

Metodologías, normas y estándares de seguridad orientada a aplicaciones web

Breve descripción:

En este componente se presentan los fundamentos necesarios para el desarrollo de una auditoría de la seguridad en aplicaciones web, mediante el reconocimiento de estándares, metodologías, técnicas y herramientas necesarias para la evaluación de la seguridad.

Tabla de contenido

Introducción	1
1. Fundamentos de seguridad en aplicaciones web	3
1.1. Conceptos	3
1.2. Infraestructura de aplicaciones web	6
1.3. Normatividad y estándares para la gestión de la seguridad	9
1.4. Metodologías para la gestión de la seguridad en aplicaciones web	20
2. Vulnerabilidades en aplicaciones web	23
2.1. Metodologías para la gestión de vulnerabilidades	23
2.2. Fundamentos de “pentesting”	25
2.3. OWASP - top 10 de vulnerabilidades	27
2.4. Herramientas especializadas	29
2.5. Análisis de resultados	34
Síntesis	38
Material complementario	39
Glosario	40
Referencias bibliográficas	41
Créditos	42

Introducción

Le damos la bienvenida al componente formativo denominado Metodologías, normas y estándares de seguridad orientada a aplicaciones web, el cual hace parte del programa de formación técnico en “Seguridad de aplicaciones web”, para lo cual se invita a observar el siguiente video:

Video 1. Metodologías, normas y estándares de seguridad orientada a aplicaciones web.



[Enlace de reproducción del video](#)

Síntesis del video: Metodologías, normas y estándares de seguridad orientada a aplicaciones web

Metodologías, normas y estándares de seguridad orientada a aplicaciones web: en los últimos años se ha presentado un factor que ha impulsado la adopción de soluciones tecnológicas, en su mayoría de tipo web. Factores políticos, sociales e incluso de salud, como la época de pandemia, obligó a las organizaciones a cambiar sus escenarios y la forma como atienden a sus clientes, proporcionando soluciones web que estén disponibles en cualquier lugar, hora y que puedan ser visualizadas desde cualquier dispositivo móvil.

Aquí es donde toman fuerza este tipo de aplicaciones. En muchas ocasiones se realiza su implementación sin contar con las mínimas condiciones de seguridad, y que una organización puede verse afectada por un incidente que puede generar pérdida, alteración o incluso robo de la información, poniendo en riesgo la confidencialidad de sus clientes y de su información.

Para no enfrentar estas vulnerabilidades que se encuentran presentes en las aplicaciones web, han surgido iniciativas que buscan reducir las brechas a la seguridad de la información. A partir de la adopción de buenas prácticas y de un modelo de evaluación permanente que permita identificar los riesgos a los que se enfrenta una aplicación web y la información que a través de ella se gestiona.

A continuación, vamos a reconocer algunas iniciativas normativas y metodológicas para la gestión de las vulnerabilidades en aplicaciones web. Bienvenidos.

1. Fundamentos de seguridad en aplicaciones web

La seguridad de la información en las organizaciones presenta una responsabilidad compartida dada por algunas normas, estándares, implementaciones de cumplimiento legal y algunas iniciativas de buenas prácticas que conllevan a generar una cultura de seguridad en los datos gestionados.

Ahora bien, existe un gran número de este tipo de iniciativas enfocadas a la seguridad perimetral, de procesos, de infraestructura, y no se le da la correspondiente importancia al desarrollo de aplicaciones las cuales son la ventana o interfaz que los usuarios o clientes van a utilizar para acceder y gestionar la información.

De acuerdo con lo anterior, a continuación, se reconocerán algunas iniciativas que han surgido y que son muy utilizadas dentro de la seguridad de aplicaciones web.

1.1. Conceptos

Dentro del ejercicio de la gestión de las vulnerabilidades en aplicaciones web se hará uso de metodologías, técnicas y herramientas especializadas, las cuales presentan términos que no son muy comunes y que es necesario reconocer para dar una adecuada interpretación de la información, así como transmitir los resultados obtenidos.

Se invita a conocer cuáles son los principales conceptos de vulnerabilidades en aplicaciones web:

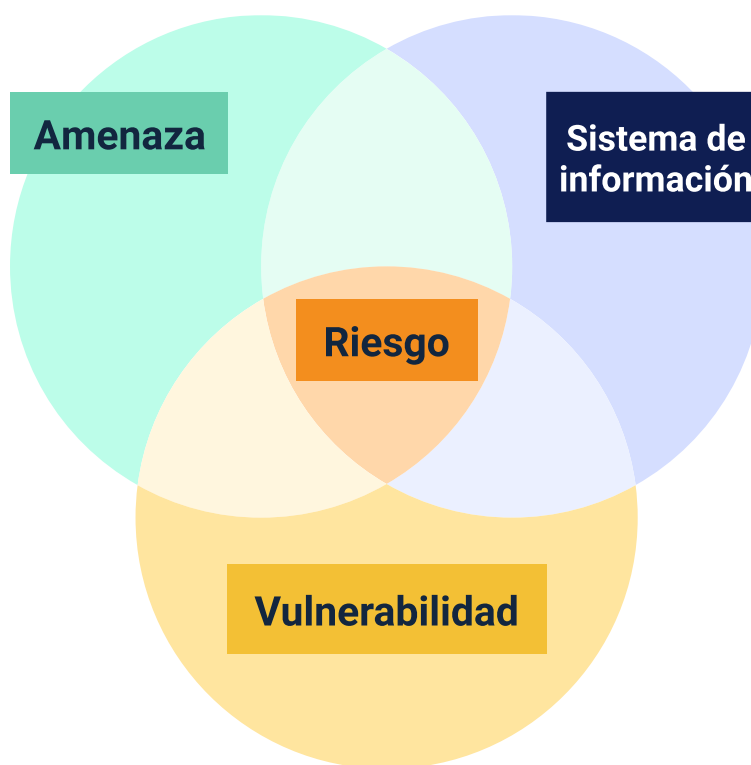
- **Vulnerabilidad.** Es una debilidad o fallo en un sistema de información que pone en riesgo la seguridad de la información.
- **Amenaza.** Es toda acción que aprovecha una vulnerabilidad para atentar contra la seguridad de un sistema de información.

- **Riesgo.** Es la probabilidad de que se produzca un incidente de seguridad, materializándose una amenaza y causando pérdidas o daños.

Los anteriores conceptos deben ser comprendidos adecuadamente, pues la presencia de una vulnerabilidad no significa que esté afectando la información, como se observa en la siguiente figura, el riesgo resulta del aprovechamiento de una vulnerabilidad y una amenaza materializada

En el siguiente recurso, se ofrece información en detalle de la relación de otros conceptos, se invita a consultarlo:

Figura 1. Relación de conceptos.



A continuación, se podrán conocer otros conceptos importantes:

- **“Bug”**. En informática se dice que hay un “bug” cuando se produce un error o fallo en un programa que tiene como consecuencia un resultado o comportamiento inesperado.
- **“Patch”**. También conocido como parche, es una pieza de programa adicional que corrige un “bug” o adiciona funcionalidades en una aplicación.
- **“Testing”**. También conocidas como pruebas, son un conjunto de técnicas y pruebas utilizadas dentro de la auditoría de aplicaciones web para verificar si estas son vulnerables o no ante dicha amenaza.
- **Medida**. Corresponde a las mediciones realizadas en una prueba de auditoría, la cual es útil para tomar decisiones.
- **Métrica**. Son instrumentos que nos permiten interpretar una medición frente a una escala, es decir, a partir de una lectura permite identificar el estado actual del objeto evaluado.

Figura 2. Uso de las medidas, métricas y KPI



Estos conceptos permiten comprender mejor el ejercicio realizado, interpretar algunos resultados obtenidos, así como generar informes técnicos con datos coherentes y que pueden demostrar el estado actual de la seguridad de una aplicación web.

1.2. Infraestructura de aplicaciones web

Las aplicaciones web han ganado terreno en la adopción e implementación de soluciones para las organizaciones, dada su flexibilidad, compatibilidad, capacidades de gestionar información y sobre todo su rapidez para la implementación dado que requiere de una infraestructura centralizada, esto conlleva a facilitar también su mantenimiento; y, por otra parte, a los usuarios finales les significa accederlas desde cualquier dispositivo haciendo uso de un navegador web y estar conectados a una red local o internet.

En el siguiente recurso se presentan algunos de los componentes que integran una infraestructura de aplicaciones web, lo cual servirá para comprender mejor su funcionamiento:

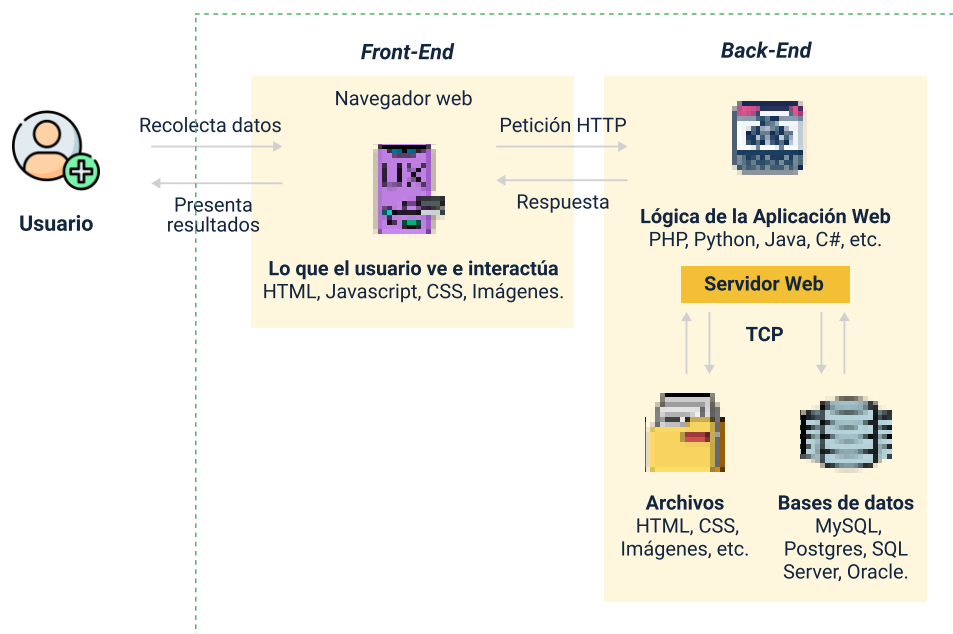
- **Usuario.** Es la persona u otra aplicación que realiza la solicitud de acceso a un recurso web, de forma habitual en diferentes situaciones, diferenciando los aspectos entre el servicio, el producto y la persona.
- **Dispositivo de acceso.** Es el dispositivo físico que utiliza el usuario para conectarse a una red y hacer uso de un navegador, a través de computadores, portátiles, tabletas y dispositivos móviles, entre otros.
- **Navegador web.** Son programas que permiten navegar dentro de una red como lo es internet y visualizar recursos enriquecidos con texto, imágenes, videos,

audio, hoy en día se cuenta con gran variedad de navegadores como, por ejemplo: Edge, Firefox, Chrome, Opera, Safari, Brave, entre otros.

- **Servidor web.** Es un programa especializado que se instala en un dispositivo “hardware” o virtual y que cuenta con recursos de procesamiento, almacenamiento, de tráfico y sirve para ofrecer alojamiento de recursos web.
- **Lenguajes de programación.** Estos permiten programar la lógica de las aplicaciones web bajo el uso de lenguajes compilados o lenguajes intérpretes, dado que el lenguaje HTML no es de alto nivel y no permite programación a un nivel superior, se debe hacer uso de estos lenguajes para generar la lógica de negocio y comportamiento que debe de seguir una aplicación web.
- **Motor de base de datos.** Este también es un programa que se aloja en un dispositivo “hardware” o virtual, y su función principal es la de almacenar la información de forma sistematizada, en estructuras organizadas como bases de datos, tablas y registros.

Todos estos componentes se interrelacionan para conformar las arquitecturas técnicas de aplicaciones web, en la siguiente figura, se aprecia su interrelación y flujo de información ante una solicitud de un recurso web.

Figura 3. Arquitectura de una aplicación web



- a. El usuario realiza la solicitud de un recurso web a través de un navegador.
- b. El navegador realiza la petición del recurso web a través del protocolo http.
- c. El servidor web recibe la petición y procesa la respuesta:
 - Procesa lógica de negocio.
 - Procesa información desde bases de datos.
 - Conformar la respuesta.
- d. El servidor devuelve la respuesta hacia el navegador.
- e. El navegador despliega resultado al usuario.

Actualmente existen diferentes tipos de aplicaciones web, entre las más comunes se resaltan:

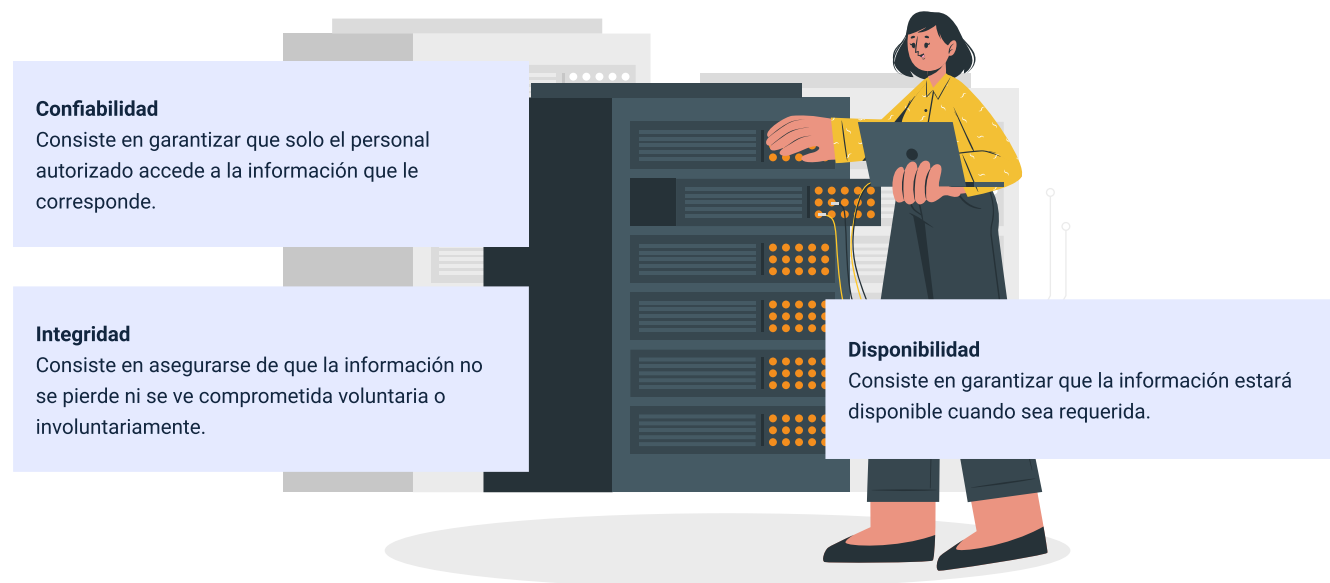
- **Hogar inteligente.** Termostatos domésticos, control de televisores, detectores de humo, cámaras de seguridad, cafeteras y otros electrodomésticos.
- **Fuera del hogar.** Automóviles en red, prendas tecnológicas y monitores de salud en la ropa, aplicaciones para ciclistas para controlar su actividad física, gafas inteligentes.
- **Ropa tecnológica (“wearables”).** Dispositivos médicos, productos de salud, marcapasos y bombas de insulina. Estos “wearables” monitorean la geolocalización, así como la frecuencia cardíaca, el pulso, calorías, patrones del sueño, etc.
- **Transporte.** Supervisión de vehículos conectados, sensores de proximidad, parqueo de vehículos, vehículos autónomos, control remoto de peajes.
- **Contadores inteligentes.** Dispositivos electrónicos de medición de energía, calefacción, climatización, agua.

Comprender la arquitectura técnica que sirven las aplicaciones web, así como sus tipos, permitirá comprender el funcionamiento y determinar un ejercicio que facilite el aseguramiento de las aplicaciones web.

1.3. Normatividad y estándares para la gestión de la seguridad

La gestión de la seguridad en las aplicaciones web es una tarea que actualmente no se le presta la debida atención, y esta no depende únicamente de las buenas prácticas de programación de “software”, en su lugar es importante tener presente que la responsabilidad de poseer una aplicación web pública, que gestione información

importante para la organización o maneje información confidencial, y ello obliga a contar con ciertos controles de seguridad que garantice los pilares de la seguridad de la información, a saber:



La imagen presenta tres conceptos clave:

- Confiabilidad, que consiste en garantizar que solo el personal autorizado accede a la información que le corresponde.
- Integridad, que se enfoca en asegurarse de que la información no se pierda ni se vea comprometida, ya sea de manera voluntaria o involuntaria.
- Disponibilidad, consiste en garantizar que la información estará disponible cuando sea requerida.

Ahora bien, para cumplir con estos tres pilares se hace necesario de acciones, y aquí es donde han surgido algunas buenas iniciativas como normatividad, estándares y buenas prácticas para una adecuada gestión de la información, entre las cuales poder revisar:

Norma ISO/IEC 27001:2013:

Ahora bien, para cumplir con estos tres pilares se hace necesario de acciones, y aquí es donde han surgido algunas buenas iniciativas como normatividad, estándares y buenas prácticas para una adecuada gestión de la información, entre las cuales poder revisar:

- Concientización de la necesidad de adoptar una cultura de seguridad.
- Delegación de responsabilidades sobre la seguridad de la información.
- Involucrar la alta dirección en la estrategia de seguridad.
- Seleccionar los controles adecuados para garantizar la seguridad de la información.
- Adoptar la seguridad como elemento esencial de los sistemas de información.
- Adoptar estrategias activas para la prevención de incidentes de seguridad activa.
- Adoptar la mejora continua en la evaluación de la seguridad.

En cuanto a la metodología para la adopción de esta norma, se incorpora el ciclo PDCA, acrónimo del inglés “Plan-Do-Check-Act” (en español: PHVA, planear, actuar verificar y actuar) la cual incorpora los siguientes ítems como se puede apreciar en la siguiente figura.

Figura 4. Fases de un SGSI



- **Planear.**

- a) Diseñar SGSI.
- b) Análisis de procesos.
- c) Definir Alcance.
- d) Elaborar política de seguridad.
- e) Identificar y evaluar inventario de activos.
- f) Realizar análisis de riesgos.
- g) Generar SoA.

- **Hacer.**
 - a) Generar plan de mitigación de riesgos.
 - b) Aplicar plan de mitigación de riesgos.
 - c) Implementar controles seleccionados.
 - d) Administración de cambio.
- **Verificar.**
 - a) Revisiones gerenciales.
 - b) Revisiones independientes.
 - c) Auditorías internas.
 - d) Revisiones técnicas.
- **Actuar.**
 - a) Implementar mejoras.
 - b) Tomar acciones preventivas y correctivas.
 - c) Comunicar resultados de las acciones tomadas.

Ahora bien, este estándar es base para la adopción de un SGSI, lo que conlleva a que, si una organización está certificada o está pensando en certificar sus procesos bajo esta norma, debe tener presente la seguridad de la información que se gestiona a través de aplicaciones web, de acuerdo con las siguientes recomendaciones:

- Los elementos y componentes de las aplicaciones web deberán ser objeto de análisis, planificación y evaluación para que sean cubiertos por el SGSI, esto involucra que también se le haga un análisis de riesgos.
- Dado el cambio tecnológico continuo, se requiere que las aplicaciones web sean evaluadas permanentemente según los nuevos retos, tecnologías y amenazas.

Norma ISO/IEC 27002:2013

Este estándar establece una guía para la implementación de controles de seguridad en una organización, a partir de sus objetivos, controles y estrategias de implementación.

Este estándar está compuesto por 14 dominios, 35 objetivos de control y 114 controles, con los cuales se busca adoptar medidas de seguridad en una organización.

A continuación, se exponen algunos de los controles asociados directamente con la seguridad de la información en aplicaciones web:

- Documento con política de seguridad de la información.
- Asignación de responsabilidades relativas a la seguridad.
- Formación y concienciación sobre la seguridad de la información.
- Notificación de los eventos de seguridad de la información.
- Gestión de los derechos de propiedad intelectual.
- Protección de activos de documentación de la organización.
- Protección de datos y privacidad de la información de carácter sensible de índole personal.

Desde el punto de vista de aporte de este estándar en la seguridad web, se pueden referenciar algunas medidas importantes que ayudan a gestionar la seguridad de los datos en las aplicaciones web, así:

- **Control de acceso al código fuente.** Se debe gestionar la seguridad del código fuente, las personas que acceden a partir de medidas de protección.

- **Procedimientos de control de cambios.** Cualquier cambio realizado en un sistema puede afectarlo negativamente, se sugieren una serie de recomendaciones para la gestión del control de cambios.
Para conocer una serie de recomendaciones para la gestión del control de cambios, haz clic en el botón
 - a. Mantener un control centralizado de cambios, con control de autorizaciones de actualización.
 - b. Cualquier cambio deberá ser evaluado y aprobado antes de ser incluido oficialmente, además, todo cambio deberá ser motivado por una necesidad justificada.
 - c. Todo cambio deberá de realizarse con el menor trastorno y efecto para la organización.
 - d. Deberá contarse con un plan de retroceso ante problemas causados por los cambios.
- **Revisión técnica tras cambios en las plataformas.** Todo sistema deberá ser evaluado después de la aplicación de los cambios, para verificar que no existan fallas.
- **Fugas de información.** Se deben evitar que sucedan fugas de información a partir de la inadecuada aplicación de cambios en los sistemas de información, es necesario adoptar los controles para la detección de fugas.
- **Desarrollo externo de aplicaciones.** Si la organización considera el desarrollo por terceros o externos, se deberá velar porque cumpla con las condiciones y requisitos de seguridad y teniendo en cuenta las condiciones de propiedad intelectual, calidad del desarrollo, certificaciones, entre otros.

- **Gestión de vulnerabilidades.** Se deben adoptar mecanismos y planes para la gestión de vulnerabilidades en cualquier desarrollo o cambio realizado. Haz clic en el botón Ver más, para conocer qué se debe hacer antes de ser puesto en producción.
 - a. Determinar las vulnerabilidades de mayor impacto para ser atendidas con prioridad.
 - b. Establecer los roles y asignar responsabilidades para la gestión de la seguridad.
 - c. Validar las vulnerabilidades identificadas con mecanismos o servicios centralizados y actualizados.
 - d. Cualquier parche generado para solucionar posibles vulnerabilidades deberá ser probado en entornos de prueba antes que en producción para identificar problemas, compatibilidad y evitar fallas generadas por dicha actualización.

Norma ISO/IEC 25010:2013

Este estándar aplica para la industria del desarrollo de “software”, así como para las organizaciones que realizan desarrollos propios para gestionar su información, es así como esta norma busca mejorar la calidad en los productos de desarrollo de “software” a partir de las siguientes características:

- **Adecuación funcional del “software”.** El sistema satisface las necesidades declaradas cuando se utiliza en condiciones específicas.
- **Eficiencia del desempeño del “software”.** También conocido como performance, se debe evaluar el producto “software” ante diferentes

escenarios, por ejemplo: volumen, carga, estrés, escalabilidad y rendimiento.

- **Compatibilidad.** Necesidad del producto final para funcionar ante diferentes características de “software” y “hardware”.
- **Usabilidad.** Donde entra en juego la interfaz y experiencia de usuario para aprender a usar rápidamente el “software”.
- **Confiabilidad.** Se basa en evaluar la capacidad del sistema para trabajar sin fallas durante un periodo de tiempo específico en un entorno específico.
- **Seguridad.** Nos ayuda a analizar y verificar la seguridad de nuestro producto “software” mediante diferentes pruebas como: pruebas de penetración, vulnerabilidad, “ethical hacking” o “static analysis”.
- **Mantenibilidad.** El sistema debe ser fácil de mantener, analizarlo, cambiarlo y probarlo.
- **Portabilidad.** Esto se refiere a la capacidad que tiene el producto “software” para cambiar entre versiones de un sistema operativo a otro sin mayores complicaciones.

Esto fortalece los procesos de desarrollo de “software” dentro las organizaciones y en los procesos de desarrollo en sitio en las organizaciones, que buscan mejorar los niveles de seguridad.

Para reforzar esta temática, se sugiere observar el siguiente video.

Video 2. Norma ISO



[Enlace de reproducción del video](#)

Síntesis del video: Norma ISO

Norma ISO: la información se ha convertido en uno de los activos de mayor valor para las organizaciones hoy en día. Gran parte de esta información es gestionada a través de las aplicaciones, en su mayoría de tipo web. Es donde se ha identificado que actualmente no se cuenta con los niveles de seguridad robusto que garanticen la seguridad de los datos que a través de ella circulan. A partir de estas fallas, han surgido iniciativas interesantes para la reducción de brechas que afectan la seguridad de la información. Vemos iniciativas de normas y estándares como los de la familia ISO 27001 y su anexo, que equivale a la Norma ISO 27002. Las cuales nos

dictan los lineamientos para la adopción de sistemas de gestión de seguridad de la información y que abarcan directamente la seguridad de los datos, desde la tecnología, las personas y los procesos.

Asimismo, encontramos la norma ISO 25010, la cual está enfocada en la gestión de la calidad de productos “software”. La traemos a colación dado que gran parte de la responsabilidad de la seguridad de las aplicaciones recae sobre el sector del desarrollo de “software”. Encontramos también algunas metodologías y resaltamos OWASP la cual es una metodología para la auditoría y revisión técnica de la seguridad propia de este tipo de aplicaciones. Se ha convertido en una de las más utilizadas, dada su compatibilidad, actualización y practicidad en su adopción.

Aunque esta metodología busca también que el factor de seguridad esté presente en todas las etapas de los ciclos de vida del desarrollo de “software”, no es una obligación. Su plan de pruebas puede ser aplicado a cualquier proyecto tipo web. Esta iniciativa de OWASP produce también informes periódicos en los que se pueden apreciar cuáles son las vulnerabilidades más presentes. Lo podemos encontrar en las publicaciones de OWASP top 10. Finalmente, el plan de pruebas que nos sugiere esta metodología es un compendio de validaciones mínimas que, apoyadas en el uso de herramientas especializadas, nos permiten identificar si una aplicación cuenta con vulnerabilidades presentes; permitiendo establecer un nivel de riesgo y tomar acciones para reducir la posibilidad de que se materialice y ocasione algún daño o pérdida en la información. Los invitamos a continuar con la lectura y desarrollo de las actividades propuestas. Bienvenidos.

1.4. Metodologías para la gestión de la seguridad en aplicaciones web

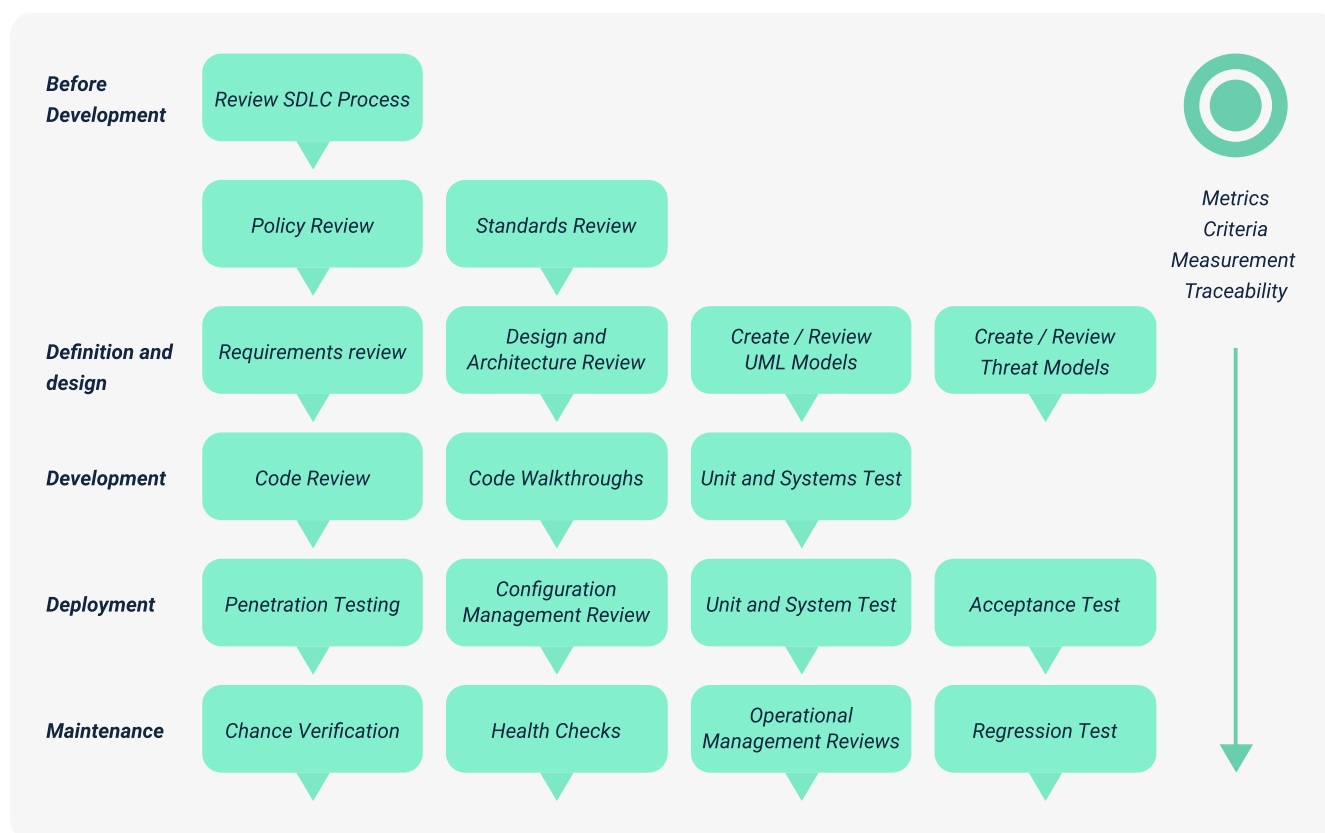
Existen normas y estándares que orientan en la adopción de iniciativas de seguridad en las organizaciones y que impactan directamente en la seguridad de las aplicaciones web y en la información gestionada a través de ellos, ahora bien, desde el punto de vista práctico existen iniciativas metodológicas para la gestión de las vulnerabilidades en aplicaciones web específicamente, una de ellas es OWASP.

OWASP (“Open Web Application Security Project”) es una metodología de seguridad para la auditoría web, orientada al análisis de seguridad de estas aplicaciones. Esta metodología está siendo utilizada en la mayoría de los trabajos de auditoría de seguridad, ya que permite a partir de la auditoría realizar una evaluación de riesgos.

Parte de la revisión de los controles definidos por la metodología, la cual permite al auditor garantizar que una revisión de la plataforma se realiza de forma adecuada, garantizando que todos los vectores de ataque han sido analizados y que los fallos de seguridad han sido detectados.

Este proceso ayuda a mejorar la seguridad y la protección de los sistemas informáticos en las organizaciones.

Figura 5. Flujo de trabajo de prueba SDLC genérico

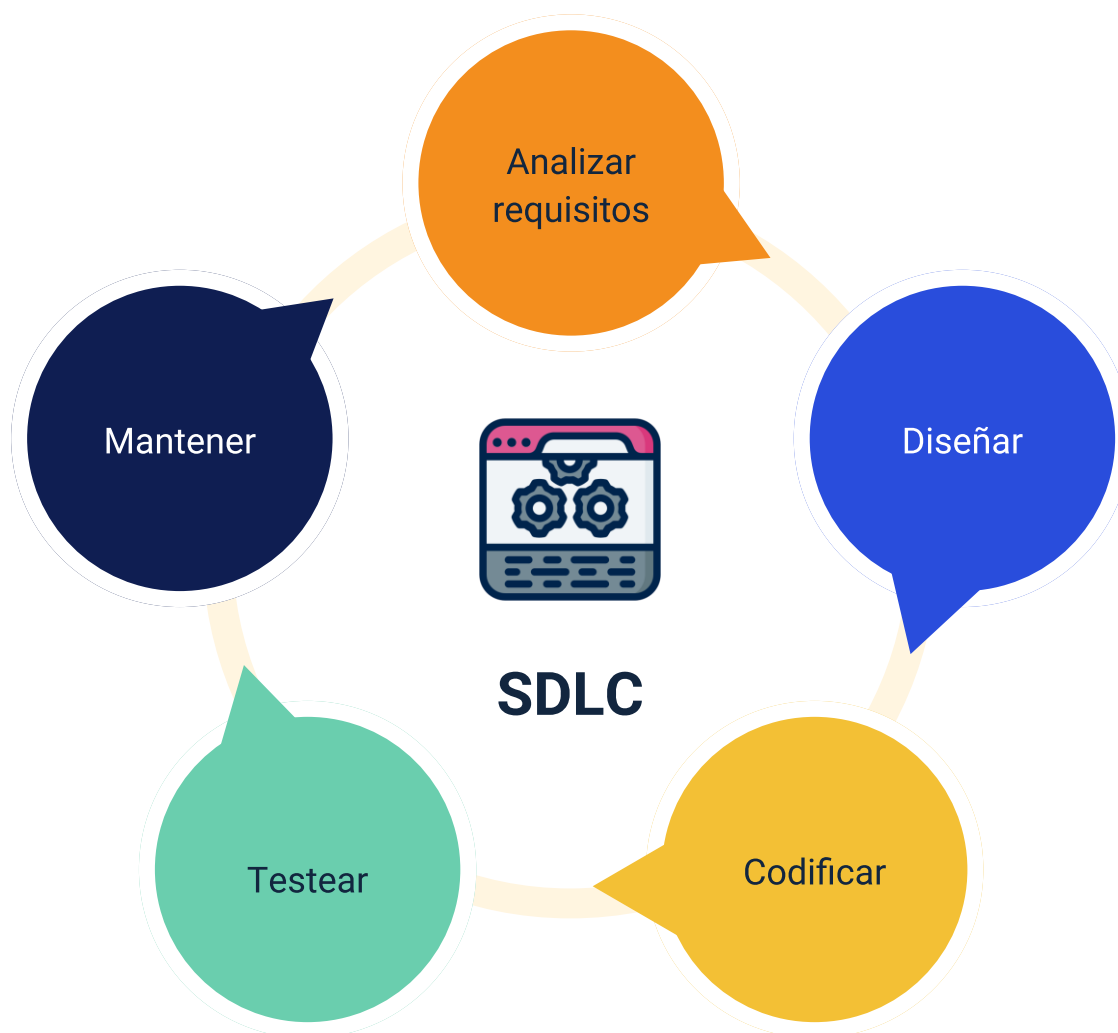


Actualmente la mayoría de los desarrolladores de “software” no realizan la validación de vulnerabilidades en sus productos si no hasta que este ya es funcional, pero esto no es tan eficiente, por ello se sugiere la adopción de un Ciclo de Vida de Desarrollo de “software” (SDLC), el cual lleva a tener presente el factor de seguridad en cada una de las fases del desarrollo, así:

Con la adopción de los SDLC o Ciclo de Vida de Desarrollo de “software”, se busca que el factor de la seguridad esté presente desde el momento de la concepción del “software” permitiendo identificar los requisitos funcionales de la necesidad que motiva la creación de una aplicación, mantener los componentes seguros desde el diseño y prototipado, buenas prácticas de programación, desarrollar un plan de

pruebas de seguridad, así como mantener un plan de mantenimiento y actualizaciones que garanticen su funcionalidad y seguridad de la información en el tiempo, como se puede apreciar en la siguiente figura.

Figura 6. Ciclo de vida SDLC genérico



La metodología OWASP, es abierta y su actualización se puede consultar en su página oficial <https://owasp.org/www-project-web-security-testing-guide/stable/>

2. Vulnerabilidades en aplicaciones web

Actualmente se cuenta con una amplia gama de recursos normativos y estándares para la gestión de vulnerabilidades, algunas de ellas con un nivel de cobertura más amplio dado su capacidad de aplicación dentro de las necesidades de una organización. Cada una de estas normas tiene como talante la gestión proactiva de las vulnerabilidades antes de una ocurrencia de un incidente que ponga en riesgo la información administrada

2.1. Metodologías para la gestión de vulnerabilidades

OWASP propone una metodología para la identificación de vulnerabilidades basada en pruebas de seguridad sobre las aplicaciones web, con el cual se busca encontrar las debilidades presentes en un sistema.

Este modelo de pruebas está conformado por:

- **Probador/Auditor.** Quien realiza las pruebas y validaciones.
- **Herramientas y metodología.** Guía de pruebas de OWASP.
- **Aplicación.** Objeto que se desea probar.

A su vez, las pruebas se pueden clasificar así:

- **Pruebas pasivas.** El probador intenta comprender la lógica de la aplicación y explora la aplicación web como un usuario. Se pueden utilizar herramientas para la recopilación de información. Por ejemplo, se puede utilizar un “proxy” HTTP para observar todas las solicitudes y respuestas. Al final de esta fase, el probador generalmente debe comprender todos los puntos de acceso y la funcionalidad del sistema (por ejemplo, encabezados HTTP, parámetros, “cookies”, API, uso / patrones de tecnología, etc.).

- **Pruebas pasivas.** Esto puede indicar un formulario de autenticación donde la aplicación solicita un nombre de usuario y una contraseña.

Los siguientes parámetros representan dos puntos de acceso a la aplicación:

<https://www.example.com/appx?a=1&b=1>

En este caso, la aplicación muestra dos puntos de acceso (parámetros y).

Todos los puntos de entrada encontrados en esta fase representan un objetivo para las pruebas. Hay que realizar un seguimiento del directorio o árbol de llamadas de la aplicación y de todos los puntos de acceso puede ser útil durante las pruebas activas.

- **Pruebas activas.** Durante estas pruebas, un probador comienza a utilizar las metodologías descritas en las siguientes secciones, las cuales están divididas en 12 categorías:
 - Recopilación de información.
 - Pruebas de administración de configuración e implementación.
 - Pruebas de administración de identidades.
 - Pruebas de autenticación.
 - Pruebas de autorización.
 - Pruebas de gestión de sesiones.
 - Pruebas de validación de entrada.
 - Manejo de errores.
 - Criptografía.
 - Pruebas de lógica empresarial.

- Pruebas del lado del cliente.
- Pruebas de API.

2.2. Fundamentos de “pentesting”

Dentro del ejercicio de la gestión de vulnerabilidades, cobran vital importancia las metodologías para el descubrimiento e identificación de estas, ya que orientan y dan un mapa de ruta de cómo se debe realizar el ejercicio, uno de estos es el “pentesting” o prueba de penetración, que no es más que un ejercicio de descubrimiento apoyado en técnicas y herramientas especializadas para la validación y búsqueda de vulnerabilidades en un sistema o aplicación.

Existen los siguientes tipos de “pentesting”:

- **Prueba en aplicación web.** Es una exploración profunda en una aplicación de tipo web, realizando análisis extremadamente detallados, para identificar vulnerabilidades que pueden poner en riesgo la información.
- **Prueba de “Client Side”.** En este tipo de prueba, es posible analizar “software”, programas de creación de contenido y web “browsers” (como Chrome, Firefox, Explorer, etc) en ordenadores de los usuarios, para la búsqueda de vulnerabilidades.
- **Prueba en red inalámbrica.** Se examinan las redes inalámbricas validando protocolos, puntos de acceso y credenciales administrativas.
- **Prueba de Ingeniería Social.** Se analizan las informaciones y datos confidenciales que pueden ser objeto de robo por medio de manipulación psicológica, aprovechándose del ingenio o desconocimiento del usuario.

Para cualquiera de los tipos de “pentesting”, se sugiere un ciclo de vida como se muestra en la siguiente figura, en donde se resalta:

- **Reconocimiento.** Identificar los activos de información, aplicaciones, dispositivos, información de red, credenciales de acceso, información.
- **Análisis de vulnerabilidades.** Realizar mediante la aplicación de técnicas y herramientas especializadas, la búsqueda de posibles vulnerabilidades que pueden afectar un activo de información.
- **Explotación.** Permite validar si una vulnerabilidad es explotable o puede permanecer en estado inactivo.
- **Escalar privilegios o postexplotación.** Permite evaluar los posibles daños que pueden causar si una vulnerabilidad se materializa como amenaza.
- **Informe.** Generar el reporte del ejercicio realizado, hallazgos y detalles de las vulnerabilidades encontradas.

Ahora bien, este ejercicio se puede realizar de manera exploratoria sin conocimiento del escenario e información de acceso, y dependiendo de esto, las pruebas se pueden clasificar en:

- **Caja negra.** Aquí los auditores no cuentan con información previa sobre el escenario en el cual se va a trabajar, se desarrollan ejercicios de exploración e intrusión, este tipo de pruebas por lo general son realizadas por personas externas a la organización.
- **Caja blanca.** Los auditores cuentan con datos del escenario e incluso credenciales de acceso a los sistemas de información, este tipo de pruebas lo realizan principalmente personal interno de la organización.

- **Caja gris.** Se cuenta con parte de la información, pero se requiere detallar ciertos aspectos técnicos.

2.3. OWASP - top 10 de vulnerabilidades

De acuerdo con la comunidad de OWASP, este genera reportes sobre las vulnerabilidades que más se encuentran en las aplicaciones web y en su último reporte se puede observar el siguiente comportamiento del top 10 de las vulnerabilidades identificadas.

- **Pérdida del control de acceso (“Broken Access Control”).** El control de acceso permite cumplir una política de permisos y roles, es decir, que un usuario pueda acceder a determinados lugares. Estas restricciones implican que los usuarios no puedan actuar fuera de los permisos y, además, llevar un control de quien accede a cada recurso.
La vulnerabilidad “Broken Access Control” permite que un usuario sin privilegios pueda acceder a un recurso al que no tendría que acceder.
- **Fallos criptográficos (“Cryptographic Failures”).** Hay ciertos datos que deben estar cifrados, como credenciales de acceso, datos bancarios, información confidencial de la empresa, etc., ya que, aparte de que la ley lo exija, lo que un ciberdelincuente pueda hacer con los datos resulta catastrófico para la empresa. Para que estos sean vistos únicamente por las personas autorizadas de la empresa, hay que aplicarles un cifrado con algoritmos y protocolos estándares y robustos.
- **Inyección (“Injection”).** Esto sucede cuando un ciberdelincuente puede enviar datos dañinos a un cliente y desde el año 2021, el Cross-site Scripting forma parte de esta categoría, como una vulnerabilidad de

seguridad inyectando un script malicioso en un sitio web para luego ser procesado y ejecutado.

- **Diseño inseguro (“Insecure Design”).** A la hora de desarrollar una aplicación web es primordial incluir la seguridad de la aplicación desde la fase del diseño, centrándose la detección de riesgos relacionados con el diseño y las fallas arquitectónicas, además de dar más fuerza al modelado de amenazas, patrones de diseño seguros y arquitecturas de referencia.
- **Configuración de seguridad defectuosa (“Security Misconfiguration”).** En el entorno de la aplicación web los ciberdelincuentes intentarán acceder mediante cuentas por defecto, versiones obsoletas con vulnerabilidades sin actualizar, directorios desprotegidos, etc. Por ello, tiene que estar todo bien configurado y evitar usar credenciales por defecto, como puede ser en el caso del servidor, aplicaciones o dispositivos.
- **Componentes vulnerables y obsoletos (“Vulnerable and Outdated Components”).** Un ciberdelincuente podrá comprometer un sistema mediante vulnerabilidades ya conocidas en componentes comunes, como por ejemplo la versión del sistema operativo o aplicaciones instaladas en el servidor, entre otras.
- **Fallos de identificación y autenticación (“Identification and Authentication Failures”).** Esto sucede cuando en las interfaces de acceso no se controla el número de intentos de autenticación, hay una baja complejidad de las contraseñas o no se implanta un sistema multifactor “2FA”.

Esto podría permitir a un ciberdelincuente realizar ataques de fuerza bruta o diccionario para ingresar en él o cuando la aplicación permite utilizar contraseñas débiles.

- **Fallos en el “software” y en la integridad de los datos (“software” and Data Integrity Failures”).** Muchas aplicaciones se actualizan de manera automática. Cuando estas actualizaciones no son verificadas los ciberdelinquentes podrían modificarlas cargando sus propias actualizaciones y distribuyéndolas.
- **Fallos en el registro y la supervisión de la seguridad (“Security Logging and Monitoring Failures”).** La falta de registros sobre eventos, los denominados logs, en la aplicación o en el sistema, como inicios de sesión (tanto válidos como fallidos). Por ejemplo, que estos registros no se almacenen remotamente impide que se puedan detectar las infracciones.
- **Falsificación de solicitud del lado del servidor (“Server-side Request Forgery o SSRF”).** Cuando nuestra aplicación web obtiene un recurso externo y este no valida la URL un ciberdelincuente podría modificarla con fines malintencionados y realizar peticiones no autorizadas.

2.4. Herramientas especializadas

Estas son un elemento fundamental en el ejercicio de la búsqueda y gestión de vulnerabilidades, dado que permiten agilizar, automatizar y realizar cientos de validaciones en segundos, y generando informes muy completos sobre los hallazgos encontrados.

A continuación, se exponen algunas de las herramientas más comunes, aunque más adelante se profundizará sobre su utilidad y manejo en el ejercicio de la auditoría.

Dentro del universo de herramientas a utilizar, están las siguientes categorías:

- Herramientas de pruebas de seguridad de aplicaciones estáticas (SAST).
- Herramientas de pruebas dinámicas de seguridad de aplicaciones (DAST), principalmente para aplicaciones web.
- Herramientas interactivas de pruebas de seguridad de aplicaciones (IAST), principalmente para aplicaciones web y API web.
- Mantener actualizadas las bibliotecas de código abierto, para evitar el uso de componentes con vulnerabilidades conocidas (OWASP Top 10-2017 A9)
- Herramientas de calidad de código estático.

Herramientas SAST

En el siguiente recurso podrá conocer las herramientas más importantes SAST.

- **Escaneo de código de GitHub.** Es un servicio gratuito de análisis estático de código abierto que utiliza GitHub Actions y CodeQL.
Sirve para escanear repositorios públicos en GitHub. Soporta C/C++, C#, Ruby (beta), Java, JavaScript/TypeScript, Python y Go.
- **Análisis estático de Coverity Scan.** Se puede conectar a Travis-CI para que se realice automáticamente con recursos en línea. Soporta más de una docena de lenguajes de programación.
- **Reshift.** Es una herramienta de CI/CD que utiliza el análisis de código estático para buscar vulnerabilidades y utiliza el aprendizaje automático para dar una

predicción sobre falsos positivos. Soporta Java con soporte futuro para NodeJS y JavaScript planeado.

- **WhiteSource Free Tools.** “Suite” de herramientas para detectar vulnerabilidades en código fuente.
- **HCL AppScan CodeSweep.** Esta es una versión de la edición de la comunidad SAST de HCL AppScan.

La herramienta actualmente es compatible con Python, Ruby, JS (Vue, Node, Angular, JQuery, React, etc.), PHP, Perl, Go, TypeScript y con otros nuevos lenguajes que se agregan con frecuencia.

Herramientas DAST

Si un proyecto tiene un componente de aplicación web, se recomienda ejecutar análisis automatizados contra él para establecer las vulnerabilidades.

- **OWASP ZAP.** Es una herramienta gratuita y de código abierto con todas las funciones que incluye escaneo automatizado en busca de vulnerabilidades y herramientas para ayudar a los expertos a realizar pruebas manuales de lápiz de aplicaciones web.
- **StackHawk.** Esta es una herramienta con soporte comercial construida sobre OWASP ZAP y optimizada para ejecutarse en CI/CD (casi todos los CI soportados) para probar aplicaciones web durante el desarrollo y en CI/CD. StackHawk es gratuito para proyectos de código abierto y de uso gratuito en una sola aplicación.

- **Arachni – Arachni.** Es un escáner con soporte comercial, pero es gratuito para la mayoría de los casos de uso, incluido el escaneo de proyectos de código abierto.
- **VWT Digital's Sec-helpers.** Colección de ayudantes dinámicos relacionados con la seguridad. Sec-helpers es un conjunto de pruebas y validadores útiles para garantizar la seguridad de un dominio determinado.
- **OWASP Purpleteam.** Una regresión de seguridad que prueba SaaS y CLI, perfecta para insertar en sus canalizaciones de compilación. No necesita escribir ninguna prueba. Esta herramienta únicamente necesita proporcionar un archivo de trabajo que le diga a Purpleteam lo que desea que se pruebe. Tiene dos ambientes principales y un “local cloud”.

Herramientas IAST

Estas herramientas suelen estar orientadas a analizar aplicaciones web y API web, pero eso es específico del proveedor. Puede haber productos IAST que también realicen un buen análisis de seguridad en aplicaciones que no sean web.

Contrast Community Edition (CE). Versión con todas las funciones para 1 aplicación y hasta 5 usuarios (algunas funciones Enterprise deshabilitadas). Contrast CE solo admite Java y .NET.

A continuación podrá conocer algunas de las herramientas disponibles para la auditoría de aplicaciones web y para proyectos de desarrollo de aplicaciones web, y vale la pena aclarar que algunas de ellas son “open source”, o tienen una posibilidad de ser utilizados cuando son proyectos abiertos, pero para proyectos comerciales o de gran tamaño requieren de una licencia o contrato de uso.

- **“Software” de código abierto (OSS). Herramientas de seguridad de “software” de código abierto (OSS).** OSS se refiere a las bibliotecas o componentes de código abierto que los desarrolladores de aplicaciones aprovechan para desarrollar rápidamente nuevas aplicaciones y agregar características a las aplicaciones existentes.
 - “Plugin Maven Versions”.
 - “Dependabot”.
- **Componentes vulnerables conocidos. Detección de componentes vulnerables conocidos.** Como alternativa y para tratar de mantener todos sus componentes actualizados, un proyecto puede monitorear específicamente si alguno de los componentes que utilizan tiene componentes vulnerables conocidos:
 - OWASP.
 - GitHub
 - Debricked
- **Calidad de código. Herramientas de calidad de código.** Los proyectos de desarrollo de “software” consideran el uso de herramientas de buena calidad de código. Algunos sugeridos por OWASP son:
 - SpotBugs
 - SonarQube
 - DeepScan

- **Entornos DevOps/CI. Herramientas de seguridad integradas en entornos DevOps/CI.** Permiten a las empresas conseguir una mayor eficiencia, productividad y agilidad, conectando diferentes aplicaciones de “software” y servicios, interfaces de programación de aplicaciones (API) datos y dispositivos para automatizar los procesos empresariales.
 - a) Grendel-Scan
 - b) Vega
 - c) Wapiti

Otras herramientas de seguridad gratuitas de código abierto son:

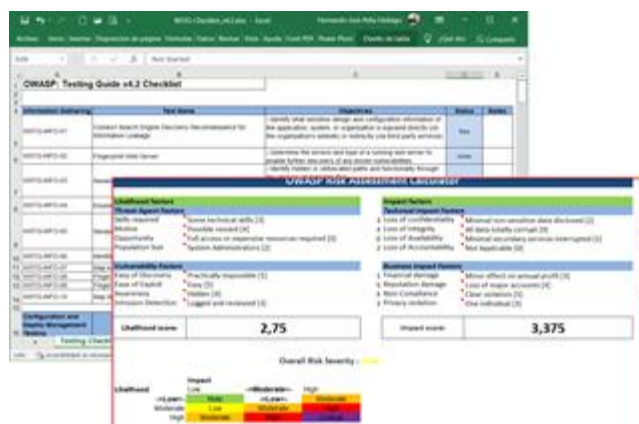
- a) Skipfish.
- b) SQLMap
- c) Graber.
- d) Ratproxy.
- e) Wfuzz.

2.5. Análisis de resultados

Una vez realizado el plan de pruebas programado con las herramientas y técnicas seleccionadas, se deben documentar los hallazgos encontrados en el ejercicio, para ello se emplean instrumentos que permitan registrar, analizar y evaluar el estado actual de la seguridad de las aplicaciones web.

Desde el proyecto OWASP, se han propuesto varios instrumentos para registrar la información a manera de “checklist”, a saber:

Figura 7. Instrumento “checklist” para registro de la evaluación.



Instrumento “checklist” para registro de la evaluación

Este permite obtener un informe del riesgo que pueden llegar a presentar una determinada aplicación web, registrando cada una de las aplicaciones web que se desea evaluar, y registrar el resultado de las pruebas practicadas, esta información será útil para generar un informe final del nivel del riesgo presente en las aplicaciones.

Para revisar un ejemplo puede ingresar al enlace:

<https://docs.github.com/es/get-started/writing-on-github/working-with-advanced-formatting/about-task-lists>

Figura 8. OWASP Risk Assessment Calculator.



- **Instrument OWASP Risk Assessment Calculator.** Esta herramienta es una calculadora de evaluación de riesgos OWASP y se puede encontrar un instrumento de evaluación en línea disponible en <https://javierolmedo.github.io/OWASP-Calculator/>

Es importante resaltar que el ejercicio de pruebas y la obtención del indicador de riesgo es un producto importante, pero ante la organización o los propietarios de la información se deben presentar reportes técnicos, con evidencias, detalles de las pruebas realizadas y resultados obtenidos, así como una serie de recomendaciones para subsanar dichas vulnerabilidades.

Desde el proyecto OWASP, en su metodología en el capítulo 5, sugiere las partes mínimas de un informe. <https://owasp.org/www-project-web-security-testing-guide/v42/5-Reporting/README>

Tabla 1. Estructura general de un reporte de “pentesting”

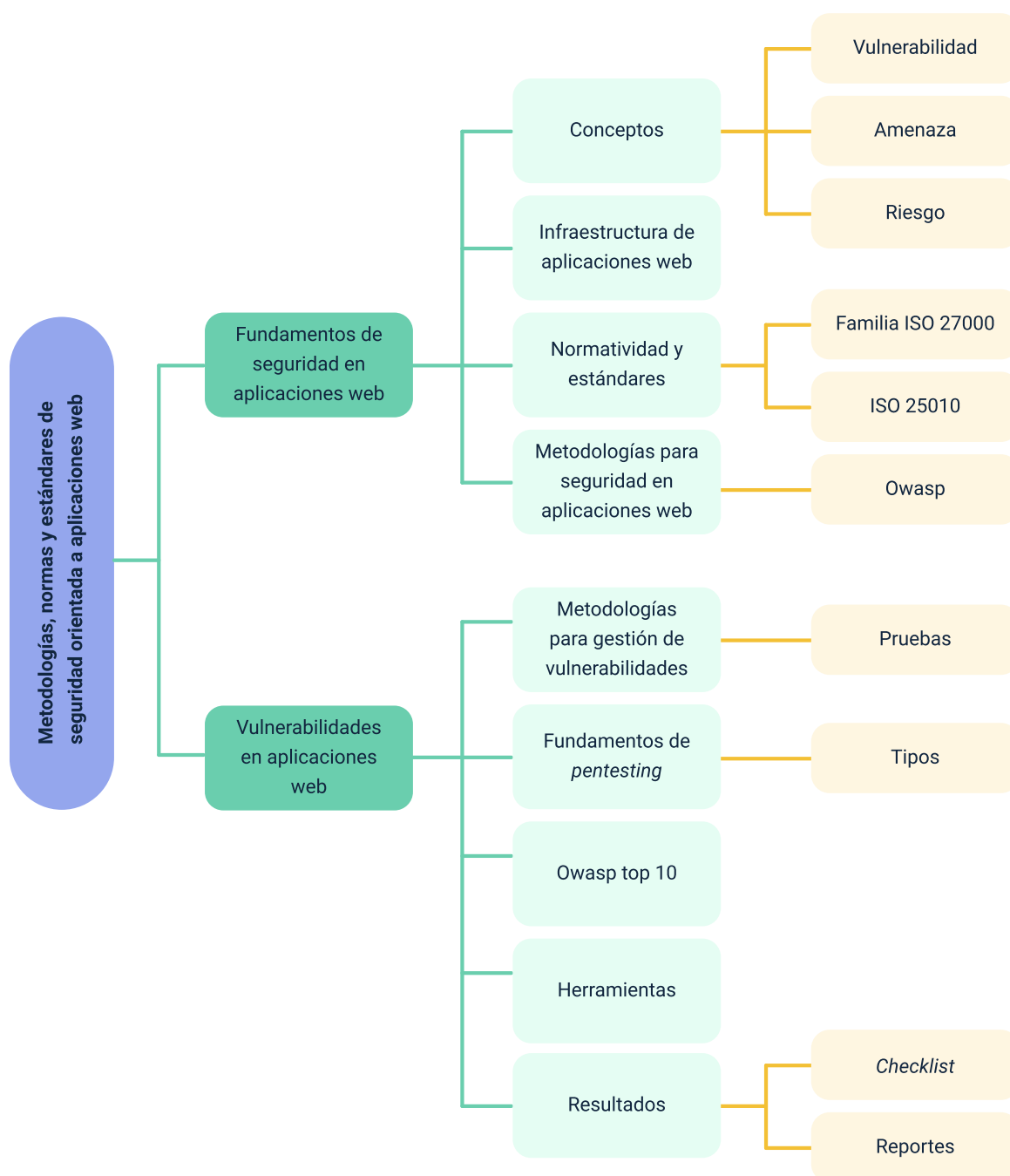
01. Introducción	02. Resumen ejecutivo
1. Control de versiones.	a. Conclusiones 1. Resumen de conclusiones. 2. Detalles de hallazgos. b. Apéndices
2. Tabla de contenido.	
3. Equipo de trabajo.	
4. Ámbito de aplicación.	
5. Limitaciones.	
6. Cronología.	
7. Descargo de responsabilidades.	

Modelo de “pentesting”

Para realizar una presentación y sustentación ante la organización del ejercicio realizado, y argumentando con detalles y pruebas, las debilidades que presentan y un plan de mejoramiento que permita la reducción de la brecha a la seguridad en aplicaciones web, se puede recurrir al modelo de reporte de “pentesting” ingresando al siguiente enlace: <https://owasp.org/www-project-web-security-testing-guide/v42/5-Reporting/README>

Síntesis

A continuación, se muestra un mapa conceptual con los elementos más importantes desarrollados en este componente.



Material complementario

Tema	Referencia	Tipo de material	Enlace del recurso
Análisis de resultados	OWASP. (2022). Web Security Testing Guide 4.2. OWASP.	PDF	https://github.com/OWASP/wstg/releases/download/v4.2/wstg-v4.2.pdf
Análisis de resultados	OWASP. (2022). Web Security Testing Guide Checklist. OWASP.	Página Web	https://github.com/OxRadi/OWASP-Web-Checklist
Análisis de resultados	Olmedo, J. (2021). OWASP risk assessment calculator.	Página Web	https://javierolmedo.github.io/OWASP-Calculator/

Glosario

2FA: término que representa doble factor de autenticación, solución que agrega una capa de seguridad en sistemas de autenticación basados en componentes externos a un sistema, por ejemplo, un “token”, una aplicación, tarjeta inteligente, etc.

“Checklist”: lista de chequeo que sirve para registrar un proceso de auditoría.

GUI: interfaz gráfica de usuario, por ejemplo, una aplicación, o aplicación móvil.

KPI: “Key Performance Indicator” o indicador clave de desempeño, que representa las métricas de eficacia de las acciones adoptadas y así determinar si son efectivas o no.

URL: “Uniform Resource Locator” (localizador de recursos uniforme), representa una dirección que apunta hacia un recurso único en la web.

Referencias bibliográficas

Guijarro, O., J., Caparrós, R., J., & Cubero, L., L. (2019). *DevOps y seguridad cloud*. Editorial UOC. <https://elibro-net.bdigital.sena.edu.co/es/ereader/senavirtual/128889>

Incibe. (2022). *Top 10 vulnerabilidades web de 2021*. Incibe. <https://www.incibe.es/protege-tu-empresa/blog/top-10-vulnerabilidades-web-2021>

MinTIC. (2016). *Guía de gestión de riesgos*. MinTIC. https://www.mintic.gov.co/gestionti/615/articles-5482_G7_Gestion_Riesgos.pdf

MinTIC. (2016). *Guía para la gestión y clasificación de activos de información*. MinTIC. https://www.mintic.gov.co/gestionti/615/articles-5482_G5_Gestion_Clasificacion.pdf

MinTIC. (2016). *Guía para la implementación de seguridad de la información en una mipyme*. MinTIC. https://www.mintic.gov.co/gestionti/615/articles-5482_Guia_Seguridad_informacion_Mypimes.pdf

MinTIC. (2020) *Anexo 3 - Resolución MinTIC 1519 del 2020, Condiciones mínimas técnicas y de seguridad digital*. https://gobiernodigital.mintic.gov.co/692/articles-178659_Condiciones_minimas.pdf

Ortega, C., J. M. (2018). *Seguridad en aplicaciones web Java*. RA-MA Editorial. <https://elibro-net.bdigital.sena.edu.co/es/ereader/senavirtual/106511>

OWASP. (2022). *OWASP Top 10:2021*. <https://owasp.org/Top10/>

OWASP. (2022). *OWASP Web Security Testing Guide*. <https://owasp.org/www-project-web-security-testing-guide/latest/>

Créditos

Nombre	Cargo	Centro de Formación y Regional
Claudia Patricia Aristizábal	Líder del Ecosistema	Dirección General
Rafael Neftalí Lizcano Reyes	Responsable de Línea de Producción	Centro Industrial del Diseño y la Manufactura - Regional Santander
Hernando José Peña Hidalgo	Experto temático	Centro de teleinformática y producción industrial - Regional Cauca
Paula Andrea Taborda Ortiz	Diseñadora instruccional	Centro de la Industria, la Empresa y Los Servicios CIES - Regional Norte de Santander
Carolina Coca Salazar	Asesora metodológica	Centro de Diseño y Metrología - Regional Distrito Capital
José Gabriel Ortiz Abella	Corrector de estilo	Centro de Diseño y Metrología - Regional Distrito Capital
Juan Daniel Polanco Muñoz	Diseñador de Contenidos Digitales	Centro Industrial del Diseño y la Manufactura - Regional Santander
Edward Leonardo Pico Cabra	Desarrollador Fullstack	Centro Industrial del Diseño y la Manufactura - Regional Santander
María Natalia Maldonado Delgado	Diseño web	Centro Industrial del Diseño y la Manufactura Regional Santander
Juan Daniel Polanco Muñoz	Validación Diseño web	Centro Industrial del Diseño y la Manufactura Regional Santander
Carlos Mauricio Gomez Delgado	Desarrollo front-end	Centro Industrial del Diseño y la Manufactura Regional Santander
Andrea Botello	Validación de Maquetación	Centro Industrial del Diseño y la Manufactura Regional Santander

Nombre	Cargo	Centro de Formación y Regional
Eulises Orduz	Actividades didácticas	Centro Industrial del Diseño y la Manufactura Regional Santander
Lina Marcela Perez	Validación Contenido	Centro Industrial del Diseño y la Manufactura Regional Santander
Zuleidy María Ruíz Torres	Revisión de guion audiovisual	Centro Industrial del Diseño y la Manufactura Regional Santander
María Carolina Tamayo López	Locución	Centro Industrial del Diseño y la Manufactura Regional Santander
Wilson Andrés Arenales Cáceres	Ilustración	Centro Industrial del Diseño y la Manufactura Regional Santander
John Jairo Arciniegas González	Producción audiovisual	Centro Industrial del Diseño y la Manufactura Regional Santander
Gilberto Junior Rodríguez Rodríguez	Validación audiovisual	Centro Industrial del Diseño y la Manufactura Regional Santander
Zuleidy María Ruiz Torres	Validador de Recursos Educativos Digitales	Centro Industrial del Diseño y la Manufactura - Regional Santander
Luis Gabriel Urueta Alvarez	Validador de Recursos Educativos Digitales	Centro Industrial del Diseño y la Manufactura - Regional Santander
Daniel Ricardo Mutis Gómez	Evaluador para contenidos inclusivos y accesibles	Centro Industrial del Diseño y la Manufactura - Regional Santander