Sesión 5 : Principios de Código Seguro

Angel Martinez

angel.martinez@wizeline.com

Acerca de mí

Angel Martínez

Staff Software Engineer

- 8 años de experiencia
- 5 años en Wizeline
- Diseño de arquitectura y soluciones

Puntos importantes

Identifícate en Zoom utilizando tu nombre y apellido.

Mantén tu micrófono apagado durante el transcurso de la sesión.

Utiliza el chat para hacer tus preguntas durante la sección de Q&A.

Procura enfocar tus preguntas al tema presentado.

Apaga tu cámara en caso de tener problemas con tu conexión.

Código de conducta

Sé respetuoso, no hay malas preguntas o ideas.

Sé cordial y paciente.

Sé cuidadoso con tus palabras.

Objetivo

Al final de esta sesión podrás:

- Visualizar un conocimiento general acerca de código seguro, reconociendo la importancia y los principios bajo los cuales se rige

Table of Contents

Tabla de Contenidos

Introducción

90M Usuarios

En 2018, Cambridge Analytics accedió y comercializó alrededor de 90M de registros esto fue posible a que no existían reglas de cumplimiento.

¿Qué es código seguro?

- Adaptación de mejores prácticas de seguridad para la **prevención, identificación y mitigación** de vulnerabilidades a nivel código en nuestro SDLC.
- **“Las malas prácticas de codificación segura, incrementa la probabilidad de ataques”**

Importancia

Mitigan de riesgos de seguridad

Reduce la explotación de vulnerabilidades así como el comprometer la **confidencialidad, integridad, y disponibilidad.**

Protegen la información del usuario

Asegura que la información de carácter personal, financiero, privado, permanece protegido de **acceso no autorizado, filtraciones de datos, y modificaciones no deseadas.**

Previenen ataques cibernéticos

Protegen contra ataques comunes de código.

Importancia

Cumplimiento de normativas

Se alinea a estándares, normativas y regulaciones enfocadas en protección de datos y derecho de privacidad.

Integración en el SDLC

Integra prácticas a nivel seguridad al SDLC

Adapta a amenazas

Establece medidas de mitigación con base al descubrimiento y evolución de vulnerabilidades nuevas o existententes.

Tabla de Contenidos

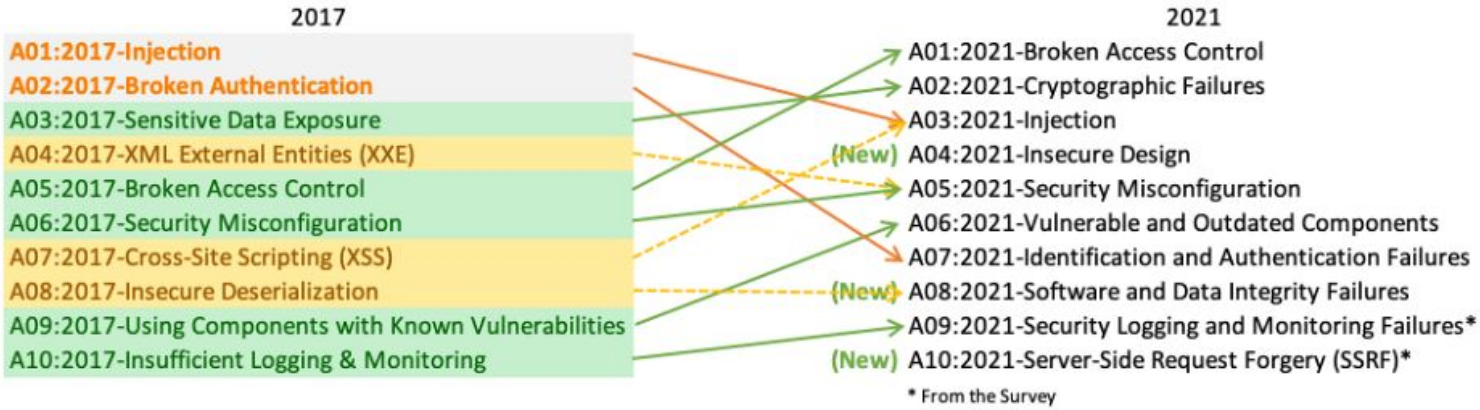

OWASP

¿Qué es OWASP?

- Open Worldwide Application Security Project **(OWASP)** es una comunidad enfocada en ayudar a las organizaciones a desarrollar, comprar y mantener aplicaciones y APIs confiables.

OWASP Top 10 Project

- **OWASP Top 10 Project,** es un estándar que representa un consenso sobre los riesgos más críticos para las aplicaciones Web.



A01:2021 – Broken Access Control

Escenario Ideal

- Los usuarios no pueden actuar fuera de los permisos que se definieron.

Impacto

- Divulgación, manipulación o eliminación de la información
- Actuar fuera del alcance definido
- Violación del **principio de mínimo privilegio**
- Ignorar controles de acceso
- Elevación de privilegios

Mecanismos de Prevención

- Negar por defecto
- Implementación de mecanismos de control de acceso
- Registro de fallas de acceso
- Límite de peticiones de APIs
- Invalidación de sesiones después del logout

A02:2021 – Cryptographic Failures

Escenario Ideal

- Información no es transmitida en texto claro
- Se hace uso de algoritmos criptográficos robustos
- Manejo apropiado de llaves, no se reusan y son rotadas constantemente
- No se usan funciones débiles de hashing

Impacto

- Exposición de información sensible
- Apertura a una posible brecha de seguridad

Mecanismos de Prevención

- No almacenamiento de información sensitiva no necesaria
- Cifrado en tránsito y en reposo
- Uso de algoritmos robustos y recomendados
- Uso de salts para hashing

A02:2021 – Cryptographic Failures

Escenario Ideal

- Información no es transmitida en texto claro
- Se hace uso de algoritmos criptográficos robustos
- Manejo apropiado de llaves, no se reusan y son rotadas constantemente
- No se usan funciones débiles de hashing

Impacto

- Exposición de información sensible
- Apertura a una posible brecha de seguridad

Mecanismos de Prevención

- No almacenamiento de información sensitiva no necesaria
- Cifrado en tránsito y en reposo
- Uso de algoritmos robustos y recomendados
- Uso de salts para hashing

A03:2021 – Injection

Escenario Ideal

- Información proporcionada por el usuario no se considera segura
- Se valida toda la información proporcionada

Impacto

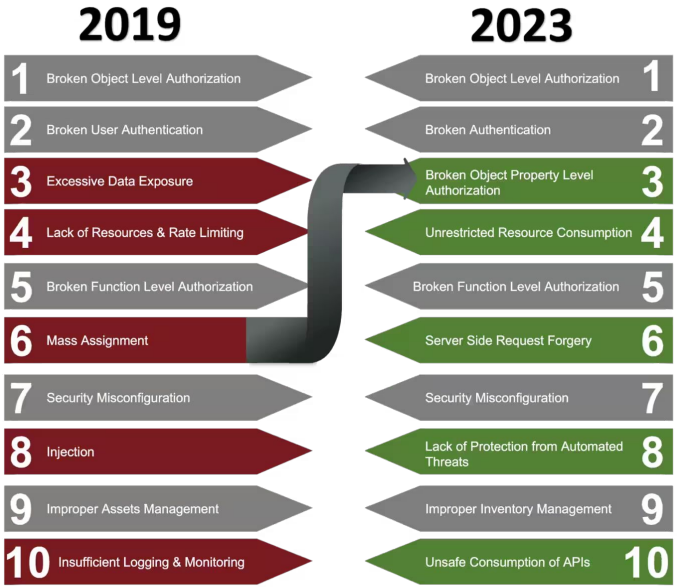
- Alterar la ejecución y el comportamiento del programa
- Extracción de información sensible

Mecanismos de Prevención

- Validación, filtrado, escapado y sanitizado de entradas
- Uso de consultas dinámicas, procedimientos almacenados, ORMs
- Testing automatizado para los parámetros de entrada
- Validación de lado del cliente y servidor

OWASP API Security Project

- El **OWASP API Security Project,** se enfoca en estrategias y soluciones para comprender y mitigar vulnerabilidades y riesgos de seguridad en APIs.



API1:2023 Broken Object Level Authorization

Escenario Ideal

- Permisos otorgados a recursos necesarios
- Validación de permisos previo a interacción

Impacto

- Divulgación, destrucción o modificación de la información
- Acceso a funcionalidad o recursos vulnerables

Mecanismos de Prevención

- Mecanismos de control de acceso y autorización
- Uso de GUIDs en lugar de ids genéricos

API2:2023 Broken Authentication

Escenario Ideal

- Autenticación no puede ser omitida utilizando fuerza bruta
- Solicita re-autenticación obligatoria para operaciones sensibles

Impacto

- Divulgación de detalles de autenticación
- Se vulnera el principio de no repudio
- Permite la ejecución de operaciones sensibles

Mecanismos de Prevención

- Uso de captchas y mecanismos de bloqueo
- Políticas de contraseñas seguras
- Validación de contraseñas para operaciones sensibles
- Validación de token
- Cifrado de datos sensibles
- Uso de estándares y buenas prácticas

API3:2023 Broken Object Property Level Authorization

Escenario Ideal

- Acceso limitado a propiedades específicas de los objetos

Impacto

- Exposición de propiedades sensibles que no deberían ser legibles a usuarios
- Alteración de propiedades sensibles (metadata)

Mecanismos de Prevención

- Verificación de propiedades a retornar
- Evitar el uso de métodos genéricos para retornos
- Implementar schemas de retorno
- Permitir actualización parcial solo a propiedades específicas
- Evitar funciones automáticas de enlace

Tabla de Contenidos

Principios de Código Seguro

¿Qué son los principios de código seguro?

- Guías, principios, estándares a implementar en el código, están diseñados para minimizar las fallas de seguridad.

Principle	Description
Secure Dependencies	Keep 3rd party libraries constantly updated
Avoid Hardcoded Secrets	Avoid sensitive information hardcoded
Regular Security Testing	Conduct regular security testing
Secure Development Lifecycle (SDLC)	Integrate security testing in your SDLC
Use Strong Encryption	Implement a strong encryption algorithm to protect data
Defense in Depth	Implement multiple security layers
Fail Securely Principle	Design the system to fail secure despite incidents

¿Qué son los principios de código seguro?

Principle	Description
Input Validation	User input is validated and sanitized before being processed
Output Encoding	Encode data before displaying it or being interpreted as code
Authentication and Authorization	Implement authorization (who) and authorization controls (what)
Minimize Attack Surface	Remove unnecessary features, components, and service
Secure Configuration	Do not consider default configuration as secure
Least Privilege	Limit the permissions and access to specific users
Secure Communication	Use protected protocols to transmit data over the network
Error Handling and Logging	Handling errors and avoid disclosing sensitive data

Principio de Privilegio Mínimo

Definición

- Se otorga el mínimo necesario de permisos para la ejecución de tareas o flujos.
- **Ejemplo:**
  Las interfaces UI a mostrar se basan en roles

Beneficios

- Reduce el impacto y el alcance en caso de una brecha de seguridad
- Se evita entidades con más privilegios que los necesarios

Implementación

- Habilitar permisos granulares bien definidos
- Uso de roles, permisos, políticas
- Revisiones periódicas y actualización de permisos
- Aislamiento de entidades

Defensa a Profundidad

Definición

- Inclusión de capas múltiples de seguridad
- **Ejemplo:**
  Un firewall actúa como puerta principal para proteger aplicaciones de amenazas de seguridad

Beneficios

- Ofrece una defensa multicapa robusta

Implementación

- Uso de herramientas como IDS, Antivirus, Firewalls, etc
- Implementación de seguridad a nivel:
  - Red
  - Aplicación
  - Información

Sanitización de Entradas

Definición

- Validación sintáctica y semántica, escapado y limpieza de información proporcionada o mostrada al usuario
- **Ejemplo:**
  El campo correo de inicio de sesión debe contener @

Beneficios

- Protección contra ataques de inyección

Implementación

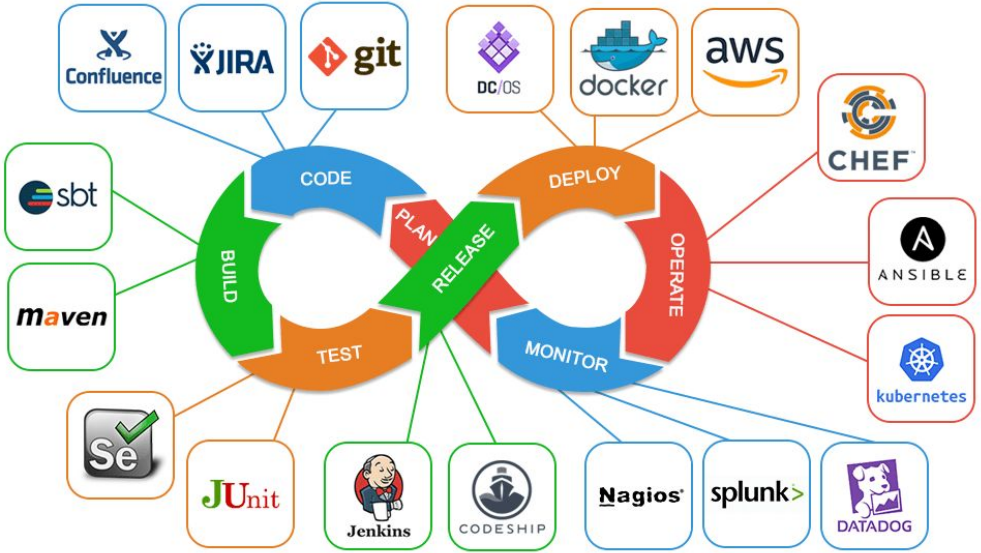
- Validación de entrada mediante:
  - Regex
  - Librerías
  - Funciones Built-in
  - Frameworks
- Uso de procedimientos almacenados o ORMs
- Validación bidireccional

Tabla de Contenidos

DevSecOps

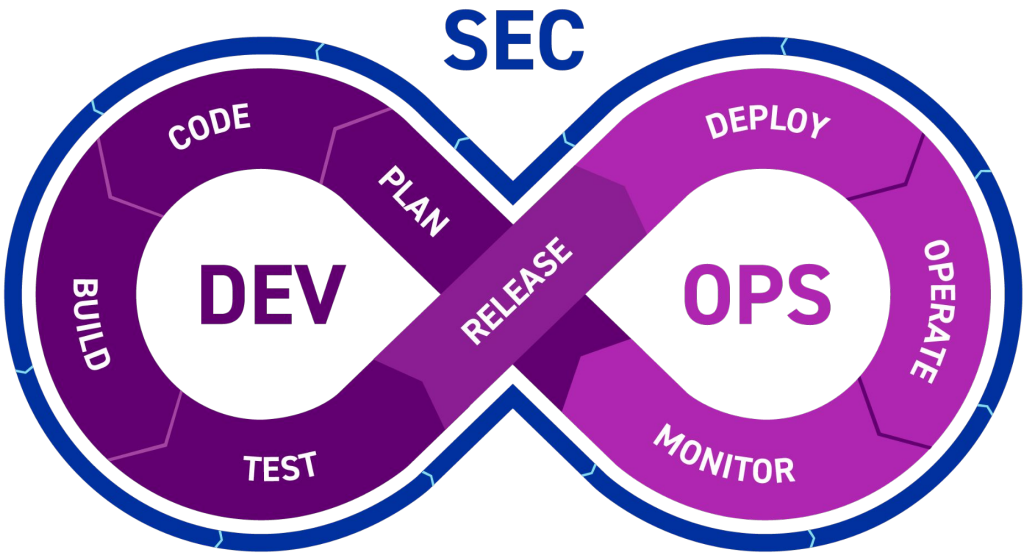
¿Qué es DevOps?

- DevOps es una metodología que se centra en la colaboración entre IT (Operaciones) y Desarrollo, para el desarrollo de software de manera conjunta

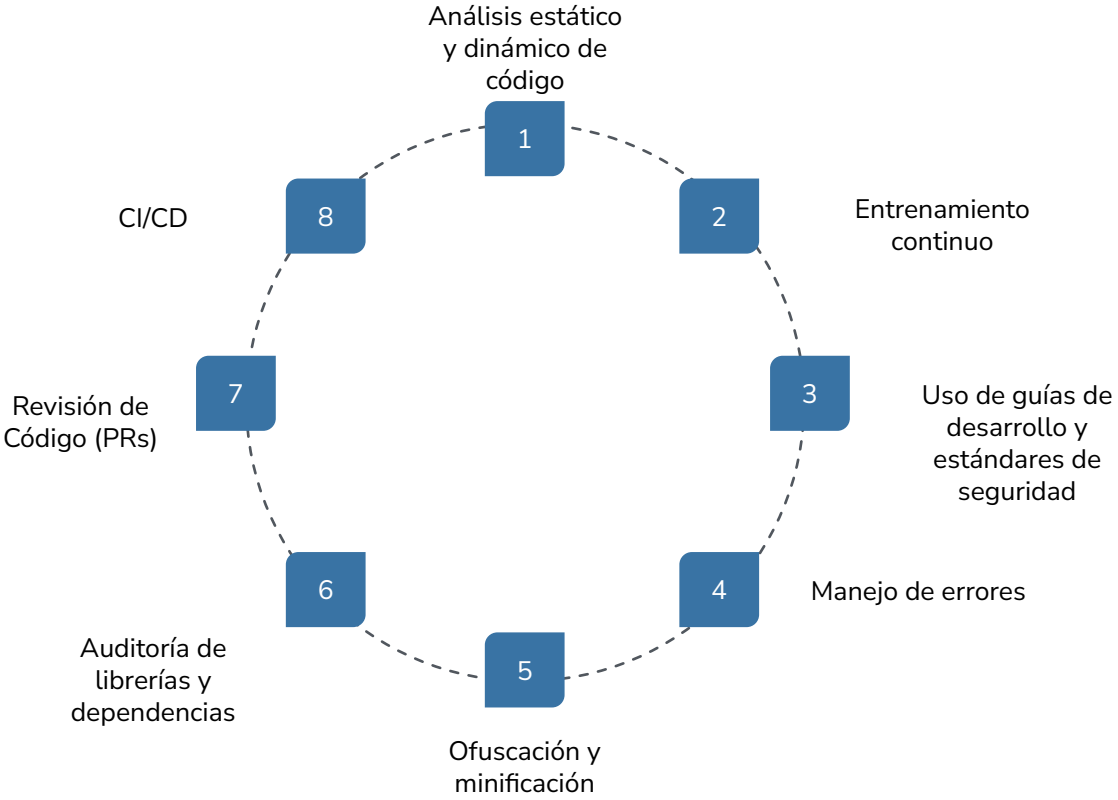


¿Qué es DevSecOps?

- DevSecOps es una extensión que incorpora la fase de seguridad al SDLC, permitiendo la detección oportuna de incidentes de seguridad, vulnerabilidades y fallos de cumplimiento de manera temprana.



Seguridad en nuestro SDLC



Recapitulación

Hemos hablado acerca de:

- Autorización
- Autenticación
- Consecuencias
- Ataques comunes
- Técnicas de prevención
- Ejemplo

¿Preguntas?

Retroalimentación

Déjanos saber tu retroalimentación

Gracias.