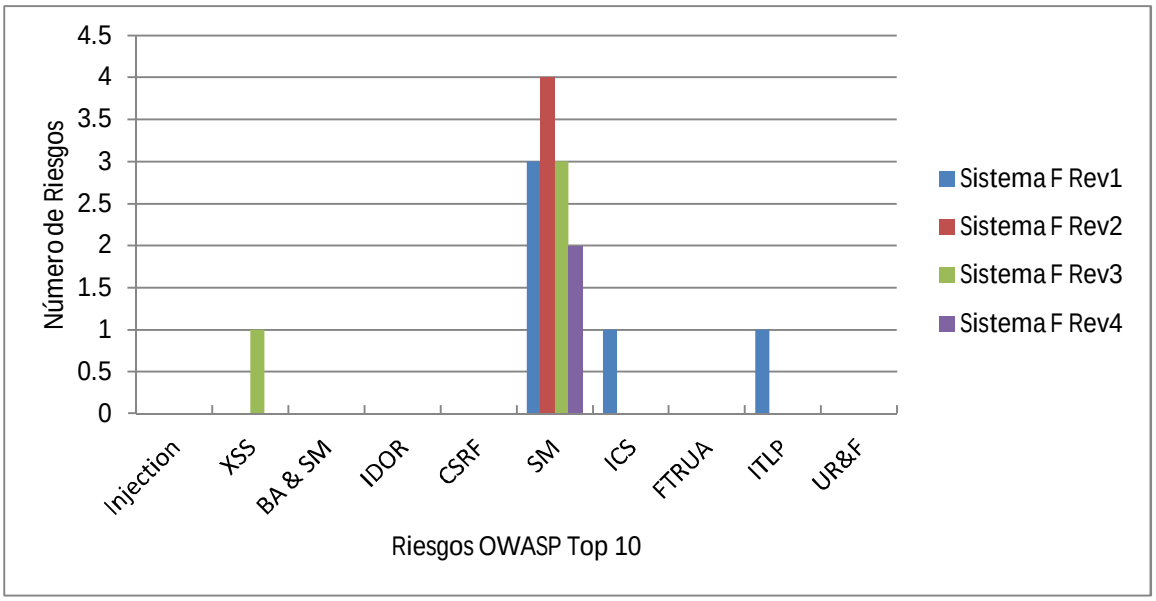


Gráfico 4_4 – Riesgos por revisión sobre el sistema F

Se determinó que el riesgo que representan los dos problemas de seguridad restantes se encuentran dentro del nivel de riesgo aceptable para la institución, por lo que serían considerados para su remediación durante una fase posterior de mantenimiento al equipo que aloja la aplicación. Así, el sistema se consideró seguro para la operación y funcionamiento.

Como respuesta y para fortalecer el proceso de integración de seguridad a los sistemas desarrollados dentro de la institución, Se elaboró el primer borrador sobre mejores prácticas de seguridad que incorpora recomendaciones tanto de configuración (para garantizar un nivel de seguridad mínimo de los equipos y sistemas) como de programación (para evitar algunas de las fallas más comunes que propician la aparición de problemas o huecos de seguridad en las aplicaciones).

Dicho documento tiene como base varias guías y publicaciones de reconocidos organismos internacionales en materia de seguridad informática, como el SANS Institute, el Center for Internet Security, el Departamento de Defensa de los Estados Unidos, el National Institute of Standards and Technology, así como los mismos proveedores de las tecnologías empleadas como Microsoft, Apache Software Foundation, entre otros.

Adicionalmente, también se generaron guías para la instalación y configuración de herramientas de interés particular para el área de sistemas de la institución, iniciando así la existencia de una base de conocimientos con la que podrán contar los desarrolladores para generar los futuros sistemas.

De esta forma, en esta primera etapa del proyecto se han identificado los problemas de seguridad presentes en los 10 sistemas de mayor riesgo y se han puesto en marcha los pasos para implementar los mecanismos de remediación correspondientes para reducir la

posibilidad de exponer a la institución a incidentes de seguridad informática. Se cuenta además con una base de conocimiento que conforme se desarrolla facilitará el apego a la normatividad interna, así como a estándares y/o mejores prácticas internacionales en caso de ser requerido.

4.1 Trabajo Futuro

Contar con una estrategia para la evaluación de las aplicaciones que se desarrollan dentro de la institución es sólo el primer paso hacia la creación de un ciclo de desarrollo de software seguro. Alcanzar ese objetivo es una tarea mucho más grande que el proceso de auditoría de las aplicaciones, y requiere de un cambio en la cultura de la organización para que exista un deseo común de priorizar la seguridad. Dicho cambio debe contemplar un análisis eficiente de los procesos y modelos de negocio de la organización, así como la capacitación efectiva del personal para que la selección de tecnologías empleadas sea la mejor opción para resolver una necesidad particular.

A medida que madura la metodología y se van incorporando mecanismos de evaluación dentro de los procesos de desarrollo de software con los que se cuenta, se espera que el balance de trabajo se distribuya de manera que la cantidad de actividades referentes a la seguridad de las aplicaciones comience a ser más alto dentro del área de desarrollo y menor dentro de la Dirección de Seguridad Informática.

De manera ideal, una vez que se cuente con la capacitación adecuada del personal y se hayan incorporado los mecanismos para evaluar el software al proceso de desarrollo del mismo, la función de la Dirección de Seguridad Informática será más de supervisión y guía (en caso de ser necesario) al tiempo que el proceso de "interiorización" de la seguridad en la construcción de aplicaciones se ha consolidado dentro del ciclo de desarrollo de software.

Conclusiones

Ante la creciente complejidad requerida por los sistemas web para ofrecer los servicios y funcionalidades que demanda una organización, la forma más eficiente de mantenerse por delante de los problemas de seguridad en las aplicaciones es construir software de forma segura desde el inicio.

El reto que enfrentan todas las instituciones que integren desarrollo de software en sitio radica en que la mayoría de los desarrolladores no cuentan con conocimientos sólidos de seguridad, y culturalmente el código seguro no suele ser identificado como una prioridad relacionada a la entrega de funcionalidad en tiempo, en forma y dentro del presupuesto.

Con la primera etapa del proyecto de auditoría y código seguro, se ha logrado sentar las bases para el desarrollo de una estrategia de evaluación de la calidad de los sistemas producidos bajo el contexto de la seguridad de la información, de manera que se pueda garantizar que éstos cumplen con los requerimientos de confidencialidad, integridad y disponibilidad necesarios dentro de la institución. En consecuencia, el proyecto contribuye a evitar que las aplicaciones (tanto aquellas basadas en web como las "stand alone") sean puestas en funcionamiento con una serie de deficiencias en los mecanismos de aseguramiento de los datos, que de ser explotadas pueden exponer información sensible de la institución a riesgos innecesarios como la revelación y/o robo de información confidencial.

Llevar a cabo la implementación del proyecto de auditoría y desarrollo seguro me permitió experimentar de primera mano que la seguridad informática es un proceso y no un producto. Dicho proceso requiere de una participación integral de todas las áreas involucradas en el procedimiento de creación del sistema, donde en esta primera etapa el trabajo recae principalmente en el área de seguridad informática de la institución para alertar y crear conciencia en desarrolladores y administradores sobre la seguridad de la información de los sistemas de los cuales son responsables.

Por otro lado, aprendí que un estado de seguridad no implica necesariamente cero riesgos, sino una combinación estable de controles que ofrezcan un nivel de seguridad aceptable a la vez que permitan un funcionamiento y ofrecimiento de servicios consistente, fluido y dentro de las medidas de usabilidad requeridas por la organización. Pude comprobar que la misma naturaleza del proceso de identificación y remediación de riesgos presenta grandes retos, como aquellos de involucran la toma de decisiones para asumir riesgos cuando no es factible aplicar los mecanismos de remediación debido al impacto que generan sobre el modelo de negocio de la institución.

Referencias

[1] “Fallas en el sistema de Check-in de Aeroméxico provoca caos en AICM”.
Periódico Excélsior en línea. (Última consulta octubre de 2011).
http://www.excelsior.com.mx/index.php?m=nota&id_nota=739030.

[2] 10 realidades de la seguridad de sitios web y su impacto en las empresas (Whitehat).
“10 Important Facts About Website Security and How They Impact Your Enterprise”.
(Última consulta: noviembre de 2011)
http://img.en25.com/Web/WhiteHatSecurityInc/WP10facts0111.pdf

[3] Artículo sobre las vulnerabilidades como función de la calidad del software.
“Vulnerability as a Function of Software Quality”. (Última consulta: septiembre de 2011).
https://www.giac.org/paper/gsec/647/vulnerability-function-software-quality/101493

[4] Artículo sobre un tipo de vulnerabilidad (Time Of Check, Time of Use) debida a suposiciones falsas acerca de un sistema.
“A Tour of TOCTTOUs” SANS Institute (2003) (Última consulta: noviembre de 2011)
http://www.sans.org/reading_room/whitepapers/securecode/tour-tocttous_1049

[5] Estadísticas de seguridad en sitios web hacia el invierno de 2011.
WhiteHat Website Security Report (Última consulta: octubre de 2011)
https://www.whitehatsec.com/home/assets/WPstats_winter11_11th.pdf?doc=WPstats_winter11_11th

[6] Página principal del Proyecto Abierto de Seguridad en Aplicaciones Web (OWASP)
Open Web Application Security Project (OWASP) (Última consulta: octubre de 2011)
https://www.owasp.org/index.php/Main_Page

[7] Página principal del proyecto para la guía de prueba de aplicaciones web OWASP
“OWASP Testing Guide Project” (Última consulta: septiembre de 2011)
https://www.owasp.org/index.php/Category:OWASP_Testing_Project

[8] Guía de prueba de aplicaciones web OWASP versión 3
“OWASP Testing Guide v3” (Última consulta: septiembre de 2011)
https://www.owasp.org/images/5/56/OWASP_Testing_Guide_v3.pdf

[9] 10 principales riesgos de seguridad en aplicaciones web (OWASP top 10)
“OWASP Top Ten Project” (Última consulta: septiembre de 2011)
https://www.owasp.org/index.php/Category:OWASP_Top_Ten_Project
http://owasptop10.googlecode.com/files/OWASP%20Top%2010%20-%202010.pdf

[10] Sistema Común de Calificación de Vulnerabilidades
Common Vulnerability Scoring System v2.0 (Última consulta: septiembre de 2011)
http://www.first.org/cvss/cvss-guide.pdf

[11] Artículo sobre los retos de la calidad del software.
“The Software Quality Challenge” (Última consulta: septiembre de 2011)
http://www.semat.org/pub/Main/PubsandRefs/Humphrey_SoftwareQuality.pdf

[12] Center for Internet Security benchmarks (Último acceso: noviembre de 2011)
http://benchmarks.cisecurity.org/en-us/?route=downloads.benchmarks

[13] Escáner de vulnerabilidades Nessus. (Último acceso: noviembre de 2011)
http://www.tenable.com/products/nessus

[14] Análisis estático de código .NET
“Microsoft FxCop” (Última consulta: agosto de 2011)
http://msdn.microsoft.com/en-us/library/bb429476%28v=vs.80%29.aspx

Anexo A. Guía de Mejores Prácticas para la fortalecer la seguridad de la información en los sistemas institucionales

Presentación

Como resultado del proceso de auditoría de sistemas web y código seguro, así como los correspondientes análisis dinámicos de los sistemas de la institución que ofrecen sus servicios a través de Internet y/o la red interna, se ha desarrollado una serie de recomendaciones y buenas prácticas que deben ser consideradas por el área de Desarrollo de Sistemas para la mejora de la seguridad de los sistemas web.

Bases de Datos

Base de datos MySQL

- ***Establecer contraseña para usuario root (o cuenta de administración si ésta ha sido renombrada)***

  Por defecto, la cuenta root de MySQL contiene una contraseña en blanco, por lo que justo después de instalar es absolutamente necesario crear y asignar una contraseña para dicho usuario. También es recomendable renombrar la cuenta para obtener un nivel adicional de ofuscación si una persona malintencionada intenta realizar un ataque de fuerza bruta sobre la contraseña de la cuenta root.

  Para cambiar la contraseña del usuario root, se debe ejecutar el siguiente comando:

  *UPDATE user SET password = PASSWORD('**contraseña**') WHERE user = 'root';*

  Donde ***contraseña*** es la contraseña en texto claro que se le asignará al usuario root.

  NOTA: La función PASSWORD() regresa una representación hash de la cadena especificada, y es la función que se utiliza para almacenar los hashes de las contraseñas en la tabla user.

  Para cambiar el nombre por defecto de la cuenta de administración (root), se deben ejecutar los siguientes comandos:

  *update user set user="**nombreCuenta**" where user="root";*

  flush privileges;

donde ***nombreCuenta*** es el nombre por el que se desea reemplazar el del usuario root.

- ***Eliminar la base de datos que se instala por defecto***

Por defecto, MySQL instala una base de datos de prueba llamada "test", misma que debe ser eliminada en caso de que aún exista. Para comprobar la presencia de dicha base de datos, se ejecuta el comando:

show databases like 'test';

Y en caso de que la base de datos aún se encuentre presente, se deberá eliminar mediante el comando:

drop database test;

- ***Eliminar cuentas anónimas o cuentas con contraseña en blanco***

Las cuentas anónimas se pueden entender como usuarios que no tienen nombre (''). Éstas cuentas permiten acceso por defecto y los permisos asignados a estas cuentas en ocasiones pueden ser usadas por otro usuarios. Evitar que se usen este tipo de cuentas garantiza que sólo los elementos autorizados puedan interactuar con la base de datos.
Por otro lado, las contraseñas en blanco permiten a un usuario acceder a la base sin necesidad de usar la contraseña para autenticarse, por lo que ponen en riesgo los beneficios de seguridad que establecen los mecanismos de autenticación.

Para verificar la existencia de cuentas anónimas se debe ejecutar el comando:

Select user from mysql.user where user='';

Para verificar la existencia de cuentas con contraseña en blanco, se debe ejecutar el comando:

Select user, password from mysql.user where length(password) = 0 or password = null;

Si alguna de estas consultas devuelve resultados es necesario eliminar las cuentas correspondientes.

- ***Restringir o deshabilitar el acceso remoto***

  Limitar la accesibilidad del socket de red de MySQL puede reducir la exposición derivada de una vulnerabilidad al evitar que hosts no autorizados puedan establecer comunicación con el servicio.

  Por defecto, MySQL usa el puerto 3306, por lo que existen dos posibles escenarios:

  1. Si el servidor de base de datos corre de forma independiente en un equipo distinto al de la aplicación, entonces se debe restringir el acceso a este puerto usando un FireWall de host, como por ejemplo IPTables.

  2. Si tanto el servidor de la aplicación como la base de datos corren sobre el mismo equipo, se deben deshabilitar las conexiones de red de MySQL para que en su lugar se usen los sockets de comunicación de UNIX. Esto evita que el servidor de MySQL esté esperando conexiones de red y por tanto, reduce la superficie de ataque al servidor.

  A pesar de que normalmente el archivo de configuración de MySQL contiene la directiva que previene las conexiones de red hacia el servicio, se debe verificar que el archivo my.cnf tenga la línea:

  Bind-address = 127.0.0.1

  O en versiones anteriores:

  Skip-networking

- ***Ejecutar MySQL como un usuario no privilegiado***

  La cuenta con la que se ejecuta el servicio de MySQL debe ser la de un usuario común para reducir el daño potencial al sistema operativo y otros procesos en caso de que llegara a suceder un ataque satisfactorio sobre MySQL. Por ello hay que asegurarse de que el servicio no lo ejecuta el usuario root.

  Para verificar con qué usuario se está ejecutando el servicio de MySQL, se puede usar el comando:

  ps auwwfx | grep mysql

- ***Revisar los privilegios de los usuarios***

  En apego al principio de mínimo privilegios, se debe verificar que los usuarios cuentan únicamente con los privilegios indispensables para poder realizar las acciones requeridas. Algunos de los aspectos que se deben cuidar son:

- **Permiso de GRANT**

  Este privilegio permite a los usuarios otorgar privilegios a otros usuarios. Esto debe estar limitado al administrador de la base de datos y a los dueños de los datos (tablas).

  Para ver qué usuarios tienen este privilegio, se pueden ejecutar las siguientes consultas:

  *Select * from user*
  where Grant_priv = 'Y';

  *Select * from db*
  where Grant_priv = 'Y';

  *Select * from host*
  where Grant_priv = 'Y';

  *Select * from tables_priv*
  where Table_priv = 'Grant';

- **Permiso ALTER**

  Este privlegio permite a los usuarios acceder para hacer cambios a la estructura de la base de datos a nivel global, de base de datos y de tabla.

  Para listar los usuarios con estos privilegios, se pueden realizar las siguientes consultas:

  *Select * from user*
  where Alter_priv = 'Y';

  *Select * from db*
  where Alter _priv = 'Y';

  *Select * from host*
  where Alter_priv = 'Y';

  *Select * from tables_priv*
  where Table_priv = 'Alter';

- **Permisos para insertar, eliminar, actualizar.**

  Estos privilegios pueden llegar a ser peligrosos si han sido otorgados a usuarios que no los requieren; por ello se debe de seguir el principio de mínimos privilegios.

  Para listar los usuarios con este tipo de privilegios, se pueden realizar las siguientes consultas:

  ```
  Select * from user where
  Select_priv  = 'Y' or Insert_priv  = 'Y'
  or Update_priv = 'Y' or Delete_priv  = 'Y'
  or Create_priv = 'Y' or Drop_priv    = 'Y'
  or Reload_priv = 'Y' or Shutdown_priv = 'Y'
  or Process_priv = 'Y' or File_priv    = 'Y'
  or Grant_priv   = 'Y' or References_priv = ‘Y'
  or Index_priv = 'Y' or Alter_priv = 'Y';

  Select * from host
  where Select_priv  = 'Y' or Insert_priv  = 'Y'
  or Create_priv = 'Y' or Drop_priv    = 'Y'
  or Index_priv = 'Y' or Alter_priv = 'Y';
  or Grant_priv   = 'Y' or References_priv = ‘Y'
  or Update_priv = 'Y' or Delete_priv  = 'Y';

  Select * from db
  where Select_priv  = 'Y' or Insert_priv  = 'Y'
  or Grant_priv   = 'Y' or References_priv = ‘Y'
  or Update_priv = 'Y' or Delete_priv  = 'Y'
  or Create_priv = 'Y' or Drop_priv    = 'Y'
  or Index_priv = 'Y' or Alter_priv = 'Y';
  ```

- ***Cifrar la información sensible mediante las funciones integradas en MySQL***

  Siempre que la base de datos guarde información sensible, ésta debe almacenarse de forma cifrada. MySQL provee funciones inherentes al manejador de bases de datos que proveen la funcionalidad para cifrar y descifrar información mediante AES.

  Se debe recordar, sin embargo, las **contraseñas y otra información sensible** que se provee como argumento a las funciones de cifrado **se envían al servidor de MySQL como “texto en claro”** a menos que se use una conexión SSL. Además, **dichos valores aparecerán en cualquier bitácora (log) de MySQL que registre la operación**, por lo que para poder evitar este tipo de exposición se pueden cifrar del lado del cliente antes de enviarlos al servidor. Las mismas

consideraciones aplican para las llaves de cifrado, por lo que se puede forzar a que las aplicaciones usen procedimientos almacenados para cifrar y descifrar esos valores del lado del servidor.

Más información sobre los mecanismos de cifrado y compresión se puede encontrar en:

http://dev.mysql.com/doc/refman/5.1/en/encryption-functions.html

- ***Activar las bitácoras del manejador de la base de datos***

Aunque la mayor parte de los scripts para el inicio de MySQL ya traen configuradas las opciones de registro en bitácoras, se debe garantizar que éstas se están generando (normalmente en el directorio /var/log/mysql/) pues esta información es de gran importancia para el análisis de patrones y detección de incidentes de seguridad.

Dentro del archivo de configuración de MySQL se deben configurar las rutas de las bitácoras que llevarán el registro de los sucesos de la base de datos. De manera general, la bitácora activada por defecto es la de errores (log_error), y dependiendo del desempeño requerido en la base de datos, se puede activar la bitácora general (general_log_file) en el archivo *my.cnf*.

Para conocer más acerca de las bitácoras de un servidor MySQL, se puede consultar el recurso:

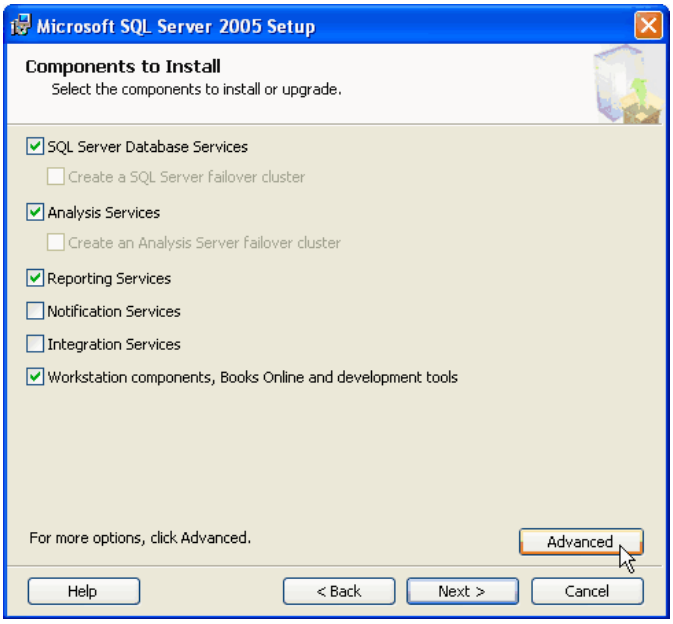
http://dev.mysql.com/doc/refman/5.1/en/server-logs.html

SQL Server

Nota: Estas recomendaciones toman como base SQLServer 2005, pero en su mayoría aplican también a versiones más recientes de la herramienta.

- ***Minimizar el área de ataque***

Es indispensable determinar desde la misma de instalación de SQL Server 2005 los componentes que se van utilizar dependiendo de las necesidades de la organización, lo que permitirá limitar aún más las posibilidades de que un ataque pueda ser satisfactorio.



Si posteriormente se decide habilitar características opcionales, ya sea por servicio o por instancia se podría emplear la utilidad *SQL Server 2005 Surface Area Configuration*.

Las opciones recomendadas para este asistente son:

Opción	Acción / Configuración Recomendada
Ad Hoc Remote Queries	Deshabilitar consultas Ad Hoc Remotas si éstas no son requeridas.
CLR Integration	Deshabilitar integración CLR si no se requiere.
DAC	Deshabilitar remote Dedicated Administrator

	Connection si no se requiere.
Database Mail	Deshabilitar Database Mail donde no se requiere el envío de mensajes.
Native XML Web Services	No configurar XML Web Services endpoints si no son requeridos.
OLE Automation	Deshabilitar automatización OLE si se requiere.
Service Broker	No configurar Service Broker endpoints si no se requieren.
SQL Mail	No habilitar SQL Mail si no es requerido o si es posible usar Database Mail en su lugar.
Web Assistant	Deshabilitar Web Assistant si no es requerido
xp_cmdshell	Deshabilitar el procedimiento almacenado xp_cmdshell si no es requerido.
Ad Hoc Data Mining	Deshabilitar las consultas ad hoc data mining si éstas no son requeridas.
Anonymous Connections	Deshabilitar las conexiones anónimas a los servicios de Análisis si éstos no son requeridos.
Linked Objects TO other instances	Deshabilitar los objetos ligados a otras instancias si éstos no son requeridos.
Linked Objects FROM other instances	Deshabilitar los objetos ligados desde otras instancias si éstos no son requeridos.
User-Defined Functions	Deshabilitar el cargado de funciones COM definidas por el usuario si éstas no son requeridas.
Scheduled Events and Report Delivery	Deshabilitar scheduled events and report delivery si no es requerido
Web Service and HTTP Access	Deshabilitar Web Service and HTTP access si no es requerido
Windows Integrated Security	Habilitar la seguridad integrada de Windows para las conexiones de fuentes de datos de reporte.

- ***Ejecutar los servicios de SQL Server mediante cuentas limitadas***

SQL Server 2005 se ejecuta como un conjunto de servicios de Windows, a la vez que cada servicio puede ser configurado para utilizar su propia cuenta de servicio. Cuando se realice la configuración de las cuentas de servicio, se debe considerar el principio del menor privilegio; esto es, NO DEBEN ejecutarse los servicios de SQL Server con cuentas que presenten privilegios administrativos o de red como las siguientes:

- Cuenta Network Service.
- Cuenta Local System.
- Cuenta de usuario local con privilegios administrativos.
- Cuenta de dominio con privilegios administrativos.

Las opciones más recomendables son:

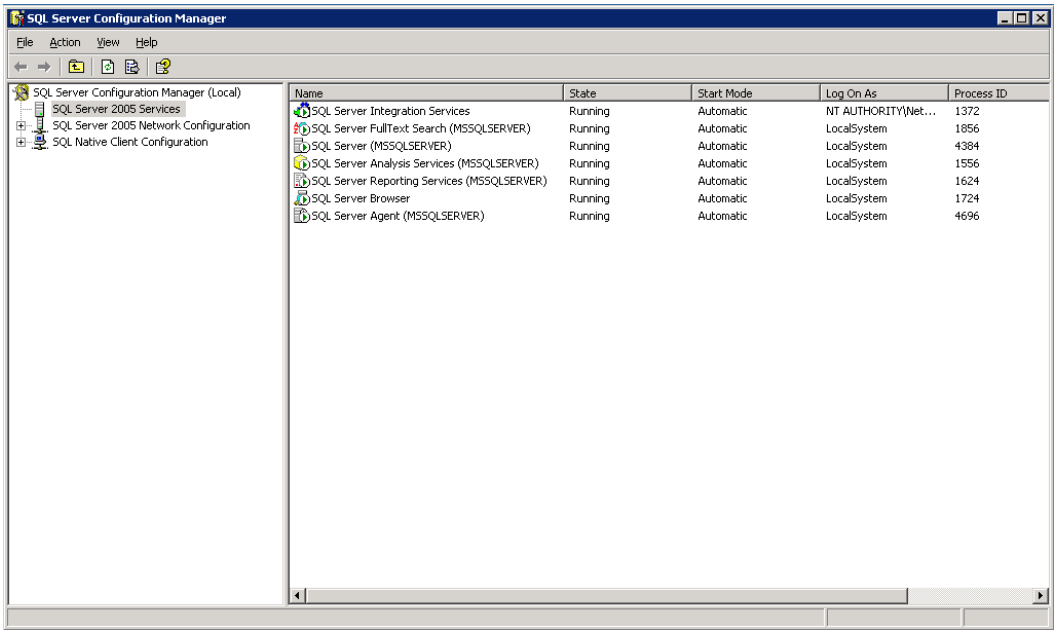
- Utilizar una cuenta de usuario local o una cuenta de usuario de dominio que no tenga privilegios administrativos. Si el servidor en el que se está ejecutando SQL Server es parte de un dominio y se debe acceder a recursos de dicho dominio como carpetas compartidas o realizar otras conexiones a otros servidores de SQL Server, la mejor opción es utilizar una cuenta de dominio.

- Si el servidor no es parte de un dominio (por ejemplo, un servidor ejecutado en la red de perímetro o DMZ, en una aplicación Web), o no necesita acceder a recursos de dominio, es recomendable utilizar una cuenta de usuario local que no tenga privilegios de administrador. También se debería utilizar una cuenta individual por cada uno de los servicios de SQL Server.

De esta forma, si algún servicio de SQL Server fuera comprometido la exposición del compromiso estaría limitada, además, se debe establecer una política sobre el cambio de contraseñas de dichas cuentas.

No es recomendable asignar ningún privilegio especial a la cuenta de servicio de SQL Server, estos serán asignados por la membrecía de grupos; también, se deben administrar los privilegios a través de los grupos creados a partir de la instalación de SQL Server y no a través de cuentas de usuario individual.

Siempre se debe utilizar la herramienta *SQL Server Configuration Manager* para cambiar las cuentas de servicio.



Para conocer más sobre el cambio de las cuentas de servicio se sugiere la lectura del siguiente documento:

- *How to change SQL Server service account*
  http://www.sqlserver.in/index.php/administration/35-administration-queries/111-how-to-change-sql-server-service-account.html

- ***Modo de autenticación***

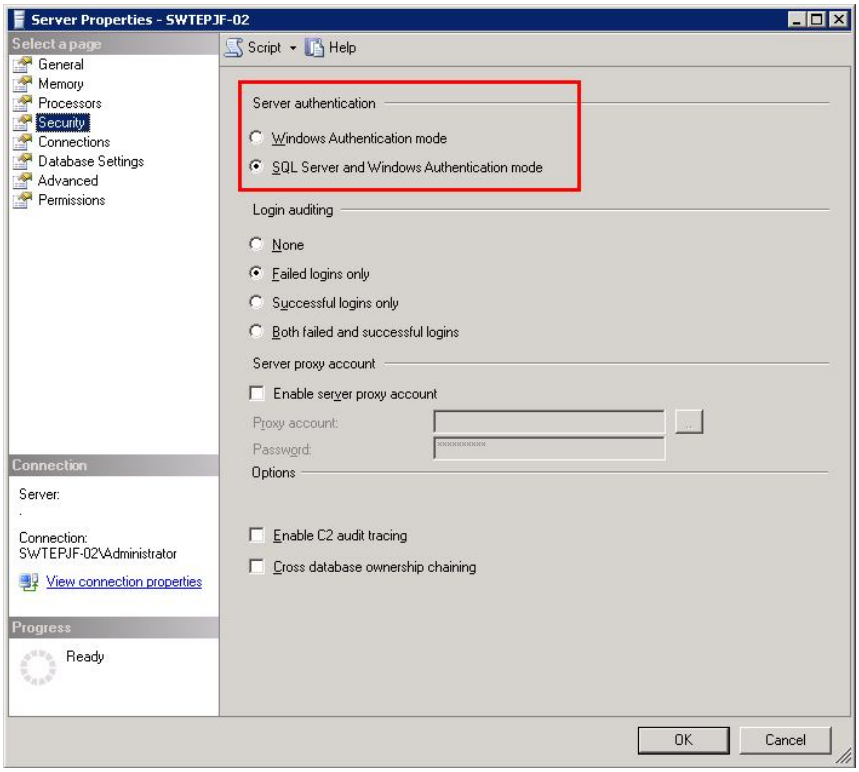
  SQL Server tiene dos modos de autenticación: *Autenticación de Windows* y *Autenticación en Modo Mixto*.

  - En modo de Autenticación de Windows, es posible iniciar sesión en SQL Server a través de cuentas de grupo y usuario de Windows específicas. Las credenciales de Windows son utilizadas en el proceso; esto es, ya sea credenciales NTLM o Kerberos. Las cuentas de Windows utilizan una serie de mensajes cifrados para autenticarse con SQL Server; ninguna contraseña es pasada a través de la red durante el proceso de autenticación.

  - En Autenticación en Modo Mixto, tanto las cuentas de Windows y las cuentas especificas de SQL Server (conocidas como logins) son permitidas para iniciar sesión. Cuando se utilizan logins de SQL, las contraseñas de dichos logins son pasadas a través de la red para realizar la autenticación. Esto hace que los logins de SQL sean menos seguros que los de Windows.

  De acuerdo con lo anterior, es recomendable siempre utilizar el modo de Autenticación de Windows cuando sea posible, y solo utilizar la Autenticación en Modo Mixto cuando la aplicación no soporte la Autenticación de Windows. La ventaja de utilizar el método de Autenticación de Windows es que es posible aplicar políticas de contraseñas a dichas cuentas y facilita la administración de las identidades.

  Si se instala el SQL Server en modo de Autenticación de Windows, la cuenta *sa* es deshabilitada y el sistema le establece una contraseña aleatoria. Si posteriormente se decide cambiar a la Autenticación en Modo Mixto y rehabilitar la cuenta *sa*, no se podrá conocer la contraseña, por lo que es recomendable cambiar dicha contraseña posterior a la instalación.



Para cambiar el Modo de Autenticación de SQL Server se puede utilizar la herramienta *Microsoft SQL Server Management Studio* o seguir las opciones descritas en:

- *How to: Change Server Authentication Mode*
  http://msdn.microsoft.com/en-us/library/ms188670.aspx

- *How to: Connect to SQL Server Using SQL Authentication in ASP.NET2.0*
  http://msdn.microsoft.com/en-us/library/ff648340.aspx

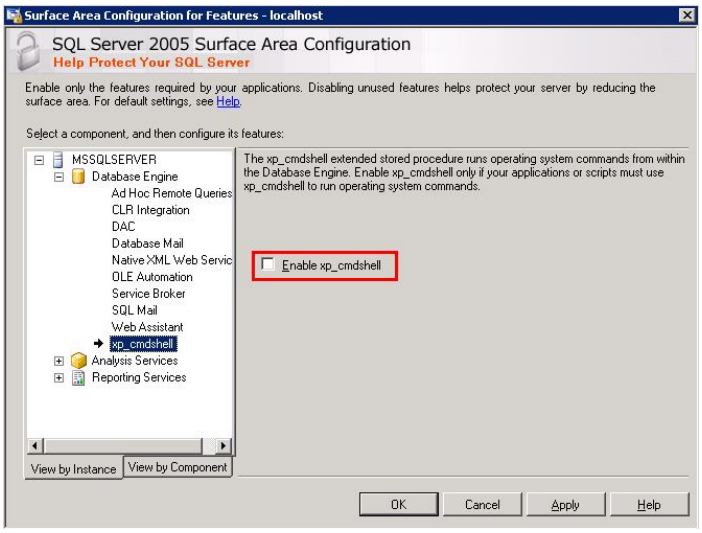
- *How To: Connecto to SQL Server Using Windows Authentication in ASP.NET 2.0*
  http://msdn.microsoft.com/en-us/library/ff647396.aspx

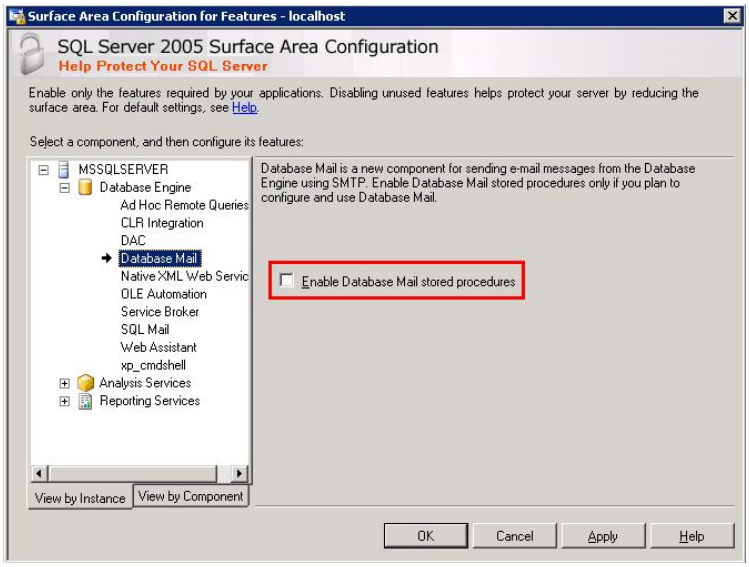
o ***No habilitar los stored procedures de sistema***

SQL Server utiliza stored procedures de sistema para realizar algunas tareas administrativas. Estos procedimientos la mayoría de las veces comienzan con el

prefijo *xp_* o *sp_*. Debido a que algunos de estos stored procedures interactúan con el sistema operativo o ejecutan código fuera de los permisos de SQL Server normales, que pueden constituir un riesgo de seguridad.

Los stored procedures de sistema como *xp_cmdshell* o *sp_send_dbmail* están deshabilitados de forma predeterminada y deberían permanecer así a menos que exista una razón para utilizarlos.





En caso de que se requieran usar estos u otros tipos de procedimientos almacenados de sistema, es recomendable usar el asistente de configuración "Surface Area Configuration" disponible a través de las herramientas de configuración de SQL Server 2005.

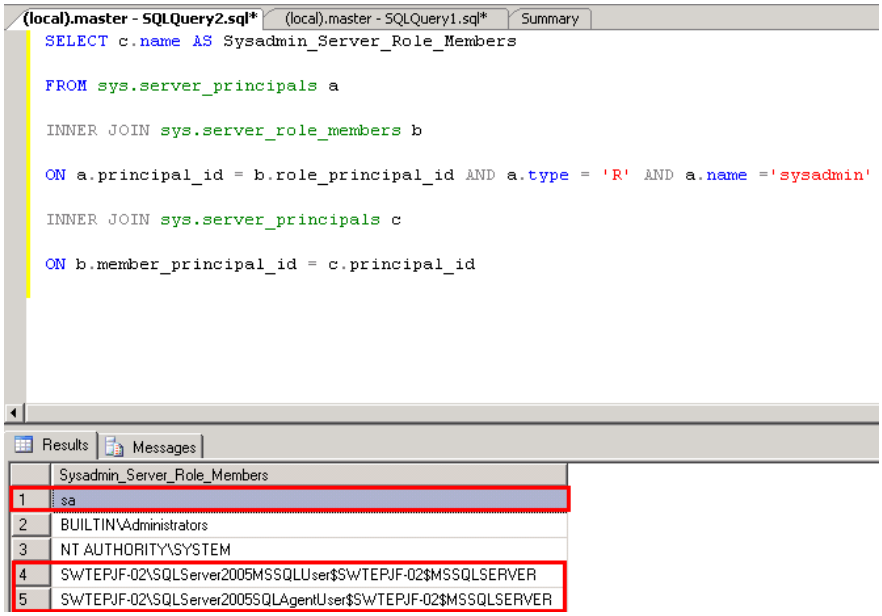
- ***Emplear usuarios con privilegios limitados en la base de datos***

Es altamente recomendable que el usuario que se emplee para conectarse a la base de datos de algún sistema tenga privilegios de acuerdo a lo que se requiera. Esto es, si el sistema sólo tiene funciones de extraer información, es recomendable que dicho usuario únicamente cuente privilegios de ***db_datareader*** y de esta forma lo único que pueda ejecutar sea la instrucción SELECT sobre cualquier tabla o vista de la base de datos. De esta forma, en caso de que la cuenta sea comprometida, la exposición a la información estaría limitada.

La contraseña establecida para este usuario tiene que ser compleja y con una longitud considerable, con el propósito de que sea más difícil descifrar en caso de algún ataque de fuerza bruta o de diccionario.

Los permisos se pueden otorgar a través de la herramienta *Microsoft SQL Server Management Studio*.

Adicionalmente, es necesario revisar la existencia de la cuenta de administrador por defecto (sa) así como los usuarios con privilegios de sysadmin.

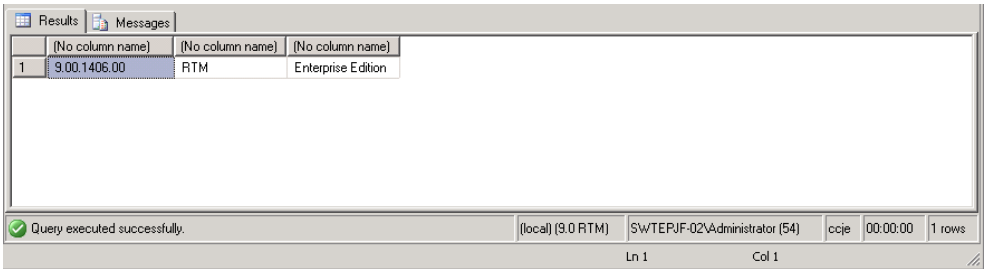


- ***Instalar las actualizaciones de seguridad más recientes relacionadas con el manejador de base de datos SQL Server 2005: Service Pack y hotfixes.***

Para conocer cuál es la versión y edición actual de SQL Server se puede realizar la siguiente consulta en el manejador de base de datos:

```
SELECT      SERVERPROPERTY('productversion'),      SERVERPROPERTY
('productlevel'), SERVERPROPERTY ('edition')
```

A continuación se muestra el resultado de la consulta:



Con base en el resultado se define el nivel de Service Pack que presenta el manejador SQL Server 2005 y puede encontrarse en cualquiera de los documentos mencionados a continuación:

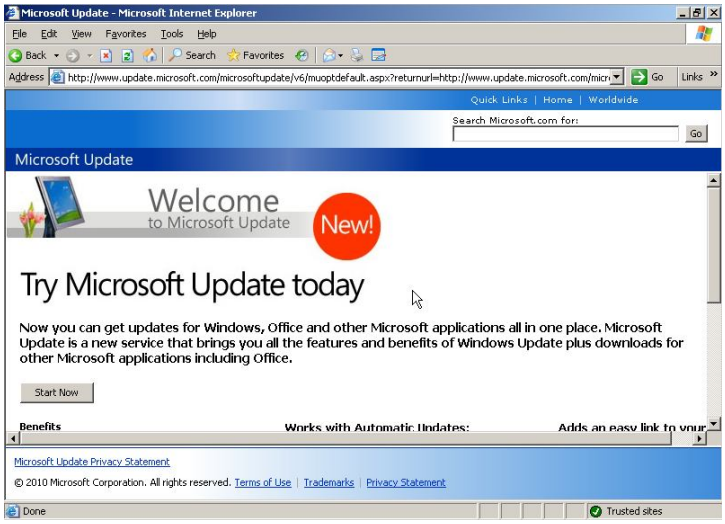
- *How to identify your SQL Server version and edition*
  http://support.microsoft.com/kb/321185/en-us

- *SQL Server Version Builds*
  http://www.bigdatabaselist.com/wiki/SQL_Server_Version_Builds

- *SQL Server Builds*
  http://sqlserverbuilds.blogspot.com/

Existen dos formas de aplicar las actualizaciones de seguridad:

- **Manual**. Empleando herramientas como Windows Update o Microsoft Update. La herramientas más recomendable es *Microsoft Update* ya que permite buscar e instalar actualizaciones relacionadas con el sistema operativo, Microsoft Office y aplicaciones como IIS, SQL Server, etc.



- **Automática.** Empleando herramientas como WSUS (Windows Server Update Services) o SCCM (System Center Configuration Manager) que permiten la instalación automática o semiautomática de actualizaciones de seguridad.

  - *Windows Server Update Services*
    http://technet.microsoft.com/en-us/wsus/default.aspx

  - *System Center Configuration Manager*
    http://www.microsoft.com/systemcenter/en/us/configuration-manager.aspx

- ***En la medida de lo posible, actualizar el manejador de base de datos a la versión más reciente.***

  SQL Server 2005 ya es un producto que tiene 5 años de haber sido liberado al mercado y su última actualización más importante el Service Pack 3 fue liberada el 15 de diciembre de 2008, hace casi dos años. También, es importante señalar que la fase de soporte técnico principal de SQL Server 2005 ya caducó (los 5 primeros años de vida del producto de un total de 10) y comenzó con su fase de soporte técnico extendido (los siguientes 5 años).

  Es importante señalar que las nuevas versiones del manejador de base de datos como SQL Server 2008 R2 presentan mejores características de seguridad, de alta disponibilidad y desempeño, por lo que sería recomendable utilizar este tipo de versiones de SQL.

  A continuación se mencionan algunos documentos sobre el ciclo de vida de SQL Server 2005:

  - *Ciclo de vida del soporte técnico de Microsoft*
    http://support.microsoft.com/gp/lifecycle/es-es

  - *Ciclo de vida del soporte técnico de Microsoft – SQL Server 2005*
    http://support.microsoft.com/lifecycle/?p1=2855

- ***Aplicar políticas de contraseñas***

  Si se emplea Autenticación de Windows es posible usar políticas de seguridad que se deben aplicar a nivel de sistema operativo. Estas políticas pueden ser aplicadas mediante las herramientas *Domain Security Policy* (en el caso de que el equipo pertenezca a dominio) o *Local Security Policy* (en el caso de que sea un equipo independiente). Las bitácoras de SQL también soportan la aplicación

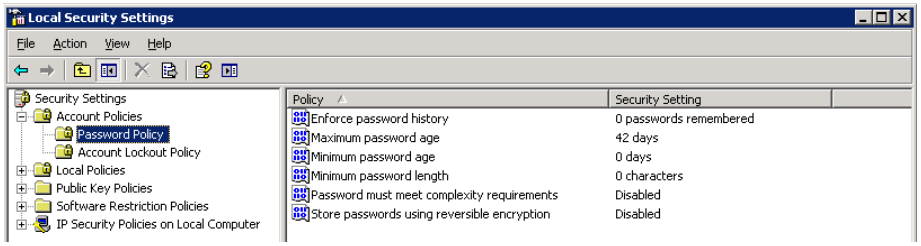
de políticas de contraseñas siempre y cuando el sistema operativo donde se instale sea Windows Server 2003 o superior. Las políticas de inicio de sesión caen en dos categorías: Directivas de Contraseñas y Directivas de Bloqueo de Cuenta.

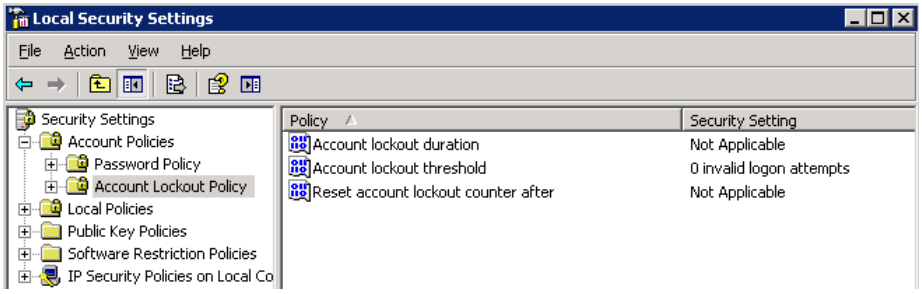
Las Directivas de Contraseñas incluyen:

- Forzar el historial de contraseñas.
- Vigencia mínima y máxima de la contraseña.
- Las contraseñas deben cumplir con los requerimientos de complejidad.
- Almacenar las contraseñas utilizando cifrado reversible (Nota: esta configuración no aplica a SQL Server).

Las Directivas de Bloqueo de Cuenta incluyen:

- Umbral de bloqueos de la cuenta (Número de inicios de sesión inválidos antes del bloqueo).
- Duración del bloqueo de cuenta (Cantidad de tiempo que estará bloqueada)
- Restablecer la cuenta de bloques después de *n* minutos.





Para saber más acerca de las políticas de cuenta, se sugiere la lectura del siguiente documento:

- Account Passwords and Policies in Windows Server 2003 http://technet.microsoft.com/en-us/library/cc783860%28WS.10%29.aspx

Servidores Web

Servidor web Apache 2.x

- ***Utilizar los binarios precompilados para la versión del Sistema Operativo***

  Salvo situaciones muy particulares, se recomienda usar los binarios del servidor Apache proporcionados por quienes ofrecen el Sistema Operativo sobre el que se montará el servidor. Dentro de las ventajas que de usar dichos binarios, se encuentran:

  - ✓ Facilidad de instalación.
  - ✓ Adecuación a detalles y configuraciones particualres del sistema operativo.
  - ✓ Los binarios han sido probados y han pasado por pruebas de calidad.
  - ✓ Las actualizaciones de seguridad para dichas particularidades son ofrecidas por quienes proveen el sistema operativo, por lo que hay que buscar en menos lados.
  - ✓ La aplicación de los parches de seguridad serán más sencillos de instalar, además de que quienes ofrecen el sistema operativo ya habrán verificado el problema, verificado la firma de la descarga, evaluado el impacto, etc.
  - ✓ Se pueden obtener actualizaciones de forma automática, reduciendo el margen de riesgo.

- ***Deshabilitar los módulos de apache que no se requieren***

  Es importante deshabilitar los módulos que no son utilizados por el servidor para reducir el área de impacto del servidor web, al tiempo que se mejora el desempeño del mismo. Si se dejan habilitados módulos que no se usan, se proveen rutas de ataque potenciales en contra del servidor, por lo que se deben definir de forma clara los módulos que se requieren para el funcionamiento del sistema.

  Si no se está seguro de si se requiere o no un módulo, se puede consultar la documentación de Apache en ***http://httpd.apache.org/docs/2.2/mod/*** y posteriormente deshabilitar el módulo para probar el funcionamiento del servidor.

- ***Definir las cuentas de usuario y grupo para Apache***

  Una de las formas de reducir la exposición de un servidor web a un ataque es crear un id y un grupo únicos para la aplicación del servidor, de forma que no se da acceso innecesario a otros servicios. El userid y el grupo “nobody” NO debe

ser usado para ejecutar el servidor web, pues dicha cuenta suele ser usada por otros servicios que se ejecutan como demonios dentro del sistema.

Por otro lado, el userid usado para el usuario de apache debe ser un valor único entre 1 y 499, pues estos ids bajos no son utilizados por usuarios resgulares. **Como medida extra de seguridad** y bajo el supuesto de que dicha cuenta jamás será usada para acceder y tener acceso a la consola, se sugiere no crear un directorio home ni un Shell válido a la cuenta del servicio de apache. Un ejemplo para crear dicha cuenta sería:

```
# groupadd apache
# useradd apache -g apache -d /dev/null -s /sbin/nologin
```

- ***Restringir el acceso al directorio raíz***

Las directivas "Allow" y "Deny" otorgan o restringen el acceso respectivamente. Sin embargo, la directiva "Order" suele ser un poco más compleja de entender, especialmente porque en este caso el orden sí importa. Esta directiva indica si primero se evaluarán las directivas "allow" o primero las directivas "deny", por lo que se debe ser cuidadoso.

Para evitar cualquier forma de ***directory traversal*** que se salga de la ruta especificada bajo "document root", se debe colocar una directiva que restrinja el acceso al directorio raíz del sistema operativo. Se debe recordar que la característica de acceso por defecto de Apache indica que menos que se haga algo para evitarlo, si el servidor es capaz de encontrar la ruta hacia un archivo a través de las reglas de mapeo de la URL, entonces lo hará disponible a los clientes.

La directiva que se debe implementar es:

```
<Directory />
Options None
AllowOverride None
deny from all
</Directory>
```

Los parámetros "order" y "allow/deny" deben ser aplicados en cualquier lugar del sitio web que se desee proteger. Es recomendable usar, siempre que sea posible, direcciones IP para prevenir ataques de tipo DNS spoofing.

- ***Control de funcionalidad de directorios mediante la directiva "Options"***

Esta medida se suma a la "seguridad por capas" y previene la habilitación accidental de algún servicio o característica. La directiva "Options" controla las funciones extendidas que aplica el servidor web a los archivos y/o directorios.

Entre las opciones más importantes se encuentran:

- ***All*** – Indica que todas las funcionalidades extras están disponibles excepto "Multiviews".
- ***ExecCGI*** - Permite la ejecución de scripts CGI dentro del directorio. Esta caracterísitica **sólo debe aplicarse al directorio cgi-bin asignado.**
- ***FollowSymLinks*** - Esta opción le permite al servidor seguir las ligas simbólicas que se encuentran en el directorio. Si se deben usar ligas simbólicas, se debe tener en consideración que sería posible para un atacante obtener acceso a áreas fuera de "root document" si esta opción está activada.
- ***Indexes*** - Esta opción le indica al servidor que debe crear de forma automática una página que liste todos los archivos dentro del directorio si no existe una página por defecto (como por ejemplo un archivo index.html). **El listado de directorios no debe ser permitido**, puesto que revela demasiada información a un atacante (como pudiera ser la estructura de directorios, convenciones de nombrado de archivos, módulos adicionales, etc). NOTA: Esta directiva es redundante si se ha deshabilitado el módulo autoindex (mod_autoindex).
- ***AllowOverride*** - Indica al servidor cómo manejar el control de acceso desde un archivo *.htaccess*. Cuando el servidor encuentra un archivo .htaccess, éste debe saber qué directivas declaradas dentro de ese archivo pueden sobreescribir la información de acceso anterior.
  Cuando a esta directiva se le da el valor "None", entonces se ignoran los archivos .htaccess por completo, por lo que el servidor ni siquiera intentará leer el archivo.
  Cuando se le da el valor "All" cualquier directiva que contenga el contexto htaccess se permite dentro de dichos archivos.

  A pesar de que la funcionalidad de los archivos htaccess es relevante, desde una perspectiva de seguridad, su presencia descentraliza los controles de acceso del archivo de configuración del servidor y hace más compleja la administración de la seguridad dentro del servidor.

- ***Protección contra el filtrado de información***

Poder identificar los detalles del servidor incrementa de forma significativa la eficiencia de un ataque debido a que las vulnerabilidades de seguridad suelen depender mucho de versiones específicas de un software o sus configuraciones.

Una forma de proteger un servidor Apache es limitar la información que provee acerca de la versión del servidor mediante las siguientes directivas:

- ***Servertokens*** - Esta configuración ayuda a esconder la versión del software del servidor web, así como las versiones de los módulos. El

valor recomendado es "Prod" (ProductOnly), que sólo muestra la leyenda "Apache" en la cabecera de respuesta HTTP del servidor.

Si se desea cambiar el token del servidor por completo, se debe usar ModSecurity con la directiva SecServerSignature

- ***ServerSignature*** - Para proteger la versión del software que usa el servidor, también se debe cuidar que no se despliegue información en las páginas de error que se generan dentro del sistema. La directiva "ServerSignature" indica a Apache que anexe cierta información en el pié de las páginas de error. Si no se deshabilita esta opción, se reduce la ventaja adquirida al cambiar la directiva "ServerTokens".

Se debe recordar que si no existen dichas directivas en el archivo de configuración, el valor por defecto es "Full" para ServerTokens y On para ServerSignature, por lo que deben agregarse o modificar su valor a :

```
ServerTokens Prod
ServerSignature Off
```

- ***Usar páginas de error personalizadas***

Cada tipo de servidor web tiene un estilo distinto de páginas de error. El servidor envía estás páginas en situaciones en las que, por ejemplo, no se puede encontrar una página. Debido a ello, un atacante puede determinar el tipo de software que usa el servidor al generar una solicitud a una página inexistente.

Existen dos formas de evitar el despliegue de esta información:

1. Editando las páginas de error para que todas tengan un estilo que concuerde con la plantilla del sitio web. Esto puede incluir cambio del esquema de colores y la alteración del texto del mensaje que se despliega.
2. Para efectos de conseguir un engaño, editar las páginas de error de forma que éstas imiten exactamente el estilo de un servidor web distinto.

Las directivas que deben agregarse son, por ejemplo:

```
ErrorDocument 400 /error400.html
ErrorDocument 401 /error401.html
ErrorDocument 403 /error403.html
ErrorDocument 404 /error404.html
ErrorDocument 405 /error405.html
ErrorDocument 500 /error500.html
```

Donde cada número identifica el número de error y se provee la ruta que contiene la página personalizada para dicho error.

- ***Bitácoras***

  Las bitácoras del servidor son un recurso invaluable por muchas razones. Éstas pueden ser usadas para determinar cuáles son los recursos que se usan más, detectar cualquier problema potencial antes de que se convierta en un problema serio, y más importante, permiten detectar comportamientos anormales que pudieran ser un indicio de que un ataque ocurrió o está por ocurrir.

  Por lo anterior, es importante considerar las siguientes directivas para el registro de bitácoras:

  - ***LogLevel*** - Esta directiva controla la cantidad de información que se registra en la bitácora de error, en un estilo por niveles similar al de la herramienta syslog (*emerg, alert, crit, error, warn, notice, info* y *debug*). **Se sugiere usar el valor "notice"** para que toda la información con nivel de "notificación" o superior sea registrado en la bitácora.

  - ***ErrorLog*** - Esta directiva establece el nombre del archivo en el que el servidor registrará todos los errores que genere. Se debe asegurar que hay espacio adecuado en el disco o partición que almacenará todas las bitácoras, además de que el rotado de bitácoras (log rotation) esté configurado. Por ningún motivo se deben almacenar las bitácoras de Apache en la partición raíz del sistema operativo, pues podría generarse una denegación de servicio del servidor web mediante la saturación del espacio disponible en la partición raíz, provocando que el sistema completo falle.

  - ***LogFormat*** - Esta directiva permite definir exactamente el tipo de información que será registrada en la bitácora. La estructura básica de la directiva es: *LogFormat* ***especificación_formato nombre_formato***, donde la especificación del formato se compone de una serie de reemplazo de variables. El nombre que se le asigna al formato se usa en otras directivas como la de "CustomLog".

- ***Eliminar contenido por defecto***

  Dependiendo de la forma en que se haya instalado el servidor web, en ocasiones se pueden encontrar aplicaciones de prueba o características que pueden ser explotadas de forma remota y que pueden proveer diferente niveles de acceso al servidor. Puesto que la principal función de dichas aplicaciones es demostrar las

capacidades del servidor web, suele no dársele mucha importancia a la seguridad.

De manera general, para el caso de los servidores Apache y **en caso de que se encuentren presentes**, se debe procurar la eliminación de los siguientes archivos:

- ***Archivos HTML por defecto***

  Por defecto, Apache instala una serie de archivos dentro del directorio "document root" que tienen la finalidad de brindar ayuda al administrador Web una vez que se ha instalado Apache de forma satisfactoria. Dentro de estos archivos se encuentra el que contiene el mensaje que se despliega en caso de que no se haya creado un archivo index, la biblioteca de documentación de Apache. A pesar de que estos archivos son útiles, no forman se apegan al objetivo de seguridad de esconder el tipo de software que está ejecutando el servidor web.

- ***Scripts CGI de ejemplo***

  De forma continua, los atacantes intentarán explotar programas CGI ubicados dentro del servidor web; inclusive usarán estos programas para propósitos de reconocimiento del servidor o para explotar el servidor o sistema operativo de forma directa. Los programas CGI tienen una larga historia llena de problemas bugs asociados a la aceptación inadecuada de información proporcionada por los usuarios, por lo que se debe garantizar que no existen programas CGI que pudieran ser usados con fines maliciosos. Además, se debe revisar de forma periódica el código de programas CGI que se creen dentro del servidor.

- ***Archivos del manual de Apache***

  Es necesario quitar el directorio del manual de Apache y los archivos que estén presentes en el "ServerRoot". Este directorio virtual no será accesible a los clientes si se quitaron las directivas por defecto en el archivo de configuración durante la minimización de la superficie de ataque.

- ***Código fuente de Apache***

  Para las versiones compiladas de Apache, es necesario eliminar el código fuente del servidor de producción. Esto evitará que cualquier usuario compile o recompile una nueva versión de Apache.

NOTA: Las distribuciones basadas en Debian, como Ubuntu, mantienen una organización particular de los archivos de configuración para Apache 2, cuyas rutas rutas por defecto son, de manera general:

Elemento	Ubicación
ServerRoot	/etc/apache2
DocumentRoot	/var/www
Archivos de Configuración	/etc/apache2/apache2.conf /etc/apache2/ports.conf
Configuración del Host Virtual por defecto	/etc/apache2/sites-available/default /etc/apache2/sites-enabled/000-default
Ubicación de los módulos	/etc/apache2/mods-available /etc/apache2/mods-enabled
Bitácora de error	/var/log/apache2/error.log
Bitácora de acceso	/var/log/apache2/access.log
Cgi-bin	/usr/lib/cgi-bin
Binarios (apachectl)	/usr/sbin
Start / Stop	/etc/init.d/apache2 (start|stop|restart|reload|force-reload|start-htcacheclean|stop-htcacheclean)

Aún así, las siguientes explicaciones son pertinentes:

- Normalmente, el archivo principal de configuración de Apache se llama httpd.conf; sin embargo, en las distribuciones basadas en Debian este archivo sólo existe por cuestione de compatibilidad y suele encontrarse vacío (el archivo de configuración principal es *apache2.conf*). Cabe destacar que aún es posible agregar directivas de configuración en httpd.conf puesto que *apache2.conf* incluye su contenido cuando arranca el servidor.

- El archivo *ports.conf* contiene las directivas “Listen” que indican al servidor Apache las direcciones IP y los puertos a los que debe escuchar.

- El mejor lugar para agregar configuraciones particulares es el directorio ***conf.d*** Los archivos en este directorio son incluidos como parte de la configuración global y por tanto se aplican a todos los hosts virtuales. Lo anterior sugiere que si existen características que serán compartidas por todos los sitios, incluso se puede colocar una copia de dichas configuraciones en este directorio (pues se convierte en una localidad central que puede ser compartida entre todos los sitios) y crear un archivo con el Alias adecuado para mapear el recurso a la mismo espacio de URL para todos los sitios.

- Las distribuciones de Linux basadas en Debian aprovechan el hecho de que los archivos de configuración pueden incluir a su vez, otros archivos de configuración y por ello crean dos directorios no estándar ***/etc/apache2/mods-enabled*** y ***/etc/apache2/mods-available***. Cada que se instala un módulo desde un paquete Debian, el módulo agrega uno odos

archivos en el directorio *mods-available*, donde el archivo obligatorio *nombreModulo.load* contiene la directiva "Load" de apache para cargar el módulo en el servidor web. El archivo opcional *nombreModulo.conf* contiene directivas de configuración adicionales para la operación del módulo.

- ***Usar el Firewall de Aplicación modSecurity***

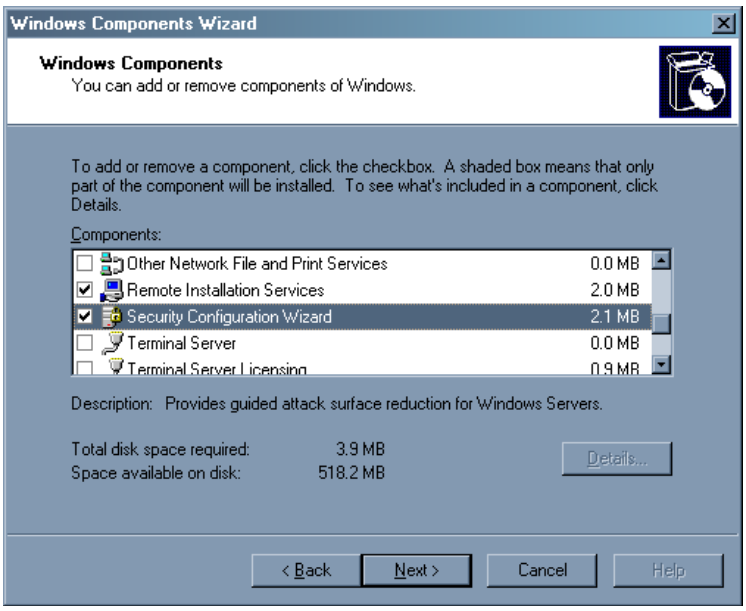
  Referirse al apartado de instalación y configuración de modSecurity.

Internet Information Services (IIS)

- ***Utilizar Security Configuration Wizard***

  Para limitar la superficie de ataque sobre el servidor de IIS (Internet Information Services) es posible utilizar la herramienta *Security Configuration Wizard*. Esta herramienta es un asistente que ayuda a asegurar un servidor modificando ciertas partes del registro, deshabilitando servicios innecesarios, quitando aplicaciones MS que no se usen y por supuesto habilitaría el firewall de Microsoft y cerraría los puertos que no son necesarios.

  La herramienta permite elegir el tipo de servicio que ofrecerá el servidor y en base a esto limitar la ejecución de servicios. Esta herramienta no viene instalada de forma predeterminada en Windows Server 2003 pero puede ser instalada y configurada posterior a su instalación.



Para obtener más información sobre esta herramienta consulte el siguiente documento:

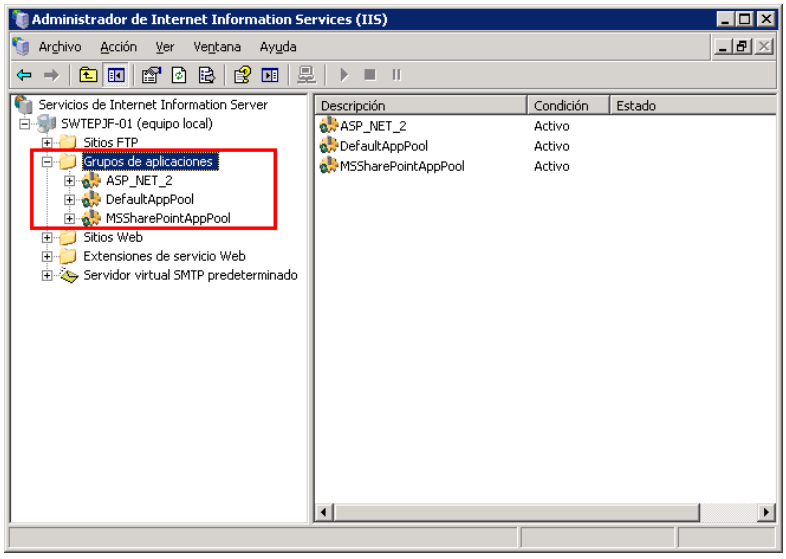
- Security Configuration Wizard for Windows Server 2003
  http://www.microsoft.com/windowsserver2003/technologies/security/configwiz/default.mspx

- The Security Configuration Wizard
  http://redmondmag.com/articles/2005/06/01/the-security-configuration-wizard.aspx

- ***Configurar grupos de aplicaciones en IIS 6.0***

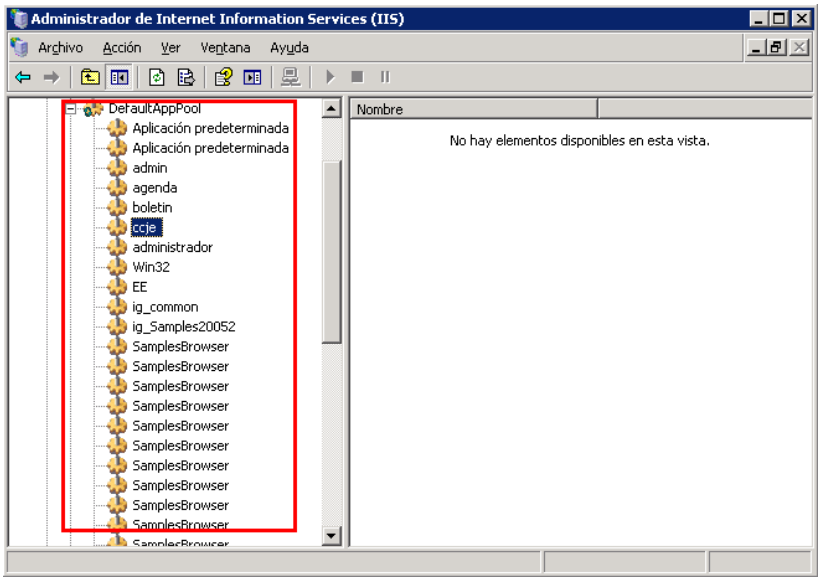
Un grupo de aplicaciones es una configuración que vincula una o varias aplicaciones con un conjunto de uno o varios procesos de trabajo. Como las aplicaciones de un grupo de aplicaciones están separadas unas de otras mediante límites de procesos de trabajo, una aplicación de un grupo no se ve afectada por los problemas que puedan causar las aplicaciones de otros grupos.

Mediante la creación de nuevos grupos de aplicaciones y la asignación a ellos de sitios y aplicaciones Web, puede hacer que el servidor sea más eficaz y confiable, además de conseguir que las demás aplicaciones estén siempre disponibles, incluso cuando se producen problemas en el proceso de trabajo que da servicio al nuevo grupo de aplicaciones.



Actualmente, en el servidor Web ya existen grupos de aplicaciones creados, pero la mayoría de los sistemas pertenecen al grupo de aplicaciones predeterminado ***DefaultAppPool***, si el proceso de trabajo asignado a este pool de aplicaciones fallara, afectaría a todas las aplicaciones en ***DefaultAppPool***.

Por lo tanto, como se menciono anteriormente, asignando cada sistema a un grupo de aplicaciones específico, en el caso de que falle el proceso de trabajo asignado a ésta se garantiza la continuidad del funcionamiento de las demás aplicaciones en el servidor.



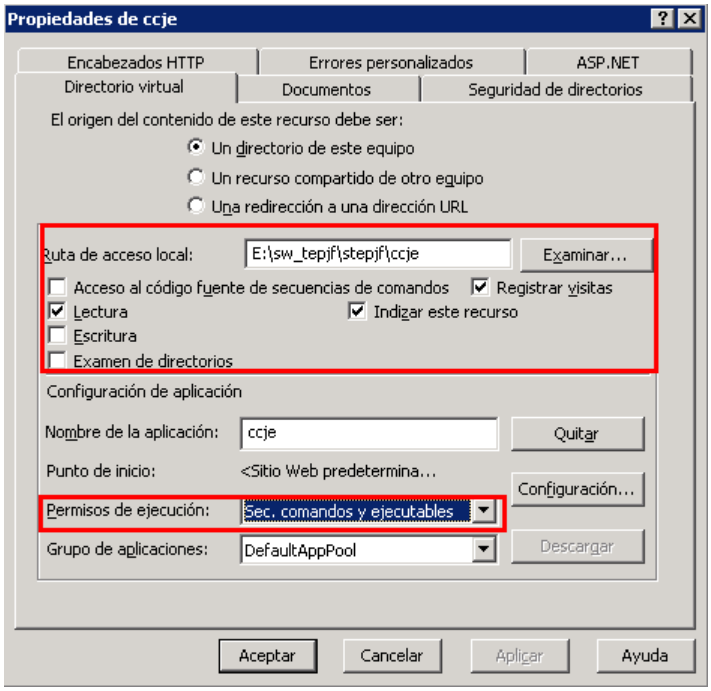
- *Creating Application Pools in IIS 6.0 (IIS 6.0)*
  http://www.microsoft.com/technet/prodtechnol/WindowsServer2003/Library/IIS/c7f5c204-e7a0-452f-8652-56caf4238a4b.mspx?mfr=true

- ***Permitir solo acceso de lectura al sitio Web***

Se debe establecer en los sitios Web utilizados por los sistemas del Tribunal Electoral una configuración que solo permita privilegios de ***Lectura*** a la ruta de acceso local del sitio para limitar la exposición a algún tipo de escritura y/o enumeración de documentos del sitio.

Salvo casos justificados, es recomendable limitar los permisos de ejecución a ***Sólo secuencias de comando***, lo que evita la ejecución de archivos ejecutables dentro del espacio designado para el sistema web.

Como puede observarse a continuación, este **NO** es el caso para CCJE, que aunque sí tiene permisos de sólo lectura a la ruta de acceso local del sitio, su permiso de ejecución es el de ***Secuencias de comando y ejecutables.***



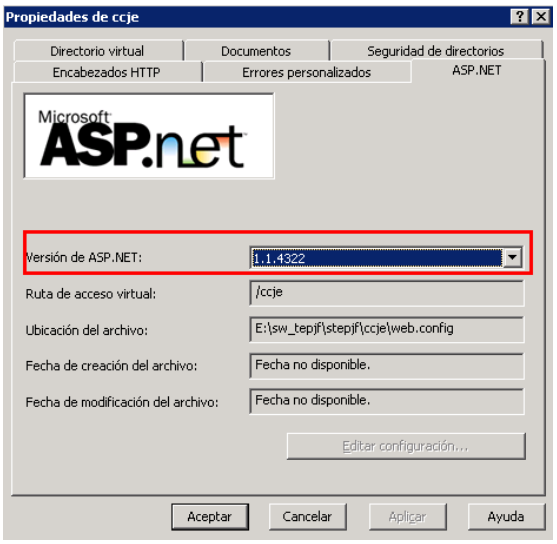
Por ello que se sugiere la justificación correspondiente o la revaloración del permiso de ejecución, de forma que en caso de no ser necesario, se regrese al valor sugerido de ***Sólo secuencias de comando***.

Desarrollo

Tecnología .NET

- ***Actualizar a una versión más reciente de .Net Framework***

  Algunos de los sistemas se encuentran desarrollados en bajo tecnología de Microsoft .NET Framework 1.1.4322.2443 o 2.0.50727 por lo que debería pensarse en actualizar a una versión de .Net Framework más actual.



Las ventajas que pueden observarse al hacer esta migración son las siguientes:

- Las versiones .NET Framework 1.1 y 2.0 ya son antiguas, lo que significa que sus problemas de funcionamiento y seguridad son bien conocidos, a diferencia de las versiones más recientes de este producto.

- El utilizar versiones más actuales de .NET Framework mejora la compatibilidad de ejecución de las aplicaciones en sistemas operativos más recientes como Windows Server 2008. Aunque las versiones antiguas de .NET Framework como la 1.1 aún pueden ejecutarse en estos sistemas actuales, no resulta una buena práctica realizar estas acciones debido a que un Framework superior tiene más potencial y mejores características de seguridad. Sin embargo, se debe evaluar el costo que representa dicha actualización para la organización, ya que se debe invertir tiempo y esfuerzo para lograr que la aplicación pueda ejecutarse en un Framework más actual, aunque la mayoría de las veces esto representa cambios simples.

- Mejora el rendimiento de las aplicaciones, el manejo de memoria y presenta, además, mejoras en el acceso y modelado de los datos.

- ***Cifrar las cadenas de conexión***

A lo largo del análisis de código de diversas páginas en diferentes secciones del portal del Tribunal Electoral, se encontró que de manera recurrente las cadenas de conexión a las bases de datos se almacenan en texto claro. Lo anterior deja vulnerable información sensible como:

- Dirección IP o nombre del servidor de la base de datos.
- ID de la conexión a la base de datos.

- Nombre de usuario.
- Contraseña de acceso.

El cifrado de este tipo de información se vuelve de vital importancia para evitar que las personas que obtengan acceso a estos archivos (o incluso al archivo de configuración de una aplicación web de .NET – web.config --) obtengan los parámetros para establecer la conexión.

- ***Prevenir ataques Cross-Site Scripting en ASP.Net***

La vulnerabilidad de Cross-site scripting (XSS) fue de las que más se presentaron en los sistemas del Tribunal Electoral, la cual, explora debilidades en la validación de la página Web inyectando código script del lado del cliente.

De acuerdo al análisis, esta vulnerabilidad se presenta en los sistemas debido a la inadecuada validación de los parámetros de entrada, falta de codificación en las salidas y en la confianza que se tiene sobre los datos obtenidos desde una base de datos.

Una medida importante para prevenir este tipo de ataque es asumir que todas las entradas a la aplicación Web son maliciosas, limitar y validar todas las entradas en la aplicación y codificar todas las salidas que podrían, potencialmente incluir caracteres HTML.

Para mayor información se puede consultar el documento:

- *How To: Prevent Cross-Site Scripting in ASP.NET*
  http://msdn.microsoft.com/en-us/library/ff649310.aspx

- ***Prevenir ataques de SQL Injection en ASP.Net***

Una de las principales vulnerabilidades encontradas en las aplicaciones del Tribunal Electoral fue la de SQL Injection. Es importante señalar que este tipo de ataques permiten a un usuario malicioso ejecutar comandos SQL en la base de datos utilizada por la aplicación Web con los privilegios otorgados por la cuenta de usuario empleada por dicha aplicación.

Los errores más comunes detectados durante el proceso de auditoría y las pruebas de penetración fueron:

- Una débil validación de los parámetros de entrada.
- Construcción dinámica de sentencias SQL sin el empleo de parámetros seguros.

- Uso de logins de base de datos con privilegios excesivos.

Para proteger las aplicaciones de ataques SQL Injection pueden realizarse los siguientes pasos:

- Validar los parámetros de entrada de acuerdo con el tipo, longitud, formato y rango que requiere el dato.
- Utilizar parámetros a través de Stored Procedures para disminuir el riesgo.
- En caso de no poder utilizar Stored Procedures, se debería utilizar parámetros cuando se construyen sentencias SQL dinámicas.

Para mayor información se puede consultar el documento:

- How To: Protect From SQL Injection in ASP.NET
  http://msdn.microsoft.com/en-us/library/ff648339.aspx

o ***Establecer el manejo adecuado de errores***

En la mayoría de los sistemas del Tribunal Electoral se encontró que debido al manejo inapropiado de errores en posible revelar información sensitiva del sistema como tecnología empleada por el servidor Web, ruta de ubicación de archivos, etc.

Es indispensable que los sistemas cuenten con el manejo de errores ya que no solo ayuda a mejorar la seguridad si no también agiliza la identificación y solución de problemas en las aplicaciones.

Para mayor información se puede consultar el documento:

- *Error Handling in ASP.NET Pages and Applications*
  http://msdn.microsoft.com/en-us/library/w16865z6.aspx

Implementación de servicios en servidores UNIX/Linux

o ***Uso de jaulas o "chroot"***

Una jaula chroot presenta a una aplicación una vista muy restringida del sistema de archivos, normalmente con muchos menos privilegios dentro de éste para prevenir el impacto que pudiera generarse dentro del sistema en caso de que la aplicación haya sido vulnerada o comprometida por un atacante.

Lo anterior se logra a través de la creación de una nueva estructura de directorios restringida que se encuentra como un subdirectorio bajo el directorio raíz (root) real del sistema. Mediante la llamada al sistema "chroot", se cambia el directorio raíz para el proceso actual y todos sus subprocesos; es decir, Esta nueva estructura, es vista como "/" por el proceso del servicio que se ejecuta, limitando así el acceso del servicio o proceso al sistema de archivos.

Es posible configurar jaulas para varios servicios, entre los que se destacan el servidor web Apache y el manejador de bases de datos MySQL. Para su correcta configuración, es necesario tomar en cuenta los siguientes puntos:

- Dependiendo del servicio que se vaya a colocar dentro de la jaula, existirán diferentes bibliotecas que deberán ser copiadas dentro del sistema de archivos restringido.
- Los scripts CGI que no estén escritos de forma adecuada y permitan el acceso al servidor no funcionarán.
- Si se desea usar características de Perl u otros lenguajes de programación que sean necesarios para el funcionamiento del servicio, se deberán copiar los binarios y/o bibliotecas correspondientes a una estructura adecuada dentro de la jaula (por ejemplo, ***/bin/mail***, ***/bin/ls***, etc).
- Lo mismo aplica para SSL, PHP, LDAP, PostgreSQL y otros programas o servicios.
- Chroot suele requerir un buen número de pruebas para garantizar un funcionamiento adecuado para la seguridad y funcionalidad del servicio.

Para profundizar en el tema de aseguramiento del manejador de bases de datos MySQL y el servidor web Apache con el uso de jaulas chroot se pueden consultar los siguientes recursos de Symantec:

- ✓ Securing MySQL: Step-by-step
  http://www.symantec.com/connect/articles/securing-mysql-step-step

- ✓ Securing Apache: Step-by-step
  http://www.symantec.com/connect/articles/securing-apache-step-step

Adicionalmente, el Open Web Application Security Project ofrece un documento referente al hardening de MySQL, donde se describe el procedimiento para crear una jaula chroot para el manejador de la base de datos:

- ✓ http://www.owasp.org/index.php/OWASP_Backend_Security_Project_MySQL_Hardening#Designing_a_chroot-jail

Anexo B. Instalación sugerida de ModSecurity sobre servidor web Apache 2.x

ModSecurity es un módulo de FireWall de aplicación web gratuito y de código abierto que se integra al servidor web Apache. Existen dos versiones del módulo, una para cada rama de Apache (Apache 1.3 y Apache 2) y son casi idénticas en su funcionamiento. En la rama de Apache 2, mod_security usa un API de filtrado avanzado que permite la intercepción del cuerpo de la respuesta a las peticiones y es más eficiente en términos de consumo de memoria.

Entre las ventajas que aporta mod_security, se encuentran:

- ✓ Intercepción de solicitudes HTTP antes de que sean completamente procesadas por el servidor web.
- ✓ Intercepción del cuerpo de la solicitud (por ejemplo, el payload del método POST).
- ✓ Intercepción, almacenamiento y opcionalmente la validación de los archivos subidos al servidor.
- ✓ Realización opcional de acciones anti-evasivas.
- ✓ Análisis de la solicitud a través del procesamiento de un conjunto de reglas definidas en la configuración.
- ✓ Intercepción de las respuestas HTTP antes de ser enviadas al cliente (sólo Apache 2).
- ✓ Análisis de la respuesta mediante el procesamiento de un conjunto de reglas definidas en la configuración.
- ✓ Realización de una acción o ejecución de un script externo cuando una solicitud o respuesta no pasan el análisis (un proceso llamado detección).
- ✓ Dependiendo de la configuración, bloqueo de la ejecución de una solicitud o envío de una respuesta (un proceso llamado prevención).
- ✓ Implementación de bitácoras de auditoría del servidor web.

Instalación del módulo

A continuación se describe la instalación del módulo para un sistema operativo Linux basado en Debian. Esta configuración supone que la instalación del servidor web Apache

2.2.x fue realizada mediante los binarios precompilados para la distribución, y que se descargó ModSecurity desde la página www.modsecurity.org y se ha comprobado la validez de los archivos descargados como se indica en el tutorial de instalación de la página.

Adicionalmente, ***realiza la instalación de forma independiente*** con una estructura de directorios **bajo la ruta /opt** del sistema, que puede ser creada como se muestra:

#carpeta para binarios
mkdir -p /opt/modsecurity/bin

#carpeta de archivos de configuración
mkdir -p /opt/modsecurity/etc

#carpeta de bitácoras de auditoría
mkdir -p /opt/modsecurity/var/audit

#carpeta de tados persistentes
mkdir -p /opt/modsecurity/var/data

#bitácoras
mkdir -p /opt/modsecurity/var/log

#archivos temporales
mkdir -p /opt/modsecurity/var/tmp

#archivos subidos
mkdir -p /opt/modsecurity/var/upload

Se supone, además, que ***el usuario y el grupo del servidor web Apache es www-data***, y que las operaciones se están llevando a cabo con el ***usuario root***.

1. Activar el módulo unique_id de apache
   a2enmod unique_id

2. Reiniciar el servidor
   apache2ctl restart

3. Verificar que se ha cargado el módulo
   apache2ctl -t -D DUMP_MODULES

4. Instalar las bibliotecas adicionales requeridas para usar ModSecurity. Si Apache se instaló con los paquetes de Debian, harán falta libcurl, libxml2 y lua; si apache se instaló desde los archivos fuente, también hay que instalar pcre
   apt-get install libcurl3-dev liblua5.1-dev libxml2-dev

5. Asignar usuarios, grupos y permisos a los directorios

chgrp www-data /opt/modsecurity/
chgrp www-data /opt/modsecurity/var/
chgrp www-data /opt/modsecurity/var/tmp/
chown www-data /opt/modsecurity/var/audit/
chown www-data /opt/modsecurity/var/data/
chown www-data /opt/modsecurity/var/tmp/
chown www-data /opt/modsecurity/var/upload/

chmod 750 /opt/modsecurity/
chmod 750 /opt/modsecurity/bin/
chmod 700 /opt/modsecurity/etc/
chmod 750 /opt/modsecurity/var/
chmod 700 /opt/modsecurity/var/audit/
chmod 700 /opt/modsecurity/var/data/
chmod 700 /opt/modsecurity/var/log/
chmod 750 /opt/modsecurity/var/tmp/
chmod 700 /opt/modsecurity/var/upload/

6. Descomprimir modsecurity. Compilar modsecurity para generar un archivo llamado mod_security2.so que es un objeto dinámico compartido de Apache, es decir, un plugin que agrega funcionalidad al servidor web. El archivo contiene toda la funcionalidad de ModSecurity, y se puede activar como cualquier otro módulo de Apache. Adicionalmente se ejecuta el instalador sobre el candidato de instalación (make install)

./configure
make
make install

7. Copiar el archivo objeto (que se crea en la carpeta oculta "apache2/**.**libs") a la carpeta destinada para los binarios

cp apache2/.libs/mod_security2.so /opt/modsecurity/bin/
cp mlogc/mlogc /opt/modsecurity/bin/

8. Agregar ModSecurity a Apache modificando el archivo de configuración /etc/apache2/apache2.conf y agregando las líneas:

#Cargar libxml2
LoadFile /usr/lib/libxml2.so
#Cargar Lua
LoadFile /usr/lib/liblua5.1.so
#Cargar ModSecurity
LoadModule security2_module /opt/modsecurity/bin/mod_security2.so

#Indicar a Apache dónde encontrar el archivo de configuración
Include /opt/modsecurity/etc/modsecurity.conf

9. Crear el archivo "modsecurity.conf" en la ruta **/opt/modsecurity/etc** Este archivo contendrá a su vez, las directivas para incluir los archivos con las reglas de las que hará uso el módulo. Aunque pudieran colocarse aquí todas las reglas, se sugiere mantener archivos separados para una mejor administración. El contenido del archivo deberá ser algo parecido a :

<IfModule mod_security2.c>
Include ***ruta/al/archivo/de/las/reglas***
</IfModule>

Reglas

El proyecto OWASP ofrece una serie de directivas para ModSecurity llamadas "core RuleSet" que forman un conjunto de más de 120 reglas. Éstas se pueden descargar desde **http://sourceforge.net/projects/mod-security/files/modsecurity-crs/0-CURRENT/** y tienen el objetivo de proporcionar protección contra muchos ataques comunes con tan sólo incluirlos en la configuración.

Para instalar el "core RuleSet" hay que descomprimir el archivo (que a la fecha de elaboración del manual es "modsecurity-crs_2.2.0.tar.gz") y copiar los archivos de la carpeta ***base_rules*** con extensión .conf y *.data a la carpeta /opt/modsecurity/etc/, copiar el archivo de "ejemplo" para la configuración del core RuleSet llamado *modsecurity_crs_10.config.conf.example*, indicar que se deben importar esos archivos en el archivo principal de configuración de modsecurity /opt/modsecurity/etc/modsecurity.conf

1. *tar xzf modsecurity-crs_2.2.0.tar.gz*
2. *cd modsecurity-crs_2.2.0/*
3. *cp ./base_rules/modsecurity_crs_*.conf /opt/modsecurity/etc/*
4. *cp ./modsecurity_crs_10.config.conf.example /opt/modsecurity/etc/modsecurity_crs_10.config.conf*

Incluir la siguiente línea en el archivo de configuración */opt/modsecurity/etc/modsecurity.conf*:

Include /opt/modsecurity/etc/modsecurity_crs_*.conf

Para que las reglas tomen efecto, será necesario reiniciar el servidor con la directiva *apache2ctl restart*

Con esto queda finalizada la instalación de modsecurity. Sin embargo, se debe hacer notar que:

1) La aplicación de estas reglas puede resultar muy estricta y romper el funcionamiento de las aplicaciones que se tienen, por lo que debe realizarse un

período de “prueba” del módulo (ejecutar modsecurity en modo de detección únicamente)

2) Para poner la herramienta en modo de detección, se deberá asegurar que en el archivo *modsecurity_crs_10.config.conf* la línea ***SecRuleEngine DetectionOnly*** NO esté comentada y que la política por defecto permita la ejecución. Para ello se deberá cambiar la línea ***SecDefaultAction “phase:2,deny,log”*** por la línea ***SecDefaultAction "phase:1,log,auditlog,pass"***

Se deberán intentar los ataques y comprobar que éstos han sido registrados en la bitácora de errores de Apache, la bitácora de depuración de ModSecurity (si se habilitó), y la bitácora de auditoría de ModSecurity (si se habilitó).

Anexo C. Checklist mínimo para el uso de SQL Server, IIS, MySQL y Apache 2.x

A continuación se proporciona un checklist muy básico de los puntos mínimos que debieran considerarse para el uso de SQL Server, IIS, MySQL y Apache 2.x

		Valor Recomendado	*Valor Actual*
1	**SQL Server**		
1.1	*Ejecución de Servicios de SQL mediante cuentas limitadas*	Usar cuenta de usuario local o de dominio sin privilegios de admón.	
1.2	*Modo de autenticación*	Autenticación de Windows	
1.3	*Stored Procedures de Sistema*	Deshabilitados	
1.4	*Utilización de usuarios con privilegios limitados*	Renombrar la cuenta sa, limitar los usuarios con rol de sysadmin	
1.5	*Actualizaciones más recientes del sistema*	Versión 9.00.4311.00	
1.6	*Políticas de contraseñas y bloqueo de cuentas*	Políticas de contraseñas y bloqueo de cuentas configuradas	
2	**Internet Information Services**		
2.1	*Configuración grupos de aplicaciones*	Definir grupos de aplicaciones individuales para cada sitio web definido en el servidor.	
2.2	*Permitir sólo acceso de lectura al sitio web*	Permisos de lectura y sólo secuencias de comando.	

		Valor Recomendado	*Valor Actual*
3	**MySQL**		
3.1	*Establecer contraseña para la cuenta del usuario root*	La cuenta de administración (root, si no ha sido renombrada) debe tener una contraseña	
3.2	*Eliminar la base de datos por defecto*	No debe existir la base “test” que se instala por defecto.	
3.3	*Eliminar cuentas anónimas y cuentas con contraseña en blanco*	Las cuentas sin nombre de usuario (‘’) o con contraseñas en blanco deben ser eliminadas.	
3.4	*Restringir o deshabilitar el acceso*	Si la aplicación y la	

	remoto	base de datos están en el mismo servidor, la sentencia "bind-address=127.0.0.1" o "skip-networking" debe aparecer en el archivo de configuración.	
3.5	*Ejecutar MySQL como usuario no privilegiado*	Verificar que el servicio de MySQL no es ejecutado por el usuario root	
3.6	*Revisar los permisos de los usuarios.*	Variable, según los usuarios y el esquema de admón. de la base de datos. Se debe aplicar el principio de mínimos privilegios.	
3.7	*Cifrar la información sensible*	Datos como las contraseñas deben almacenarse de forma cifrada dentro de la base de datos	
3.8	*Bitácoras*	MySQL debe estar generando bitácoras (normalmente en el directorio /var/log/mysql/)	
4	**Apache 2.2.x**		
4.1	*Uso de binarios precompilados*	Instalar el servidor web a partir de los binarios disponibles para el sistema operativo de la máquina.	
4.2	*Deshabilitar módulos que no se requieren*	Variable, según objetivos y necesidades del servidor.	
4.3	*Cuentas de usuario y grupo personalizadas para apache*	El servicio no debe correr como usuario "nobody"	
4.4	*Restringir Acceso al directorio raíz*	Se debe contar con la directiva *<Directory />* *Options None* *AllowOverride None* *deny from all* *</Directory>* *En el archivo de configuración.*	

4.5	*Funcionalidad de directorios con la directiva "Options"*	Variable, según las necesidades del sitio.	
4.6	*Protección contra el filtrado/exposición de información*	Los valores de las directivas ServerTokens y ServerSignature deben ser "Prod" y "Off", respectivamente.	
4.7	*Páginas de error personalizadas*	Deben usarse páginas de error personalizadas para los errores de HTTP.	
4.8	*Bitácoras*	El registro de bitácoras debe estar habilitado, y el servicio de rotado de bitácoras activado.	
4.9	*Borrado de contenido por defecto*	En caso de existir archivos de la instalación por defecto que no se utilicen, éstos deben ser eliminados del servidor (scripts CGI de prueba, documentación, páginas HTML por defecto, etc.)	

Referencias de los anexos

- ✓ Center for Internet Security Benchmark for Apache Web Server 2.1 (january 2008)
  http://cisecurity.org

- ✓ Center for Internet Security Benchmark for MySQL Versions 4.1, 5.0 and 5.1 Community Editions (August 2007)
  http://cisecurity.org

- ✓ Center for Internet Security Configuration Benchmark for Microsoft SQL Server 2005 Version 1.2.0 (January 2010)
  http://benchmarks.cisecurity.org/tools2/sqlserver/CIS_SQL2005_Benchmark_v1.2.0.pdf

- ✓ OWASP: Reviewing MySQL Security
  http://www.owasp.org/index.php/Reviewing_MySQL_Security

- ✓ OWASP Backend Security Project: MySQL Hardening
  http://www.owasp.org/index.php/OWASP_Backend_Security_Project_MySQL_Hardening

- ✓ 10 steps to fortify the security of your MySQL installation
  http://www.net-security.org/secworld.php?id=4135

- ✓ Best Practices for UNIX chroot() Operations
  http://unixwiz.net/techtips/chroot-practices.html