UNIVERSIDAD MAYOR DE SAN ANDRÉS
FACULTAD DE CIENCIAS PURAS Y NATURALES
CARRERA DE INFORMATICA

TESIS DE GRADO

"MODELO DE SEGURIDAD EN LAS APLICACIONES WEB"

PARA OPTAR AL TÍTULO DE LICENCIATURA EN INFORMÁTICA
MENCIÓN: INGENIERIA DE SISTEMAS INFORMÁTICOS

POSTULANTE: ADALID MAMANI GUARACHI
TUTORA METODOLOGICA: Lic. MENFY MORALES RÍOS
ASESOR: Lic. FRANZ RAMIRO GALLARDO PORTANDA

LA PAZ – BOLIVIA
2015

UNIVERSIDAD MAYOR DE SAN ANDRÉS
FACULTAD DE CIENCIAS PURAS Y NATURALES
CARRERA DE INFORMÁTICA

LA CARRERA DE INFORMÁTICA DE LA FACULTAD DE CIENCIAS PURAS Y NATURALES PERTENECIENTE A LA UNIVERSIDAD MAYOR DE SAN ANDRÉS AUTORIZA EL USO DE LA INFORMACIÓN CONTENIDA EN ESTE DOCUMENTO SI LOS PROPÓSITOS SON ESTRICTAMENTE ACADÉMICOS.

LICENCIA DE USO

El usuario está autorizado a:

a) visualizar el documento mediante el uso de un ordenador o dispositivo móvil.
b) copiar, almacenar o imprimir si ha de ser de uso exclusivamente personal y privado.
c) copiar textualmente parte(s) de su contenido mencionando la fuente y/o haciendo la referencia correspondiente respetando normas de redacción e investigación.

El usuario no puede publicar, distribuir o realizar emisión o exhibición alguna de este material, sin la autorización correspondiente.

DEDICATORIA

Esta tesis es dedicada principalmente a mi familia por todo el apoyo incondicional que me brindaron en todo el trayecto de la carrera. Son ellos que me motivaron para seguir adelante en todo momento.

A mi parceira quien me apoyo y alentó a seguir hasta el final.

A todos los amigos y amigas que me apoyaron moral y emocionalmente.

AGRADECIMENTOS

En primera instancia gracias a Dios por guiarme y darme la fortaleza día a día, a pesar de las adversidades siempre me das positivismo.

A mi papá Félix Mamani Nina por guiarme constantemente, por la comprensión y por su apoyo incondicional para concluir el presente trabajo.

A mi mamá Martha Guarachi Mamani (†) porque me cuidas y me brindas protección, me guías en todo momento, gracias por toda la comprensión.

A mis hermanos Marcelino Mamani Guarachi, Wildo Mamani Guarachi y Benita Mamani Guarachi, que me enseñaron a vivir la vida como persona de bien y me brindaron todo el apoyo de manera constante alentándome a terminar mis estudios universitarios.

A mi docente Tutora Metodológica, Esp. Menfy Morales Ríos, por su guía en todo el transcurso de la elaboración del presente trabajo.

A mi docente Asesor, Lic. Franz Ramiro Gallardo Portanda, por asesoramiento y la colaboración durante la elaboración del presente trabajo.

A la Universidad Mayor de San Andrés, Facultada de Ciencias Puras y Naturales, Carrera de Informática por haberme recibido y acogido durante mis estudios académicos. También un gran agradecimiento al personal docente administrativo, por todo el conocimiento.

A todos mis amigos y amigas por su apoyo y amistad que sin ellos todo el recorrido de la carrera hubiera sido solitario, gracias por compartir sus ideas y conocimientos.

RESUMEN

La presente investigación es referente a las aplicaciones web que en la actualidad son de uso general en todos los ámbitos donde se utilizan la tecnología web. Tiene como énfasis de realizar un análisis de la seguridad en las aplicaciones web, en base a ello se plantea el Modelo de Seguridad en las Aplicaciones Web.

El modelo está dirigida a las empresas o personas dedicadas al desarrollo de las aplicaciones web, independientemente de la tecnología utilizada para su implementación.

El modelo tiene como finalidad de aportar y ayudar con la comprensión de las aplicaciones web centrándose en la seguridad, es así que el modelo de seguridad en las aplicaciones web está compuesto principalmente por cuatro capas; Capa de Presentación Visual, Capa lógica de negocio, Capa servicio de datos web y Capa de alojamiento web.

En base al modelo de seguridad establecido se realiza arquitecturas expresados en niveles y capas, viendo los aspectos de vulnerabilidad presentes en las aplicaciones.

Cada una de los niveles está compuesto por capas permitiendo aligerar la comprensión de la estructura de una aplicación web, y además en cada una de las capas están los componentes.

Capa de presentación visual, es donde se despliega la interfaz de la aplicación web, y es la capa encargada de la recopilación de datos ingresados por el usuario. Esta capa se encarga de pasar o enviar los datos filtrados a la capa lógica de negocio, donde se realiza el procesado.

Capa lógica de negocio, es donde se ejecuta los procesos en su mayoría referentes a los datos recibidos de la capa de presentación visual. Las peticiones del usuario son recibidos y se envían las respectivas respuestas tras el proceso lógico.

Capa de servicio de datos web, es donde los datos son almacenados y es la encargada de acceder a esos datos, está formada por uno o más gestores de Bases de Datos. Además reciben las peticiones de almacenamiento o recuperación de datos desde la capa lógica de negocio.

Capa de alojamiento web, en esta capa se encuentra alojada la aplicación web en su totalidad, por ello es necesario cumplir con las seguridades requeridas que puedan garantizar un buen funcionamiento de las aplicaciones web.

ÍNDICE

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

CAPITULO I

MARCO REFERENCIAL

1.1. Introducción

En la actualidad se presenta una variedad de delitos informáticos, en su mayoría son ataques a las aplicaciones web, utilizando las vulnerabilidades. La manipulación de la información puede provocar pérdidas económicas significativas brindando beneficios a los que realizan este tipo de actos.

El uso de las aplicaciones web es imprescindible en el ambiente informatizado donde se requiere información de manera rápida y en tiempo real, estas aplicaciones se adaptan en cualquier ámbito como comercio electrónico, bancos, centros educativos virtuales, redes sociales, blogs, foros, entre otros.

Estas aplicaciones webs en su mayoría necesitan de la interacción entre el usuario y el navegador web, es muy probable que la información confidencial este viajando a través de estas aplicaciones, es ahí donde es propenso la integridad de la información, si no se proporciona niveles de seguridad, es por lo que está expuesto a diversos tipos de amenazas inyección de código SQL, lo cual puede evitarse la autenticación y gestión de sesiones.

La gran mayoría de los datos sensibles del mundo están almacenados en sistemas gestores de bases de datos comerciales tales como Oracle, Microsoft SQL Server entre otros, y atacar una bases de datos es uno de los objetivos favoritos para los criminales.

Esto puede explicar por qué los ataques externos, tales como inyección de SQL, subieron 56.2% en 2012, “Esta tendencia es prueba adicional de que los agresores tienen éxito en hospedar páginas Web maliciosas, y de que las vulnerabilidades y explotación en relación a los navegadores Web están conformando un beneficio importante para ellos.

Para empeorar las cosas, según un estudio publicado en febrero de 2012 The Independent Oracle Users Group (IOUG), casi la mitad de todos los usuarios de Oracle tienen al menos dos parches sin aplicar en sus manejadores de bases de datos.

Mientras que la atención generalmente se ha centrado en asegurar los perímetros de las redes por medio de, firewalls, IDS / IPS y antivirus, cada vez más las organizaciones se están enfocando en la seguridad de las bases de datos con datos críticos, protegiéndolos de intrusiones y cambios no autorizados.

Uno de los principales problemas de seguridad al que se enfrentan las aplicaciones web son las vulnerabilidades que puedan presentar para explotar ataques de inyección en sus interfaces de interacción con el usuario.

Un ataque de inyección es un método de infiltración de código intruso que se vale de una vulnerabilidad informática en una aplicación web en el nivel de validación de las entradas de datos.

Por medio de estos ataques se puede realizar actos no autorizados como adquisición de cuentas de usuarios, acceso a información confidencial almacenada en la base de datos, cambiar configuraciones de usuario, modificar la información de una aplicación entre otros. Estos ataques son amenazas comunes en aplicaciones web y su explotación puede ocasionar serios problemas en un sistema informático es fundamental elaborar una solución al problema de este índole.

1.2. Antecedentes

La seguridad en los sistemas informáticos es de gran importancia que viene siendo objeto de estudio desde 1970. El concepto de seguridad hace referencia a las medidas y al control destinados a la protección contra la negación de servicio y la ausencia de autorización (ya sea de forma accidental o intencionada) para descubrir, modificar o destruir datos o sistemas.

Debido a la dificultad y según la mayoría de estudios e investigaciones, a la imposibilidad de garantizar esta característica en los sistemas operativos o redes de computadores, el término "seguridad" se sustituye por el de "fiabilidad". Se define la fiabilidad como la probabilidad de que un sistema se comporte tal y como se espera de él.

Es así que se tiene las siguientes investigaciones realizadas con anterioridad referentes al tema de seguridad:

- SEGURIDAD INFORMÁTICA: SUS IMPLICANCIAS E IMPLEMENTACIÓN (Cristian Fabian. Borghello), 2001, Argentina, UNIVERSIDAD TECNOLÓGICA NACIONAL;
- ANÁLISIS DE SEGURIDAD, OPTIMIZACIÓN Y MEJORA DE UN PORTAL WEB BASADO EN PHP Y MYSQL (Román Medina-Heigl

Hernández), 2002, Sevilla - España, UNIVERSIDAD DE SEVILLA ESCUELA SUPERIOR DE INGENIEROS;

- ANALIZAR EL SISTEMA DE SEGURIDAD EN SERVIDORES WEB PARA SU CORRECTA UTILIZACIÓN (Khatherine Cantos Rivera, Karla Carangui Vintimilla), 2007, Cuenca-Ecuador;
- ANÁLISIS INICIAL DE LA ANATOMÍA DE UN ATAQUE A UN SISTEMA INFORMÁTICO (Daniel Monroy López), 2009, México D.F., UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO;
- SEGURIDAD EN LOS SISTEMAS INFORMÁTICOS (José Ismael Ripoll Ripoll), 2012, Valencia – España, UNIVERSITAT POLITÉCNICA DE VALÉNCIA;
- SOFTWARE DE SEGURIDAD PREVENTIVO PARA SERVIDORES WEB (Alcides Chana Quispe), 2012, La Paz – Bolivia, UNIVERSIDAD MAYOR DE SAN ANDRÉS;
- SISTEMA DE DETECCIÓN DE ATAQUES DE INYECCIÓN DE CÓDIGO DE APLICACIONES WEB EN ANDROID, 2012, La Paz – Bolivia, UNIVERSIDAD MAYOR DE SAN ANDRÉS.

1.3. Planteamiento del problema

Los ataques son amenazas constantes en las aplicaciones web, esto ocasiona que los sistemas de información sean más vulnerables, afectando la integridad y disponibilidad de la información, por tanto es puesto en riesgo la continuidad de la actividad que realizan en la web.

Las causas de vulnerabilidad son debido a la configuración inadecuada en los proveedores de los servicios de alojamiento web.

El mal empleo de políticas de seguridad en las aplicaciones web durante la implementación puede generar vulnerabilidades que pueden ser aprovechados por los usuarios no autorizados, provocando un funcionamiento inadecuado de las aplicaciones web.

La falta de los sistemas de seguridad en las aplicaciones web, puede ocasionar un mal funcionamiento, infiltración de la información no propia, pérdida de información entre otros.

Los mecanismos de seguridad para los servidores y aplicaciones web pueden disminuir las vulnerabilidades ante los ataques, obteniendo resultados de una aplicación web funcional, menos expuesta a los ataques informáticos

1.4. Formulación del problema

¿De qué manera se puede disminuir las vulnerabilidades y prevenir los ataques a las aplicaciones web?

1.5. Objetivos

1.5.1. Objetivo general

Formular y diseñar un modelo de seguridad que permita disminuir las vulnerabilidades y los riesgos que se presentan en el desarrollo de una aplicación web.

1.5.2. Objetivos específicos

Como objetivos específicos se presenta lo siguiente:

- Realizar arquitecturas de aplicaciones web según el modelo de seguridad para una orientación esquemática.
- Evaluar técnicas y herramientas para las validaciones respectivas.
- Evaluar métodos de autenticación para evitar la vulnerabilidad en los formularios de tipo login.
- Evaluar el manejo de sesiones para la administración de las aplicación web.
- Identificar los riesgos y vulnerabilidades que afectan a las aplicaciones web para evitar las mismas.
- Establecer estrategias para las buenas practicas, con el fin de evitar futuras amenazas o inconsistencias en las aplicaciones web.

1.6. Hipótesis

“El modelo de seguridad en las aplicaciones web permitirá la disminución de las vulnerabilidades, pudiendo de esta manera incrementar la seguridad en las aplicaciones web”.

1.7. Justificaciones

1.7.1. Justificación técnica

El presente trabajo de investigación induce a la utilización de las políticas de seguridad ante las amenazas que existen en el ámbito de las aplicaciones web.

1.7.2. Justificación económica

Con la disminución de las vulnerabilidades y ataques a las aplicaciones web se permite el normal funcionamiento de las actividades en los servidores web, evitando perdidas de información y económica.

1.7.3. Justificación social

Es necesario informar a la sociedad en general de la exposición de la información en las aplicaciones web, y cuán importante es tomar medidas de seguridad para no sufrir ataques que puedan infringir principios de seguridad de la información como la integridad o la confidencialidad, llevar de esta manera a una sociedad informatizada y automatizada.

1.7.4. Justificación practica

El presente trabajo de investigación conlleva gran importancia por la información que ofrece sobre las diversas amenazas a las aplicaciones web, además prevenir a los desarrolladores de las fallas en la implementación y el buen uso de las políticas de seguridad.

1.8. Diseño metodológico

Para el desarrollo de la presente tesis se utiliza:

1.8.1. Objeto de estudio

La presente investigación muestra un análisis sobre el nivel de riesgo al que se comprometen las aplicaciones web. Realizando un estudio sobre los factores concernientes a seguridad en el ámbito de la web.

- Políticas se seguridad.
- Servidor web.
- Aplicación web.
- Software de seguridad.

1.8.2. Análisis y planificación

Metodología del marco lógico, que es considerada un instrumento de planificación que además permite estructurar los principales elementos de un proyecto, subrayando los lazos lógicos entre los insumos previstos, las actividades planeadas y los resultados esperados, también permite un diseño que satisface tres requerimientos fundamentales de calidad de un proyecto de desarrollo: coherencia, viabilidad y evaluación.

1.8.3. Universo y/o muestra

El universo de la presente investigación estuvo constituido por una encuesta a profesionales en el área del desarrollo web.

1.8.4. Metodologías para el desarrollo del modelo

El método científico es el que se emplea para el desarrollo del modelo, es un conjunto de pasos científicos bien estructurados que nos ayudan a formular, afirmar o corregir una teoría.

Se inicia con la Fase de Observación, donde el sujeto conocedor entra en contacto con el fenómeno, y sabe que algo lo induce a continuar investigando respecto al tema; el cual lleva a la Fase del Planteamiento del hipótesis, que fundamentada en conocimientos previos y en los datos por recoger, podría ser demostrada; por ultimo tenemos la Fase de Comprobación, el cual depende del grado de generalidad y sistematicidad de la hipótesis.

Durante el desarrollo se complementa con el modelo 4+1 vistas para definir a la arquitectura por medio de 5 distintas vistas, tales como: vista de escenarios, lógica, de procesos, de desarrollo y vista física. El modelo se adapta para utilizar en las aplicaciones web incluyendo una vista de seguridad.

1.9. Alcance

El modelo de seguridad en las aplicaciones web produce una disminución de las vulnerabilidades, menos ataques a las aplicaciones web, permitiendo el normal funcionamiento, viendo los factores de vulnerabilidad, políticas de seguridad y otros aspectos de seguridad.

El modelo es más enfocado a las aplicaciones web dinámicas, no tanto a las aplicaciones estáticas porque no tienen mucha interacción con datos almacenados en la Base de Datos.

CAPITULO II

MARCO TEORICO

2.1. Introducción

El presente capítulo tiene por finalidad exponer los conceptos de forma general en los cuales se basa este trabajo, considerando principalmente los tres aspectos que implica la investigación, los cuales son riesgos en el lado del cliente, en el canal de comunicación y en el servidor. También hacemos mención a las técnicas y herramientas que pueden llevar a los riesgos o a las soluciones del tema abordado.

2.2. Seguridad informática

La Seguridad Informática se refiere a las características y condiciones de sistemas de procesamiento de datos y su almacenamiento, para garantizar su confidencialidad, integridad y disponibilidad. [VVA13]

Considerar aspectos de seguridad significa, conocer el peligro**,** clasificarlo y protegerse de los impactos o daños de la mejor manera posible. Esto significa que solamente cuando estamos conscientes de las potenciales amenazas, agresores y sus intenciones dañinas (directas o indirectas) en contra de nosotros, podemos tomar medidas de protección adecuadas, para que no se pierda o dañe nuestros recursos valiosos.

En este sentido, la Seguridad Informática sirve para la protección de la información, en contra de amenazas o peligros, para evitar daños y para minimizar riesgos, relacionados con ella. [ALV05]

La seguridad informática es una disciplina que se encarga de proteger la integridad y la privacidad de la información almacenada en un sistema informático. Sin embargo, no existe una técnica que permita asegurar la seguridad de un sistema al 100%.

Un sistema informático puede ser protegido desde un punto de vista lógico o físico. Por otra parte, las amenazas pueden proceder desde programas dañinos que se instalan en la computadora del usuario o llegar por vía remota. Entre las herramientas más usuales de seguridad informática, se encuentran los programas antivirus, los cortafuegos o firewalls, la encriptación de la información y el uso de contraseñas.

Un sistema seguro debe ser íntegro con información modificable sólo por las personas autorizadas, confidencial que los datos tienen que ser legibles únicamente para los usuarios autorizados, irrefutable que el usuario no debe poder negar las acciones que realizó y tener buena disponibilidad implica que debe ser estable. [VVA13]

De todas formas, como en la mayoría de los ámbitos de la seguridad, lo esencial sigue siendo la capacitación de los usuarios. Una persona que conoce cómo protegerse de las amenazas sabrá utilizar sus recursos de la mejor manera posible para evitar ataques o accidentes. Es decir la seguridad informática busca garantizar que los recursos de un sistema de información sean utilizados tal como una organización o un usuario lo ha establecido, sin intromisiones. [VVA13]

2.1.1. Seguridad física

La seguridad física consiste en la aplicación de barreras físicas, y procedimientos de control como medidas de prevención y contra medidas ante amenazas a los recursos y la información confidencial, se refiere a los controles y mecanismos de seguridad dentro y alrededor de la obligación física de los sistemas informáticos así como los medios de acceso remoto al y desde el mismo, implementados para proteger el hardware y medios de almacenamiento de datos. Cada sistema es único, por lo tanto la política de seguridad a implementar no será única es por ello que se recomienda pautas de aplicación general y no procedimientos específicos.

La seguridad física está enfocada a cubrir las amenazas ocasionadas tanto por el hombre como por la naturaleza del medio físico donde se encuentra ubicado el centro. Las principales amenazas que se ven en la seguridad física son amenazas ocasionadas por el hombre como robos destrucción de la información , disturbios, sabotajes internos y externos, incendios accidentales, tormentas e inundaciones. [DRAG11]

2.1.2. Seguridad lógica

Consiste en la aplicación de las barreras y procedimientos que resguarden el acceso a los datos y solo se permita acceder a ellos al personal autorizado. [DRAG11]

Por lo tanto se refiere a la seguridad en el uso de software y los sistemas, la protección de los datos, procesos y programas, así como la del acceso ordenado y autorizado de los usuarios a la información. La “seguridad lógica” involucra todas aquellas medidas establecidas por la administración -usuarios y administradores de recursos de tecnología de información- para minimizar los riesgos de seguridad asociados con sus operaciones cotidianas llevadas a cabo utilizando la tecnología de información. Los principales objetivos que persigue la seguridad lógica son:

- Restringir el acceso a los programas y archivos;
- Asegurar que se estén utilizando los datos, archivos y programas correctos en y por el procedimiento correcto.

La seguridad lógica se encarga de los controles de acceso que están diseñados para salvaguardar la integridad de la información almacenada de una computadora, así como de controlar el mal uso de la información. [DRAG11]

La seguridad lógica se encarga de controlar y salvaguardar la información generada por los sistemas, por el software de desarrollo y por los programas en aplicación. [DRAG11]

Identifica individualmente a cada usuario y sus actividades en el sistema, y restringe el acceso a datos, a los programas de uso general, de uso específico, de las redes y terminales.

- La falta de seguridad lógica o su violación puede traer las siguientes consecuencias a la organización;
- Cambio de los datos antes o cuando se le da entrada a la computadora;
- Copias de programas y /o información;
- Código oculto en un programa;
- Entrada de virus.

La seguridad lógica puede evitar una afectación de perdida de registros, y ayuda a conocer el momento en que se produce un cambio o fraude en los sistemas. [DRAG11]

Un método eficaz para proteger sistemas de computación es el software de control de acceso. Los paquetes de control de acceso protegen contra el acceso no autorizado, pues piden al usuario una contraseña antes de permitirle el acceso a información confidencial. Sin embargo, los paquetes de control de acceso basados en componentes pueden ser eludidos por delincuentes sofisticados en computación, por lo que no es conveniente depender de esos paquetes por si solos para tener una seguridad adecuada. [DRAG11]

2.3. Seguridad de la información

La seguridad de la información son todas las medidas preventivas que permitan resguardar y proteger la información, manteniendo la confidencialidad, la disponibilidad e integridad. [JL08]

El concepto de seguridad de la información no debe ser confundido con la seguridad informática, ya que este último solo encarga de la seguridad en el medio de la informática y que la información se puede encontrar en diferentes medios o formas.

2.3.1. ISO 27001

ISO 27001 es una norma internacional emitida por la Organización Internacional de Normalización (ISO) y describe cómo gestionar la seguridad de la información en una empresa. La revisión más reciente de esta norma fue publicada en 2013 y ahora su nombre completo es ISO/IEC 27001:2013. La primera revisión se publicó en 2005 y fue desarrollada en base a la norma británica BS 7799-2. [KOOLE15]

ISO 27001 puede ser implementada en cualquier tipo de organización, con o sin fines de lucro, privada o pública, pequeña o grande. Está redactada por los mejores especialistas del mundo en el tema y proporciona una metodología para implementar la gestión de la seguridad de la información en una organización. También permite que una empresa sea certificada; esto significa que una entidad de certificación independiente confirma que la seguridad de la información ha sido implementada en esa organización en cumplimiento con la norma ISO 27001. [KOOLE15]

ISO 27001 se ha convertido en la principal norma a nivel mundial para la seguridad de la información y muchas empresas han certificado su cumplimiento; aquí se puede ver en la figura 2-1 la cantidad de certificados emitidos en los últimos años:

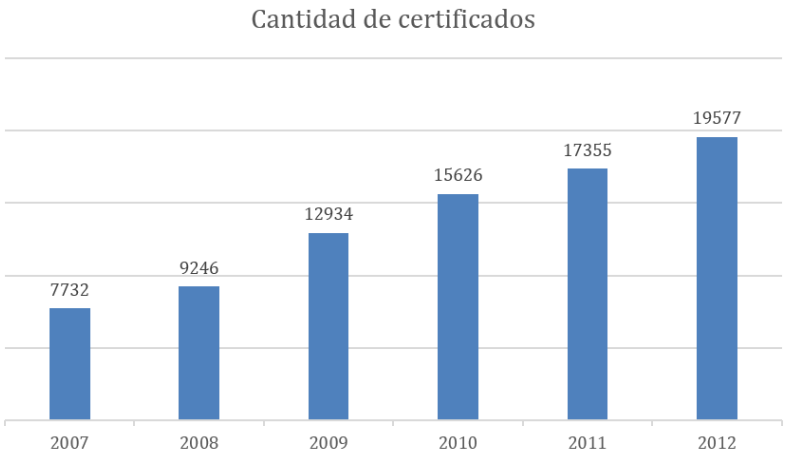

Figura 2-1: Cantidad de certificados emitidos

Fuente: Encuesta ISO sobre certificaciones de la norma para sistemas de gestión, [KOOLE15]

Establece un sistema gerencial que permite minimizar el riesgo y proteger la información de amenazas externas o internas. No está orientada a despliegues tecnológicos o de infraestructura, sino a aspectos netamente organizativos, es decir, la frase que podría definir su propósito es organizar la seguridad de la información.

Actualmente es el único estándar aceptado internacionalmente para la administración de la Seguridad de la Información y se aplica a todo tipo de organizaciones, independientemente de su tamaño o actividad. [KOOLE15]

Su objetivo principal es el establecimiento e implementación de un Sistema de Gestión de la Seguridad de la Información.

La seguridad de la información debe preservar la:

- Confidencialidad: Aseguramiento de que la información es accesible sólo para aquellos autorizados a tener acceso.
- Integridad: Garantía de la exactitud y completitud de la información y de los métodos de su procesamiento.
- Disponibilidad: Aseguramiento de que los usuarios autorizados tienen acceso cuando lo requieran a la información y sus activos asociados.

Entre otros objetivos de la norma son los siguientes:

- La definición clara y transmitida a toda la organización de los objetivos y directrices de seguridad.
- La sistematización, objetividad y consistencia a lo largo del tiempo en las actuaciones de seguridad.
- El análisis y prevención de los riesgos en los Sistemas de Información.
- La mejora de los procesos y procedimientos de gestión de la información.
- La motivación del personal en cuanto a valoración de la información.
- El cumplimiento con la legislación vigente.
- Una imagen de calidad frente a clientes y proveedores.

Propone secuencias de acciones tendientes al:

- Establecimiento-Implementación.
- Operación.
- Monitorización.

- Revisión-Mantenimiento.
- Mejora SGSI Sistema de Gestión de la Seguridad dela Información.

2.3.1.1. Cómo funciona la ISO 27001

El eje central de ISO 27001 es proteger la confidencialidad, integridad y disponibilidad de la información en una empresa. Esto lo hace investigando cuáles son los potenciales problemas que podrían afectar la información (es decir, la evaluación de riesgos) y luego definiendo lo que es necesario hacer para evitar que estos problemas se produzcan (es decir, mitigación o tratamiento del riesgo). [KOOLE15]

Por lo tanto, la filosofía principal de la norma ISO 27001 se basa en la gestión de riesgos: investigar dónde están los riesgos y luego tratarlos sistemáticamente, como se ve en la figura 2-2.

Figura 2-2: Estructura de ISO 27001

Fuente: [KOOLE15]

Las medidas de seguridad (o controles) que se van a implementar se presentan, por lo general, bajo la forma de políticas, procedimientos e implementación técnica (por ejemplo, software y equipos). Sin embargo, en la mayoría de los casos, las empresas ya tienen todo el hardware y software pero utilizan de una forma no segura; por lo tanto, la mayor parte de la implementación de ISO 27001 estará relacionada con determinar las reglas organizacionales (por ejemplo, redacción de documentos) necesarias para prevenir violaciones de la seguridad. [KOOLE15]

Como este tipo de implementación demandará la gestión de múltiples políticas, procedimientos, personas, bienes, etc., ISO 27001 ha detallado cómo amalgamar todos estos elementos dentro del sistema de gestión de seguridad de la información (SGSI).

Por eso, la gestión de la seguridad de la información no se acota solamente a la seguridad de TI (por ejemplo, cortafuegos, anti-virus, etc.), sino que también tiene que ver con la gestión de procesos, de los recursos humanos, con la protección jurídica, la protección física, etc. [KOOLE15]

2.3.1.2. Beneficios de ISO 27001

Hay 4 ventajas comerciales esenciales que una empresa puede obtener con la implementación de esta norma para la seguridad de la información:

Cumplir con los requerimientos legales**,** cada vez hay más y más leyes, normativas y requerimientos contractuales relacionados con la seguridad de la información. La buena noticia es que la mayoría de ellos se pueden resolver implementando ISO 27001 ya que esta norma le proporciona una metodología perfecta para cumplir con todos ellos.

Obtener una ventaja comercial, si su empresa obtiene la certificación y sus competidores no, es posible que usted obtenga una ventaja sobre ellos ante los ojos de los clientes a los que les interesa mantener en forma segura su información.

Menores costos, la filosofía principal de ISO 27001 es evitar que se produzcan incidentes de seguridad, y cada incidente, ya sea grande o pequeño, cuesta dinero; por lo tanto, evitándolos su empresa va a ahorrar mucho dinero. Y lo mejor de todo es que la inversión en ISO 27001 es mucho menor que el ahorro que obtendrá.

Una mejor organización, en general, las empresas de rápido crecimiento no tienen tiempo para hacer una pausa y definir sus procesos y procedimientos; como consecuencia, muchas veces los empleados no saben qué hay que hacer, cuándo y quién debe hacerlo. La implementación de ISO 27001 ayuda a resolver este tipo de situaciones ya que alienta a las empresas a escribir sus principales procesos (incluso los que no están relacionados con la seguridad), lo que les permite reducir el tiempo perdido de sus empleados. [KOOLE15]

2.3.1.3. Aplicabilidad de la norma ISO 27001 para auditoria

ISO 27001 (Ver acápite 2.1.3.) es la única norma internacional auditable que define los requisitos para un sistema de gestión de seguridad de la información (SGSI). Donde

un SGCI es una parte del sistema de gestión de una organización, basado en una aproximación de los riesgos del negocio (actividad) para establecer, implementar, operar, monitorizar, revisar, mantener y mejorar la seguridad de la información. [KOOLE15]

La creación de un SGSI es una decisión estratégica en una organización y como tal, debe ser apoyada y supervisada por la dirección. El hecho de certificar un SGSI según la norma ISO 27001 puede aportar las siguientes ventajas a la empresa que siga un Modelo de Implementación:

- Demuestra la garantía independiente de los controles internos y cumple los requisitos de gestión corporativa y de continuidad de la actividad comercial;
- Demuestra independientemente que se respetan las leyes y normativas que sean de aplicación;
- Proporciona una ventaja competitiva al cumplir los requisitos contractuales y demostrar a los clientes que la seguridad de su información es primordial;
- Verifica independientemente que los riesgos de la organización estén correctamente identificados, evaluados y gestionados al tiempo que formaliza los procesos procedimientos y documentación de protección de la información;
- Demuestra el compromiso de la cúpula directiva de su organización con la seguridad de la información;
- El proceso de evaluaciones periódicas ayudan a supervisar continuamente el rendimiento y la mejora.

En la norma ISO 27001, se menciona lo siguiente:

"La organización, establecerá, implementará, operará, monitorizará, revisará, mantendrá y mejorará un documentado SGSI en su contexto para las actividades globales de su negocio y de cara a los riesgos". [KOOLE15]

El mismo documento debe contener:

- La política de seguridad;
- Las normas o estándares de funcionamiento;
- Los procedimientos detallados;
- Guías y recomendaciones.

Para implementar el SGSI se debe utilizar el ciclo continuo PDCA10 cuyo objetivo final es asegurar la Integridad, Confidencialidad y Disponibilidad dela Información.

La metodología PDCA es la médula de la instrumentación básica de la Gestión de la Calidad Total, pero vale la pena insistir en que su extraordinaria potencialidad técnica solo podrá ser bien aprovechada si hay adecuada motivación, participación y valorización de los técnicos y funcionarios. La metodología PDCA está integrada por cuatro pasos. [KOOLE15]

- PLANEAR - PLAN (ESTABLECER EL SGSI): Establecer política, objetivos, procesos y procedimientos SGSI relevantes para manejar el riesgo y mejorar la seguridad de la información para entregar resultados en concordancia con las políticas y objetivos generales de la organización;
- HACER - DO (IMPLEMENTAR Y OPERAR EL SGSI): Implementar y operar la política, controles, procesos y procedimientos SGSI;
- CHEQUEAR- CHECK (MONITOREAR Y REVISAR EL SGSI): Evaluar y, donde sea aplicable, medir el desempeño del proceso en comparación con la política, objetivos y experiencias prácticas SGSI y reportar los resultados a la gerencia para su revisión;
- ACTUAR-ACT (MANTENER Y MEJORAR EL SGSI):Tomar acciones correctivas y preventivas, basadas en los resultados de la auditoría interna SGSI y la revisión gerencial u otra información relevante, para lograr el mejoramiento continuo del SGSI.

El uso eficiente de estos recursos, aplicando rigurosamente los procedimientos correspondientes, permitirá obtener resultados cada vez mejores, alcanzándose las metas establecidas, o por lo menos, acercándose cada vez más a ellas. Se trata de una metodología relativamente simple, pero que, si bien aplicada gerencialmente y con funcionarios motivados, será extraordinariamente efectiva. [KOOLE15]

2.4. Aplicación web

La aplicación web se define como un software dirigido a mostrar información a los usuarios de la red internet en la mayoría de los casos. Estos puede contener elementos que permiten una comunicación activa entre el usuario y la información. Esto permite

que el usuario acceda a los datos de modo interactivo, gracias a que la aplicación responde a cada una de sus acciones, como por ejemplo rellenar y enviar formularios, participar en juegos diversos y acceder a gestores de base de datos de todo tipo con los que cuenta una aplicación web. [THEOPEN09]

2.4.1. Estructura de una aplicación web

Está estructurada como una aplicación en tres capas en su forma más común. La primera capa consiste en el navegador web, la segunda capa y central u motor capaz de usar la tecnología web dinámica, y la base de datos que constituye la tercera capa. El navegador web manda peticiones a la capa central que ofrece servicio de interfaz gráfica, permitiendo interactuar con la base de datos valiéndose de las consultas y actualizaciones. [THEOPEN09]

2.4.2. Arquitectura de una aplicación web

Se utiliza el término arquitectura web, para definir una tarea que requiere conocimientos técnicos de construcción, funcionales y de diseño para sitios o páginas web. La construcción de páginas web requiere una compleja conjunción de diferentes sistemas integrados entre sí: servidores, bases de datos, organización de la información, etcétera.

Tal como en la arquitectura tradicional, actualmente el foco para el diseño y construcción de páginas web se centra en el usuario y sus requerimientos.

La arquitectura de un sitio web tiene tres componentes principales:

- Un servidor Web
- Una conexión de red
- Uno o más clientes

El servidor web distribuye páginas de información formateada a los clientes que las solicitan. Los requerimientos son hechos a través de una conexión de red, y para ello se usa el protocolo HTTP. Una vez que se solicita esta petición mediante el protocolo HTTP y la recibe el servidor web, éste localiza y envía al navegador par su despliegue.

Una importante característica de aplicaciones web es que fue proyectado para funcionar en topología de Internet. De esta forma el web incorpora naturalmente la característica de ser un ambiente distribuido y multiplataforma. [JAV10]

2.4.3. Servicios web

Es un conjunto de protocolos y estándares que se utilizan para intercambiar datos entre aplicaciones. Las distintas aplicaciones desarrolladas con diferentes tecnologías de implementación y ejecutas sobre cualquier plataforma permitiendo la utilización de los servicios web para intercambiar datos en redes internet. [THEOPEN09]

2.5. Seguridad de aplicaciones web

La seguridad es un aspecto importante para proteger la integridad y privacidad de los datos y recursos de las aplicaciones web. Debiendo designar una estrategia de seguridad para la aplicación web, que use soluciones de seguridad de eficacia probada, e implementar métodos de autenticación, autorización y validación de datos, para proteger la aplicación de una serie de amenazas. [ROM14]

2.5.1. Validación de datos

La validación de datos es un proceso que consiste en realizar un filtro de los datos proporcionados por el usuario, si estos son correctos permite el acceso caso contrario regresa mensajes de erros basándose en procedimientos definidos en la implementación.

Es una de las áreas más importantes a tener en cuenta en las aplicaciones orientadas a la web, para la no existencia de código oculto de por medio. [THEOPEN09]

2.5.2. Manejo de sesiones

Es un método ampliamente utilizado en muchos tipos de aplicaciones web, es la secuencia de páginas que usuario visita en un sitio web desde que entra hasta que lo abandona.

Tiene como objetivo primordial asegurar que solo los usuarios autenticados posean una amplia navegación y acceso a la información requerida, haciendo cumplir los controles de autorización previniendo los ataques. [THEOPEN09]

2.5.3. Ataque a la aplicación web

Se refiere a una acción o un método por el cual individuo haciendo uso de un sistema informático intenta dañar o alterar una aplicación web.

Es un intento organizado y deliberado de una o más personas para infiltrarse en las aplicaciones web de grandes organizaciones preferentemente por la confidencialidad de la información que manejan. [VVA13]

2.5.4. Riesgos en una aplicación web

El uso masivo de las aplicaciones web en la actualidad generado un riesgo, por tal motivo se debe emplear políticas de seguridad en el desarrollo, desde la fase inicial de su diseño hasta su puesta en funcionamiento, sin embargo muchas veces la seguridad no es tomada en cuenta hasta que los fallos se presentan. En el trayecto del desarrollo no solo debe dar importancia a los usuarios y sus requerimientos, sino también en los eventos que puedan inferir con la seguridad y el contenido de la información. [VVA13]

2.5.5. Vulnerabilidad

La vulnerabilidad es la debilidad de un recurso o grupo de recursos que son aprovechados por una o varias amenazas generadas a partir de la falta de seguridad en el sistema esto ocurren en dos factores Hardware o Software.

La vulnerabilidad del hardware es la facilidad de acceso a los dispositivos y la vulnerabilidad del software son las fallas o debilidades del sistema. [ROM14]

Las aplicaciones web pueden presentar diversas vulnerabilidades que van de acuerdo a los servicios que prestan. De acuerdo con la OWASP (Open Web Application Security Project) las vulnerabilidades en aplicaciones web más explotadas, fueron las siguientes:

- Cross Site Scripting (XSS).
- Ataques de inyección de código.
- Ejecución de archivos maliciosos.
- Insecure Direct Object Reference.
- Ataques Cross Site Request Forgery (CSRF).
- Pérdidas de información y errores al procesar mensajes de error.
- Robo de identidad de autentificación.
- Almacenamiento criptográfico inseguro.
- Comunicaciones inseguras.
- Acceso a URLs ocultas no restringidas de manera adecuada.

Estas vulnerabilidades son aprovechados para realizar ataques, es por eso que es necesario tomar en cuenta durante el proceso de implementación de las aplicaciones web. [ROM14]

2.6. Sistema de detección de intrusos

Uno de los mecanismos de defensa más usados para reducir el riesgo de las compañías ante ataques dirigidos hacia los bienes informáticos han sido los sistemas de detección de Intrusos o IDS (Intrusion Detection Systems). [KIO14]

Un IDS es un elemento que escucha y analiza toda la información que circula por una red de datos e identifica posibles ataques. Cuando aparece un ataque, el sistema reaccionará informando al administrador y cerrará las puertas al posible intruso reconfigurando elementos de la red como firewalls y routers.

Los IDS han sido usados ampliamente a lo largo de estos últimos años por muchas compañías porque han proporcionado una capa adicional de seguridad. Sin embargo se ha encontrado que estos elementos proporcionan seguridad reactiva, es decir para que haya protección debe existir primero un ataque. Si un ataque es pequeño y efectivo el IDS reaccionará demasiado tarde y el ataque logrará su objetivo. [KIO14]

Los IDS permiten, no solo la detección de ataques cubiertos por otros componentes de seguridad, sino la detección de intrusiones que pasan desapercibidas a otros componentes del dispositivo de seguridad. Así pues, realizan dos tareas fundamentales: la prevención y la reacción, como se observa en la figura 2-3.

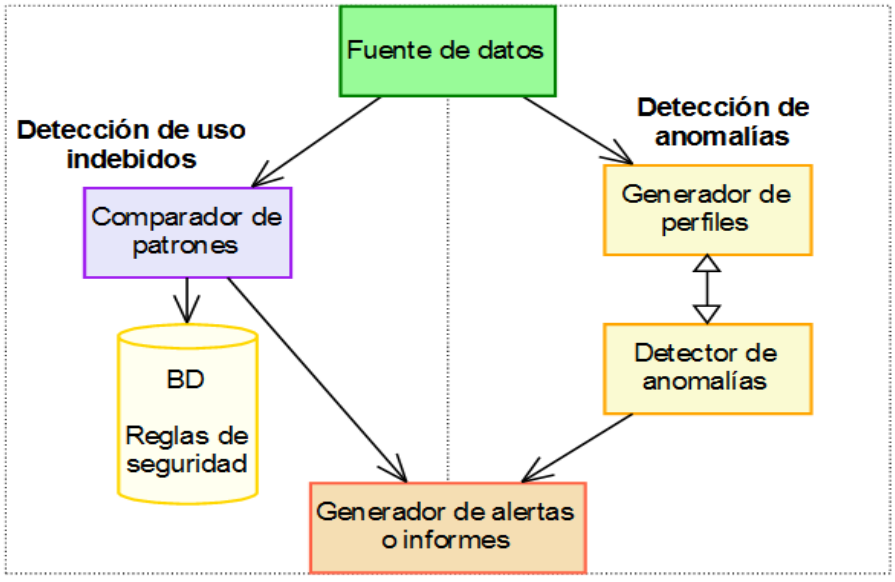

Figura 2-3: Esquema general de un IDS

Fuente: [GON10]

2.6.1. Funciones de un IDS

Estos sistemas introducen métodos de trabajo que permiten complementar y completar el trabajo realizado por otras herramientas de seguridad como los cortafuegos. Las funciones de un IDS se pueden resumir de la siguiente forma: [PFC11]

- Detección de ataques en el momento que están ocurriendo o poco después.
- Automatización de la búsqueda de nuevos patrones de ataque, gracias a herramientas estadísticas de búsqueda, y al análisis de tráfico anómalo.
- Monitorización y análisis de las actividades de los usuarios. De este modo se pueden conocer los servicios que usan los usuarios, y estudiar el contenido del tráfico, en busca de elementos anómalos.
- Auditoría de configuraciones y vulnerabilidades de determinados sistemas;
- Descubrir sistemas con servicios habilitados que no deberían de tener, mediante el análisis del tráfico y de los logs.
- Análisis de comportamiento anormal. Si se detecta una conexión fuera de hora, reintentos de conexión fallidos y otros, existe la posibilidad de que se esté en presencia de una intrusión. Un análisis detallado del tráfico y los logs puede revelar una máquina comprometida o un usuario con su contraseña al descubierto.
- Automatizar tareas como la actualización de reglas, la obtención y análisis de logs, la configuración de cortafuegos y otros.
- Logs que nos permite visualizar los acontecimientos y la funcionalidad de una aplicación.

Un IDS puede compartir u obtener información de otros sistemas como firewalls, routers y switches, lo que permite reconfigurar las características de la red de acuerdo a los eventos que se generan. También permite que se utilicen protocolos como SNMP (Simple Network Management Protocol) para enviar notificaciones y alertas a otras máquinas de la red. Esta característica de los IDS recibe el nombre de interoperabilidad.

Otra característica a destacar en los IDS, es la correlación, que consiste en la capacidad de establecer relaciones lógicas entre eventos diferentes e independientes, lo que permite manejar eventos de seguridad complejos que individualmente no son muy

significativos, pero que analizados como un todo pueden representar un riesgo alto en la seguridad del sistema. [PFC11]

La utilidad de un sistema de detección de intrusos debe ser evaluada teniendo en cuenta la probabilidad que tiene el sistema de detectar un ataque y la probabilidad de emitir falsas alarmas. Teniendo en cuenta el gran número de alertas que se generan y la gran cantidad de falsas alarmas producidas, la gestión o revisión de éstas se convierte en una tarea muy complicada y la carga de trabajo se multiplica para los administradores de sistemas. Son pues, sistemas que requieren de un mantenimiento y una supervisión constante. [PFC11]

Actualmente, hay herramientas gráficas que permiten visualizar los datos almacenados por los sistemas de detección y presentan los datos en distintas interfaces permitiendo encontrar y analizar detalles, tendencias y problemas que a simple vista no encontraríamos.

Estas herramientas es un aporte de beneficioso para detectar los ataques y prevenir de alguna manera, y realizar acciones necesarias. [PFC11]

2.6.2. Tipos de sistemas de detección de intrusos

Existen distintos tipos de IDS, atendiendo a distintas clasificaciones establecidas de acuerdo a las características que se usen para establecer dicha clasificación. Cada uno de ellos se caracteriza por diferentes aproximaciones de monitoreo y análisis y presenta distintas ventajas y desventajas como se observa en la figura 2-2. [PFC11]

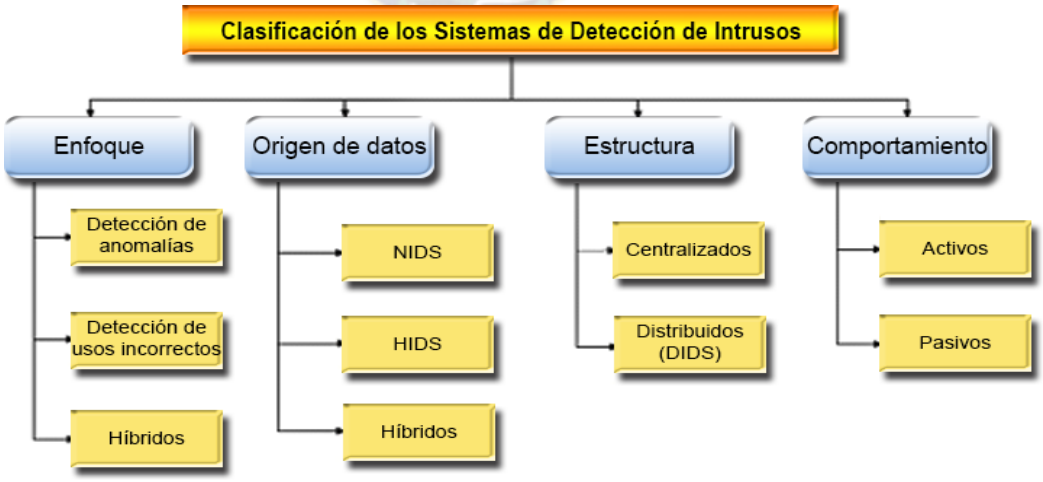

Figura 2-4: Clasificación del IDS

Fuente: [GON10]

2.6.2.1. Tipos de IDS en función del origen de los datos

Los IDS pueden clasificarse atendiendo a las fuentes de información que se utilicen. Así tenemos tres tipos: IDS basados en host, en Red y los IDS híbridos. [PFC11]

2.6.2.1.1. HIDS: Host-based Intrusion Detection Systems

Los HIDS están diseñados para monitorizar, detectar y responder a los datos generados por un usuario o un sistema en un determinado host, fueron los primeros IDS que surgieron. Estos IDS son útiles para identificar amenazas e intrusiones a nivel del host local. Monitorizan gran cantidad de eventos, analizando actividades con una gran precisión, determinando de esta manera qué procesos y usuarios se involucran en una determinada acción. Recaban información del sistema como ficheros, logs, recursos, para su posterior análisis en busca de posibles incidencias.

Sin embargo, cuando un sistema comprende cientos de hosts, los HIDS, por sí solos, no son viables para realizar una monitorización adecuada. Requieren confianza en el sistema donde se van a instalar (un sistema de detección de intrusos de host no será muy útil en un sistema que ha sido comprometido anteriormente). Impactan directamente sobre el sistema que protegen; dado que comparten los mismos recursos que el sistema y aplicaciones que protegen y son vulnerables ante un ataque directo. [PFC11]

Los HIDS utilizan las bitácoras del sistema, las cuales se generan de forma automática por diferentes aplicaciones o por el propio núcleo del sistema operativo. Como se muestra en la Figura 2-5, se hace un análisis de las bitácoras prestando especial atención a los registros relativos a demonios de red, como un servidor web o el propio inetd. Por otra parte, se usan también verificadores de integridad de determinados ficheros de importancia vital para el sistema, como el de contraseñas. [PFC11]

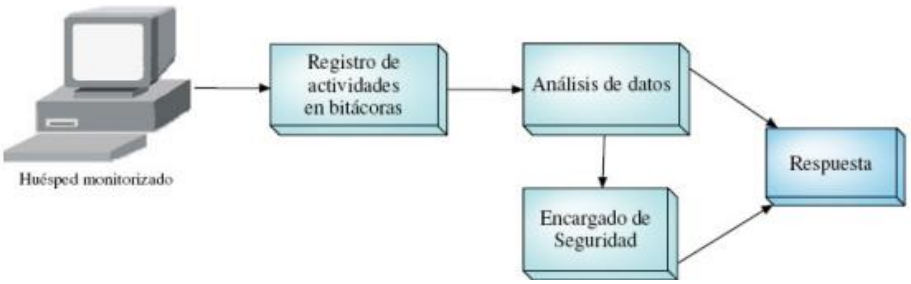

Figura 2-5: Sistema de detección de intrusos basado en host

Fuente: [PFC11]

2.6.2.1.2. NIDS: Network Intrusion Detection Systems

Los NIDS son muy parecidos a los HIDS en tanto que monitorizan, detectan y responden ante los datos generados, pero en vez de proteger un determinado host, reciben los datos de la red local donde estén instalados. Tienen la ventaja de ser, normalmente, pasivos, de forma que no interfieren en el correcto uso de la red. Actúan mediante la utilización de un dispositivo de red configurado en modo promiscuo, capturando todos los paquetes que pasan por él y almacenando la información de estos paquetes en un repositorio para su posterior análisis en busca de patrones indicativos de un ataque. Véase la figura 2-6.

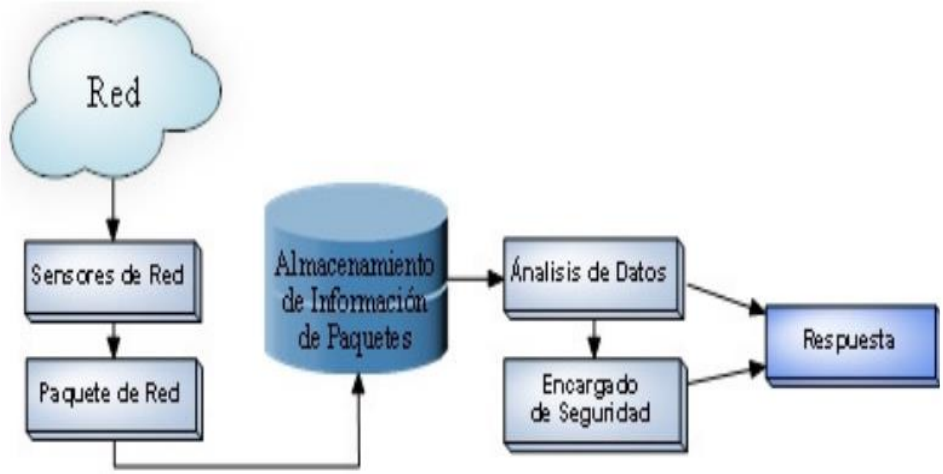

Figura 2-6: Sistema de detección de intrusos basado en red

Fuente: [PFC11]

Analizan el tráfico de red, normalmente, en tiempo real. No sólo trabajan a nivel TCP/IP, también lo pueden hacer a nivel de aplicación. Permiten la detección de ataques a un mayor nivel de abstracción, puesto que ahora no solo tienen información de un solo host, sino de múltiples. No obstante, los NIDS solo perciben información que pasa por la red, siendo inútil ante los ataques locales de los usuarios de un determinado host. [PFC11]

Un NIDS, puede protegernos contra los ataques que entran a través de los cortafuegos hasta la LAN Interna. Los cortafuegos pueden estar mal configurados, permitiendo la introducción de tráfico no deseado en nuestra red. Incluso cuando funcionan correctamente, los cortafuegos normalmente dejan pasar algún tráfico de aplicación que

puede ser peligroso, ya que sólo puede ver el tráfico que lo atraviesa desde el exterior y no respecto a la actividad de la LAN.

Podemos pensar en los NIDS y en los cortafuegos como dispositivos de seguridad complementarios. Lo que llega a los puertos de cortafuegos, permitido por las reglas, se envía a los servidores internos provocando un tráfico potencialmente dañino. Un NIDS puede comprobar dicho tráfico y marcar los paquetes sospechosos. Configurado correctamente puede hacer una doble comprobación delas reglas de nuestro cortafuegos y proporcionarnos una protección adicional para los servidores de aplicación. Bien ubicados, pueden analizar grandes redes y su impacto en el tráfico suele ser pequeño.

Los NIDS, requieren emular el comportamiento de los sistemas que protege para mantener una sincronía entre su procesamiento y el de estos equipos; si existe asincronía, el sistema puede ser evadido. También necesita capacidad de procesamiento suficiente para evitar la pérdida de paquetes y visibilidad de los equipos que vigila (configurar puertos de vigilancia en switches para permitir a este tipo de sistemas vigilar el tráfico de los sistemas que protege puede tener consecuencias de desempeño en todo el segmento de red). [PFC11]

2.7. Detección de riesgos en las aplicaciones web

En 1996 Dan Farmer realizo un estudio comparativo acerca de la detección de intruso, en los sistemas informáticos y sitios web de comercio haciendo uso de técnicas sencillas, catalogando los tipos de problema en dos grupos: [SEIN15]

- Rojo: En este grupo se encuentran los sistemas potencialmente sensibles a ataques, es decir los problemas de seguridad son conocidos por los atacantes. Podemos mencionar el servicio FTP mal configurado;
- Amarillo: Los ataques de este grupo son menos serios ya que el problema detectado no implica un daño serio e inmediato al sistema, sin embargo esto no significa que no exista el riesgo y que no cause daños como la perdida de información, modificación, o bien robo de la misma.

En la siguiente tabla 2-1 se muestran los sistemas que fueron evaluados y la categoría en la cual se asignaron, y el porcentaje de vulnerabilidad. Se observa que los sitios de noticias son mucho más vulnerables, seguidos de los sitios gubernamentales.

Tipos	Cantidad	Vulnerabilidad (%)	Amarilla (%)	Roja (%)
Bancos	660	68.34	32.73	35.61
Financieros	274	51.1	30.66	20.44
Gobiernos	47	61.1	23.4	38.30
Informativas	312	69.55	30.77	38.78
Otros	469	33.05	15.78	17.27
Total	**1762**	**56,628**	**26,668**	**30,08**

Tabla 2-1: Porcentaje de vulnerabilidad por tipo de sitio

Fuente: [SEIN15]

Haciendo un énfasis, se mención a tres principales puntos de ataque hacia una aplicación web, independientemente de la tecnología aplicada en su desarrollo:

- Ataque hacia el cliente;
- Ataque hacia el canal de comunicación;
- Ataque hacia servidores.

Estos medios al ser los principales medios de comunicación de una aplicación web son los que más sufren ataques. [SEIN15]

2.7.1. Detección de riesgos del lado del cliente

En la tabla 2-2 observa la clasificación de riesgos encontrados de lado del cliente, con la descripción respectiva.

RIESGO	CAUSAS	IMPACTO
Phishing Es la capacidad de duplicar una página web para hacer creer al visitante que se encuentra en la página original en lugar de la apropiada. Normalmente se utiliza con fines delictivos duplicando	• Falta de cultura informática; • Políticas deficientes; • Inadecuada propaganda.	• Desprestigio del sitio web; • Fraudes.

páginas web de bancos conocidos y enviando indiscriminadamente correos para que se acceda a esta página para actualizar las páginas de acceso al banco.		
Falta de control en las "Transacciones" Las transacciones son operaciones entre componentes, un ejemplo sería una transacción cuando se modifica el estado de una ID, por esto es que en una aplicación web se pone empeño en la operación de las transacciones, ya que estas conllevan a la falla de las aplicaciones.	• Transacciones limitadas.	• Robo o pérdida de la información; • Acceso a zonas no autorizadas del sistema.
Fraudes por ausencia de "Roles" Un rol se define como un perfil que designa el comportamiento de los usuarios que tienen acceso a la aplicación, estos se distinguen entre sí por el nivel de privilegios que poseen	• Ausencia de Roles	• Difícil administración de la aplicación; • Cambios sin autorización en cuentas de otros usuarios.

respecto a la Aplicación Web.		
Cross Site Scripting (CSS) Una aplicación web puede ser usada como un mecanismo para transportar un ataque al navegador del usuario final. Un ataque exitoso puede comprometer el Token de sesión del usuario final, atacar la maquina local o enmascarar contenido para engañar al usuario.	• Validadores de entradas de datos deficientes.	• Desprestigio de la aplicación web.
Ataques Unicode El ataque Unicode es utilizado para forzar a los servidores web a salir de la raíz de la aplicación y acceder a archivos residentes en otras partes de la ruta de la aplicación Esto es logrado provocando un error en el cliente.	• Validadores de entradas de datos deficientes.	• Divulgación de información privada de aplicaciones web tales como rutas, direcciones, puertos.
Inyección de código Técnica consistente en aprovechar las debilidades que ofrece la aplicación en sus	• Validaciones de entradas de datos deficientes.	• Alteración del comportamiento de la aplicación, provocando:

validaciones e ingresar código malicioso en sus campos de entrada permitiendo que estos se ejecuten en la aplicación web.		• Acceso a zonas no autorizadas de la aplicación web; • Alteración del comportamiento de la aplicación.

Tabla 2-2: Detección de riesgos al lado del cliente

2.7.2. Detección de riesgos del lado del servidor

En la siguiente tabla 2-3 se presentan los riesgos y las técnicas de ataque al lado del servidor.

RIESGO	CAUSAS	IMPACTO
Desbordamiento de buffer Debilidad ligada al servidor, ya que esta tiene el control de los procesos que ejecuta la aplicación, esta condición ocurre cuando los datos escritos en la memoria exceden el tamaño reservado en el buffer y direcciones de memoria adyacentes son sobrescritas causando que la aplicación falle o termine de manera inesperada. Los atacantes suelen corromper los buffer de memoria para interrumpir el funcionamiento de la	• Validaciones de entradas de los datos deficientes; • Mala programación.	• La parte que afecta directamente este tipo de ataques es el Back end de la aplicación Por ejemplo puede causar problemas en el funcionamiento de la ID.

aplicación en el servidor y así denegar los servicios. Esto lo logran introduciendo grandes cantidades de información al sistema web, por ejemplo, en una aplicación de Comercio Electrónico un carrito de compras con un excesivo número de productos puede ser causa inmediata para un desbordamiento de Memoria.		
Caída del servidor por mal manejo de errores. Condiciones de error que ocurren durante la operación normal que no son manejadas adecuadamente.	• Manejo inadecuado de errores.	• Si un agresor puede causar que ocurran errores que la aplicación web no maneja, este puede obtener información detallada del sistema, denegar servicios, causar que mecanismos de seguridad fallen.
Administración de autenticación y sesión de servicio interrumpida	• Mala programación.	• Fraudes en la aplicación.
Acceso y modificación a configuraciones del Servidor	• Administración de configuración Insegura.	• Fallas de seguridad en el software del servidor; • Alteraciones al

		listado de directorio o acceso no autorizado de directorio.
Generación de basura dentro del servidor	• Administración de configuración Insegura.	• Generación de archivos innecesarios por ejemplo, de respaldo o de ejemplo, incluyendo Scripts, Logs, aplicaciones archivos de configuración y páginas web.
Vulnerabilidad en los accesos a las propiedades de configuración del servidor	• Administración de configuración Insegura.	• Permisos no adecuados en archivos y directorios.
Infiltración de los usuarios malintencionados o no autorizados	• Cuentas de usuario definidas por defecto en la instalación.	• Acceso no autorizado a propiedades del servidor final.
Perdida de información del servidor web	• Funciones administrativas o de depuración que son habilitadas o accesibles	• Alto costo en recuperación de información de sistema.

Tabla 2-3: Detección de riesgos del lado del servidor

2.7.3. Detección de riesgos en el canal de comunicación

En la tabla 2-4 se muestra riesgos en el trayecto, es decir en el recorrido de la información por la red, la información solicitada puede ser interceptada y ser modificado con datos ajenos.

RIESGO	CAUSAS	IMPACTO
Robo de información en los canales de información	• Certificados SSL y opciones de encristalaron mal configurados o no habilitados.	• Alto costo en recuperación de información sensible de la aplicación que corre sobre el servidor web.
Interceptación de información sensible Las aplicaciones web frecuentemente utilizan funciones de criptografía para proteger información y credenciales. Estas funciones y el código que integran a ellas han sido difíciles de codificar adecuadamente, lo cual frecuentemente redunda en una protección débil.	• Almacenamiento de información insegura.	• Robo de información sensible.

Tabla 2-4: Detección de riegos en el canal de comunicación

2.8. Seguridad en Bases de Datos

La gran mayoría de los datos sensibles del mundo están almacenados en sistemas gestores de bases de datos comerciales tales como Oracle, Microsoft SQL Server entre otros, y atacar una bases de datos es uno de los objetivos favoritos para los criminales.

Esto puede explicar por qué los ataques externos, tales como inyección de SQL, subieron 56.2% en 2012, "Esta tendencia es prueba adicional de que los agresores tienen éxito en hospedar páginas Web maliciosas, y de que las vulnerabilidades y explotación en relación a los navegadores Web están conformando un beneficio importante para ellos. [VIL12]

Mientras que la atención generalmente se ha centrado en asegurar los perímetros de las redes por medio de, firewalls, IDS / IPS y antivirus, cada vez más las organizaciones se están enfocando en la seguridad de las bases de datos con datos críticos, protegiéndolos de intrusiones y cambios no autorizados.

En las siguientes secciones daremos las siete recomendaciones para proteger una base de datos en instalaciones tradicionales. [VIL12]

2.8.1. Inyección de código

La inyección de código es común entre los individuos que gustan por atacar las páginas web, generalmente usan esta técnica para obtener otra información que les ayude a atacar en mayor grado los sistemas.

Para ello hace uso de los errores al procesar información errónea, esto es usado por un atacante para cambiar la ejecución o funcionalidad normal, más aún puede ser usado para la propagación de algunos códigos maléficos con fines de dañar o alterar la información.

Una inyección de código puede ser de tipo SQL o Fichero, el de SQL se aprovecha de la sintaxis en la instrucción y el de fichero hace referencia a un archivo externo. [VIL12]

2.8.2. Inyección de código maliciosa

Este tipo de inyección de código malicioso se realiza de la siguiente manera.

- Instalación de malware en alguna computadora por medio de la inyección de código a un navegador web o en sus plugins;

- Instalación de malware inyectando código en aplicaciones web desarrollados en cualquiera de las plataformas disponibles;
- La inyección de código puede ser usada por medio del shell del sistema operativo, para obtener mayores privilegios de los permitidos.

Robo de sesiones desde el navegador Web usando inyección HTML/Script (Cross-site scripting). [SEIN15]

2.8.3. Inyección de código benéfico

Así como hay inyección de código maliciosa, también hay inyección de código benéfica para el programador, por ejemplo, se puede modificar una tabla de la base de datos de un sistema existente usando la inyección de datos. Básicamente la inyección de código benéfica es útil para modificar el sistema de alguna manera eficiente y con menores costos.

La inyección de código beneficioso como puede ser útil para modificar algunos componentes de una aplicación, también puede ser perjudicial en algunos casos. [SEIN15]

2.8.4. Inyección de código inesperado

Es cuando el usuario ingresa caracteres inválidos en el sistema, lo cual puede ocasionar que funcione indebidamente con comportamientos inesperados, es por eso que se recomienda que se tenga un control de caracteres en los campos donde se requiere que el usuario ingrese datos al sistema.

Atacando a través de campos de entrada incorrectamente validados, o explotando alguna vulnerabilidad presente. Con el propósito de infectar el más grande número posible de sitios web con un sencillo mecanismo, los atacantes intentarán atacar un tipo específico de clase buscando vulnerabilidades comunes en los sitios y generalmente automatizando el descubrimiento y exploración. Esto permite a los atacantes infectar sitios web con la eficiencia comúnmente encontrada en Gusanos de Internet. [SEIN15]

2.8.5. Inyección SQL

Es la inserción de código SQL por medio de entradas desde la parte del cliente hacia la aplicación en una petición. Permitiendo al atacante modificar las consultas programadas en la aplicación y ejecutar otras acciones totalmente distintas con la

intención de acceder a la información privilegiada o confidencia, borrar los datos almacenados, entre otras muchas cosas. [VIL12]

Como consecuencias de estos ataques y dependiendo de los privilegios que tenga el usuario de la base de datos bajo el que se ejecutan las consultas, se podría acceder no sólo a las tablas relacionadas con la aplicación, sino también a otras tablas pertenecientes a otras bases de datos alojadas en ese mismo servidor web.

Esto ocurre a causa ciertos caracteres en los campos de entrada de información por parte del usuario, ya sea mediante el uso de los campos de formularios que son enviados al servidor mediante POST o GET en las urls de las páginas web, lo mismos posibilitan coordinar varias consultas SQL o ignorar el resto de la consulta, para ello una alternativa es utilizar validaciones más seguras y cambios en la sintaxis de instrucción. [VIL12]

2.8.6. Encriptación de base de datos

En la actualidad la mayor parte de bases de datos contienen la información sensible, propia, o privada. Esto puede incluir la información de confidencial. La llave al mantenimiento de esta información en una manera segura es la confidencialidad. [CRIP06]

La seguridad de los datos implica protegerlos de operaciones indebidas que pongan en peligro su definición, existencia, consistencia e integridad independientemente de la persona que los accede. Esto se logra mediante mecanismos que permiten estructurar y controlar el acceso y actualización de los mismos sin necesidad de modificar o alterar el diseño del modelo de datos; definido de acuerdo a los requisitos del sistema o aplicación software. [CRIP06]

Toda encriptación se encuentra basada en un Algoritmo, la función de este Algoritmo es básicamente codificar la información para que sea indescifrable a simple vista, por ejemplo de manera que una letra "A" pueda equivaler a:"5x5mBwE" o bien a "xQE9fq", el trabajo del algoritmo es precisamente determinar cómo será transformada la información de su estado original a otro que sea muy difícil de descifrar. [CRIP06]

Una vez que la información arribe a su destino final, se aplica el algoritmo al contenido codificado "5x5mBwE" o bien a "xQE9fq" y resulta en la letra "A" o según sea el caso, en otra letra. Hoy en día los algoritmos de encriptación son ampliamente conocidos, es por esto que para prevenir a otro usuario "no autorizado" descifrar

información encriptado, el algoritmo utiliza lo que es denominado llave ("key") para controlar la encriptación y decriptación de información. Algunos algoritmos son DES (algoritmo simétrico) AES que posiblemente suplantará a DES y uno de los más conocidos RSA (algoritmo asimétrico).

2.8.6.1. Encriptación de datos en transito

La mayoría de los ambientes de base de datos utilizan TCP/IP y el servidor de base de datos escucha algunos puertos y acepta las conexiones iniciadas por los clientes de la base de datos. Mientras que los puertos son configurables, la mayoría de las veces se utiliza los puertos por defecto del servidor que está usando, por ejemplo: puerto 1433 para el servidor de Microsoft SQL, puerto 1521 para Oracle, puerto 4100 para Sybase, puerto 50000 para DB2, y puerto 3306 para MySQL. [CRIP06]

Los clientes de la base de datos se conectan con el servidor sobre estos puertos predefinidos para iniciar una comunicación, dependiendo del tipo de la base de datos y la configuración del servidor, redireccionando a otro puerto o terminando la comunicación entera sobre el mismo puerto del servidor. [CRIP06]

En un alto nivel, esto significa que con las herramientas derechas y el acceso correcto a la red, cualquiera puede golpear ligeramente en sus conversaciones de la base de datos y escuchar detrás de las puertas encendido el acceso de base de datos capturando y robando las declaraciones que se publica también los datos enviados por el servidor de base de datos. [CRIP06]

Meterse en tu base de datos de comunicaciones es relativamente fácil, porque la base de datos de comunicaciones son en su mayoría en texto legible, mediante el uso de los servicios públicos y en su mayoría simples herramientas libres, que un hacker pueda escuchar y robar información. La manera de evitar que esto suceda es encriptar las comunicaciones entre la base de datos de clientes y servidores de bases de datos. Este tipo de cifrado es llamado cifrado de los datos en tránsito porque todos, las comunicaciones entre el cliente y el servidor están encriptadas. La encriptación se produce en los extremos. [CRIP06]

Aunque el cifrado de los datos en tránsito se está convirtiendo en gran importancia, porque la mayoría no encripta los datos en tránsito, y para muchos ambientes en que está

perfectamente bien. Si usted cree que un potencial espía, es algo que no puede vivir con él, entonces se debería cifrar los datos en tránsito.

2.8.7. Cross site scripting (XSS)

Es un agujero de seguridad común en las aplicaciones web que permite inyectar código JavaScript evitando medidas de control de seguridad.

No solamente se limita a sitios web, también en aplicaciones locales o en los mismos navegadores, porque no se validan de manera correcta los datos de entrada. [OWA14]

2.9. Ataques DDOS

Para poder identificar las amenazas que puede correr una aplicación web es necesario primero conocer los tipos de ataques, las formas de acceso, el objetivo del mismo y la forma en la que opera. La forma en la que afecta el ataque DDoS (Denegación de Servicios) es en bajar los servicios que deberían de estar disponibles, el ataque corrupción de los datos consiste en modificar la información afectando así la estabilidad del negocio.

El ataque DDoS afecta a todo tipo de plataformas, una forma de evitar este tipo de ataques es monitoreando las IPs que realizan la petición a nuestra aplicación si estas sobrepasan de un número de conexiones HTTP en rango de tiempo corto es considerado un ataque. [SEIN15]

El código malicioso es un ataque informático que requiere de la intervención del usuario para propagar; Últimamente Latinoamérica es una de las regiones más afectadas por Gaobot, Spybot y Netsky P.

La forma de propagación de este último consiste en un envío masivo de sí mismo usando una extensión .zip ya que de esta forma puede evadir los filtros de seguridad, estos archivos con esta extensión .zip generalmente son confiables, por tal motivo el usuario final abre el archivo provocando la propagación del virus. [SEIN15]

2.10. Mejores prácticas en el desarrollo

Existen varias mejores prácticas que se pueden implementar para las aplicaciones web desarrolladas por las empresas dedicadas al área del desarrollado, dichas prácticas pueden ser ITIL, CMMi, ISO27001, Six Sigma, entre otros. A continuación se presenta una breve descripción de cómo apoyarse en ITIL y COBIT que son las más utilizadas en la actualidad.

2.10.1. ITIL

Las empresas de desarrollo de aplicaciones necesitan concentrarse en la calidad de los servicios que brindan, y asegurarse que los mismos estén alineados a los objetivos de la organización que las contrata.

Cuando los servicios son críticos, cada una de las actividades que se realizan deben de estar ejecutadas con un orden determinado para asegurar que la empresa encargada o el desarrollador proporcione la entrega de los servicios de forma consistente, esto brinda un mejor rendimiento en la calidad. [QUE11]

ITIL tiene que ver con todos aquellos procesos que se requieren ejecutar dentro de las organizaciones para la administración y operación de la infraestructura de las empresas encargadas de la implementación, de tal forma que se tenga una óptima provisión de servicios a los clientes bajo un esquema de costos congruentes con las estrategias del negocio.

Desarrollada su primera versión a finales de 1980, la Biblioteca de Infraestructura de Tecnologías de la Información (ITIL) se ha convertido en el estándar mundial de facto en la Gestión de Servicios Informáticos. Uno de los conceptos esenciales de ITIL es que establece que para una adecuada Gestión de Servicios en las Tecnologías de Información es necesaria una mezcla entre tres factores: Personas, Procesos y Tecnología, son las principales implicancias con las tecnologías de información. [QUE11]

Como las características que tiene ITIL podemos mencionar lo siguiente:

- Es un Framework de procesos de desarrollo no propietario;
- Es independiente de los proveedores;
- Es independiente de la tecnología;
- Está basado en "Best Practices".

Y además provee:

- Una terminología estándar;
- Las interdependencias entre los procesos;
- Los lineamientos para la implementación;
- Los lineamientos para la definición de roles y responsabilidades de los procesos;
- Las bases para la comparación de la empresa frente a las “mejores prácticas”.

En la administración o la gestión de servicios informáticos es abarcada por dos publicaciones: Entrega de Servicios y Soporte de Servicios. [QUE11]

Entrega de Servicios: Cubre los procesos necesarios para la planeación y entrega de la calidad de los servicios de tecnologías de información, estos procesos son:

- Administración de niveles de servicio;
- Administración financiera;
- Administración de capacidad;
- Administración de la continuidad de servicios;
- Administración de la Disponibilidad.

Soporte de Servicios: Proporciona los detalles de la función de mesa de servicio y los procesos necesarios para el soporte y mantenimiento de los servicios, estos procesos son los siguientes:

- Administración de incidentes;
- Administración de problemas;
- Administración de configuraciones;
- Administración de cambios;
- Administración de releases.

La Gestión de servicios organiza las actividades necesarias para administrar la entrega y soporte de servicios en procesos.

Un proceso es una serie de actividades que a partir de una entrada obtienen una salida. El flujo de la información dentro y fuera de cada área de proceso indicará la calidad del proceso en particular. [QUE11]

Existen puntos de monitoreo en el proceso para medir la calidad de los productos y provisión de los servicios.

Los procesos pueden ser medidos por su efectividad y eficiencia, es decir, si el proceso alcanzó su objetivo y si se hizo un óptimo uso de los recursos para lograr el objetivo.

Por lo que si el resultado de un proceso cumple con el estándar definido, entonces el proceso es efectivo, y si las actividades en el proceso están cumpliendo con el mínimo esfuerzo requerido y costo, entonces el proceso es eficiente. [QUE11]

2.10.2. COBIT

El estándar COBIT (Control Objectives for Information and related Technology) ofrece un conjunto de mejores prácticas para la gestión de los Sistemas de Información de las organizaciones. [MEA13]

El objetivo principal de COBIT consiste en proporcionar una guía a alto nivel sobre puntos en los que establecer controles internos con tal de:

- Asegurar el buen gobierno, protegiendo los intereses de los clientes, accionistas, empleados;
- Garantizar el cumplimiento normativo del sector al que pertenezca la organización;
- Mejorar la eficacia y eficiencia de los procesos y actividades de la organización;
- Garantizar la confidencialidad, integridad y disponibilidad de la información.

El estándar define el término control como: "Políticas, procedimientos, prácticas y estructuras organizacionales diseñadas para proveer aseguramiento razonable de que se lograrán los objetivos del negocio y se prevendrán, detectarán y corregirán los eventos no deseables" [MEA13]

Por tanto, la definición abarca desde aspectos organizativos (flujo para pedir autorización a determinada información, procedimiento para reportar incidencias, selección de proveedores) hasta aspectos más tecnológicos y automáticos (control de acceso, monitorización de los sistemas mediante herramientas automatizadas).

Por otra parte, todo control tiene por naturaleza un objetivo. Es decir, un objetivo de control es un propósito o resultado deseable como por ejemplo: garantizar la continuidad de las operaciones ante situaciones de contingencias.

En consecuencia, para cada objetivo de control de nuestra organización puede implementar uno o varios controles (ejecución de copias de seguridad periódicas, traslado de copias de seguridad a otras instalaciones) que nos garanticen la obtención del resultado deseable (continuidad de las operaciones en caso de contingencias). [MEA13]

COBIT clasifica los procesos de negocio relacionados con las tecnologías de la información en cuatro dominios:

- Planificación y organización;

- Adquisición e implementación;
- Entrega y soporte;
- Supervisión y evaluación.

En la planificación y organización, la dirección de la organización debe implicarse en la definición de la estrategia a seguir en el ámbito de los sistemas de información, de forma quesea posible proporcionar los servicios que requieran las diferentes áreas de negocio. Para ello, COBIT presenta 10 procesos:

- P01: Definición de un plan estratégico: gestión del valor, alineación con las necesidades del negocio, planes estratégicos y tácticos;
- P02: Definición de la arquitectura de información: modelo de arquitectura, diccionario de datos, clasificación de la información, gestión de la integridad;
- P03: Determinar las directrices tecnológicas: análisis de tecnologías emergentes, monitorizar tendencias y regulaciones;
- P04: Definición de procesos, organización y relaciones: análisis de los procesos, comités, estructura organizativa, responsabilidades, propietarios de la información, supervisión, segregación de funciones, políticas de contratación;
- P05: Gestión de la inversión en tecnología: gestión financiera, priorización de proyectos, presupuestos, gestión de los costos y beneficios;
- P06: Gestión de la comunicación: políticas y concienciación de usuarios;
- P07: Gestión de los recursos humanos de las tecnologías de la información: contratación, competencias del personal, roles, planes de formación, evaluación del desempeño de los empleados;
- P08: Gestión de la calidad: mejora continua, orientación al cliente, sistemas de medición y monitorización de la calidad, estándares de desarrollo y adquisición;
- P09: Validación y gestión del riesgo de las tecnologías de la información;
- P10: Gestión de proyectos: planificación, definición y asignación de recursos.

2.10.2.1. Adquisición e implementación

Con el objeto de garantizar que las adquisiciones de aplicaciones comerciales, el desarrollo de herramientas a medida y su posterior mantenimiento se encuentren

alineado con las necesidades del negocio, el estándar COBIT define los siguientes 7 procesos:

AI1: Identificación de soluciones: análisis funcional y técnico, análisis del riesgo, estudio de la viabilidad.

AI2: Adquisición y mantenimiento de aplicaciones: Diseño, controles sobre la seguridad, desarrollo, configuración, verificación de la calidad, mantenimiento.

Objetivo de control: Adquisición y mantenimiento de aplicaciones software.

Requisito de negocio: Suministrar funciones automáticas que soporten de forma efectiva los procesos de negocio. [MEA13]

Se debe comprobar si existe:

- La definición de estados específicos de los requisitos funcionales y operativos;
- Una implementación estructurada con dictámenes claros;
- Las actividades de desarrollo y pruebas están separadas.

Indicadores:

- Cantidad de aplicaciones entregadas puntualmente de acuerdo al plan;
- Cantidad de pedidos de cambios relacionados con defecto del sistema;
- Tiempo en que tardan en analizar y resolver problemas.

AI3: Adquisición y mantenimiento de la infraestructura tecnológica: Plan de infraestructuras, controles de protección y disponibilidad, mantenimiento.

AI4: Facilidad de uso: Formación a gerencia, usuarios, operadores y personal de soporte.

AI5: Obtención de recursos tecnológicos: control y asignación los recursos disponibles, gestión de contratos con proveedores, procedimientos de selección de proveedores.

AI6: Gestión de cambios: Procedimientos de solicitud o autorización de cambios, verificación del impacto y priorización, cambios de emergencia, seguimiento de los cambios, actualización de documentos.

AI7: Instalación y acreditación de soluciones y cambios: Formación, pruebas técnicas y de usuario, conversiones de datos, test de aceptación por el cliente, traspaso a producción. [MEA13]

2.10.2.2. Entrega y soporte

La entrega y soporte de servicios se encuentran constituidos por diversos procesos orientados a asegurar la eficacia y eficiencia de los sistemas de información.

DS1: Definición y gestión de los niveles de servicio: SLA con usuarios/clientes.

DS2: Gestión de servicios de terceros: gestión de las relaciones con proveedores, valoración del riesgo (non-disclousure agreements NDA), monitorización del servicio.

Objetivo de control: Gestionar los servicios prestados por las empresas que trabajan con las tecnologías de información y encargadas desarrollo de aplicaciones y de los usuarios dedicadas a esa área. [MEA13]

Requisito de negocio: Asegurar que las reglas y las responsabilidades de terceras partes están definidas de forma clara, adheridas y continuar satisfaciendo los requisitos.

Se tomará en consideración:

- Monitorización de contratos existentes;
- Acuerdos de servicio con terceras partes;
- Requisitos legales y regulados.

Indicadores:

- Cantidad de contratos de servicio actualizados.

DS3: Gestión del rendimiento y la capacidad: planes de capacidad, monitorización del rendimiento, disponibilidad de recursos.

DS4: Asegurar la continuidad del servicio: plan de continuidad, recursos críticos, recuperación de servicios, copias de seguridad.

DS5: Garantizar la seguridad de los sistemas: gestión de identidades, gestión de usuarios, monitorización y tests de seguridad, protecciones de seguridad, prevención y corrección de software malicioso, seguridad de la red, intercambio de datos sensibles.

DS6: Identificar y asignar costos

DS7: Formación a usuarios: identificar necesidades, planes de formación.

DS8: Gestión de incidentes y Help Desk: registro y escalado de incidencias, análisis de tendencias.

DS9: Gestión de configuraciones: definición de configuraciones base, análisis de integridad de configuraciones.

DS10: Gestión de problemas: identificación y clasificación, seguimiento, integración con la gestión de incidentes y configuraciones.

DS11: Gestión de los datos: acuerdos para la retención y almacenaje de los datos, copias de seguridad, pruebas de recuperación.

DS12: Gestión del entorno físico: acceso físico, medidas de seguridad, medidas de protección medioambientales.

DS13: Gestión de las operaciones: planificación de tareas, mantenimiento preventivo.

2.10.2.3. Supervisión y evaluación

Se tiene lo siguiente:

- Garantizar la alineación con la estratégica del negocio;
- Verificar las desviaciones en base a los acuerdos del nivel de servicio;
- Validar el cumplimiento regulatorio;
- Esta supervisión implica paralelamente la verificación de los controles por parte de auditores (internos o externos), ofreciendo una visión objetiva de la situación y con independencia del responsable del proceso.

ME1: Monitorización y evaluación del rendimiento

ME2: Monitorización y evaluación del control interno

ME3: Asegurar el cumplimiento con requerimientos externos

ME4: Buen gobierno. [MEA13]

2.11. Modelo 4+1 vistas

Kruchten propone el modelado de arquitecturas utilizando cuatro diferentes vistas y una vista de casos de uso para ilustrar y validar las otras vistas. Cada vista aborda un enfoque específico de la arquitectura para un conjunto particular de actores. En 4+1 View Model of Architecture en figura 2-7, Kruchten define las siguientes vistas: [MOR14]

- La vista lógica, representa los requerimientos funcionales del sistema, usando abstracciones elaboradas desde el dominio del problema. Esta vista es utilizada por el usuario final para asegurarse que todos los requerimientos funcionales han sido considerados en la implementación del sistema.

- La vista de procesos, describe los mecanismos de concurrencia y sincronización usados en el sistema. Esta vista es utilizada por los integradores del sistema para el análisis del rendimiento y escalabilidad del sistema.
- La vista de desarrollo, muestra el diseño del código en el ambiente de desarrollo. Esta vista es utilizada por los líderes de proyecto y programadores, y tiene como fin ayudar en la planeación y la evaluación del progreso del proyecto.
- La vista física, describe el mapeo del software en el hardware y refleja los aspectos de la distribución. Los Ingenieros en Sistemas desarrollan esta vista para determinar la topología y los requerimientos de comunicación entre los distintos componentes.
- Los escenarios son el termino +1 dentro del 4+1. Los escenarios ilustran las distintas decisiones que son tomadas a lo largo de las cuatro vistas. En los escenarios se capturan las características generales del sistema, como se observa en la figura 2-7. [MOR14]

Figura 2-7: Modelo 4+1 vistas de la arquitectura

Fuente: [MOR14]

Cada una de las vistas describe de forma general al sistema desde una perspectiva particular. Cada una de las vistas tiene una notación particular de representación. En nuestro caso usaremos el Lenguaje de Modelado Unificado para describir las vistas de forma ejemplificada.

En la Tabla 2-5 se muestra el mapeo de las vistas, propuestas por Kruchten, de la Arquitectura a diagramas UML. [MOR14]

4+1 Vistas	Arquitectura	Diagramas UML
Vista lógica	Lógica	Clases, de estados y colaboración
Vista de procesos	Procesos	Actividad, estados y de secuencia
Vista de desarrollo	Implementación y datos	Componentes
Vista física	Despliegue	Despliegue
Escenarios	Requerimientos	Casos de uso

Tabla 2-5: Mapeo de vistas a diagramas UML

2.12. Lenguaje de modelado unificado (UML)

Uno de los principales problemas en el desarrollo de software es la especificación de la arquitectura. Sin embargo no se ha resuelto del todo el problema debido a que no existe un estándar para este tipo de leguajes. Por lo cual el problema que ahora surge es cómo representar la arquitectura de un sistema de software, de tal manera que sea entendible por la mayoría de las personas que están involucradas en el desarrollo. [DEVA14]

Para hacer frente a este problema se propone el uso del lenguaje de modelado unificado (UML) para la representación de las arquitecturas. UML es un lenguaje estándar que nos permite modelar los componentes de un sistema de forma que sea entendible para todos los integrantes del equipo de desarrollo. [DEVA14]

El trabajo de Philippe Kruchten sobre el modelo de 4+1 vistas (Ver acápite 2.11.) (the 4+1 model view) para describir la arquitectura de software nos ayuda a analizar de una manera más comprensiva. Este enfoque utiliza diferentes vistas para separar los conceptos de cada Stakeholder. El enfoque de 4+1 vistas ha sido ampliamente usado por la comunidad de la industria de software para representar el diseño de la arquitectura de software de las aplicaciones.

En la figura 2-8 se muestra los diagramas de UML que corresponden a cada vista del modelo de 4+1 vistas.

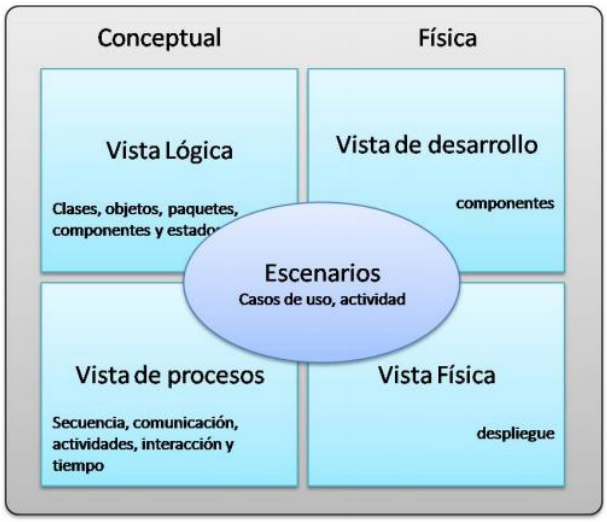

Figura 2-8: Modelo 4+1 vistas con UML

Fuente: [DEVA14]

UML es un lenguaje de modelado que utiliza conceptos orientados a objetos y tiene una sintaxis y semántica bien definidas lo que nos permite utilizarlo en todas las etapas de desarrollo y no solo en el proceso de diseño arquitectónico. De acuerdo con la especificación de UML 2 se encuentra divido en 13 tipos de diagramas básicos dentro de dos categorías clave:

- Diagramas estructurales: estos diagramas son utilizados para definir la arquitectura estática. Estos comprenden la construcción estática como las clases, objetos y componentes, y la relación entre estos elementos. Existen seis diagramas estructurales: Diagrama de paquetes, diagrama de clases, diagrama de objetos, diagrama de componentes, diagrama de despliegue y diagrama de estructura compuesta.
- Diagramas de comportamiento: estos diagramas son usados para representar la arquitectura dinámica. Estos comprenden la construcción dinámica como actividades, estados, líneas de tiempo, y los mensajes que ocurren entre diferentes objetos. Estos diagramas son usados para representar la interacción entre varios elementos del modelo y estados instantáneos sobre un periodo de tiempo. Existen siete diagramas de comportamiento: Diagramas de casos de uso, diagramas de actividades, diagramas de estados, diagramas de comunicación, diagramas de secuencia, diagramas de tiempo y diagramas de interacción.

La organización fundamental de un sistema de software puede ser representada por:

- Elementos: estructurales y sus interfaces que comprenden o forman un sistema.
- Comportamiento: representado por la colaboración entre los elementos estructurales.
- Composición: de elementos estructurales y de comportamiento dentro de los grandes sistemas.

Tales composiciones son guiadas por las capacidades deseadas (requerimientos no funcionales) como la usabilidad, resistencia, rendimiento, reusabilidad, limitaciones económicas y tecnológicas etc. Además existen temas transversales (como la seguridad y el manejo de transacciones) que aplican para todos los elementos funcionales.

Cada uno de los elementos y relaciones tiene una representación gráfica que, a su vez, se puede complementar con la especificación del elemento o relación; la especificación no es visible del todo ya que corresponde a los datos o propiedades adicionales que complementan la semántica del elemento o relación.

Por lo tanto necesitamos múltiples puntos de vista para las distintas necesidades de los actores involucrados en el desarrollo de un sistema de software.

2.13. Modelo de seguridad

Un modelo de seguridad es la presentación formal de una política de seguridad. El modelo debe identificar el conjunto de reglas y prácticas que regulan cómo un sistema maneja, protege y distribuye información delicada. [LORE06]

Los modelos se clasifican en:

- Modelo abstracto: se ocupa de las entidades abstractas como sujetos y objetos.
- Modelo concreto: traduce las entidades abstractas en entidades de un sistema real como procesos y archivos.

Además, los modelos sirven a tres propósitos en la seguridad informática:

- Proveer un sistema que ayude a comprender los diferentes conceptos. Los modelos diseñados para este propósito usan diagramas, analogías u otros aspectos que puedan ayudar a un mejor entendimiento del objeto de estudio.
- Proveer una representación de una política general de seguridad formal clara.
- Expresar la política exigida por un sistema de cómputo específico.

2.14. Metodologías de desarrollo

Una metodología es aquella guía que se sigue a fin realizar las acciones propias de una investigación. En términos más sencillos se trata de la guía que nos va indicando qué hacer y cómo actuar cuando se quiere obtener algún tipo de investigación. Es posible definir una metodología como aquel enfoque que permite observar un problema de una forma total, sistemática, disciplinada y con cierta disciplina.

Al intentar comprender la definición que se hace de lo que es una metodología, resulta de suma importancia tener en cuenta que una metodología no es lo mismo que la técnica de investigación. Las técnicas son parte de una metodología, y se define como aquellos procedimientos que se utilizan para llevar a cabo la metodología, por lo tanto, como es posible intuir, es uno de los muchos elementos que incluye.

2.14.1. RUP (Proceso Unificado de Racional)

Es un proceso de ingeniería de software que suministra un enfoque para asignar tareas y responsabilidades dentro de una organización de desarrollo. Su objetivo es asegurar la producción de software de alta y de mayor calidad para satisfacer las necesidades de los usuarios que tienen un cumplimiento al final dentro de un límite de tiempo y presupuesto previsible. Es una metodología de desarrollo iterativo que es enfocada hacia " diagramas de los casos de uso, y manejo de los riesgos y el manejo de la arquitectura" como tal.

El RUP mejora la productividad del equipo ya que permite que cada miembro del grupo sin importar su responsabilidad específica pueda acceder a la misma base de datos incluyendo sus conocimientos. Esto hace que todos compartan el mismo lenguaje, la misma visión y el mismo proceso acerca de cómo desarrollar un software.

2.14.2. XP (Programación Extrema)

Es una metodología ágil centrada en potenciar las relaciones interpersonales como clave para el éxito en desarrollo de software, promoviendo el trabajo en equipo, preocupándose por el aprendizaje de los desarrolladores, y propiciando un buen clima de trabajo. XP se basa en realimentación continua entre el cliente y el equipo de desarrollo, comunicación fluida entre todos los participantes, simplicidad en las soluciones implementadas y coraje para enfrentar los cambios. XP se define como especialmente adecuada para proyectos con requisitos imprecisos y muy cambiantes.

2.14.3. MSF (Microsoft Solution Framework)

Es una metodología realizada por Microsoft que puede aplicarse a otros proyectos de tecnologías de información y no solo al desarrollo de software, es flexible, se basa en modelos y prácticas de uso, que controlan la planificación, el desarrollo y la gestión de proyectos tecnológicos. MSF se centra en los modelos de proceso y de equipo dejando en un segundo plano las elecciones tecnológica.

2.14.4. SCRUM

Es un proceso en el que se aplican de manera regular un conjunto de buenas prácticas para trabajar colaborativamente, en equipo, y obtener el mejor resultado posible de un proyecto. Estas prácticas se apoyan unas a otras y su selección tiene origen en un estudio de la manera de trabajar de equipos altamente productivos.

En Scrum se realizan entregas parciales y regulares del producto final, priorizadas por el beneficio que aportan al receptor del proyecto. Por ello, Scrum está especialmente indicado para proyectos en entornos complejos, donde se necesita obtener resultados pronto, donde los requisitos son cambiantes o poco definidos, donde la innovación, la competitividad, la flexibilidad y la productividad son fundamentales.

2.15. Tecnologías de desarrollo

Algunas de las tecnologias de desarrollo para aplicaciones web. Ver figura 2-9

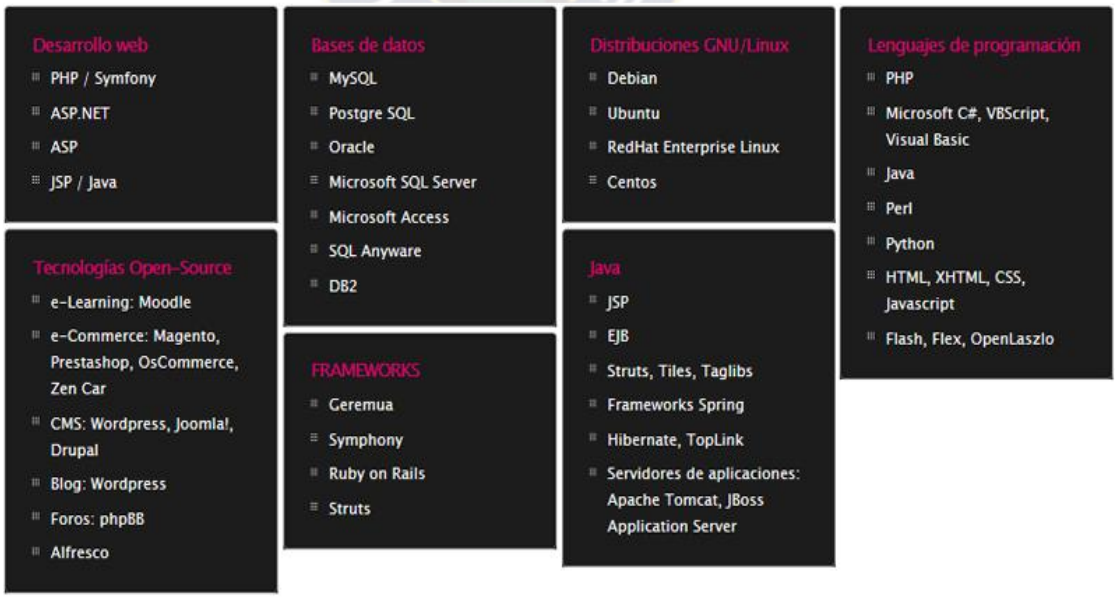

Figura 2-9: Tecnologías más usados para el desarrollo web

CAPÍTULO III

MARCO APLICATIVO

3.1. Introducción

En el presente capítulo se da a conocer el modelo de seguridad el cual describe el uso de técnicas y herramientas para minimizar las vulnerabilidades que se presentaron en el capítulo anterior (Ver acápite 2.5.5.). El modelo planteado se adapta a cualquier tecnología que nos permita desarrollar una aplicación web. Proporcionando información general acerca de las características y servicios de seguridad, principalmente en 4 capas: Capa de presentación visual, Capa lógica de negocio, Capa servicios de datos web y Capa de alojamiento web. Para un mejor entendimiento usamos el modelo 4+1 vistas descritas en el capítulo anterior (Ver acápite 2.11.), para describir cada una de las o capas.

Después de plantear el modelo de seguridad en las aplicaciones web, se presentan arquitecturas de las aplicaciones web para un mejor comprensión y entendimiento acerca de las aplicaciones web, de esta manera observar las posibles vulnerabilidades que se pueden presentar en las aplicaciones web en base al modelo de seguridad propuesto.

El modelo 4+1 vistas nos permite realizar un estudio más detallado de los componentes que contempla cada una de las capas del modelo. Además permite esquematizar con diagramas UML que nos facilita un mejor entendimiento de los componentes que interactúan en una aplicación web.

Después de observar las arquitecturas y vistas se describe las tareas y la interacción de las capas del modelo de seguridad en las aplicaciones web, sugiriendo el uso adecuado de las técnicas y herramientas que permiten cubrir las vulnerabilidades de las aplicaciones web.

Además se realiza un prototipo aplicando parcialmente el modelo de seguridad propuesto, para ver si el modelo es válido.

3.2. Modelo de seguridad en las aplicaciones web

Un modelo debe cumplir con algunas características (Ver acápite 2.13.). Esto implica al estudio de modelo abstracto y concreto, en lo abstracto tenemos a los sujetos y objetos, por otro lado en lo concreto se traduce las entidades abstractas con representaciones, es así que para una mejor comprensión y entendimiento hacemos el uso de diagramas UML para representar los procesos o el comportamiento del modelo.

La utilidad de un modelo es que ayuda la comprensión de los conceptos con diagramas para ello lo planteamos arquitecturas de las aplicaciones web, y representar las políticas de seguridad de forma esquemática.

En base a los conceptos, se realiza la representación esquemática del modelo de seguridad en las aplicaciones web con la figura 3-1, compuesto por cuatro capas que interactúan entre sí. Las capas están definidas según a la estructura arquitectónica de la aplicación web, y cada una de las capas tiene sus propios componentes.

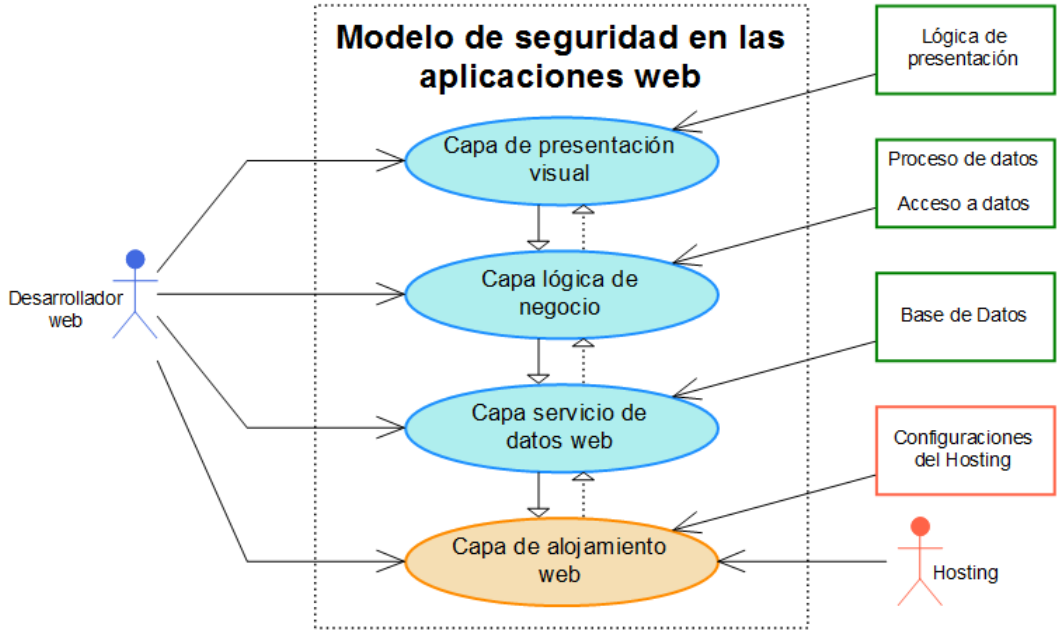

Figura 3-1: Modelo de seguridad en las aplicaciones web

El modelo es desglosado (Ver acápite 3.6.1., 3.6.2., 3.6.3. y 3.6.4) en forma textual, describiendo las funciones, las relaciones entre las capas y sus componentes. Además se muestra algunas de las técnicas y herramientas que se utilizan en cada una de las capas, sugiriendo las buenas practicas.

En la figura 3-2 se observa la vista lógica de las capas del modelo, mostrando los componentes a un nivel alto, lo cual permite ver su estructura general pero no a un nivel detallado.

En la capa de presentación visual esta la lógica de presentación y administrador de sesiones en caso de utilizarse. En la capa lógica de negocio está todo lo referente a la lógica, es decir procesado de datos o información ya que estos datos en su mayoría no deben ser legibles a la vista. En capa servicio de datos web están los gestores de bases de

datos permitiendo almacenar todos los datos y en la capa de alojamiento web esta todo lo referente a la configuración para el funcionamiento de las aplicaciones web.

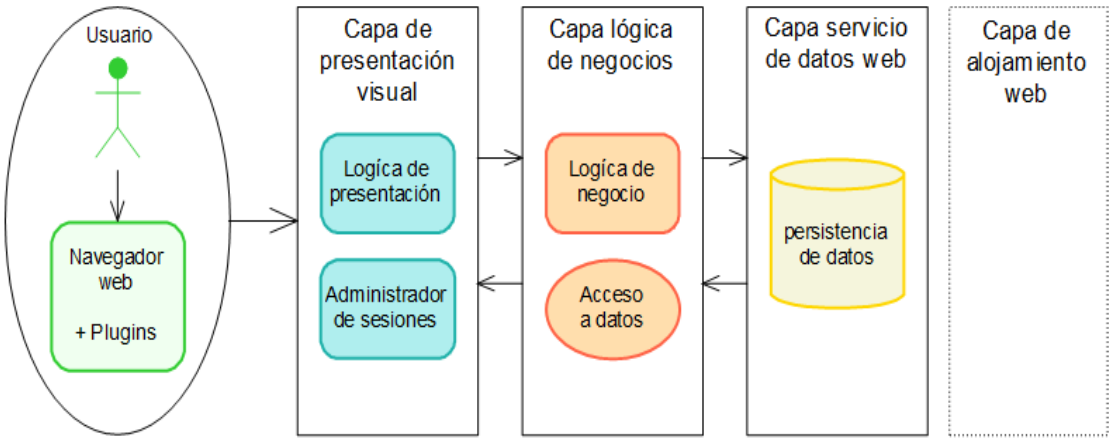

Figura 3-2: Vista lógica de las capas del modelo de seguridad

3.3. Arquitectura web

Las arquitecturas web (Ver acápite 2.4.2.) son de mucha importancia en el desarrollo de las aplicaciones web, por tanto es necesario documentar las arquitecturas que reflejan a las aplicaciones web, es así que se realizan las propuestas de las arquitecturas expresados en las figuras 3-3, 3-4, 3-5.

Un arquitectura abstrae el comportamiento de un conjunto de componentes y las relaciones que existen entre ellos, lo cual permite tener una configuración de componentes que satisfaga ciertas necesidades.

La utilización de las arquitecturas en el desarrollo de las aplicaciones web lleva a grandes ventajas, porque permite un mejor entendimiento y comprensión de los componentes que pueda tener. Existen una variedad de arquitecturas y cada uno de ellos está pensado para un determinado dominio.

Las arquitecturas que se presentan están enfocadas a las aplicaciones web, por lo tanto solo resuelven solamente este tipo de aplicaciones. En todo caso referenciamos a dos tipos de arquitecturas, los cuales son arquitecturas basadas en capas y en modelo vista controlador, los cuales describimos a continuación.

3.3.1. Arquitectura en capas

La arquitectura en capas descompone una aplicación en varias capas. El objetivo principal es separar los componentes de acuerdo a las capas del modelo de seguridad, es

decir, en las aplicaciones web hay componentes encargados de la presentación visual, otros encargados de la lógica de negocio y de servicio de datos web.

Por conveniencia y para representación esquemática hacemos la diferencia entre el término “nivel” y “capa”. El termino nivel corresponde a la representación física en que una aplicación se organiza, y el termino capa se utiliza para referenciar a las distintas partes en las que una aplicación web se divide desde el punto de vista lógico.

En la figura 3-3 se muestra una arquitectura que tiene dos niveles, la primera nivel de aplicación y el segundo nivel de datos, donde cada nivel puede tener una o varias capas. El nivel de aplicación puede estar constituido por la capa de presentación visual y por la capa de lógica de negocio y el nivel de datos puede contener sólo la capa de servicio de datos web. Además las capas interactúan entre sí por medio de una interfaz.

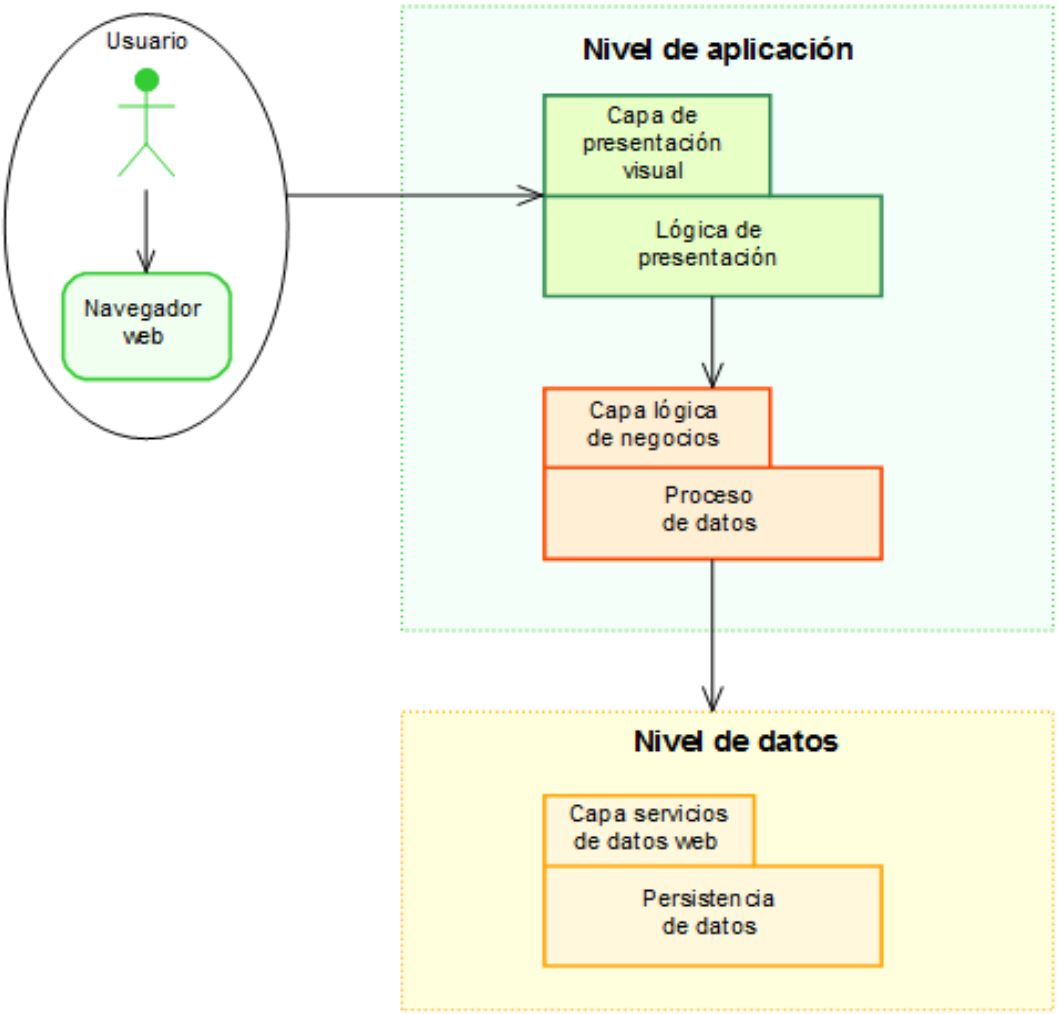

Figura 3-3: Arquitectura dividida en dos niveles y tres capas

Una arquitectura separada en niveles y capas permite modificar de forma independiente cada capa, según al nivel de complejidad que tenga la aplicación web, pudiendo incorporar más niveles o capas de acuerdo a las necesidades que se tenga. Por tanto se presenta el siguiente escenario, una aplicación que requiera interactuar con diferentes gestores de bases de datos. Por lo que se agrega otra capa en el nivel de aplicación, encargado del acceso a los datos como se observa en la figura 3-4 independientemente del tipo de gestor de la base de datos.

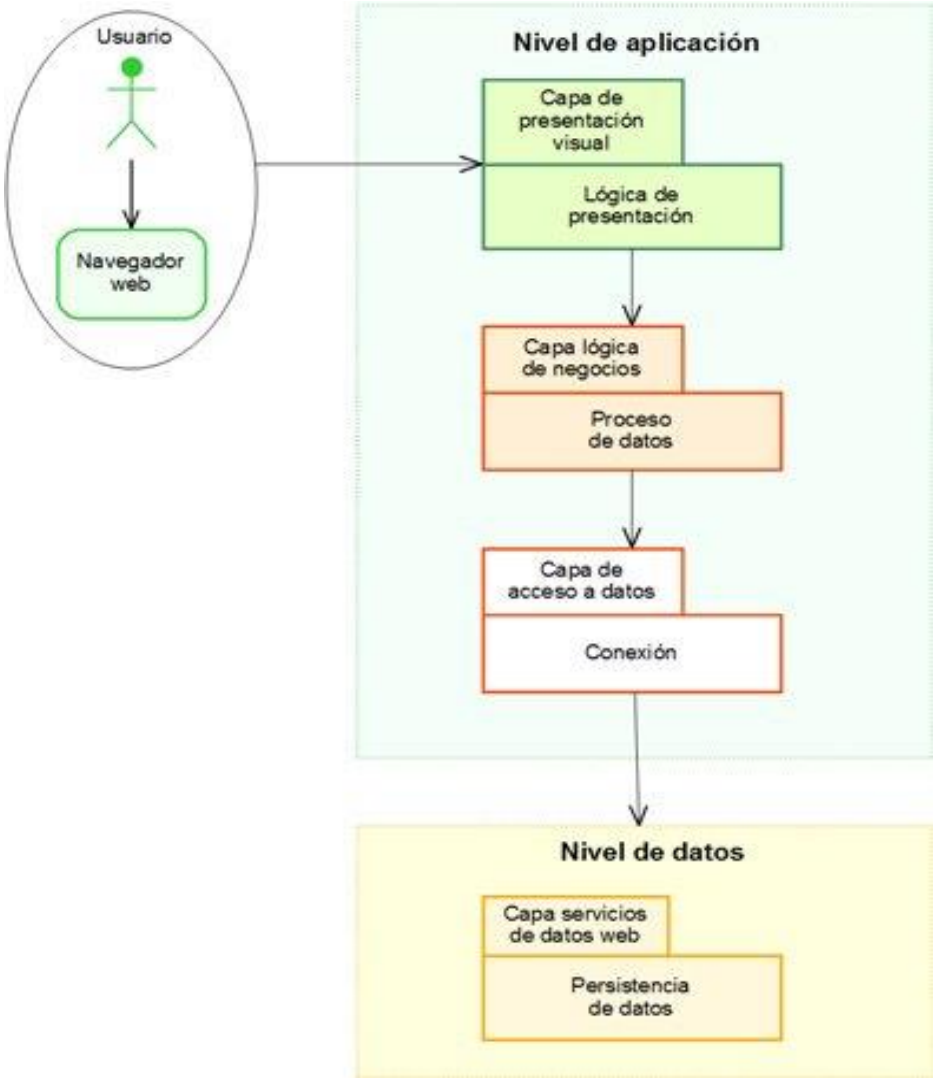

Figura 3-4: Arquitectura dividida en dos niveles y cuatro capas

En las arquitecturas del tipo cliente-servidor podemos apreciar dos niveles, cliente y servidor y cada nivel puede tener un número diferente de capas. Las aplicaciones web son aplicaciones basadas en la arquitectura cliente-servidor por naturaleza, por tanto

siempre serán aplicaciones con dos niveles como mínimo, aunque no está limitada a solo dos niveles, véase la figura 3-5.

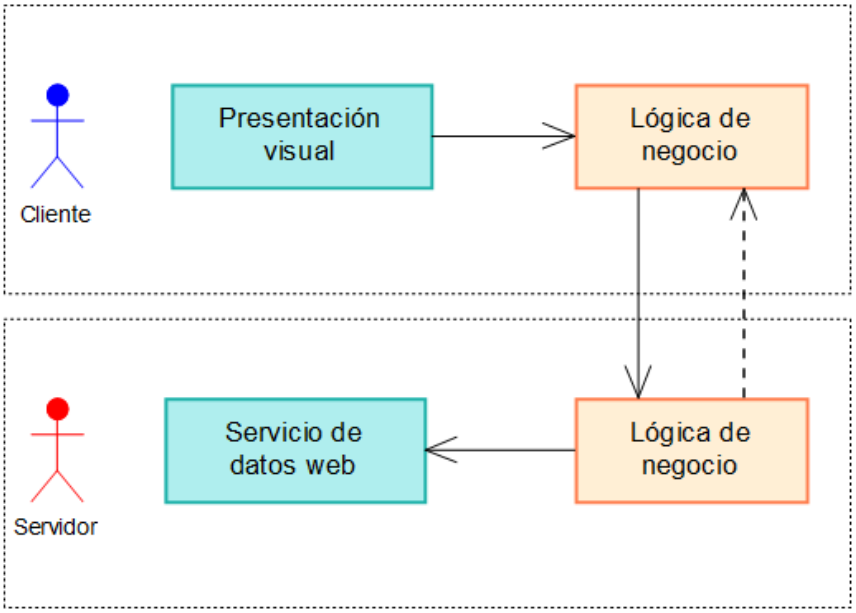

Figura 3-5: Modelo Cliente - Servidor

Como en la actualidad las aplicaciones web son utilizadas en diferentes ámbitos, es un indicio de la gran demanda de este tipo de aplicaciones, por tanto el nivel de complejidad también incrementa. Debido a lo importante que resulta satisfacer las demandas y las necesidades, es que se necesita adecuar la arquitectura para soportar las aplicaciones web complejas.

La complejidad incrementa según a las tareas establecidas en las diferentes ámbitos, estas tareas pueden ralentizar el funcionamiento óptimo de las aplicaciones. Además la mayoría de las aplicaciones necesita hacer un gran número de peticiones al servidor, en ocasiones se necesita mostrar información que no ha sido modificada y, sin embargo se envía una petición para reenviarla. Este problema causa que el tráfico en la red y en el servidor aumente de forma considerable, esto puede ocasionar que la aplicación sea vulnerable ante los ataques de tipo DOS o DOoS.

Para solucionar estos problemas es necesario incrementar otros niveles y capas para que el rendimiento sea de manera normal. Por ello en la figura 3-6 se muestra el nivel de presentación que se divide en dos capas, la capa de presentación contiene todos los elementos para presentar de una mejor forma la información y el control de presentación

que contiene lógica de negocio en su mínima expresión para determinar los elementos utilizados en la capa de presentación. En el nivel de aplicación se agrega la capa de entidades con los componentes requeridos por las de más capas.

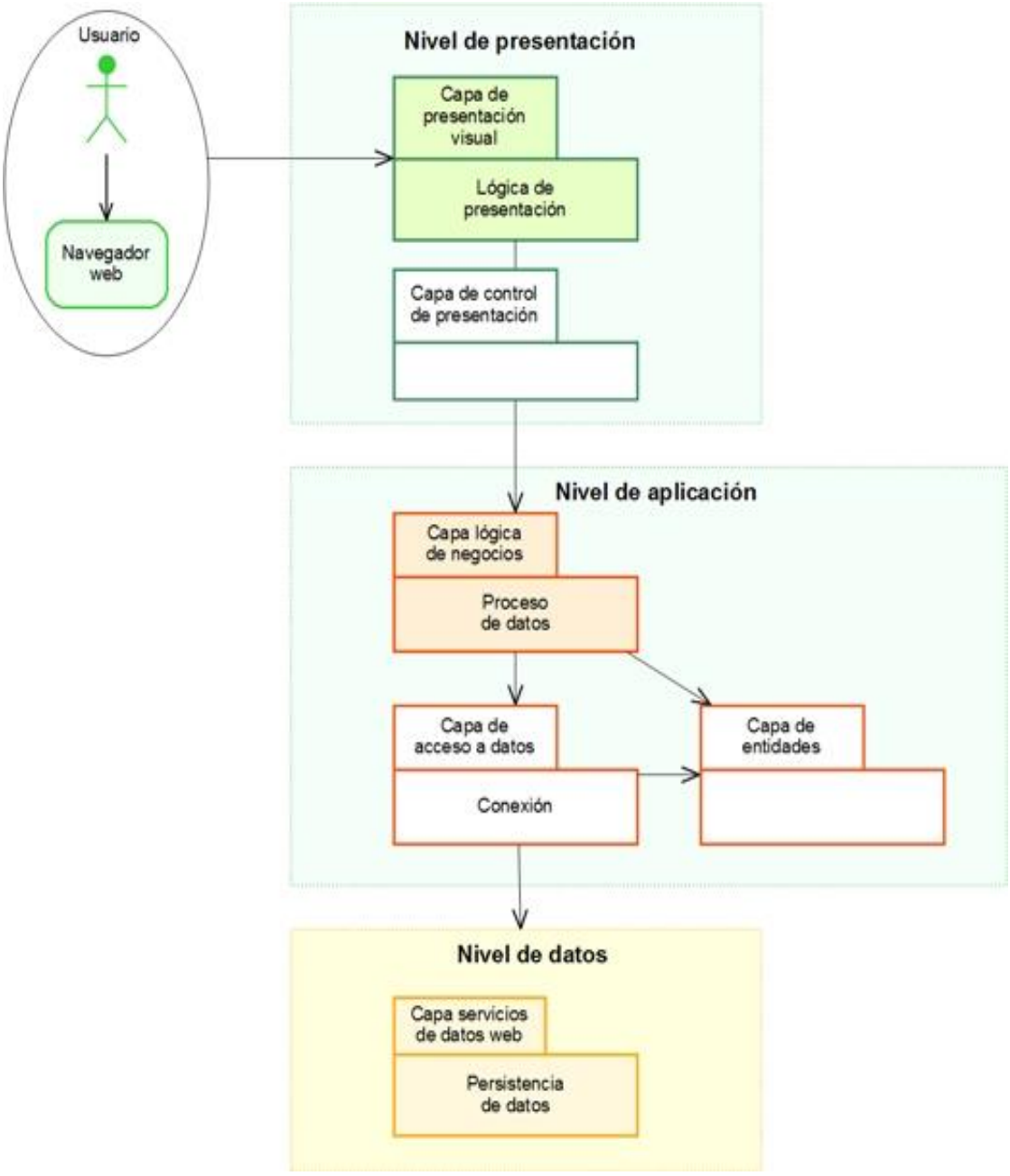

Figura 3-6: Arquitectura dividida en tres niveles y seis capas

El nivel de presentación no necesita de muchos recursos y es independiente de otros niveles, el nivel de aplicación es el encargado de proporcionar lo necesario para que el nivel de presentación funcione correctamente.

3.3.2. Arquitectura modelo vista controlador

Este tipo de arquitecturas son los que más se utilizan para el desarrollo de aplicaciones web, el modelo vista controlador (MVC) son utilizados para construir las interfaces de usuario, la importancia de esta arquitectura es la separación de los componentes de los datos de la aplicación y los componentes de la interfaz de usuario. La separación de las capas permite tener, a nivel de desarrollo, un código más claro, flexible, reusable y seguro.

Durante el desarrollo de una arquitectura en capas se puede observar que cada capa tiende a encapsular elementos que comparten ciertas características, creando capas que contienen componentes para una determinada función.

La arquitectura del MVC descompone la aplicación en capas permitiendo tener una separación entre la lógica de negocio de la aplicación, la representación visual y servicio de datos web como se observa en la figura 3-7. La arquitectura MVC identifica tres capas que son importantes para cualquier aplicación:

- Modelo, encapsula los datos de la aplicación y la lógica para interactuar con ellos de forma conjunta. Específicamente la capa lógica de negocio del modelo de seguridad propuesto.
- Vista, es la encargada de manejar la interacción con el usuario y la representación del modelo. Hace referencia a la capa de presentación visual del modelo de seguridad.
- Controlador, es el intermediario entre el modelo y la vista ante las peticiones generadas en la vista, de la capa de presentación visual. El controlador se encarga de seleccionar el modelo solicitado por el usuario y la vista hace la representación en el navegador para la visualización.

El esquema de separación que propone el modelo vista controlador por medio de las distintas capas se puede observar a nivel de diseño, identificar los componentes de cada capa y la comunicación que existe con los demás componentes.

En la figura 3-7 se observa la arquitectura de modelo vista controlador, mostrando una integración entre las capas, sin embargo no se detalla los componentes de las capas, tampoco se detalla la interacción entre las capas que componen.

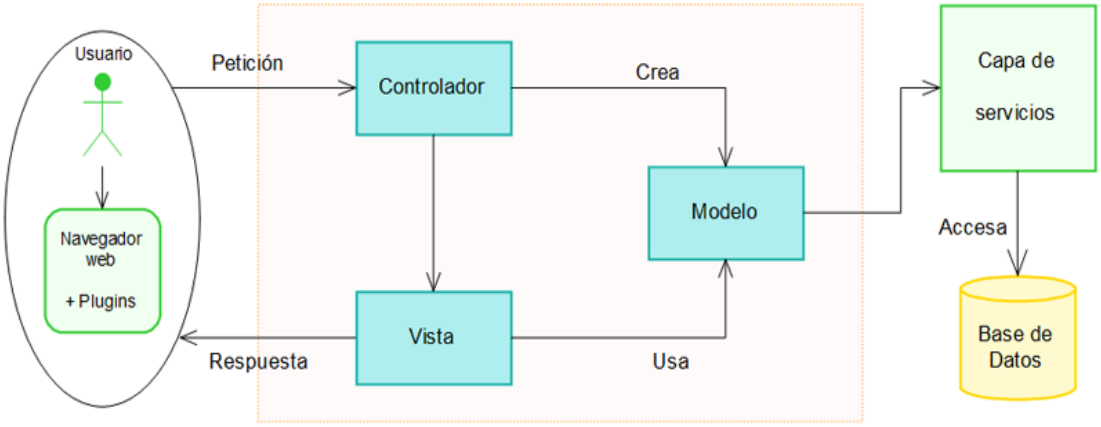

Figura 3-7: Arquitectura Modelo Vista Controlador

3.4. Arquitectura genérica para aplicaciones web

Se propone una arquitectura genérica que pueda utilizarse para modelar la mayoría de las aplicaciones web existentes. Esta arquitectura genérica contiene todos los componentes necesarios para el desarrollo de aplicaciones web, entonces debe contener componentes basados en web que permiten aprovechar al máximo la infraestructura de la web, obteniendo aplicaciones robustas.

La arquitectura ilustrada en la figura 3-8 para aplicaciones web estáticas o denominado en términos exactos páginas web, ya que una aplicación web es más dinámica.

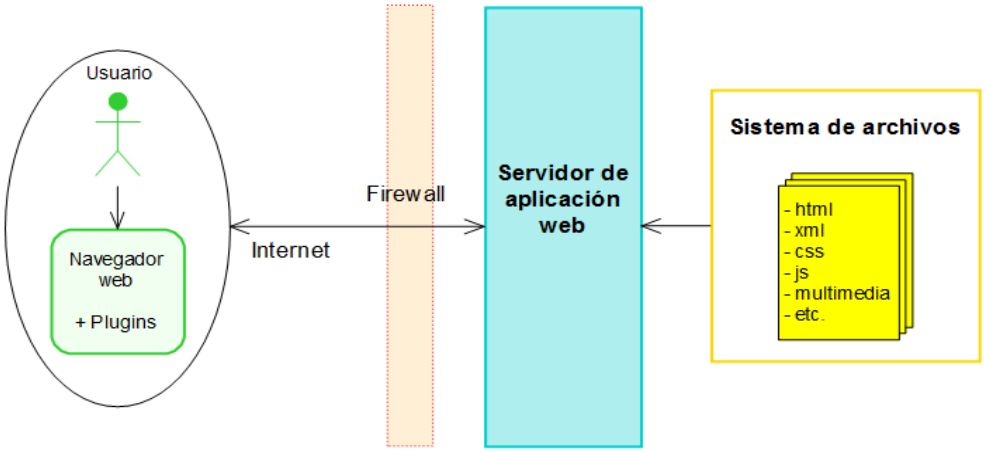

Figura 3-8: Arquitectura para aplicaciones web estáticas

La arquitectura expuesta en la Figura 3-8 satisface los requerimientos de aplicaciones web estáticas. Sin embargo, para que podamos modelar aplicaciones web dinámicas es necesario agregar componentes adicionales tales como un servidor de aplicaciones para

permitir la generación de contenidos dinámicos, tomando como base la arquitectura propuesta. Al agregar nuevos componente necesitamos considerar las interfaces que se necesitan incluir así como la relación de los componentes existentes.

Según a la necesidad se agrega un servidor de Bases de Datos y Sistemas externos, porque la mayoría de las aplicaciones web necesitan comunicarse con otro tipo de aplicaciones.

La Figura 3-9 muestra los componentes y las interfaces que se han agregado para complementar la arquitectura y poder soportar las aplicaciones web dinámicas.

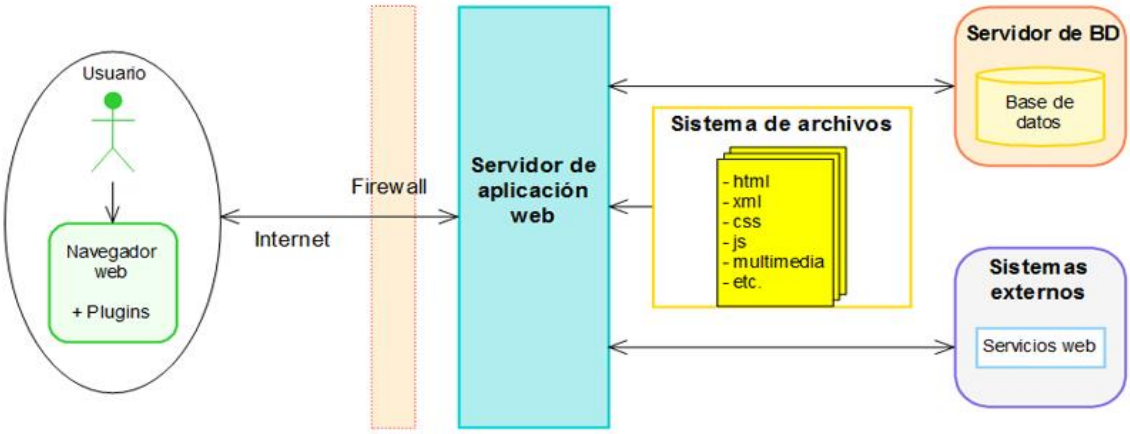

Figura 3-9: Arquitectura para aplicaciones web dinámicas

La arquitectura presentada en la figura 3-9 permite desarrollar aplicaciones web muchos más robustos en comparación con las arquitecturas presentadas anteriormente.

Una de las características que se considera importante es el manejo de contenidos multimedia en las aplicaciones web, con el manejo de interfaces más completos, se ha generado una enorme demanda de los recursos multimedia. Esta necesidad de las aplicaciones se ha convertido en un desafío, debido a que la demanda es el manejo de grandes cantidades de recursos multimedia sin presentar degradación en el rendimiento de la aplicación web.

En la figura 3-10 se observa a los nuevos componentes agregados y las relaciones que existen entre ellos, la relación del servidor multimedia va a ser directamente con el servidor de aplicación web. De la misma forma, el servidor de componentes tiene una relación directa con el servidor de aplicación web, debido a que el servidor web es el que solicita los recursos.

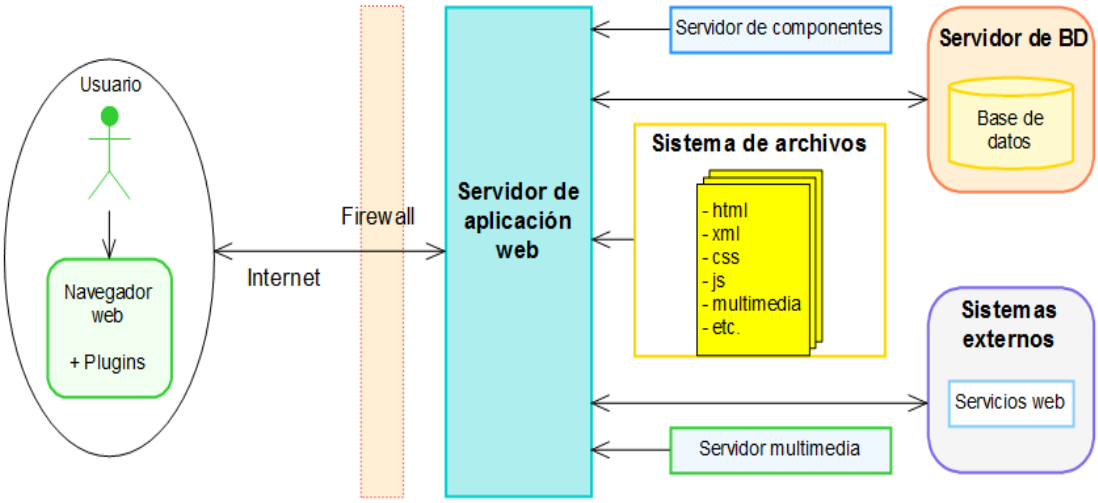

Figura 3-10: Arquitectura genérica para aplicaciones web

Después de la elaboración de las arquitecturas anteriores, se presenta una nueva arquitectura mucho más completa. La figura 3-11 muestra la arquitectura genérica con los componentes que existen dentro de cada capa, y la relación que existe entre estos. Como se observar se realiza cambios en el modelo cliente y servidor, considerando el soporte a las demandas de una aplicación web robusta.

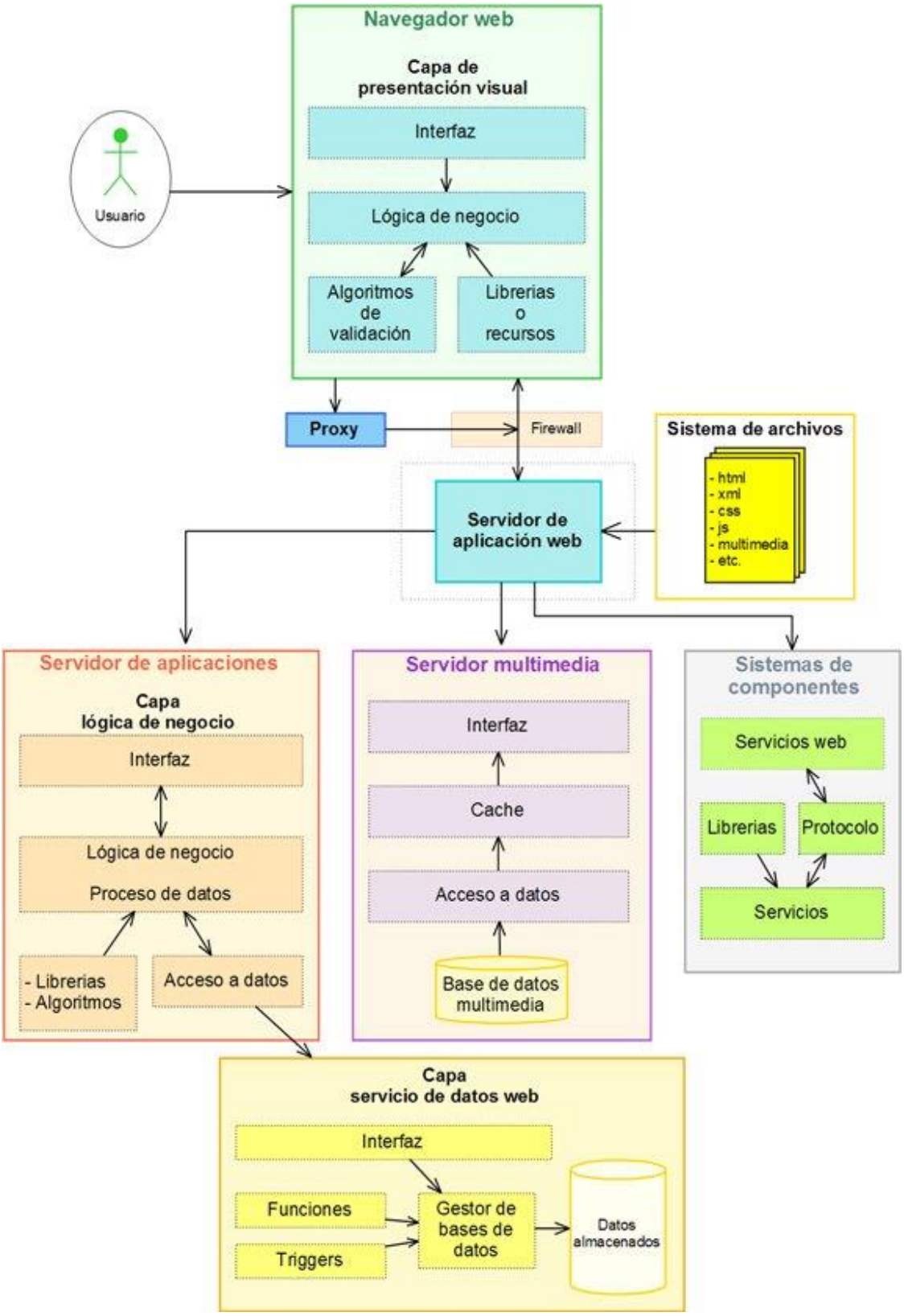

Figura 3-11: Arquitectura genérica para aplicaciones web completa

Como se observa según al modelo planteado tambien se expone las distintas arquitecturas dependiendo del requerimiento y las demasndas del cliente, incluso se muestra una arquitectura generica.

3.5. Vistas desde el punto de vista del modelo 4+1

Este modelo nos permite ver la aplicación web en diferentes vistas en el proceso de la implementación, para ello se debe realizar la vista lógica que muestra las capas de la aplicación web y los componentes que contiene. La vista de desarrollo detalla cada una de las capas y de los componentes. La vista de seguridad muestra la restricción de cada uno de los usuarios que utiliza la aplicación. La vista de procesos muestra interoperabilidad de la aplicación web con el usuario para realizar una tarea específica. Finalmente la vista física muestra a la aplicación desplegada en los componentes físicos.

3.5.1. Vista física

Dentro de esta vista se puede observar en donde están cada uno de los componentes. El objetivo principal de esta vista es brindar una guía al momento de desplegar aplicación web al usuario final como se observa en las siguientes figuras [3-12], [3-13], [3-14], [3-15] y [3-16].

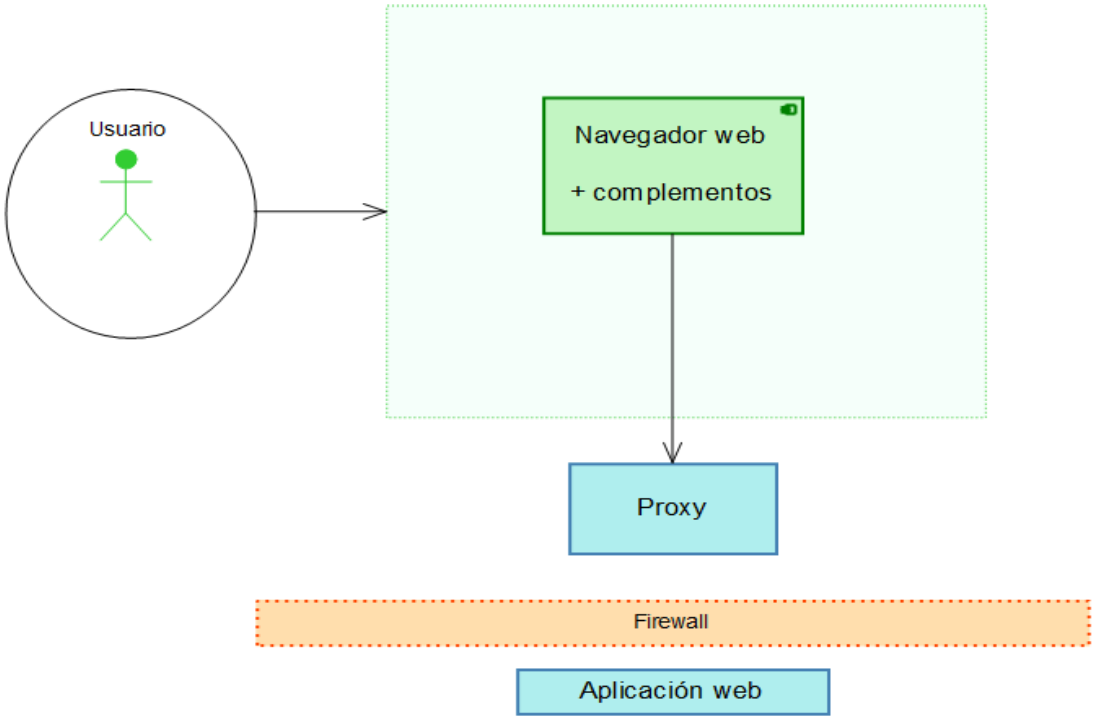

Figura 3-12: Vista del cliente navegador y la aplicación web

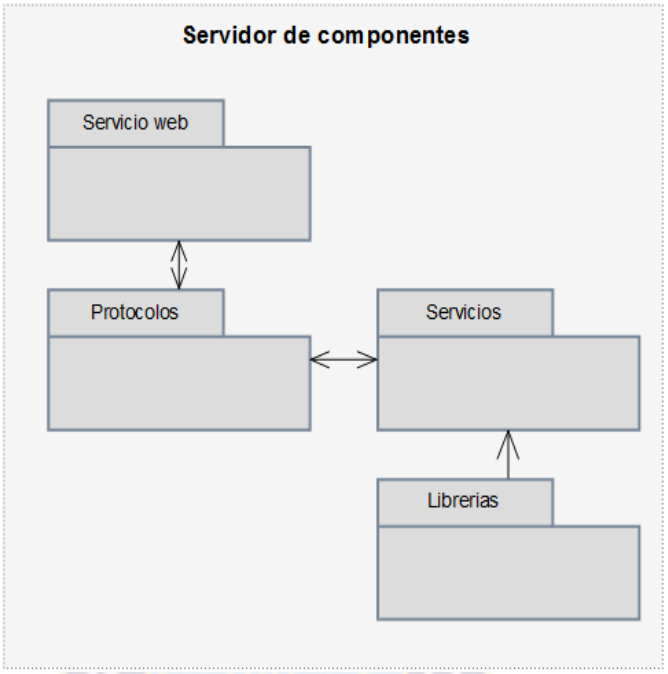

Figura 3-13: Vista física, servidor de componentes

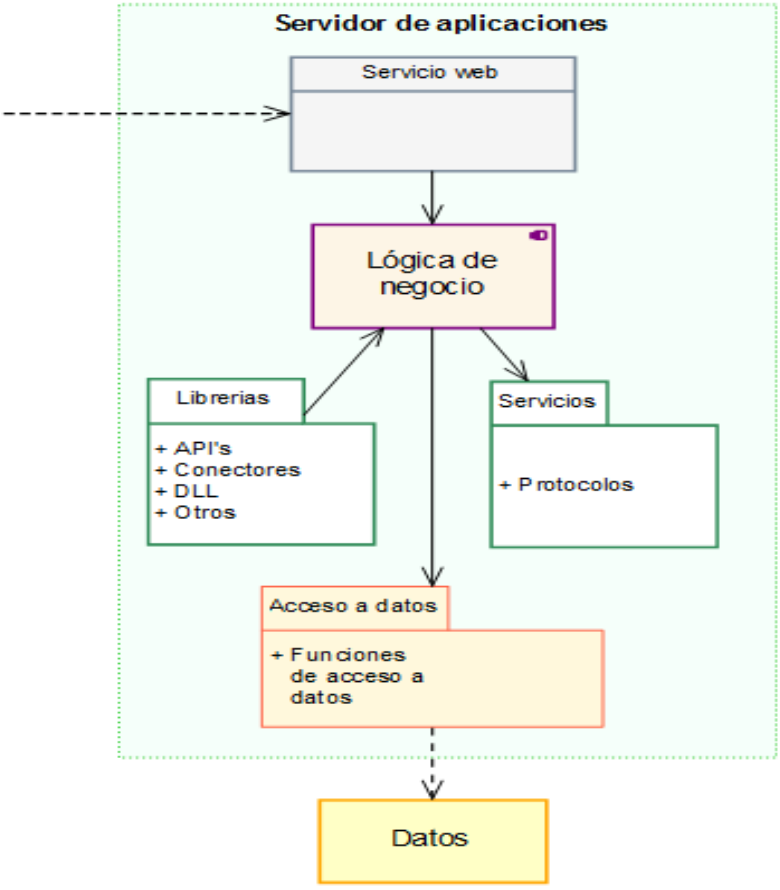

Figura 3-14: Vista física, servidor de aplicaciones

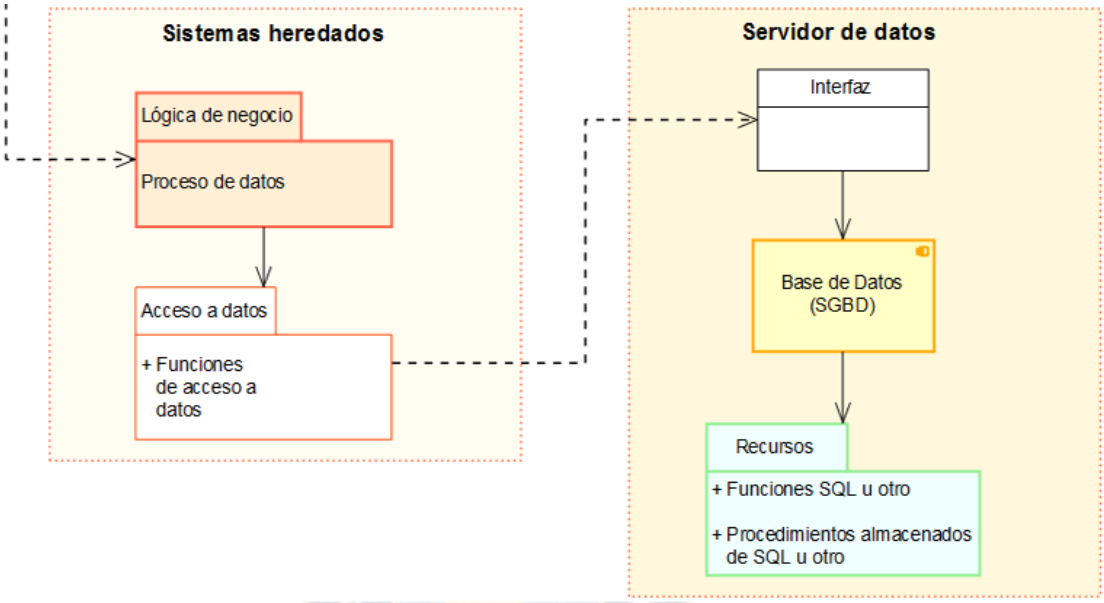

Figura 3-15: Vista física, servidor de datos

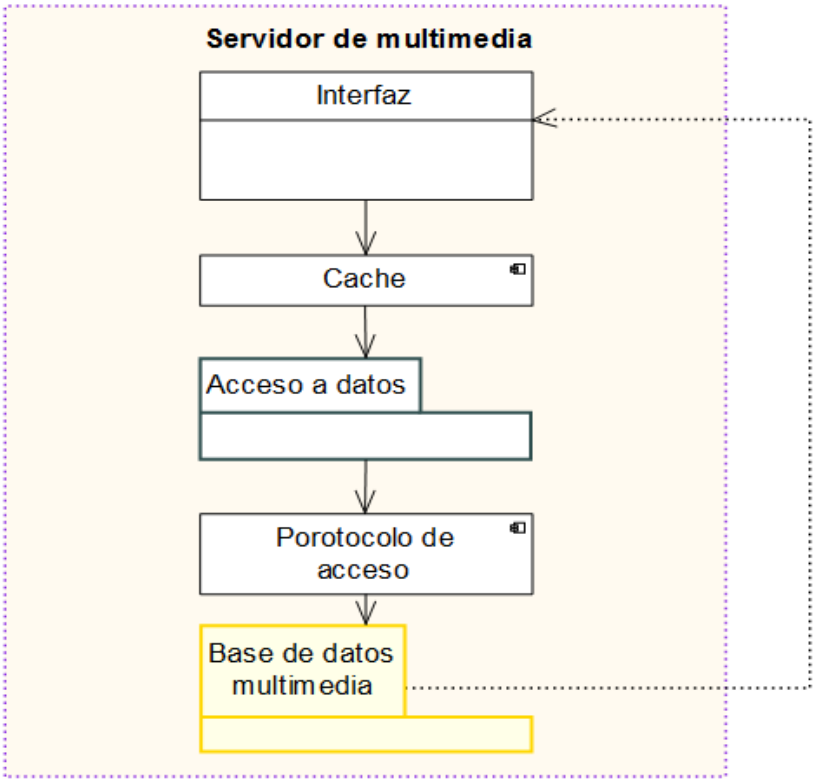

Figura 3-16: Vista física, servidor de multimedia

3.5.2. Vista lógica

La vista lógica permite definir la estructura de nuestra aplicación web. Inicialmente se define a un nivel alto, posteriormente se define a mayor detalle cada uno de los

componentes. Para ello el estudio a realizar considera una de las arquitecturas presentadas con anterioridad.

En la figura 3-17 muestra las dependencias entre las capas del modelo, el diagrama de paquetes nos permite identificar los componentes necesarios en cada capa.

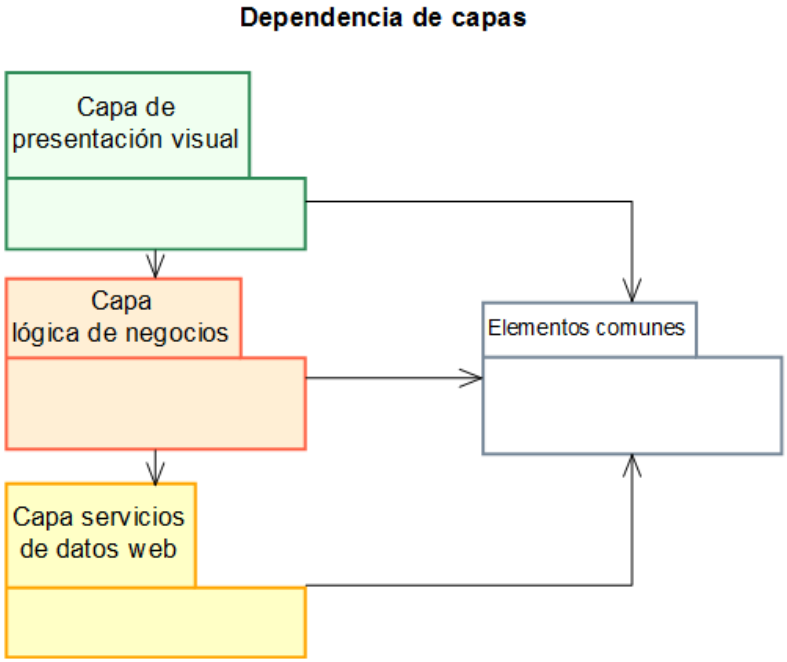

Figura 3-17: Dependencia entre las diferentes capas

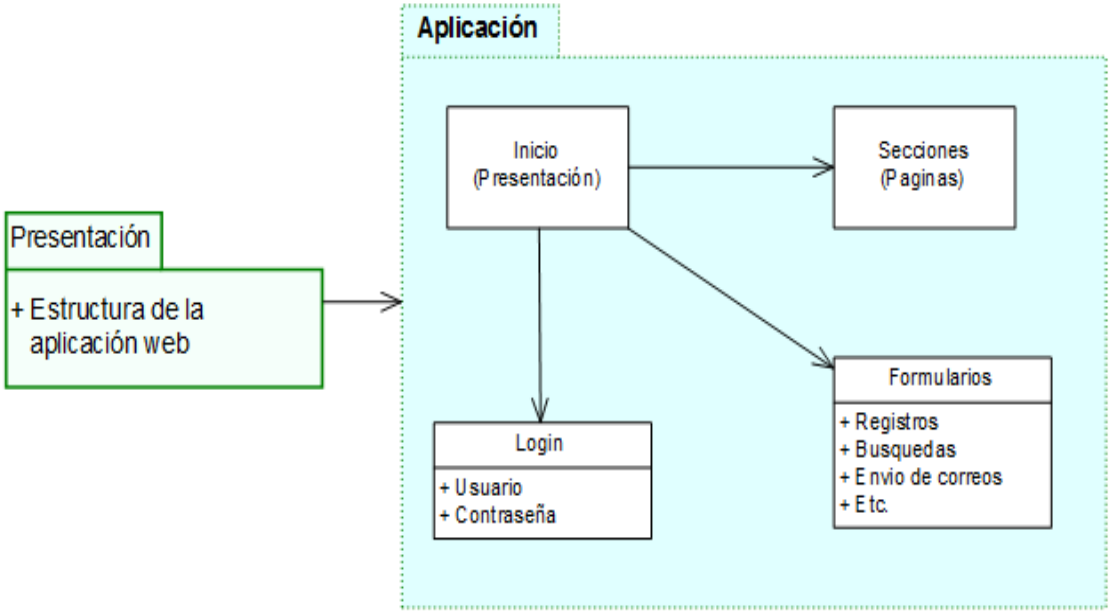

Figura 3-18: Capa de presentación de forma básica

En la figura 3-18 se muestra una ilustracion basica de una aplicación web que se visualiza al usuario final, con una estructura generica.

Pero esto dependera de los requerimientos para cada una de las aplicaciones web, puede ser de lo mas simple hasta mas complejas, pero basando en la arquitectura adoptada para su desarrollo.

3.5.3. Vista de proceso

En la Figura 3-19 se muestra la vista de procesos de la aplicación web. La vista de procesos tiene como objetivo principal modelar el proceso de negocio de la aplicación. Mostramos la interacción entre el usuario final y la aplicación web. El usuario envía la información o una petición, y la aplicación realiza un acuse dependiendo de qué tipo de información o petición haya realizada el usuario final.

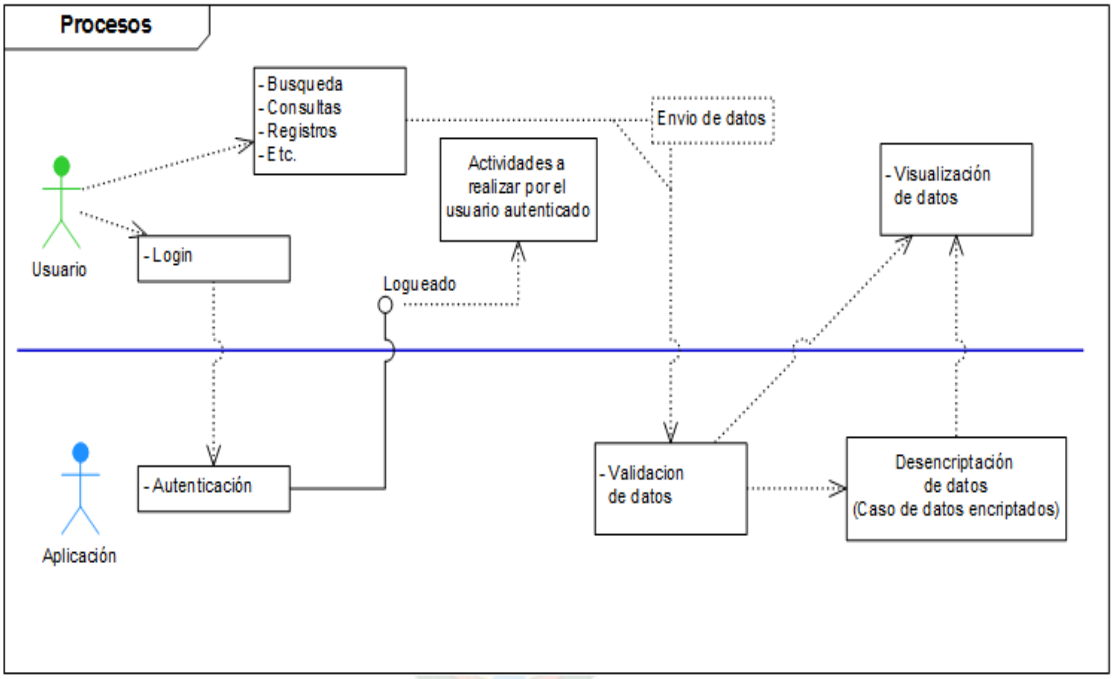

Figura 3-19: Vista de procesos con esquema básico (Ejemplo)

3.5.4. Vista de seguridad

Esta vista muestra los métodos, técnicas y componentes necesarias para brindar seguridad a la aplicación web. En la figura 3-20 se muestra el nivel de seguridad para cada una de los usuarios según a los roles que desempeña.

Los niveles de seguridad que se pueden manejar en las aplicaciones web son varias, mencionaremos los más comunes y más utilizados. El Login mediante el uso de un

nombre de usuario y una contraseña. La autenticación mediante certificados digitales y también el cifrado de los datos. Esto depende de la sensibilidad de la información de cada aplicación web.

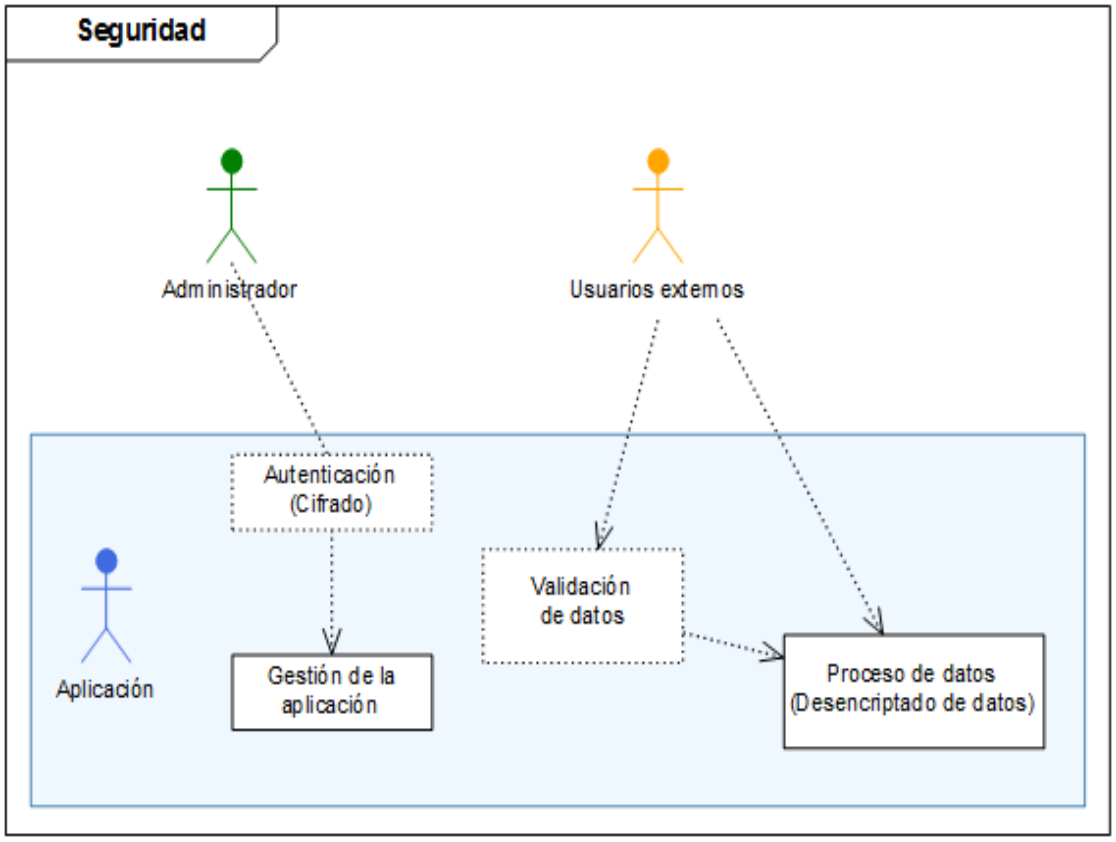

Figura 3-20: Vista de seguridad con esquema básica (Ejemplo)

3.5.5. Vista de desarrollo

Esta vista muestra todos los componentes de la aplicación y la relación que existen entre ellos. Esta vista permite tener un panorama global sobre los componentes que se van a desarrollar en ocasiones denominada también módulos de desarrollo.

Se puede realizar una variedad de esquematizaciones de los componentes por lo tanto no se puede mostrar un esquema de manera específica, esto se realiza según la arquitectura adoptada.

3.6. Especificaciones del modelo

Después de haber presentado las arquitecturas según a las capas y realizado las vistas para una visión general de la aplicación web, vemos el procedimiento adecuado de la utilización de las técnicas y herramientas en las capas del modelo de seguridad.

Antes de realizar el desglose de las técnicas y herramientas se lista algunos complementos de OWASP referentes a la seguridad web, que permite desarrollar las aplicaciones webs más seguras en sus diferentes ámbitos.

- OWASP AntiSamy
- OWASP CAL9000
- OWASP CLASP
- OWASP Encoding
- OWASP Enterprise Security API
- OWASP Insecure Web App
- OWASP LAPSE
- OWASP Pantera Web Assessment Studio
- OWASP SQLiX
- OWASP Validation

3.6.1. Capa de presentación visual

Es donde se despliega la interfaz de la aplicación web, y es la capa encargada de la recopilación de datos ingresados por el usuario que utiliza la aplicación web. Posteriormente se realiza un filtrado previo para comprobar que los datos a procesar son válidos y no erróneas.

Esta capa se encarga de pasar o enviar los datos filtrados a la capa lógica de negocio donde se realiza el procesado. A continuación se detallan algunas de las técnicas y herramientas aplicados en esta capa.

3.6.1.1. Validación de datos con JavaScript

La principal utilidad de JavaScript es la validación de formularios que permiten ingresar datos a los usuarios. Antes de enviar los datos de un formulario al servidor, se recomienda validar mediante JavaScript los datos ingresados por el usuario, para verificar la no existencia de errores, además se puede notificar de forma instantánea, sin necesidad de esperar la respuesta del servidor acortando el tiempo del procesamiento.

Notificar los errores de forma inmediata mediante JavaScript mejora la satisfacción del usuario que utiliza la aplicación web y ayuda a reducir la carga de procesamiento en el servidor.

Normalmente, la validación consiste en llamar a una función de validación que utiliza las expresiones regulares que comprueban si los datos ingresados cumplen con las restricciones impuestas por la aplicación.

3.6.1.2. Validación AntiSamy

Es una API para asegurarse que las entradas HTML/CSS del usuario estén en cumplimiento con las reglas de la aplicación. Es decir, ayuda a asegurarse que los clientes no provean código malicioso en el HTML. El termino código malicioso en términos de aplicaciones web es generalmente relacionado solo con JavaScript. CSS, hojas de estilo en cascada son consideradas maliciosas cuando invocan a JavaScript. Sin embargo, hay muchas situaciones donde HTML y CSS pueden ser usados de una forma maliciosa.

Por tanto, nos ayuda a reforzar los puntos débiles en un ataque de inyección de JavaScript y CSS basándose en el uso de expresiones regulares. Además es aplicable en diferentes lenguajes de programación.

3.6.1.3. Tecnología Ajax

Hay dos disponibilidades de Ajax, una es creando un objeto con JavaScript y la otra el de JQuery, con esto agilizamos él envió los datos bajo un formato, pero el empleo de esta tecnología puede ser de forma errónea. Por tanto se debe tomar algunas medidas restrictivas:

- Implantar controles de seguridad como pueden ser autenticación, autorización, e incluso registro de operaciones, siempre en el lado del servidor.
- No almacenar datos sensibles o confidenciales en el lado del cliente, este hecho haría que obtenerlos por un atacante fuera algo potencialmente sencillo.
- Utilización de métodos criptográficos para transmitir datos sensibles o confidenciales entre el cliente y el servidor.
- Se recomienda usar Ajax del Framework JQuery.
- Toda petición realizada con AJAX y que acceda a recursos protegidos deberán encontrarse autenticadas.
- Utilizar métodos POST en vez de métodos GET.

- La lógica de negocio debe tener el principio de mínima exposición y encontrarse siempre en el lado del servidor.
- Ajax solo debe ser usado para enviar datos al servidor, para la validación u otro tipo de proceso.
- Utilizar el método adecuado para renderizar los tipos de datos recibidos del servidor por ejemplo JSON.
- No utilizar un Servidor Proxy para llamar a otros sistemas externos como Servicios Web.

3.6.1.4. Utilización del algoritmo de cifrado

Son algoritmos que se encargan de cifrar y descifrar datos en la aplicación web, para la utilización es necesario asignar una clave secreta para el cifrado tanto como para descifrado. La clave secreta se debe conocer en la capa de presentación visual y en la capa lógica de negocio.

Para aplicar un algoritmo de cifrado determinado junto con una clave, a una determinada información que se quiere transmitir confidencialmente, es necesario definir qué tipo de criptografía a utilizar, ya sea simétrica y asimétrica.

Algunos de los algoritmos comunes son MD5, SHA, SHA1, existe otros algoritmos que se pueden emplear. Pero lo más recomendable es utilizar un algoritmo propio y personalizado.

3.6.1.5. Control de actividades

Es una técnica utilizada para el seguimiento de las acciones realizadas durante el uso de la aplicación web, consiste en generar una pila de control del lado del servidor para identificar al usuario autenticado.

Cada acción que el usuario realiza se valida del lado del servidor comparando la pila generado con anterioridad, si los datos son incorrectos se invalida la petición y la autenticación se anula.

3.6.2. Capa lógica de negocio

Es la capa encargada de todo tipo de lógica de negocio o procesado de datos, es donde se ejecuta los procesos en su mayoría referentes a los datos recibidos de la capa de presentación visual. Las peticiones del usuario son recibidos y se envían las

respectivas respuestas tras el proceso lógico. Es por lo que se denomina capa de negocio, porque es donde se establecen las reglas que deben cumplirse.

Inicialmente la capa lógica de negocio se comunica con la capa de presentación visual, para recibir las peticiones del usuario y presentar posteriormente los resultados. También se comunica con la capa servicios de datos web, para la interacción con uno o más gestores de bases de datos, con propósito de almacenar o recuperar los datos necesarios. A continuación se detallan algunas de las técnicas y herramientas aplicados en esta capa.

3.6.2.1. Validación de datos en el servidor

No es suficiente realizar la validación de datos en la capa de presentación visual es con objeto de disminuir la cantidad de peticiones y además pueden ser interceptados con facilidad y manipulados, lo cual lleva a utilizar un conjunto de librerías para validar los datos en el servidor, siempre será necesario realizar las validaciones de las entradas de formularios u otro tipo de entradas de datos.

Las validaciones generalmente son de verificar números, cadenas de textos, etc., utilizando expresiones regulares.

3.6.2.2. Limitación de intentos durante la autenticación

Es una técnica utilizada para la disminución de ataque de fuerza bruta, es decir, cuando el usuario no autorizado intenta acceder a la aplicación web sin contar con los datos correctos, por ende es necesario limitar el número de intentos para acceder.

Esto permite anular los intentos continuos para acceder a la aplicación web, disminuyendo la probabilidad de éxito de este tipo de ataques, por ejemplo si el usuario falla tres o cinco veces en ingresar los datos de autenticación serán bloqueados automáticamente. Incluso es necesario tener registros de los intentos fallidos, cuyos registros pueden ser el identificador de usuario y la identificación de IP de la máquina.

3.6.2.3. Cifrado de datos en servidor

Los algoritmos de cifrado como tal se pueden aplicar en ambos casos: cliente y servidor, pero siempre es recomendable realizarlo en el servidor, porque estas no pueden ser interceptados.

Es necesario cifrar los datos confidenciales ya sea de manera síncrona o asíncrona según al requerimiento con las claves respectivas.

3.6.2.4. Transacción de los datos

Para garantizar la transferencia de datos entre la capa lógica de negocio y la capa servicio de datos web, es necesario contar con librerías que permitan manejar la transacción de datos a la base de datos y viceversa de forma segura.

Es recomendable utilizar la programación orientada a objetos para la respectiva conexión con la Base de Datos, de esta manera permitir la integridad de los datos sin corromper a la Base de Datos.

3.6.2.5. Proceso de datos en SQL

Utilización de las librerías correspondientes al lenguaje de programación en el cual se utiliza para la inserción, modificación y consulta de datos. A esto se lo denomina binding para construir las sentencias SQL, estos componentes nos ayudan a evitar la inyección de SQL.

Mencionar que en algunas tecnologías de desarrollo web ya viene incorporado con el uso de Framework, y en caso de no ser así se recomienda construir de forma genérica.

3.6.3. Capa servicios de datos web

Es la capa en donde persisten los datos y es la encargada de acceder a los mismos, está formada por uno o más gestores de Bases de Datos que realizan todo el almacenamiento de datos, reciben las peticiones de almacenamiento o recuperación de datos desde la capa lógica de negocio.

Cada Base de Dato deber tener asignados un usuario con los roles respectivos, independiente del administrador. A continuación se observa algunas de las técnicas y herramientas referentes a esta capa.

3.6.3.1. Servicios web

Un servicio web puede ser consumida por otras aplicaciones, y es la encargada de exponer los datos por medio de lógica de negocio a consumidores de este tipo de servicios. Básicamente sirve para intercambiar información entre dos aplicaciones independientemente al lenguaje en el que se encuentren implementados, cuyos servicios se encuentran desarrollados en un estándar llamado Web Services Description Language y utilizan el protocolo SOAP en su mayoría, no es el único protocolo utilizado para este tipo de servicios.

Se debe tener precaución respecto al formato de datos que maneja este tipo de servicios, porque puede ser tratadas y expuestas de manera errónea. Por lo general es usado el XML en su respuesta.

3.6.3.2. Credenciales de acceso

La función de las credenciales de acceso o de autenticación es prevenir el robo de identidad de una manera fácil. Esta función trabaja de dos maneras. Primero, le permite verificar si la aplicación web es legítima, y no una aplicación falsa o copia, posteriormente pasa a la autenticación, si los pares de datos son correctos usuario y contraseña, la aplicación permite el acceso, caso contrario hace una solicitud de pregunta secreta.

La pregunta secreta es un mecanismo que permite verificar la veracidad del usuario autorizado para el acceso a la aplicación web. Ahora las preguntas pueden ser expresados mediante una imagen o frase. También se puede relacionar un usuario a una dirección IP, con esto se puede disminuir la suplantación de identidad.

3.6.3.3. Limitar las peticiones

Esta técnica permite limitar la cantidad de peticiones al usuario, generalmente son usados para limitar las descargas. En caso de tener una cantidad elevada de peticiones puede causar que la aplicación web sea propenso a un ataque de tipo denegación de servicio.

Este tipo de ataques pueden provocar inestabilidad de las aplicaciones web, incluso la caída del servidor. Por tal motivo se debe considerar la incorporación de esta técnica de limitar las peticiones permitiendo el rendimiento y estabilidad.

3.6.4. Capa de alojamiento web

En esta capa se encuentra alojada la aplicación web en su totalidad, por ello es necesario cumplir con las seguridades requeridas que puedan garantizar un buen funcionamiento de las aplicaciones web.

La primera medida de seguridad a realizar en contar con un proveedor de servicio hosting de renombre, estos proveedores en lo mínimo deben contar con un buen soporte y facilitar las configuraciones de seguridad necesarias. Para el propósito se presenta algunas de las técnicas y herramientas que son recomendables para esta capa, de esta manera garantizar aplicaciones web estables.

3.6.4.1. Balanceo en la carga de datos

En las aplicaciones web donde hay un gran número de peticiones, es necesario equilibrar la carga de datos de manera que los servidores compartan el trabajo. Consiste realizar un cluster ya sea de tipo servidor de aplicaciones o base de datos, es decir repartir las peticiones entre servidores. En caso de que un servidor falle o cae los demás siguen trabajando.

3.6.4.2. Componentes ORM

Al utilizar los componentes ORM se permite disminuir el acceso constante a la base de datos debido a que se almacenan los registros de las últimas consultas. Las nuevas peticiones primero acuden al cache, si es que los datos consultados se encuentran será generado la respuesta respectiva a la petición, caso contrario se acude a la Base de Datos y almacenarlo en dicho cache, con esto se aligera el trabajo a la Bases de Datos.

3.6.4.3. Servidor de cache

Un servidor de cache es de mucha utilidad cuando se trata de almacenar las ultimas peticiones realizadas por medio de un navegador web, la información que se guardan generalmente son las imágenes, códigos scripts entro otros.

El propósito de este servidor es la disminución de la carga a la aplicación web, por tanto al momento de realizar alguna petición por primera vez, algunos de los elementos quedan almacenados, por lo cual para las siguientes peticiones en primera instancia acudo al servidor cache y lo despliega los resultados sin necesidad de que la petición acuda al servidor de aplicaciones.

El uso de los servidores de cache aligeran el generando de una nueva respuesta por cada petición, acortando el tiempo de respuesta a las peticiones realizadas.

3.6.4.4. Corta fuegos como seguridad

Un corta fuegos en una aplicación web es esencial, porque tiene por objetivo principal de proteger de los diversos ataques realizados por personas ajenas, además permite monitorear el tráfico de HTTP.

Algunos de los antivirus utilizan los cortafuegos que permiten analizar en tiempo real el tráfico de los datos. Existen una variedad de corta fuegos disponibles con diferentes propósitos, pero con el objetivo de no permitir el ingreso de elementos maliciosos, sin necesidad de hacer cambios a la aplicación.

3.6.4.5. Implementación de SSL

Permite tener confidencialidad en las transmisiones de nuestra capa lógica de negocio a la capa de presentación visual y viceversa.

Proporciona sus servicios de seguridad cifrando los datos intercambiados entre el servidor y el cliente con un algoritmo de cifrado simétrico, típicamente el RC4 o IDEA, y cifrando la clave de sesión de RC4 o IDEA mediante un algoritmo de cifrado de clave pública, típicamente el RSA.

Un componente muy común para la generación de los certificados digitales es la utilización de OpenSSL.

3.7. Aplicación del Modelo de seguridad

En este apartado se realiza la aplicación del modelo de seguridad con el propósito de evaluar la funcionalidad, para ello se realiza un prototipo práctico en el cual la aplicación demande el modelo de seguridad en las aplicaciones web; las técnicas y herramientas para minimizar las vulnerabilidades explicados durante el desarrollo del presente trabajo.

El modelo de seguridad proporciona información respecto a las vulnerabilidades, tomando en cuenta las principales características de seguridad referentes a la web. Para ello se establece los componentes del modelo en capas: Capa de Presentación Visual, Capa Lógica de Negocio, Capa Servicio de Datos Web y Capa de Alojamiento Web, sin restricción a las tecnologías con el que se desarrolle una aplicación web. Además se mencionan las ventajas y desventajas del uso del modelo de seguridad.

Para observar las técnicas y herramientas sugeridas, se realiza un análisis en cada una de las capas. A continuación se presenta las tablas 3-1, 3-2, 3-3, y 3-4.

Capa de presentación visual:

Procedimiento	Aplicación	Descripción	Resultado
Utilización de JavaScript para las validaciones.	Correcto.	JavaScript es de mucha utilidad, cuando se trata de validar los datos en el lado del cliente con expresiones regulares.	Disminuye los posibles ataques de código malicioso.

Uso adecuando de JavaScript, sin lógica de negocio y servicios web.	Correcto.	El uso de JavaScript únicamente para la construcción de todos los componentes HTML, sin realizar ningún tipo de lógica de negocio. No se deben llamar a servicios web externos en esta capa.	Disminuye los ataque de tipo SQL inyección, si se maneja lógica de negocio.
Herramientas para evitar la inyección de CSS y JavaScript.	Correcto.	Utilización de los Framework como Extjs, el cual no permite la visualización del código JavaScript o CSS ingresado por el usuario.	Disminuye la inyección XSS, con código JavaScript y CSS.
Uso de herramientas de encriptación de datos en el lado del cliente.	Incorrecto.	No se debe usar ninguna herramienta o algoritmo para encriptar los datos que se envían desde el cliente hacia el servidor y viceversa.	Cualquier persona mal intencionada puede obtener la información de la transacción por medio de un Sniffer.
Técnica para evitar las manipulaciones, denominados Anti-Tampering.	Incorrecto.	No se usa ninguna técnica para disminuir la manipulación de los campos ocultos.	Cualquier persona mal intencionada puede modificar los parámetros ocultos y alterar la información para su conveniencia.

Tabla 3-1: Procesos en la capa de presentación visual

Capa lógica de negocio:

Procedimiento	Aplicación	Descripción	Resultado
Validación de los formulario en el servidor.	Correcto.	Se toman en cuenta que la validación en la capa de presentación visual no es suficiente, para el funcionamiento correcto es recomendable validar.	Evita la inyección de códigos inesperados dentro de la aplicación.
Limitación de autenticación, es decir el número de intentos de acceso.	Incorrecto.	No se limitan los intentos de acceso.	Cualquier usuario mal intencionado puede recurrir a un ataque de fuerza bruta y acceder a la aplicación.
Limitación de la cantidad de peticiones.	Correcto.	Se revisa la eficiencia de la aplicación web, para observar la cantidad de peticiones que soporta.	Disminución de un ataque de tipo DDoS.
Cifrado de datos en el servidor.	Correcto.	Utilización de algoritmos de cifrado, para que los datos no sean legibles a la vista, hay que dar prioridad a los datos confidenciales.	Disminución de la manipulación de los datos confidenciales.
Flujo de datos en la capa lógica de negocio.	Correcto.	Las conexiones con el gestor de BD deben de permitir transacciones seguras y eficientes.	Permitir que los datos fluyan de manera segura.

Proceso de datos, es decir, sentencias SQL.	Correcto.	Para este proceso es recomendable la utilización de Programación Orientado a Objetos.	Disminuye un ataque de inyección de SQL.

Tabla 3-2: Procesos en la capa lógica de negocio

Capa servicios de datos web

Procedimiento	Aplicación	Descripción	Resultado
Autenticación de usuarios.	Correcto.	Al realizar web servicies siempre se valida a la aplicación cliente que realiza la transacción.	Disminución de un ataque DDoS.
Identificación por IP de usuario.	Incorrecto.	Debido al negocio no se puede amarrar una IP específica a un cliente.	Suplantación de identidad.
Limitación de consumo.	Incorrecto.	No se toma en cuenta la cantidad de peticiones que realiza el usuario. En excepción a limitar, a la cantidad de descargas a realizar.	Ataque tipo Dos o DDoS.
Asignación de roles a los usuarios, a cada BD.	Correcto.	Es fundamental que asignemos uno o más usuarios a cada BD.	Disminuye el riesgo de exponer datos de la BD.

Tabla 3-3: Procesos en la capa servicio de datos web

Capa de alojamiento web

Procedimiento	Aplicación	Descripción	Resultado
Estructura del balanceo de carga.	Incorrecto.	Hasta ahora no se ha llevado una arquitectura de ese tipo debido a las dimensiones de la aplicación.	Ataque de tipo DOS y DDoS.
Componentes ORM	Correcto.	permite vincular los objetos usados en nuestro modelo de la aplicación con la base de datos relacional.	Ataque de tipo DOS y DDoS.
Servidor de cache.	Incorrecto.	No se ha implementado un servidor de cache.	Ataque de tipo DOS.
Componentes Firewall de aplicación.	Incorrecto.	No se ha implantado un firewall de ampliación.	Ataque DOS.
Implementación de SSL HTTP.	Correcto.	En el desarrollo siempre se revisa que la implementación funcione de manera correcta con HTTPS, que es más segura.	Funcionalidad de la aplicación en un protocolo seguro.

Tabla 3-4: Procesos en la capa de alojamiento web

3.7.1. Prototipo

En el prototipo se abordan las vulnerabilidades más comunes que se encuentran en las aplicaciones web, como SQL inyección, Cross Site Scripting en sus tres casos y la encriptación de los datos durante el tránsito, como también en el almacenamiento, para ello se puede utilizar un algoritmo de cifrado síncrona o asíncrona como se sugiere en el apartado que se trata el tema de cifrado (Ver acápite 3.6.1.3. y 3.6.2.2.).

Las herramientas para el desarrollo del prototipo son las siguientes: HTML5, CSS3, JAVASCRIPT, JQUERY, JQUERY-UI, PHP y MYSQL, estas herramientas son de software libre y son utilizadas generalmente para el desarrollo de las aplicaciones web en su mayoría a excepción de PHP y MYSQL que pueden ser sustituidos por JAVA o Microsoft .NET. Existen una variedad de Framework como Laravel en PHP, Spring en JAVA y ASP.NET MVC Framework de Microsoft .NET.

En primera instancia se presenta el diseño de interfaz del prototipo, simulando la aplicación web con características respectivas como se observa en la figura 3-21.

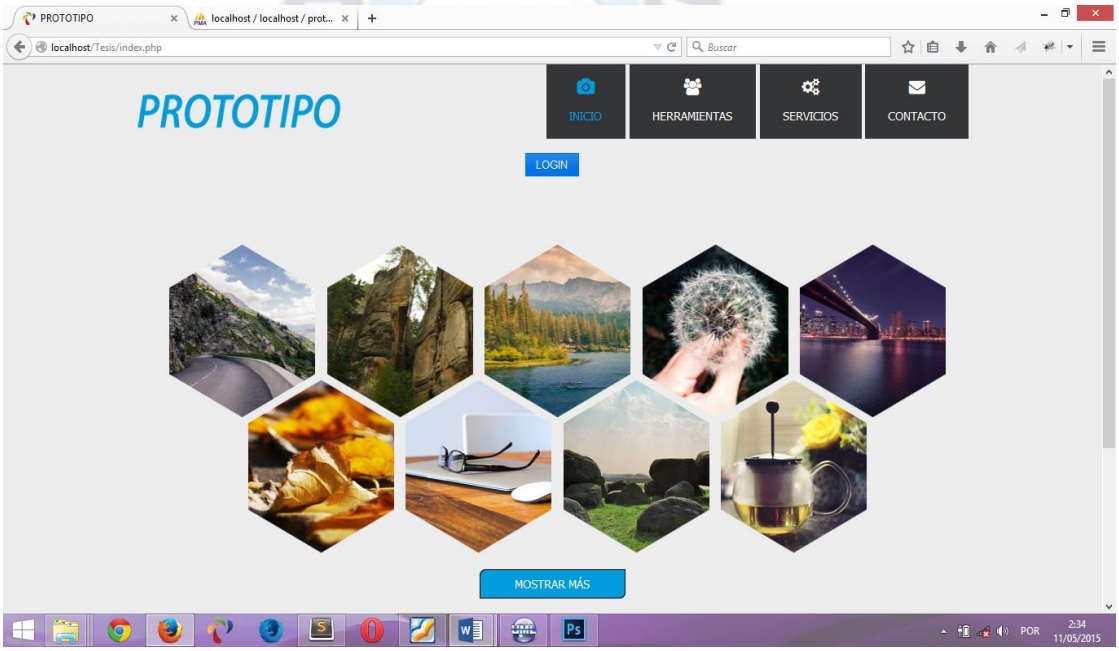

Figura 3-21: Interfaz general del prototipo

Con esta práctica vamos a realizar en un principio las pruebas de SQL inyección (Ver acápite 2.8.5.), según a la propuesta de este tipo de vulnerabilidades.

Para ello se implementa un formulario de autenticación, el cual permite la identifacación de los usuarios autorizados. Es asi que la aplicación no debe admitir el

acceso a los usuarios ajenos, por lo que el proceso de los datos ingresados se realiza con el parametro de enlace (bindValue) que nos ofrece el PDO.

Para utilizar el bind del PDO tambien es necesario que la conexión con el gestor de base de datos sea del mismo tipo, de la siguiente manera:

```
const USER = "root";
const PASS = "clave";
private static $inst = null;
$dsn="mysql:host=NOMBREHOST;dbname=BASEDATOS";
self::$inst=new PDO($dsn, self::USER, self::PASS);
self::$inst->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);
```

Despues de ver la respectiva conexión que perimite la fluides de los datos (Ver acápite 3.6.2.4.) se hace el procesado de auntenticación con el proceso de dato en SQL (Ver acápite 3.6.2.5.), como se observa a contunuación:

```
$datos = $pdo->prepare("SELECT * FROM tabla WHERE user = ? AND pas = ?");
$datos->bindValue(1, “Usuario”);
$datos->bindValue(2, “Clave”);
```

En la figura 3-22 se observa el formulario de autenticación o incio de sesión. Cuyo formulario es desplegado haciendo clic en el boton “LOGIN” presentando dos entradas de datos; el primero el campos de “Usuario”, el segundo el campo de “Clave o Contraseña” y un boton de accion para ejecutar la instrucción de inicio de sesion.

En dicho formulario se realizan las diferentes pruebas de vulnerabilidad de tipo SQL inyección.

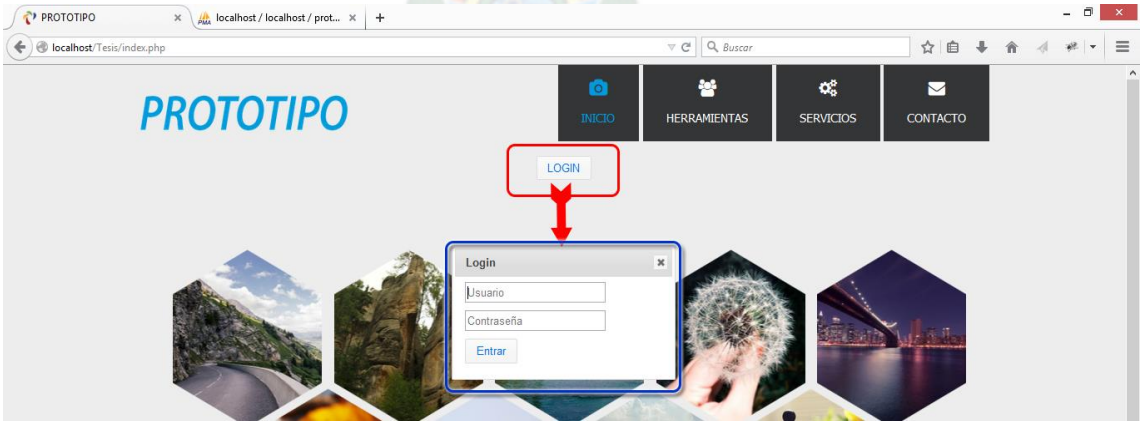

Figura 3-22: Formulario de inicio de sesión

En la figura 3-23 se muestra el resultado de la autenticación correcto, sin intento de vulnerar con SQL inyección. Por lo tanto no genera ninguna alerta de notificación, es asi que la aplicación indica que la sesion a sido iniciada correctamente y muestra el usuario autenticado.

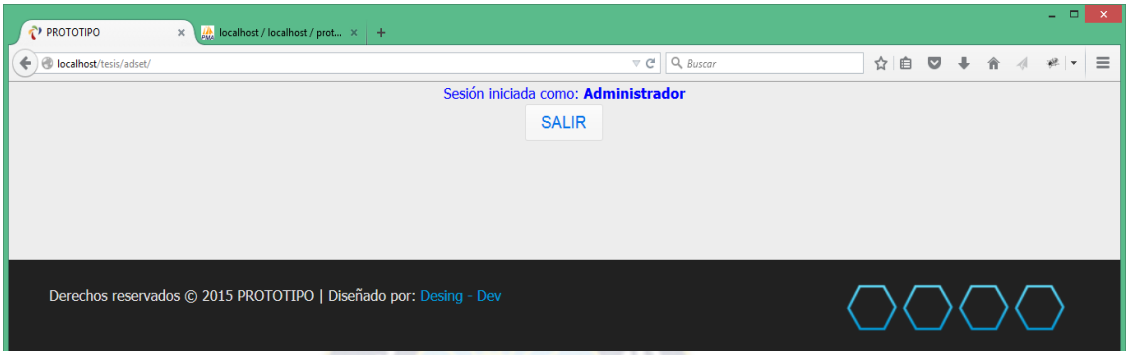

Figura 3-23: Inicio de sesión con datos correctos

Ahora se procede a intento de vulnerar con SQL inyección (Ver acápite 2.8.5.) como se observa en la figura 3-24, las instrucciones que se usan son los siguientes.

- admin /* En caso de conocer el usuario
- admin -- En caso de conocer el usuario
- ' or '1'='1
- ' or 1=1 ---
- ' having 1=1 ---
- ' or users.userName like 'a%' ---
- ' or 1=1; drop table users; --

Estas instrucciones son alguna de las mas utilizadas para vulnerar, en la figura 3-24 se observa una alerta de notificacion, indicando que el usuario no esta autorizado.

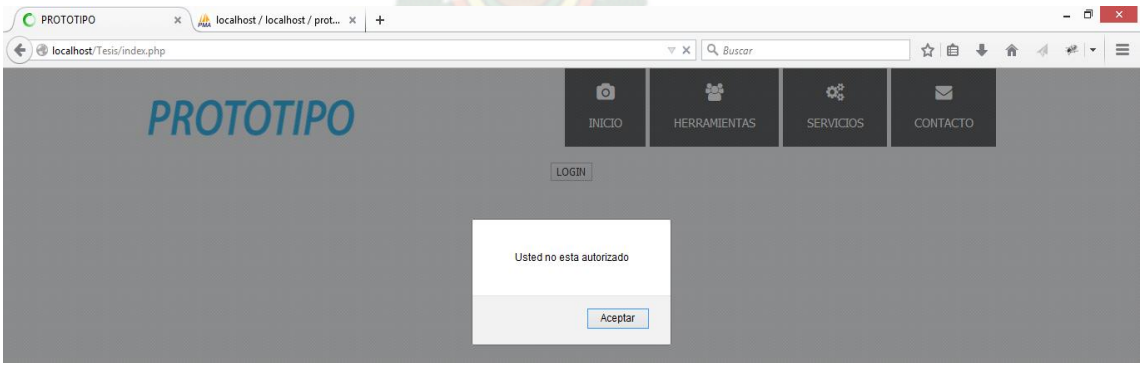

Figura 3-24: Alerta de usuario no autenticado

Es asi que se observa la funcionalidad de las técnicas y herramientas sugeridas contribuyendo a que las aplicaciones sean menos vulnerables. Ademas mancionar que es recomendable guardar las actividades que realiza el usuario autenticado en las aplicaciones.

Para ver esquematicamente la funcionalidad veace la figura 3-25 en el cual se observa el comportamiento durante el proceso de autenticación.

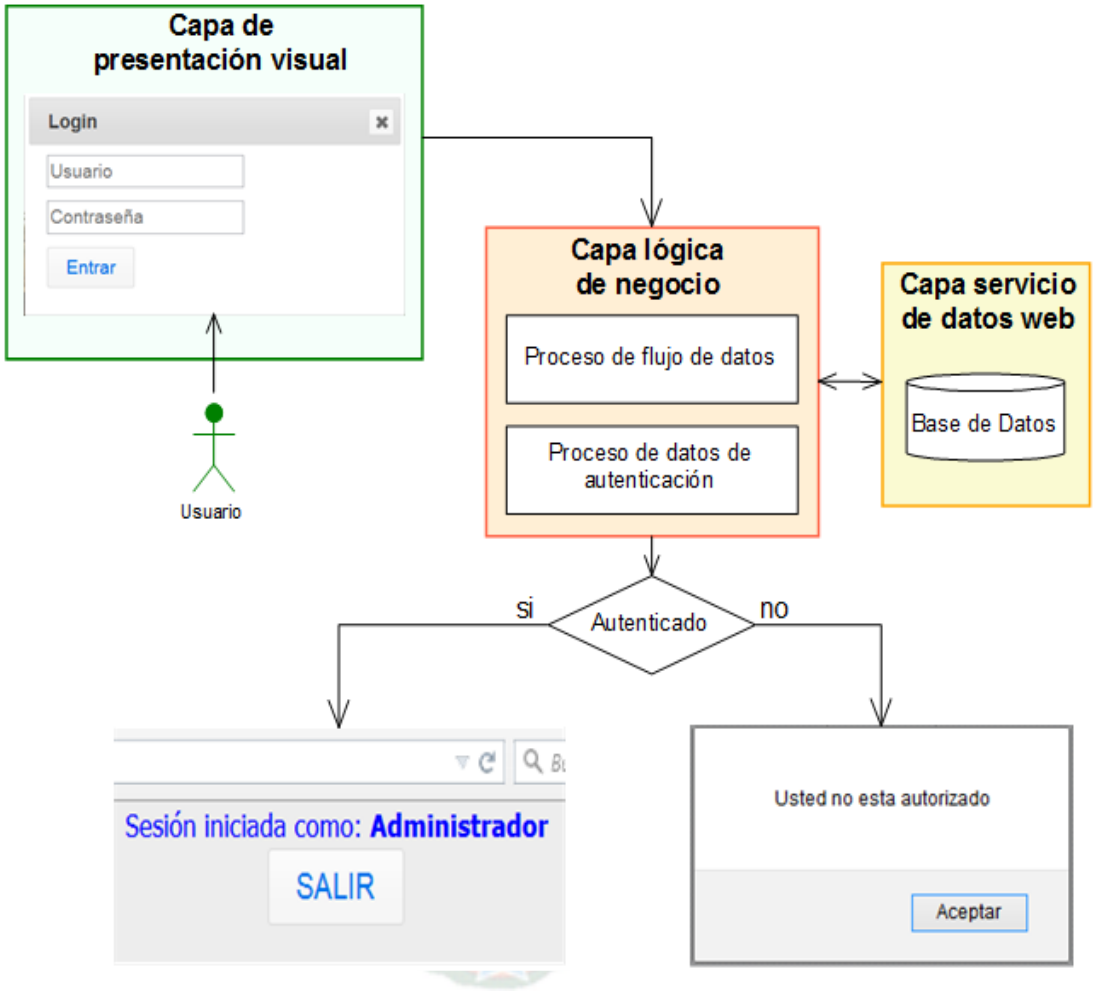

Figura 3-25: Proceso de autenticación

Ahora se realiza las pruebas de vulnerabilidad de tipo Cross Site Scripting (XSS), para ello usaremos el código de JavaScript, tomando encuenta tres formas de vulnerar listados a continuación:

- El Tipo-0, que se utiliza para ejecutar código remotamente o de forma local.
- El Tipo-1, ataque no-persistente o reflejado utilizado en páginas no estáticas.

- Y el Tipo-2, ataque persistente, donde se inyecta código en aplicaciones.

Suponiendo que la aplicación es vulnerable, hacemos la prueba de la siguiente manera, pasamos un valor como parametro, en este caso es un codigo JavaScript; http://prototipo.com/ejemplos/index.php?id=<script>alert('Si_el_mensaje_de_alerta_es_lanzado,_es_un_indicio_de_la_vulnerabilidad_existente');</script>, como se observa en la figura 3-26.

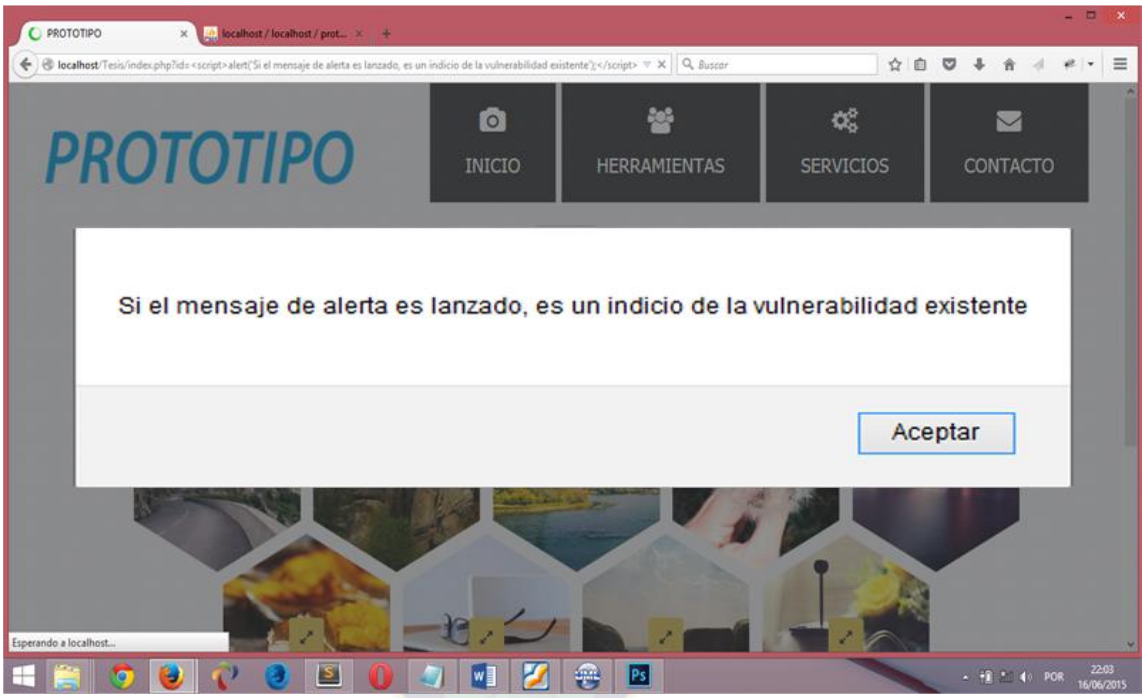

Figura 3-26: Aplicación web vulnerable a XSS

La diferencia entre el Tipo-0 y los otros tipos de ataque, es que el código se inyecta a través de la URL pero no se incluye en el código fuente de la aplicación. Es decir, el código nunca llegó a la aplicación web, solo se ejecutó en el navegador.

Se puede modificar la información a cualquier elemento HTML contenedor que tenga un identificador ya sea por ID, NAME o CLASS. Estos identificadores pueden ser obtenidos con el código JavaScript o JQuery.

El propósito es realizar la inserción de la información modificada con el DOM por ejemplo de la siguiente manera:

```
document.getElementById('contenidoxss').innerHTML ='TEXTO MODIFICADO';
$('#contenidoxss').text('TEXTO MODIFICADO');
```

Con cualquiera de los códigos el resultado es un texto modificado, resaltado con el color rojo. Como se observa en la figura 3-27.

Figura 3-27: Vulnerabilidad XSS de Tipo-0

El de Tipo-1 probablemente es la vulnerabilidad más común que hay, por lo usual se encuentra en motores de búsqueda, es una vulnerabilidad no persistente o reflejada. Utilizando otra aplicación falsa a la original, pero con el mismo contenido. El comportamiento es de la siguiente manera como se observa en la figura 3-28.

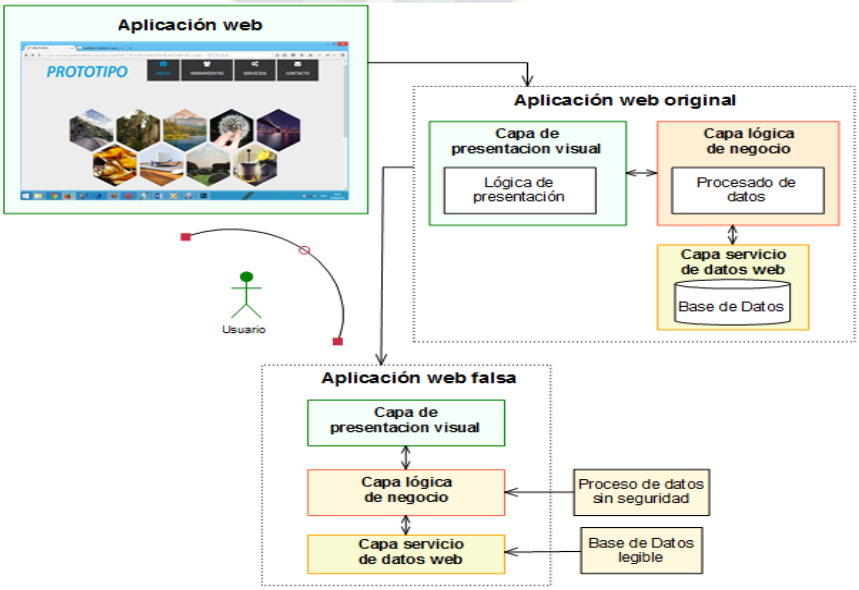

Figura 3-28: Comportamiento de la vulnerabilidad XSS de Tipo-1

La vulnerabilidad Tipo-2 permite hacer los ataques más peligrosos y poderosos a las aplicaciones web. Es conocida como vulnerabilidad persistente o almacenada. La diferencia a los anteriores, es que la inyección la información proporcionada es almacenada en la base de datos, después es mostrada a otros usuarios que visiten la aplicación, por eso se dice que la vulnerabilidad persiste. Esto es lo poderoso, la inyección se quedará siempre en alguna parte de la web, no estará en una página que se genera cada vez que se pasa información al servidor.

También se puede utilizar para obtener información sobre las Base de Datos realizando de la siguiente forma:
http://prototipo.com/ejemplos/index.php?id==-2_union_select 1,2,_group_concat(table_name),_4,5_from_information_schema.tables_where table_schema=database()-- El resultado que nos despliega, son las tablas que tiene la Base de Datos, como se observa en la figura 3-29.

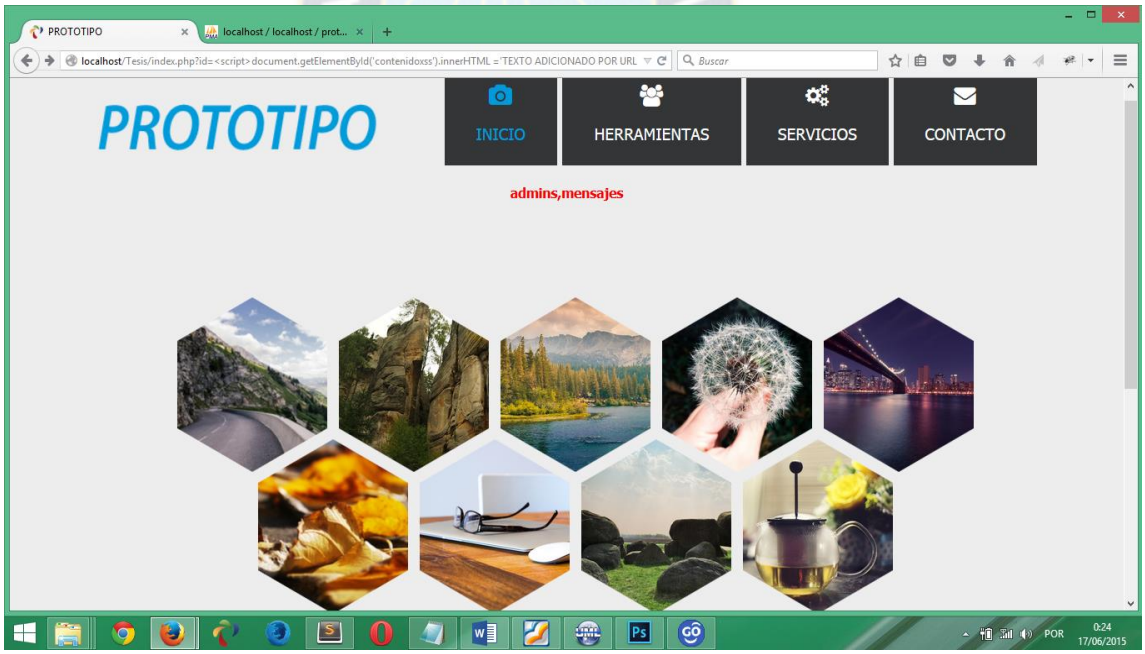

Figura 3-29: Vulnerabilidad XSS de Tipo-2

Esquematicamente su coportamiento es como se observa en la figura 3-30, el ataque del Tipo-2 aprovecha la vulnerabilidad en URL de las aplicaciones web. La istruccion que vulnera es por envio de parametros con el metodo GET, para que el servidor procese estos datos y como resultado proporcione informacion referente a la Base de Datos, pudiendo extraer incluso los datos de las tablas.

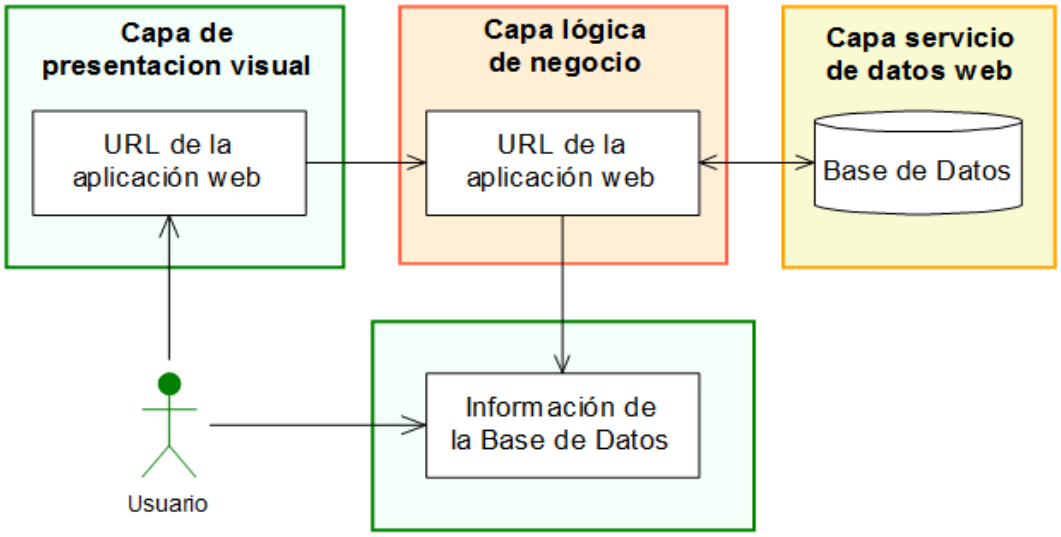

Figura 3-30: Comportamiento de la vulnerabilidad XSS de Tipo-2

Pasamos a realizar el encriptado o cifrado (Ver acápite 3.6.1.4., 3.6.2.3.) de los datos de forma síncrona, solo con una clave unica para el cifrado y descifrado.

Para el envio de datos utilizamos un formulario de "Contactos" del prototipo, como se observa en la figura 3-31. En el cual se puede apreciar dos formularios, uno para el envio de datos y el otro para la recuperacion del dato cifrado.

Los datos enviados son cifrados por un algoritmo con una clave secreta que puede ser de 8 o 24 caracteres, la misma clave es requerida para descifrar la informacion.

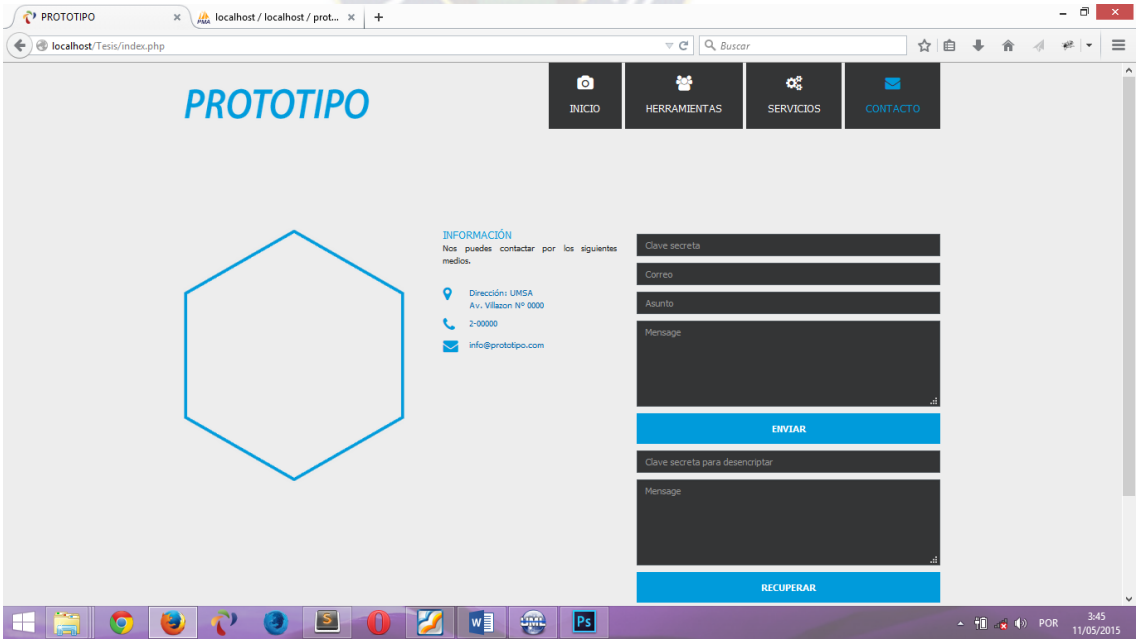

Figura 3-31: Formulario de contacto para envió de datos

Los campos de entrada de datos del formularion son validados (Ver acápite 3.6.1.1., 3.6.2.1.), es asi que no permite procesar las entradas de datos vacias, por lo tanto como se aprecia en figura 3-32 indica que el correo debe ser ingresado obligatoriamente, haciendo notar con un color rojiso como alerta.

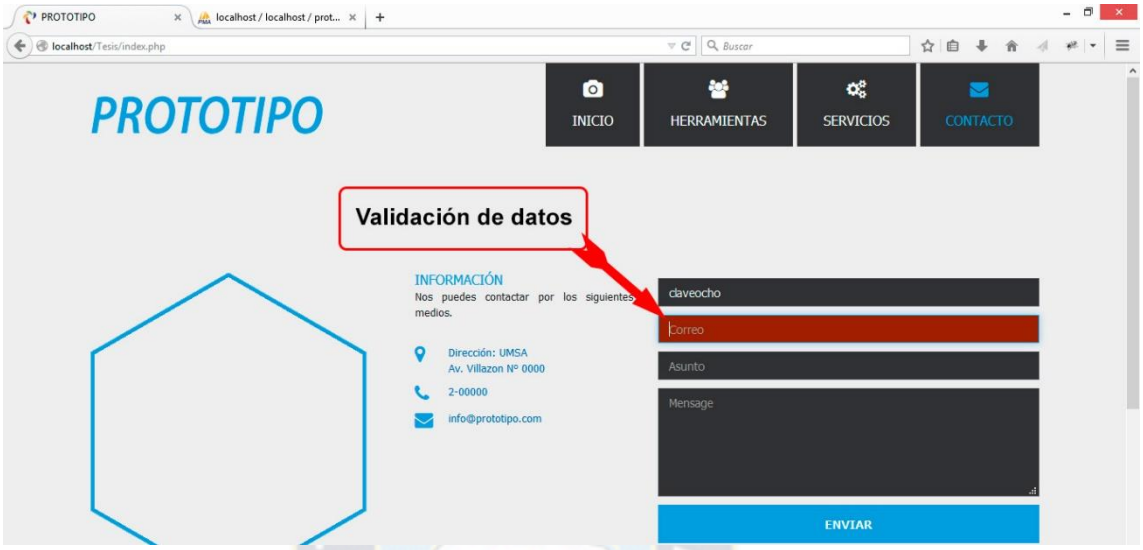

Figura 3-32: Validación de datos con resalte rojo como alerta de requerimiento

El rellenado de datos se observa en la figura 3-33, en el primer campo de texto pertenece a la clave secreta con “claveocho” de 8 caracteres, cuyo clave sera requerido para la obtencion de la información cifrada, en el segundo campo es requerido un correo electronico para su identificación, en el tercer campo el asunto y por ultimo esta el campo de mensaje que será cifrado

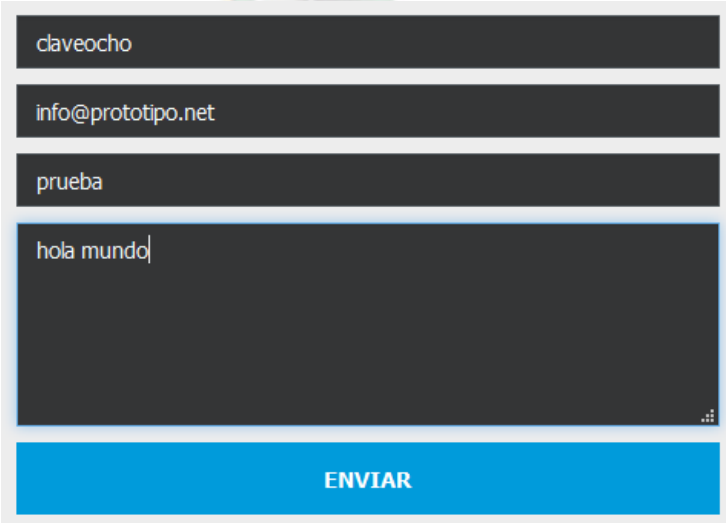

Figura 3-33: Datos a enviar para el almacenamiento en la Base de Datos

Durante el proceso del envio y cifrado de los datos, se observa como se envia cada uno de los datos ilustrados en la figura 3-34. El mensaje es enviado dos veces la primera sin el cifrado y la segunda cifrada.

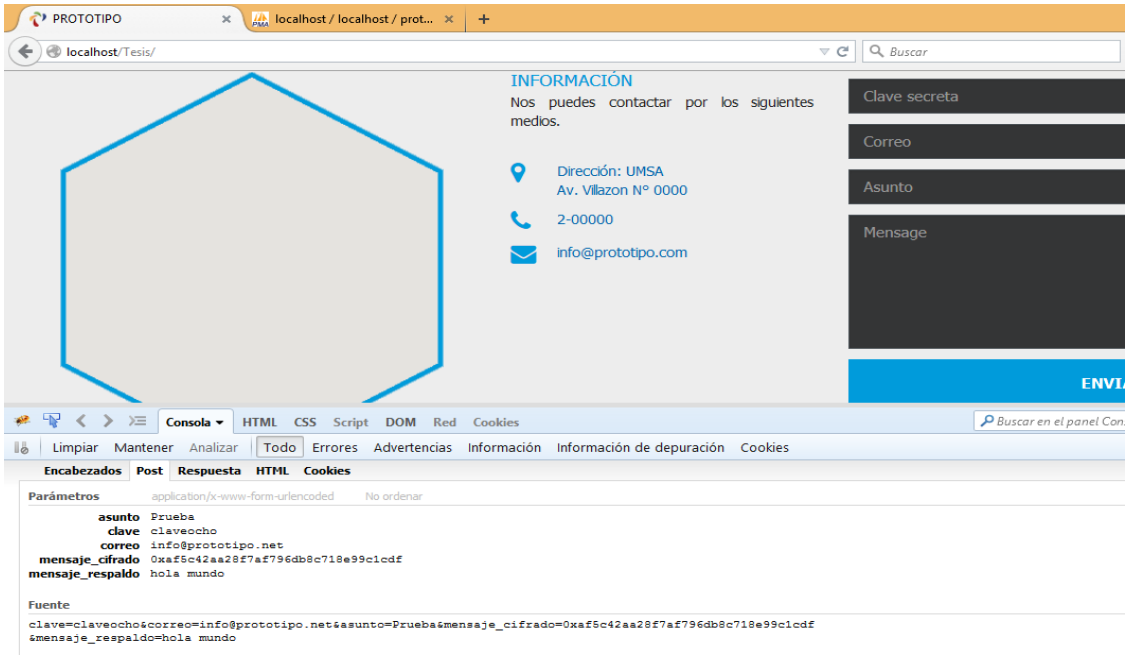

Figura 3-34: Visualización de los datos previos al envió a la Base de Datos

En la figura 3-35 se muestra el almacenamiento de los datos, enfocando en las columnas de “clave”, “mensaje” y “respaldo” se observa que en “clave” se guardan las claves secretas (claveocho), en “mensaje” el dato cifrado (hola mundo) y en el “respaldo” el mismo dato que del mensaje pero sin el cifrado respectivo.

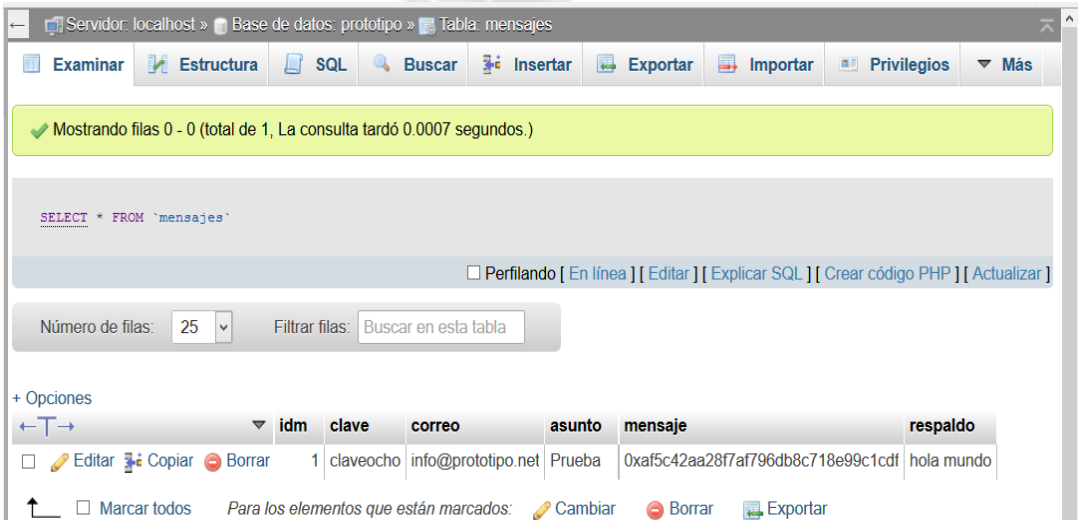

Figura 3-35: Datos almacenados en la Base de Datos

Para la obtención de la información cifrada, es necesario ingresar en el formulario de obtención de datos, la clave secreta correcta. Como se observa en la figura 3-36.

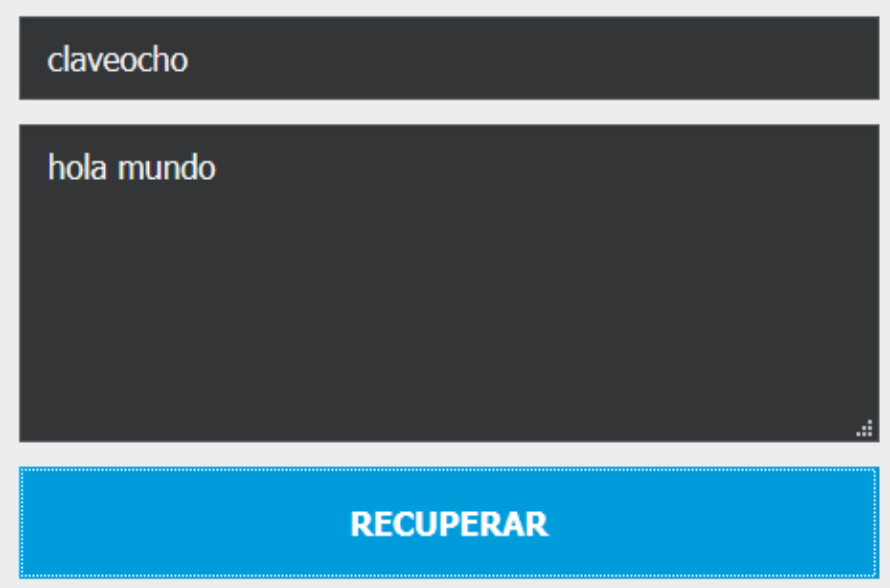

Figura 3-36: Datos recuperados con la Clave Secreta correcta

En caso de que la clave ingresada es incorrecta, se despliega un mensaje de error como se observa en la figura 3-37.

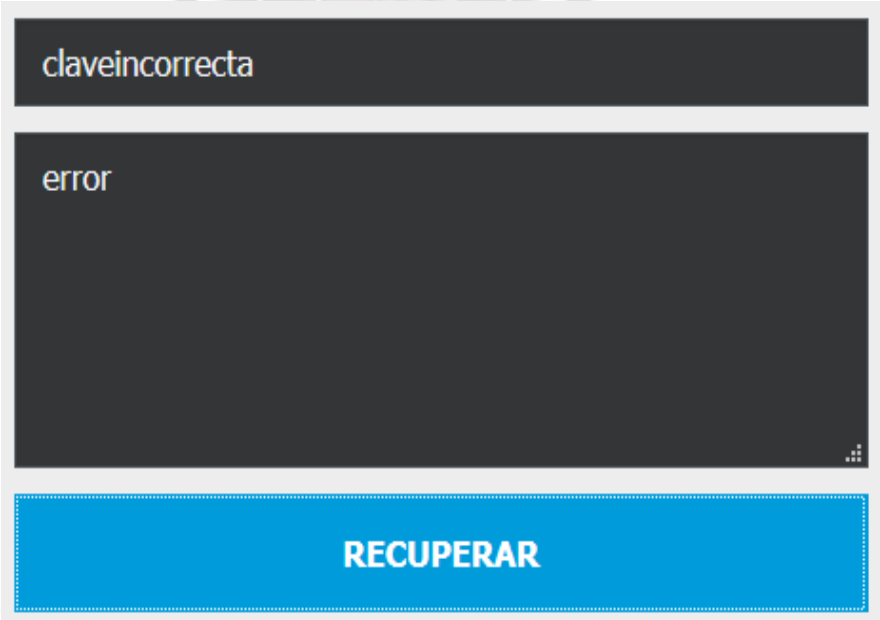

Figura 3-37: Datos recuperados con la Clave Secreta incorrecta

Por lo tanto la información cifrada no puede ser obtenida sin la clave secreta, por lo que la información almacenada en la Base de Datos es más segura.

Esquemáticamente se tiene el siguiente comportamiento el proceso de cifrado, como se observa en la figura 3-38.

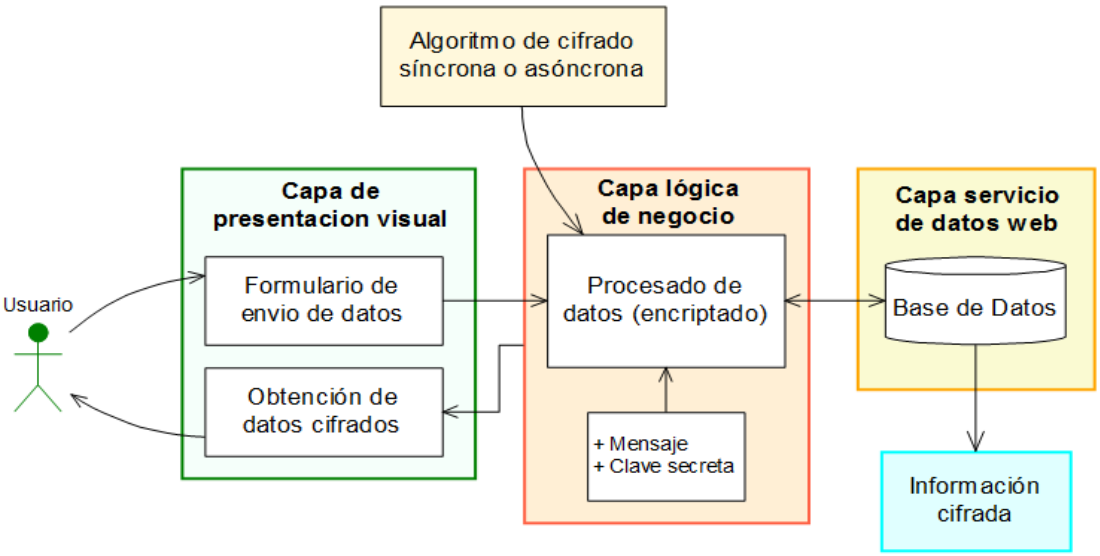

Figura 3-38: Comportamiento del proceso de cifrado de datos

3.7.2. Resultados del prototipo

Después de aplicar el modelo de seguridad en las aplicaciones web en el prototipo, se observa que no se aplica todas las propuestas realizadas, por tanto se tiene resultados en un 80% respecto a su totalidad de la propuesta.

Para la autenticación de usuarios, en la capa lógica de negocio se evalúa cada parámetro recibido y además se realiza el preparado antes de ejecutar la instrucción de SQL, de esta manera se evita SQL inyección.

Respecto a las vulnerabilidades de tipo XSS mostrados en el prototipo la solución dada es, no pasar los datos o parámetros por medio de URL. Por lo que es recomendable usar la tecnología Ajax, proporcionada por la Biblioteca JQuery.

Para el cifrado de los datos en caso particular del prototipo, se utiliza el algoritmo de cifrado simétrica en la capa lógica de negocio. El cifrado se puede realizar en la capa de presentación visual utilizando el mismo algoritmo, pero no es recomendable por que puede ser interceptado y modificado, es así que todo el proceso de datos de esta índole se realiza en la capa lógica de negocio.

El cifrado se realiza con una clave secreta, es la única llave para el respectivo descifrado de la información.

3.7.3. Ventajas al aplicar el modelo de seguridad

Según el modelo de seguridad en la aplicaciones web propuesto, se presenta las siguientes ventajas según a las capas:

Capa de presentación visual:

- El uso adecuado de las técnicas y herramientas de seguridad para el desarrollo web, permiten disminuir las vulnerabilidades.
- Encriptado o cifrado de datos en la capa de presentación visual, permite disminuir los ataques para obtener información durante la transacción por medio de sniffer. Aunque no es recomendable ni seguro realizarlo en esta capa, pero optimiza el rendimiento de la aplicación web.
- El uso de Framework Extjs permite optimizar la aplicación web, evitando la inyección de código malicioso.
- Con el uso de las técnicas de anti manipulación se evita la modificación de los parámetros ocultos, y la alteración o el cambio de la información.

Capa lógica de negocio:

- La validación de formularios disminuye la inyección de código no esperado en las aplicación web.
- La restricción de la cantidad de peticiones disminuye el ataque de denegación de servicio, es decir la aplicación web funciona sin truncamiento.
- La restricción de la cantidad de intentos en la autenticación, previene los ataques de fuerza bruta.
- El cifrado de los datos permite proteger la información confidencial en las aplicaciones web. Por tanto los datos de autenticación obligatoriamente deben ser cifrados por su seguridad.
- El uso adecuado de las técnicas y herramientas para el procesado de datos disminuye que la aplicación sea vulnerable, de esta manera evitamos los códigos maliciosos.
- El uso de los componentes de SQL de tipo "bindValue, bindParam" permite disminuir los ataques de tipo inyección de SQL, es más recomendable utilizar PDO en el trato de los datos.

Capa servicio de datos web:

- La restricción de las peticiones al servidor permite la disminución de los ataques tipo DOS o DDoS.
- Para evitar la suplantación de identidad, es recomendable identificar al usuario por medio de IP asociado. Pero no es una solución óptima.
- La asignación de usuarios con roles a las Bases de Datos, no permite el acceso directo a los datos almacenados.
- El almacenamiento de los datos en la Base de Datos es un mecanismo recomendable para evitar modificaciones en la información confidencial.

Capa de alojamiento web:

- La utilización de los certificados SSL HTTPS se aumenta la seguridad de la aplicación. Porque la aplicación web se ejecuta bajo un protocolo seguro.
- La implementación de los Servidores de Cache evita los ataques de denegación de servicio, ya que disminuye el número de consultas al servidor.
- El uso de ORM permite la integración completa de los elementos de la Base de Datos, con los componentes en el que han sido implementados.
- La utilización de Firewall reduce la probabilidad de sufrir ataques de denegación de servicio.

3.7.4. Desventajas al aplicar el modelo de seguridad

El modelo de seguridad tiene las siguientes desventajas:

- Inversión económica al personal del área de seguridad en las aplicaciones web.
- Inversión del tiempo para la implementación del modelo de seguridad.
- La complejidad de la culturalización de las buenas prácticas.

CAPITULO IV

MARCO DE PRUEBAS Y METRICAS

4.1. Introducción

En esta etapa del estudio se reproduce lo descrito en el Capítulo III mediante diferentes escenarios del modelo de seguridad en las aplicaciones web.

Para la obtención de datos estadísticos se optó por la técnica de recolección de datos por medio de la encuesta, planteados a los usuarios o entes de interés, dedicados al desarrollo de las aplicaciones web.

4.2. Comprobación de hipótesis

Para la comprobación de hipótesis, es necesario establecer el entorno de prueba para la recolección de datos, con los que se realiza la prueba de hipótesis. Para ello se da los niveles de ponderación, los mismos serán utilizados para los cálculos de la muestra, esto se realiza según a los valores de ajuste establecidos como se observa a continuación en la tabla 4-1.

Descripción	Escala	Ponderación
Pésimo	1	0
Malo	2	0
Regular	3	2
Bueno	4	5
Muy bueno	5	10

Tabla 4-1: Valores de ajuste

Los datos para la muestra son obtenidos bajo la técnica de recolección de datos, por medio de la encuesta realizada a la empresa OSPLEX “Servicios Integrales en Informática”, los mismos se dedican al desarrollo de software, en diferentes índoles como ser; sistemas a medida, aplicaciones web y móvil.

La encuesta fue realizada a dos grupos de desarrolladores web de la misma empresa, el primer grupo utiliza como plataforma base la tecnología Microsoft .NET, y el segundo utiliza las tecnologías de software libre.

Los dos grupos suman una cantidad de 20 desarrolladores, por lo tanto la muestra tiene como N = 20, y los resultados obtenidos se muestran a continuación, como se observa en la tabla 4-2.

Encuesta	Observados	Esperados
1	92	90
2	97	90
3	95	90
4	92	90
5	87	90
6	95	90
7	95	90
8	90	90
9	87	90
10	85	90
11	95	90
12	87	90
13	92	90
14	95	90
15	92	90
16	85	90
17	97	90
18	97	90
19	95	90
20	85	90
Total promedio	91.75	90

Tabla 4-2: Factores de ponderación total

La Prueba de Hipótesis para medias usando Distribución t de Student, se usa cuando se cumplen las siguientes dos condiciones:

- Es posible calcular las media y la desviación estándar a partir de la muestra.
- El tamaño de la muestra es menor a 30.

El procedimiento obedece a los 5 pasos esenciales que se presenta en el siguiente apartado (Ver acápite 4.2.1.).

4.2.1. Planteamiento de la hipótesis nula y alternativa

Primer paso: Se identifica la hipótesis nula y la hipótesis alternativa, bajo la siguiente notación: Hipótesis alternativa denotado por Hi, es la hipótesis planteada en el Capítulo I y la Hipótesis nula denotado por Ho, es exactamente lo contrario de la hipótesis planteada.

Hi: "El modelo de seguridad en las aplicaciones web permitirá la disminución de las vulnerabilidades, pudiendo de esta manera incrementar la seguridad en las aplicaciones web".

Ho: "El modelo de seguridad en las aplicaciones web no contribuyo en la disminución de las vulnerabilidades, por lo cual no se pudo incrementar la seguridad en las aplicaciones web"

Segundo paso: Determinar nivel de significancia, es el rango de aceptación de hipótesis alternativa. Para ello se considera:

- 0.05 para proyectos de investigación
- 0.01 para aseguramiento de calidad

Por lo tanto para nuestro caso: **$\alpha = 0.05$** este valor utilizamos por conveniencia.

Tercer paso: Evidencia muestral, cálculo de la media y la desviación estándar a partir de la muestra.

Según a los datos estadísticos tenemos los siguientes datos:

$$\bar{x} = \frac{\sum_{i=1}^{n} x_i}{n}$$

$$\bar{x} = \frac{92+97+95+92+87+95+95+90+87+85+95+87+92+95+92+85+97+97+95+85}{20}$$

$$\bar{x} = \frac{1835}{20}$$

$\bar{x} = 91.75$ Media

$$S_x = \sqrt{\frac{1}{n-1}\sum_{i=1}^{n}(x_i - \overline{x})^2}$$

$$S_x = \sqrt{\frac{1}{20-1}(353.75)} = \sqrt{18.61842105}$$

$S_x = 4.31$ Desviación estándar

Esto lo verificamos con una calculadora de Desviación estándar como se observa en la figura 4-1.

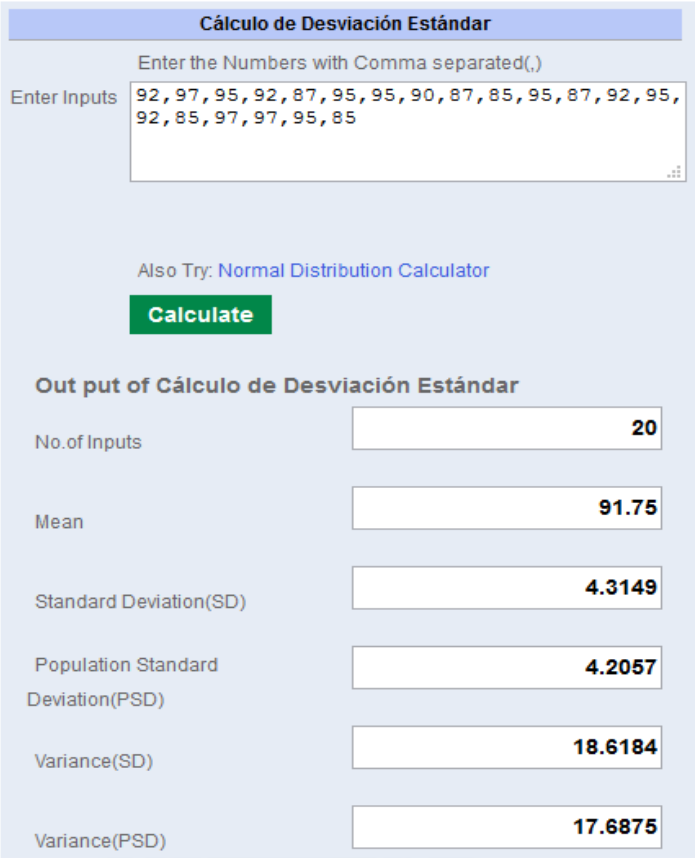

Figura 4-1: Calculadora de Desviación estándar

Fuente: [NCA13]

Cuarto paso: Se aplica la distribución t de Student para calcular la probabilidad de error (P) por medio de la fórmula como se observa a continuación.

Fórmula de t de Student:

$$t^* = \frac{\overline{x} - \mu}{s_x / \sqrt{n}}$$

Grados de libertad: gl = n – 1

Media: $\bar{x} = 91.75$

Valor a analizar : $\mu = 90$

Desviación estándar : $S_x = 4.31$

Tamaño de la muestra : n = 20

4.2.2. Calculo de la prueba estadística t de Student

Remplazando en la fórmula de t de Student

$$t^* = \frac{\overline{x} - \mu}{Sx/\sqrt{n}} = \frac{91.75 - 90}{4.31/\sqrt{20}} = 1.813$$

$$gl = n - 1 = 20 - 1 = 19$$

Esto lo podemos verificar con una calculadora de t de Student como se observa en la siguiente figura 4-2.

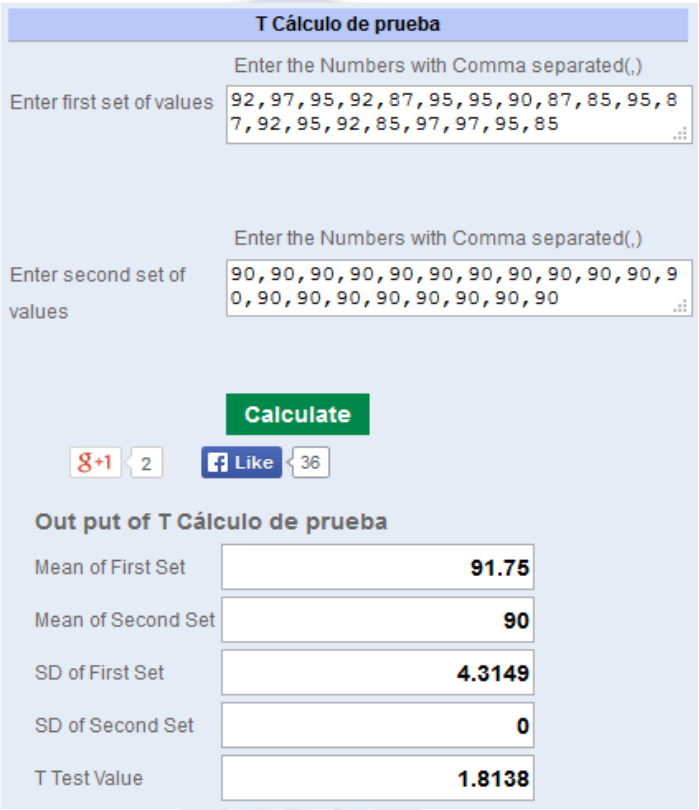

Figura 4-2: Calculadora de t de Student

Fuente: [NCA13]

Quinto paso: en base a la evidencia disponible se acepta o se rechaza la hipótesis alternativa, bajo las siguientes condiciones:

- Si la probabilidad de error (P) es mayor que el nivel de significancia. Se rechaza hipótesis alternativa.
- Si la probabilidad de error (P) en menor que el nivel de significancia: se acepta hipótesis alternativa.

Según a los datos calculados con la fórmula de t de Student procedemos a determinar si la hipótesis alternativa es válida o no. Viendo el resultado del cálculo se observa que: **t*=1.813**, se halla en la tabla de distribución de t de Student con el valor más próximo al resultado, tomando en cuenta la columna de grados de libertad.

***P** = 0.043*

***α** = 0.05*

*Siendo que: **P** < **α***

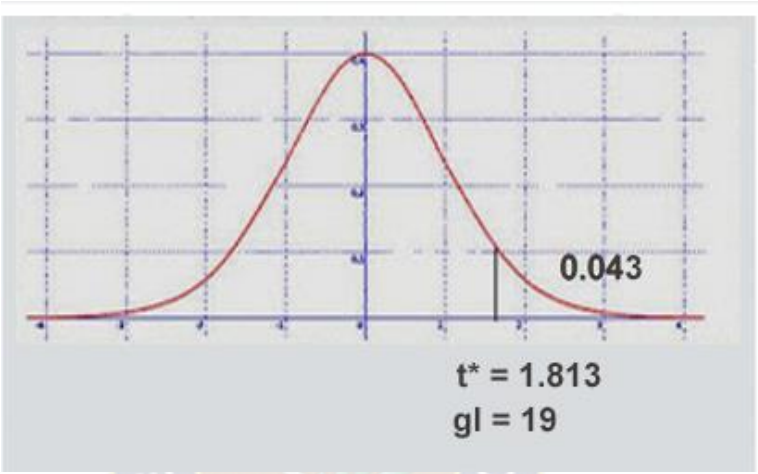

Figura 4-3: Grafica del resultado de t de Student

Entonces se concluye que la hipótesis Hi es aceptable, por lo tanto el modelo de seguridad en aplicaciones web propuesta es válida, porque contribuye en la disminución de las vulnerabilidades en aplicaciones web, en el proceso del desarrollo.

CAPITULO V

CONCLUSIONES Y RECOMENDACIONES

5.1. Introducción

En este capítulo se establecen las conclusiones y recomendaciones finales del presente trabajo y los posteriores trabajos que se pueden hacer, en base a la investigación realizada.

5.2. Conclusiones

El presente trabajo de investigación fue según a los objetivos propuestos en un principio, y las conclusiones a las que se llegaron en relación al modelo de seguridad en las aplicaciones web son las siguientes:

El objetivo general descrito en el Capítulo I, se cumplió con la elaboración del modelo de seguridad en aplicaciones web, planteados en 4 capas para tener aplicaciones menos vulnerables. Los cuales describen las arquitecturas, características, y otros aspectos que deben considerarse a la hora de implementar una aplicación web, de esta forma evitar las diferentes vulnerabilidades.

En relación a los objetivos específicos:

El primer objetivo específico: "Realizar arquitecturas de aplicaciones web según el modelo de seguridad para una orientación esquemática", se cumple en el Capítulo III, donde se describen las arquitecturas en dos patrones multicapa y modelo vista controlador, ilustrados con las figuras 3-3, 3-4, 3-5 y 3-6 respectivamente. Además se presenta arquitecturas genéricas esquematizados con las figuras 3-8, 3-9, 3-10 y 3-11. Más la arquitectura del modelo vista controlador figura 3-7.

El segundo objetivo específico: "Evaluar técnicas y herramientas para las validaciones respectivas", se cumplió en el Capítulo III (Ver acápite 3.6.1., 3.6.2.) en el cual se realiza un desglose de las técnicas y herramientas para la validación de datos en el lado del cliente y servidor, en el modelo de seguridad en aplicaciones web hace referencia a la capa de presentación visual y capa lógica de negocio.

El tercer objetivo específico: "Evaluar métodos de autenticación para evitar la vulnerabilidad en los formularios de tipo Login", se cumplió en el apartado del Capítulo III (Ver acápite 3.6.2.4. y 3.6.2.5.) los cuales tratan de la restricción de intentos de acceso al sistema de gestión o administración de una aplicación web, y cuando se habla de autenticación no se debe admitir las instrucciones de inyección SQL en un formulario tipo Login.

El cuarto objetivo específico: “Evaluar el manejo de sesiones para la administración de las aplicación web”, se cumplió en el apartado del Capítulo III (Ver acápite 3.6.3.2.) el cual permite tomar en cuenta la identificación con credenciales de acceso, esta información es de mucha utilidad cuando se quiere ver las actividades realizadas por los administradores o por usuarios infiltrados en la aplicación.

El quinto objetivo específico: “Identificar los riesgos y vulnerabilidades que afectan a las aplicaciones web para evitar las mismas”, se cumplió en los apartados del Capítulo II (Ver acápite 2.7., 2.7.1., 2.7.2. y 2.7.3.), haciendo énfasis en, la detección de riesgos del lado del cliente, detección de riesgos del lado del servidor y detección de riegos en el canal de comunicación. Son los tres principales puntos de ataque o vulnerabilidad que puede dañar una aplicación web.

El sexto objetivo específico: “Establecer estrategias para las buenas practicas, con el fin de evitar futuras amenazas o inconsistencias en las aplicaciones web”, se cumple en el Capítulo II y III puesto que se aborda la temática referente al modelo de seguridad en aplicaciones web, lo cual nos lleva a buenas practicas durante el desarrollo de una aplicación web. Apoyándonos con ITIL, CMMi, ISO27001, Six Sigma, entre otros como se menciona en el Capítulo II (Ver acápite 2.10.).

El modelo de seguridad en las aplicaciones web, puede ser aplicado en cualquier plataforma de desarrollo por lo que no está obligado usar alguna tecnología en especial, en el caso del prototipo de este trabajo se utilizó un conjunto de tecnologías, pero el modelo puede ser aplicado a otras tecnologías sin mayores dificultades.

Se debe dar la importancia a implementar un modelo de seguridad en las aplicaciones web dentro de las empresas, ya que existe una dependencia hacia la tecnología, si bien las amenazas, en cuanto a tecnologías de información, siempre van un paso más adelante que las soluciones, entonces se debe implementar las soluciones adecuadamente y aprovechar al máximo.

Por ende, existe la necesidad de crear una cultura sobre ello y así mismo, volverlo un hábito por medio de la auditoría en informática y la implementación de modelos de seguridad.

La aplicación del modelo de seguridad propuesto en este trabajo ayuda a las empresas a ser eficiente, y como consecuencia disminuye los costos económicos.

5.3. Sugerencias y recomendaciones

En base al contenido de la presente tesis.

- Construcción de herramientas de desarrollo, como Framework que permitan realizar aplicaciones web más seguras.
- Modelos de seguridad más flexibles y de fácil aplicabilidad para tecnologías de información.
- Estándares y normas de seguridad aplicables en el proceso de desarrollo de las aplicaciones web.
- Software de seguridad preventivo para las aplicaciones web.

Bibliografía

Libros

- [ALE07] Alexander, Alberto G (2007), "DISEÑO DE UN SISTEMA DE GESTIÓN DE SEGURIDAD DE INFORMACIÓN", Alfaomega.
- [CRIP06] Criptored, (2006), "SEGURIDAD INFORMÁTICA Y CRIPTOGRÁFICA".
- [DEVA14] Debrauwer Laurent - Van Der Heyde Fien, (2014), "INICIACIÓN, EJEMPLOS Y EJERCICIOS CORREGIDOS", 3ª edición, ENI.
- [GIM08] Giménez María Isabel, (2008). "UTILIZACIÓN DE SISTEMAS DE DETECCIÓN DE INTRUSOS COMO ELEMENTO DE SEGURIDAD PERIMETRAL".
- [JL08] J.L. Altero (2008), "SEGURIDAD EN LA INFORMACIÓN", Ediciones paraninfos, S.A.
- [QUE11] Quesnel Jacques, (2011), "NORMAS Y MEJORES PRÁCTICAS PARA AVANZAR HACIA ISO 20000", Nueva edición, ENI.
- [THEOPEN09] The Open Web Aplication Security Project, Mayo 2009. "UNA GUÍA PARA CONSTRUIR APLICACIONES Y SERVICIOS WEB SEGUROS", Segunda edición, Editorial Black Hat.
- [VVA13] VV.AA., (2013), "SEGURIDAD INFORMÁTICA, ETHICAL HACKING", Nueva edición, ENI.

Revistas y artículos

- [GON10] Gonzáles Gómez Diego, (2010), "SISTEMA DE DETECCION DE INTRUSOS", México.
- [LORE06] López Barrientos María Jaquelina y Quezada Reyes Cintia, (2006), "FUNDAMENTOS DE SEGURIDAD INFORMÁTICA", México, Universidad Nacional Autónoma de México, Facultad de Ingeniería.
- [MOR14] Morales González, (2014). "LA IMPORTANCIA DEL DESARROLLO PARA EL BUEN DISEÑO DEL SOFTWARE", Tecnológico de Estudios Superiores de Ecatepec.

- [VIL12] Villalobos Murillo Johnny, (2012). "PRINCIPIOS BASICOS DE SEGURIDAD EN BASES DE DATOS", Seguridad, Defensa Digital.

Enlaces web

- [DRAG11] Dragonjar, (2011), "NETWORK AND SYSTEM SECURITY", http://www.dragonjar.org/101-libros-de-seguridad-informatica-gratis.xhtml, editorial Sysgress, 18/04/15 (15:22:10)
- [IMH14] Instituto Máquina Herramienta, (2013). "FIREWALL O CORTAFUEGOS", http://www.imh.eus/es/comunicacion/dokumentazio-irekia/manuales/seguridad-basica-en-internet/firewall-cortafuegos/referencemanual-all-pages, http://www.imh.eus, 15/05/15 (06:24:11)
- [JAV10] Javajan, Webs for designers and you, (2010), "ARQUITECTURA DE UN SITIO WEB" http://www.guiadiseño.com/05_arquitectura.php, 20/05/15 (07:35:08)
- [KOOLE15] Dejan Kosutic, Oscar, Rhand Leal, (2015), " ISO 27001 STANDARD", http://www.iso27001standard.com/es/que-es-iso-27001, 20/05/15 (22:14:43)
- [KIO14] Kioskea.net, (2014). "SISTEMA DE DETECCIÓN DE INTRUSIONES", http://es.kioskea.net/contents/detection-171307462#162, Kioskea.net, 08/05/15 (11:32:06)
- [LOMAMOGL10] Iren Lorenzo-Fonseca, Francisco Maciá-Pérez, Francisco José Mora-Gimeno, Diego Marcos-Jorquera, Juan Antonio Gil-Martínez, Rogelio Lau-Fernández, (2010). "SISTEMA DE DETECCIÓN DE INTRUSIONES", https://www.dtic.ua.es/grupoM/recursos/articulos/JDARE-09-B.pdf, Centro de Estudio de Ingeniería y Sistema – Instituto Superior Politécnico José Antonio Echevarría, 12/05/15 (21:45:22)
- [MEA13] Rolling Meadows, (2013), "COBIT 5", http://www.isaca.org/About-ISACA/Press-room/News-Releases/Spanish/Pages/ISACA-tiene-ya-disponible-la-version-en-espanol-de-COBIT-5.aspx, http://www.isaca.org, 16/05/15 (23:21:38)

- [OWA14] owasp.org, (2010). "CROSS SITE SCRIPTING (XSS)", https://www.owasp.org/index.php/Cross-site_Scripting_%28XSS%29, owasp.org, 12/05/15 (22:27:48)
- [PFC11] PFC Marisa, (2011), "INTRODUCCIÓN A LOS IDS", http://www.adminso.es/index.php/Archivo:Pfc_Marisa_Capitulo1.pdf, 20/05/15 (23:24:13)
- [ROM14] MSc. Romaniz C. Susana, (2014). "SEGURIDAD DE APLICACIONES WEB: VULNERABILIDADES EN LOS CONTROLES DE ACCESO", http://sedici.unlp.edu.ar/bitstream/handle/10915/21581/1927+-+Seguridad+de+aplicaciones+web+vulnerabilidades+en+los+controles+de+acceso.pdf;jsessionid=8F904076D591AFBCFC689D7AF53F5169?sequence=1, Facultad Regional Santa Fe - Universidad Tecnológica Nacional, 02/05/15 (10:08:34)
- [SEIN15] Segu - Info, (2015). "SEGURIDAD INFORMÁTICA - IMPLICANCIAS E IMPLEMENTACIÓN", http://www.segu-info.com.ar/tesis/, http://www.segu-info.com.ar, 15/05/15 (07:30:42)
- [NCA13] ncalculators, (2013), "CALCULATORS STATISTICS", http://es.ncalculators.com/statistics/, 26/05/15 (21:45:12)

Tesis

- [ALV05] Álvarez Basaldúa Luis Daniel, (2005). "SEGURIDAD EN INFORMÁTICA (AUDITORIA EN SISTEMAS)", Universidad Iberoamericana, México.
- [TAH11] Tahuiton Mora Juan, (2011). "ARQUITECTURA PARA APLICACIONES WEB", Instituto Politécnico Nacional, México.
- [CHA13] Chana Quispe Alcides, (2013). "SOFTWARE DE SEGURIDAD PREVENTIVO PARA SERVIDORES WEB", UMSA, La Paz - Bolivia.

Anexo I: Tabla t de Student

Tabla distribución t de Student con N grados de libertad

n* \ *p	**0,60**	**0,70**	**0,75**	**0,80**	**0,85**	**0,90**	**0,95**	**0,99**	**0,995**
1	0,324 919	0,726 543	1,000 001	1,376 382	1,962 612	3,077 685	6,313 749	31,820 96	63,655 90
2	0,288 675	0,617 214	0,816 497	1,060 660	1,386 206	1,885 619	2,919 987	6,964 547	9,924 988
3	0,276 671	0,584 390	0,764 892	0,978 472	1,249 778	1,637 745	2,353 363	4,540 707	5,840 848
4	0,270 722	0,568 649	0,740 697	0,940 964	1,189 567	1,533 206	2,131 846	3,746 936	4,604 080
5	0,267 181	0,559 430	0,726 687	0,919 543	1,155 768	1,475 885	2,015 049	3,364 930	4,032 117
6	0,264 835	0,553 381	0,717 558	0,905 703	1,134 157	1,439 755	1,943 181	3,142 668	3,707 428
7	0,263 167	0,549 110	0,711 142	0,896 030	1,119 159	1,414 924	1,894 578	2,997 949	3,499 481
8	0,261 921	0,545 934	0,706 386	0,888 890	1,108 145	1,396 816	1,859 548	2,896 468	3,355 381
9	0,260 956	0,543 480	0,702 722	0,883 404	1,099 716	1,383 029	1,833 114	2,821 434	3,249 843
10	0,260 185	0,541 528	0,699 812	0,879 057	1,093 058	1,372 184	1,812 462	2,763 772	3,169 262
11	0,259 556	0,539 937	0,697 445	0,875 530	1,087 667	1,363 430	1,795 884	2,718 079	3,105 815
12	0,259 033	0,538 618	0,695 483	0,872 609	1,083 212	1,356 218	1,782 287	2,680 990	3,054 538
13	0,258 591	0,537 504	0,693 830	0,870 151	1,079 469	1,350 172	1,770 932	2,650 304	3,012 283
14	0,258 212	0,536 552	0,692 417	0,868 055	1,076 280	1,345 031	1,761 309	2,624 492	2,976 849
15	0,257 885	0,535 729	0,691 197	0,866 245	1,073 531	1,340 605	1,753 051	2,602 483	2,946 726
16	0,257 599	0,535 010	0,690 133	0,864 667	1,071 137	1,336 757	1,745 884	2,583 492	2,920 788
17	0,257 347	0,534 378	0,689 195	0,863 279	1,069 034	1,333 379	1,739 606	2,566 940	2,898 232
18	0,257 123	0,533 815	0,688 364	0,862 049	1,067 169	1,330 391	1,734 063	2,552 379	2,878 442
19	0,256 923	0,533 314	0,687 621	0,860 950	1,065 507	1,327 728	1,729 131	2,539 482	2,860 943
20	0,256 742	0,532 863	0,686 954	0,859 965	1,064 016	1,325 341	1,724 718	2,527 977	2,845 336
21	0,256 580	0,532 455	0,686 352	0,859 075	1,062 670	1,323 187	1,720 744	2,517 645	2,831 366
22	0,256 432	0,532 085	0,685 805	0,858 266	1,061 449	1,321 237	1,717 144	2,508 323	2,818 761
23	0,256 297	0,531 747	0,685 307	0,857 530	1,060 337	1,319 461	1,713 870	2,499 874	2,807 337
24	0,256 173	0,531 438	0,684 850	0,856 855	1,059 319	1,317 835	1,710 882	2,492 161	2,796 951
25	0,256 060	0,531 154	0,684 430	0,856 236	1,058 385	1,316 346	1,708 140	2,485 103	2,787 438
26	0,255 955	0,530 891	0,684 043	0,855 665	1,057 523	1,314 972	1,705 616	2,478 628	2,778 725
27	0,255 858	0,530 649	0,683 685	0,855 138	1,056 727	1,313 704	1,703 288	2,472 661	2,770 685
28	0,255 768	0,530 424	0,683 353	0,854 648	1,055 989	1,312 526	1,701 130	2,467 141	2,763 263
29	0,255 684	0,530 214	0,683 044	0,854 192	1,055 303	1,311 435	1,699 127	2,462 020	2,756 387
30	0,255 606	0,530 019	0,682 755	0,853 768	1,054 663	1,310 416	1,697 260	2,457 264	2,749 985
31	0,255 532	0,529 836	0,682 486	0,853 370	1,054 065	1,309 463	1,695 519	2,452 825	2,744 036
32	0,255 463	0,529 665	0,682 234	0,852 998	1,053 504	1,308 573	1,693 888	2,448 678	2,738 489
33	0,255 399	0,529 504	0,681 997	0,852 649	1,052 979	1,307 737	1,692 360	2,444 795	2,733 286
34	0,255 338	0,529 353	0,681 774	0,852 322	1,052 485	1,306 951	1,690 923	2,441 147	2,728 393