



Unidad 2 / Escenario 3

Lectura fundamental

# Seguridad en las etapas de análisis, diseño e implementación

## Contenido

- 1 Introducción
- 2 Seguridad en la etapa de análisis
- 3 Seguridad en la etapa de diseño
- 4 Seguridad en la etapa de implementación

**Palabras clave:** análisis, requerimientos, diseño, riesgos, modelamiento, amenazas, implementación, codificación.

## 1. Introducción

En la Unidad 1 comprendimos cuáles son los conceptos de básicos de seguridad, el modelo de desarrollo de software en cascada y diferentes modelos de seguridad en el ciclo de desarrollo. Con lo aprendido hasta el momento, tenemos las bases suficientes para iniciar una profundización en los temas de seguridad aplicados al desarrollo de software, especialmente utilizando como referencia el modelo de desarrollo en cascada.

En este sentido, la Unidad 2 nos presenta un detalle de las actividades de seguridad para cada una de las etapas de dicho modelo, en específico el escenario 3 nos permitirá aprender las actividades de seguridad para las primeras etapas del modelo: análisis, diseño e implementación.

Repasemos cuáles son las actividades de seguridad para las primeras 3 fases del ciclo de desarrollo de software según el modelo en cascada:

## 2. Seguridad en la etapa de análisis

Durante la etapa de análisis del modelo de desarrollo en cascada se definen los requerimientos, tanto funcionales como no funcionales. La seguridad no puede ser una excepción, por lo que en esta etapa se definen los requisitos de seguridad que debe satisfacer el desarrollo.

Desde la perspectiva de seguridad, existen dos tipos de requisitos que se deben o pueden definir, los relacionados con los conceptos básicos que aprendimos en el escenario 1 y/o los requisitos contractuales o legales de seguridad que le sean aplicables al desarrollo. En las siguientes secciones observaremos que son estos requerimientos y veremos ejemplos de los mismos.

### 2.1. Requerimientos de los conceptos básicos

- » Confidencialidad: Los requerimientos de confidencialidad en un desarrollo dependen de la clasificación que tenga la información que se manejará en el mismo. Generalmente las empresas definen diferentes niveles de confidencialidad para la información según su importancia para el negocio, un ejemplo de dicha clasificación se observa a continuación:
- Información Pública: información que está al alcance de cualquier persona y en general se puede decir que no tiene ningún tipo de restricción de acceso.

- Información Interna: información que solo debe ser accedida por todas las áreas y colaboradores internos de la organización.
- Información confidencial: información que solo debe ser accedida por algunas áreas de la organización.
- Información secreta: información que solo debe ser accedida por algunas personas de la organización.

De igual manera, los requerimientos de seguridad para confidencialidad deben definirse para los 3 posibles estados de la información:

- Información en tránsito.
- Información en procesamiento.
- Información almacenada.

Teniendo en cuenta estas definiciones, observemos algunos ejemplos de requerimientos de confidencialidad que pueden ser incluidos en el software:

- Las contraseñas no deben almacenarse en texto claro en la base de datos, es decir deben tener algún tipo de cifrado.
- La aplicación debe funcionar bajo el protocolo HTTPS para proteger los datos de tarjeta de crédito.
- Los datos con información clasificada como “secreta” deben ser enmascarados.

» Integridad: Los requerimientos de integridad se deben definir para:

- El software: es decir que cumpla con el objetivo para el cual está siendo diseñado.
- Los datos: el software debe proteger los datos con los que trabaja.

Teniendo en cuenta estas definiciones, observemos algunos ejemplos de requerimientos de integridad que pueden ser incluidos en el software:

- Todos los formularios de entrada deben ser validados antes de que el software los acepte como válidos para procesarlos.
- El instalador de la aplicación debe publicarse junto con su hash MD5 para que el usuario pueda comprobar su completitud y exactitud.
- El sistema debe comprobar que durante la transmisión la información no ha sido alterada.

- » Disponibilidad: Los requerimientos de disponibilidad dependen de los siguientes conceptos:
  - MTD: Maximum Tolerable Downtime, tiempo máximo que una organización tolera cuando el software no está disponible.
  - RTO: Recovery Time Objective, tiempo en el cual se debe recuperar el software.  
Los nuevos, es decir el tiempo porcentual de disponibilidad de la aplicación:
    - 99.9%: 43.2 minutos de indisponibilidad al mes.
    - 99.99%: 4.32 minutos de indisponibilidad al mes.
    - 99.999%: 25.9 segundos de indisponibilidad al mes.
    - 9.9999%: 2.59 segundos de indisponibilidad al mes.
- A mayor cantidad de nuevos, mayor es la inversión que se debe realizar en infraestructura tecnológica.
- Teniendo en cuenta estas definiciones, observemos algunos ejemplos de requerimientos de disponibilidad que pueden ser incluidos en el software:
  - El sistema debe permitir máximo 100 usuarios concurrentes.
  - El software y las bases de datos deben ser replicados en al menos 2 centros de datos diferentes.
  - El software debe asegurar una disponibilidad de 4 nuevos (99.99%).
- » Autenticación: Los requerimientos de autenticación se definen dependiendo de la cantidad de factores que se requieran incluir:
  - Único: algo que conoce o algo que sabe o algo que es.
  - Múltiple: algo que conoce y algo que sabe y algo que es, o al menos alguna combinación de 2 de ellos.

Teniendo en cuenta estas definiciones, observemos algunos ejemplos de requerimientos de autenticación que pueden ser incluidos en el software:

- La longitud de las contraseñas de acceso deben ser de al menos 10 caracteres alfanuméricos.
- El software debe soportar single-sign-on (SSO) con la aplicación de nómina.
- El acceso al módulo de pagos se realiza a través de token, previamente validado el usuario y contraseña.

» Autorización: Los requerimientos de autorización se definen para las necesidades CRUD:

- Create: crear
- Read: leer
- Update: actualizar
- Delete: borrar

Otro concepto importante a la hora de definir los requisitos de autorización es la ley del menor privilegio, la cual establece que los usuarios tienen permisos basados únicamente en sus necesidades de saber/conocer.

Teniendo en cuenta estas definiciones, observemos algunos ejemplos de requerimientos de autorización que pueden ser incluidos en el software:

- La aplicación debe permitir crear perfiles de acceso.
- Usuarios no autenticados tendrán perfil de solo lectura en la página expuesta al público.
- El acceso a calificar los trabajos solo se permite para los tutores.
- » No Repudio: Los requerimientos de no repudio se definen con base en las necesidades de trazabilidad en cuanto a:
  - Quién
  - Qué
  - Donde
  - Cuando

Teniendo en cuenta estas definiciones, observemos algunos ejemplos de requerimientos de no repudio que pueden ser incluidos en el software:

- Los intentos de autenticación fallidos deben registrarse en el log de auditoría.
- Todas las consultas a los datos financieros de los clientes deben registrar el usuario, la dirección IP, la fecha y la hora.
- La traza en el sistema permite identificar el precio anterior y en nuevo precio del producto.

¿Sabía qué...?



Uno de los ataques más utilizados por los criminales para acceder sin autorización a los datos es la inyección de código o SQLInjection.

## 2.2. Requerimientos legales y/o contractuales

Dependiendo de la situación, es posible que las aplicaciones deban cumplir con ciertos requisitos de seguridad establecidos por la ley de un país o por un convenio o acuerdo contractual. En todo caso, estos requisitos están enmarcados en los conceptos básicos de seguridad de la información.

### Ejemplos de requisitos legales:

- Circular 042 de 2012 (Superintendencia Financiera de Colombia, 2012), es una circular emitida por la Superintendencia Financiera de Colombia, la cual establece requisitos de seguridad, dentro de ellos varios aplicables al software.
- SOX: Ley Sarbanes-Oxley (107th United States Congress, 2002), aplicable a aquellas empresas que cotizan en la bolsa de USA. Dicha ley también incluye requisitos de seguridad, orientados a la protección de los inversionistas mediante el aseguramiento de la integridad de los estados financieros.
- Ley de protección de datos personales 1581 de 2012 (Congreso de Colombia, 2012): orientada a proteger la privacidad de los datos personales, lo cual incluye requisitos de seguridad para la protección de las bases de datos relacionadas.

Ejemplos de requisitos contractuales, es decir pueden hacer parte integral de los contratos cumplir con:

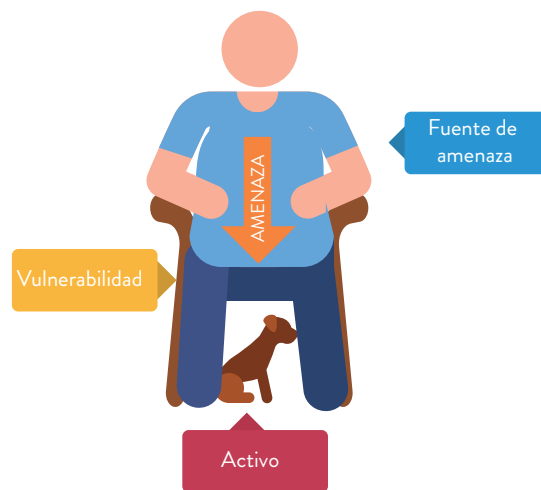
- PCI DSS (PCI Security Standards Council, 2016): es un estándar para la seguridad de los datos de tarjetas débito y crédito.
- ISO/IEC 27001 (International Organization for Standardization, 2013): sistemas de gestión de seguridad de la información, incluye un dominio dedicado a la seguridad de los sistemas de información.

### 3. Seguridad en la etapa de diseño

En la etapa de diseño, la actividad principal de seguridad a incluir es el análisis de riesgos, que en términos de desarrollo de software es conocido como modelamiento de amenazas. Observemos los principales conceptos relacionados:

- Activo de información: información que tiene valor para una organización.
- Amenaza: posibilidad de un evento no deseado, que puede resultar en daño a un sistema u organización.
- Vulnerabilidad: debilidad de un activo o de control que puede ser explotado por una o más amenazas.
- Riesgo: probabilidad de que amenazas exploten vulnerabilidades de un activo de información y por ende causa un daño (impacto) para la organización. Matemáticamente es la multiplicación de la probabilidad por el impacto.
- Tratamiento: actividades para llevar el riesgo a un nivel aceptable. Las posibles opciones para el tratamiento de riesgos son: aceptar, mitigar, transferir y evitar.

Observemos gráficamente un resumen de estos conceptos, donde el riesgo es que la mascota salga lastimada:



**Figura 1. Conceptos de riesgos**

Fuente: elaboración propia

Como se mencionó anteriormente, el riesgo es función de la probabilidad y el impacto. Observemos a continuación el significado de esos conceptos:

- » Probabilidad: que tan probable es que ocurra una amenaza, puede ser determinable según la cantidad de veces que ha sucedido ese evento en el pasado y/o según qué tan reproducible (R), explotable (E) y descubrible (DI) es la amenaza. Ejemplo basado en eventos:

**Tabla 1. Matriz ejemplo para la definición de probabilidad**

Valor cualitativo	Valor cuantitativo	Definición
Alto	3	Ocorre una vez por semana.
Medio	2	Ocorre una vez al mes.
Bajo	1	Ocurrió una vez en el año.

Fuente: elaboración propia

Impacto: daño que sufre el negocio, se puede medir en “drivers” de negocio como impacto económico, reputacional, legal, etc. Ejemplo basado en impacto económico:

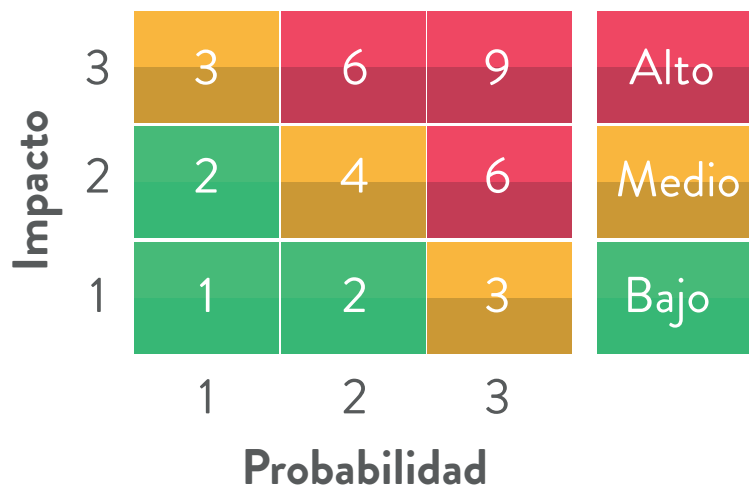
**Tabla 2. Matriz ejemplo para la definición de impacto**

Valor cualitativo	Valor cuantitativo	Definición
Alto	3	Se pierden más de 30 millones.
Medio	2	Se pierden entre 10 y 29 millones.
Bajo	1	Se pierden hasta 9 millones.

Fuente: elaboración propia

A partir de los valores de probabilidad e impacto, se calcula el riesgo mediante una multiplicación. Los valores de riesgo se pueden representar mediante un diagrama o mapa de calor. Observemos un ejemplo, donde se definieron 3 niveles de riesgo:






**Figura 2. Diagrama de calor**

Fuente: elaboración propia

Según lo que defina la empresa, se realiza el tratamiento de los riesgos. Un ejemplo puede ser que la empresa decida tratar los riesgos altos y medios, pero decida no tratar los riesgos bajos.

¿Sabía qué...?

---



La “atacabilidad” o que tan atacable es el software es conocido como la superficie de ataque y depende de la cantidad de entradas o salidas que tenga el software.

### Modelamiento de amenazas

En desarrollo de software, consiste en determinar cuáles son las amenazas que requieren mitigación y la forma de hacerlo, permitiendo de esta manera tomar decisiones sobre la identificación de los riesgos de una aplicación. El proceso para realizar modelado o modelamiento de amenazas es el siguiente:

- A. Paso 1: Diagramar la arquitectura de la aplicación.

Para esto debe tener en cuenta:

- Identificar las topologías físicas y lógicas
- Identificar flujos de datos
- Identificar los actores humanos y no humanos del sistema
- Identificar los límites de confianza (punto en el cual los niveles de confianza o privilegios cambian, por ejemplo existe un límite de confianza entre Internet y la red LAN)

B. Paso 2: Identificar amenazas.

Para la identificación de amenazas, se propone la utilización de la metodología STRIDE:

**Tabla 3. Ejemplo de metodología STRIDE**

Categoría	Concepto	Definición	Ejemplos
Spoofing	Autenticación	Suplantación de algo o alguien.	Pretender ser Bill gates, Microsoft.com o ntdll.dll
Tampering	Integridad	Manipulación de datos.	Modificar una DLL o un DVD o los paquetes que pasan por una red LAN.
Repudiation	No Repudio	Pretender negar haber realizado una acción.	“Yo no envié ese correo”, “Yo no modifiqué ese archivo”, “Yo no visité ese sitio web”
Information Disclosure	Confidencialidad	Divulgación o fuga de información.	Permitir a alguien leer el código fuente de Windows, publicar una lista de clients de un sitio web.
Denial of Service	Disponibilidad	Denegación de servicio.	Provocar la caída de la red, reducir dramáticamente los recursos del servidor, realizar una consulta mal hecha a una base de datos.
Elevation of Privilege	Autorización	Elevación de privilegios.	Un usuario normal se vuelve administrador, permitir que un usuario de una aplicación pueda ejecutar comandos en el servidor.

Fuente: elaboración propia

Otra técnica muy utilizada son los casos de abuso de software.

C. Paso 3: Priorizar e implementar controles.

La priorización se puede realizar mediante el cálculo del riesgo, como se observa a continuación:

**Tabla 4. Priorización**

Categoría	Probabilidad	Impacto	Riesgo
Spoofing	2	3	6
Tampering	1	3	3
Repudiation	3	3	9
Information Disclosure	3	1	3
Denial of service	1	1	1
Elevation of privilege	1	2	2

Fuente: elaboración propia

En el ejemplo anterior, “Repudiation” es la prioridad ya que representa el mayor riesgo con un valor de 9.

D. Paso 4: Documentar.

Cada amenaza debe quedar debidamente documentada, incluyendo al menos:

- ID de la amenaza
- Descripción de la amenaza
- Objetivos de la amenaza
- Técnicas de ataque
- Impacto
- Riesgo
- Controles

## 4. Seguridad en la etapa de implementación

### 4.1. Codificación Segura

La codificación segura consiste en una serie de prácticas orientadas a prevenir el éxito de los ataques más comunes realizados contra las aplicaciones. A continuación observaremos las diferentes técnicas:

- » Validación de entradas: Se deben verificar todas las entradas de la aplicación con el objetivo de garantizar:
  - Que el tipo y formato del dato sea correcto.
  - Que el dato caiga dentro del rango esperado y permitido de valores.
  - Que no se interprete como código en caso de que sea una inyección.

Para tal fin, se pueden crear listas:

- Blancas: por ejemplo permitir el caracter @ y .com en el campo de correo electrónico.
- Negras: no permitir ciertas entradas, por ejemplo patrones como 1=1, comentarios SQL (--) o comilla simple (').
- » Canonicalización: Consiste en convertir data que tiene más de una posible representación en una forma estándar o forma más simple. Por ejemplo, la forma canónica para `http://pagina.com`, `http://www%2epagina%2ecom` y `http://200.100.100.1` es `http://www.pagina.com`.
- » Sanitización: Proceso de convertir algo que es considerado peligroso en su forma inocua. Veamos 2 técnicas utilizadas:
  - Stripping (“pelar”): eliminar caracteres potencialmente peligrosos como >, <, /, (, ), etc. Por ejemplo si se recibe un ataque XSS mediante la cadena `<script>alert('Ataque XSS');</script>`, mediante el stripping quedaría `scriptalertAtaque XSSscript`, lo cual es inofensivo.
  - Substitution: sustituir caracteres potencialmente peligrosos por algo que no lo sea. Por ejemplo ante una inyección SQL como `'Or 1=1 --`, se substituye la comilla simple por la doble, quedando una cadena de caracteres inofensiva `"Or 1=1 --`.
- » Manejo de errores y excepciones: Los mensajes de error son las primeras fuentes de información que un atacante busca en un software. Los mensajes de error de la aplicación no deben ser detallados, deben entregar la menor cantidad de información posible:

- Ante una autenticación fallida, responder “Usuario o Password incorrecto”
- Ante un recurso inexistente no revelar información útil para el atacante, como la versión del servidor web.
- » APIs seguras: Las API (Application Programming Interfaces) utilizadas se deben verificar antes de utilizarlas, pues muchas de ellas contienen vulnerabilidades que pueden poner en riesgo la aplicación.
- » Gestión de Sesiones: Las sesiones se deben gestionar adecuadamente para evitar ataques que permitan suplantación, como los ataques Man-In-The-Middle. Se deben seguir las siguientes recomendaciones:
  - Todas las sesiones deben manejar tokens de sesión únicos por usuario.
  - Cifre el tráfico de la aplicación para que no se le pueda hacer sniffing a los datos de sesión.
  - Las sesiones deben tener un tiempo de vida definido.
  - Cerrar el navegador no debe ser requisito para finalizar una sesión.
- » Concurrencia: El software debe controlar la cantidad de usuarios concurrentes para evitar problemas de disponibilidad.
- » Criptografía: Para proteger la confidencialidad de los datos, se hace necesario aplicar criptografía dependiendo de estado de los mismos:
  - En movimiento: utilización de protocolos seguros como SSL, HTTPS, SFTP, SSH
  - Almacenados: utilización de algoritmos de cómo 3DES o AES en sus versiones más actualizadas.

## 4.2. Revisión de Código

La revisión de código tiene por objetivo inspeccionar si existen vulnerabilidades que puedan ser explotadas.

- » Revisión de Código Estático:
  - Inspección del código sin estar siendo ejecutado
  - Puede ser manual o automatizado.
  - Tipos de escáner:

- Analizadores de código fuente
- Analizadores binarios (código objeto)
- Los analizadores buscan patrones en el código para cruzarlo contra un listado de vulnerabilidades conocidas.
- El análisis estático no requiere de un ambiente productivo de la aplicación.
- Se deben analizar los resultados para descartar falsos positivos.
- » Revisión de Código Dinámico:
  - Inspección del código cuando está siendo ejecutado
  - Busca garantizar que el software:
    - Funciona como se diseñó
    - No es propenso a errores
    - No es propenso a ser explotado
  - Los escáneres de este tipo de análisis son conocidos como analizadores de código dinámico.
  - El análisis dinámico requiere de un ambiente productivo de la aplicación o un espejo del mismo.
  - Se deben analizar los resultados para descartar falsos positivos.

¿Sabía qué...?



La revisión de código es una forma temprana de identificar que la codificación es realmente segura.

## En síntesis...

Las tres primeras etapas de seguridad en el ciclo de desarrollo y sus actividades permiten definir e implementar seguridad a la aplicación.



# Referencias bibliográficas

Superintendencia Financiera de Colombia. (2012). *Circular Externa*

042 de 2012. Recuperado de [https://m.superfinanciera.gov.co/descargas?com=institucional&name=pubFile31164&downloadname=ce042\\_12.doc](https://m.superfinanciera.gov.co/descargas?com=institucional&name=pubFile31164&downloadname=ce042_12.doc).

107th United States Congress. (2002). *Sarbanes–Oxley Act of 2002*. Recuperado de <https://www.gpo.gov/fdsys/pkg/PLAW-107publ204/pdf/PLAW-107publ204.pdf>.

Congreso de Colombia. (2012). *ley estatutaria 1581 de 2012*. Recuperado de [http://www.secretariasenado.gov.co/senado/basedoc/ley\\_1581\\_2012.html](http://www.secretariasenado.gov.co/senado/basedoc/ley_1581_2012.html).

PCI Security Standards Council. (2016). *Requisitos y procedimientos de evaluación de seguridad*. Recuperado de [https://www.pcisecuritystandards.org/documents/PCI\\_DSS\\_v3-2\\_3\\_es-LA.pdf?agreement=true&time=1512747614564](https://www.pcisecuritystandards.org/documents/PCI_DSS_v3-2_3_es-LA.pdf?agreement=true&time=1512747614564).

International Organization for Standardization. (2013). *Sistemas de Gestión de Seguridad de la Información*. Recuperado de <https://www.iso.org/obp/ui/#iso:std:iso-iec:27001:ed-2:v1:en>.



## INFORMACIÓN TÉCNICA



FACULTAD DE  
**INGENIERÍA, DISEÑO  
E INNOVACIÓN**

**Módulo:** Seguridad en el Ciclo de Desarrollo

**Unidad 2:** Etapas de seguridad en el ciclo de desarrollo de software

**Escenario 3:** Seguridad en las etapas de análisis, diseño e implementación

**Autor:** Miguel Ángel Zambrano Puentes

**Asesor Pedagógico:** Edwin Mojica Quintero

**Diseñador Gráfico:** Brandon Steven Ramírez Carrero

**Asistente:** Ginna Quiroga

*Este material pertenece al Politécnico Gran Colombiano. Por ende, es de uso exclusivo de las Instituciones adscritas a la Red Ilumino. Prohibida su reproducción total o parcial.*