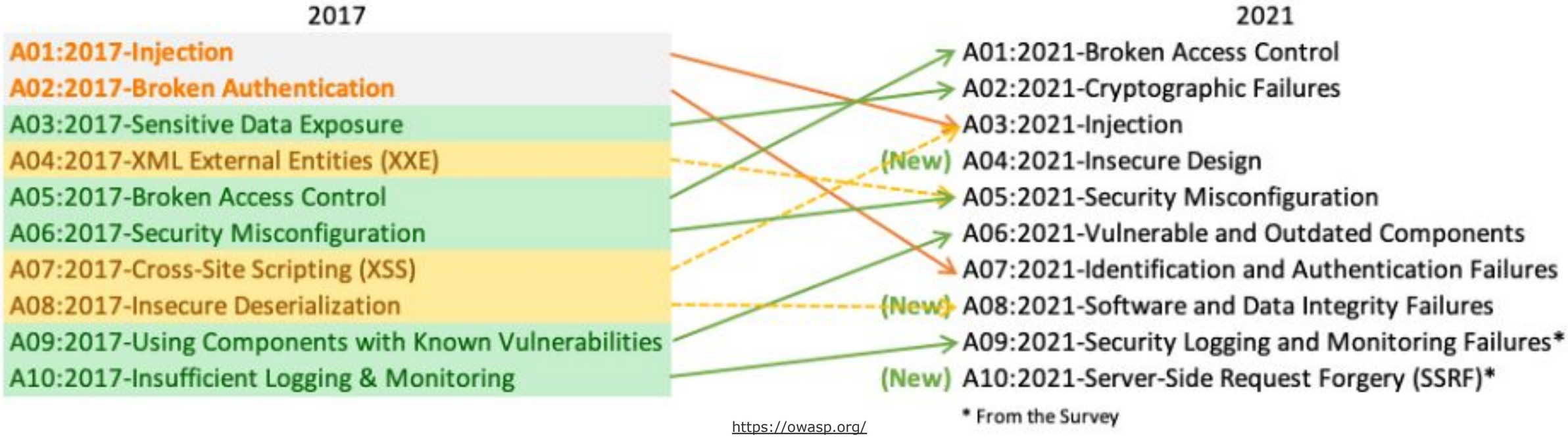
THE BRIDGE

CYBER SECURITY

OWASP

OWASP

- **OWASP**
  - **Open Web Application Security Project** (Proyecto Abierto de Seguridad de Apps Web) dedicado a determinar y combatir las causas que hacen que el software sea **inseguro**.
  - Proporcionan **guías**, **herramientas**, conferencias, **documentación.**
  - **OWASP Top Ten** de los riesgos de seguridad en Aplicaciones se actualiza cada cierto tiempo por un grupo de especialistas.
  - La documentación se consigue en https://owasp.org/Top10/



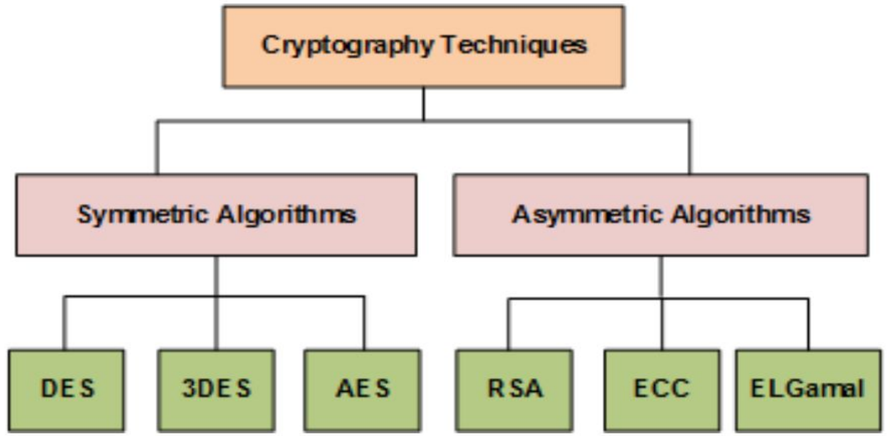
https://owasp.org/

OWASP Top 10

- **Control de acceso**
  - Debilidades en la implementación de controles de autenticación y autorización.
  - Principio de Mínimo privilegio.
  - Registrar los fallos de control de acceso y alertas.

https://cuadernosdeseguridad.com/2021/06/control-de-accesos-genetec/

- **Criptográfico**
  - Esquemas de cifrado obsoletos y/o inseguros.
  - Claves de cifrado débiles
  - Encriptar los datos en tránsito con protocolos seguros, priorizando el cifrado por parte del servidor
  - Evitar el almacenamiento de datos sensibles que no son necesarios o eliminarlos lo antes posible



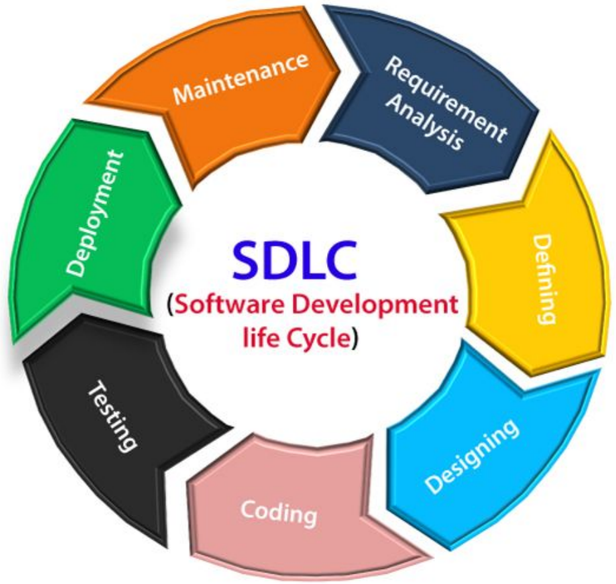
https://www.researchgate.net/figure/Taxonomy-of-cryptography-techniques_fig4_318200344

OWASP Top 10

- **Inyección**
  - los atacantes pueden aprovechar la ausencia de un correcto filtrado o saneado de los datos de entrada, para alterar el código de las funciones de la aplicación.
  - Los datos que introduce el usuario no son validados, filtrados o saneados.
  - Las consultas dinámicas son utilizadas directamente en el intérprete.
  - Los datos hostiles se utilizan dentro de los parámetros de búsqueda para extraer registros sensibles y se procesan o concatenan directamente

https://www.dnsstuff.com/sql-injection

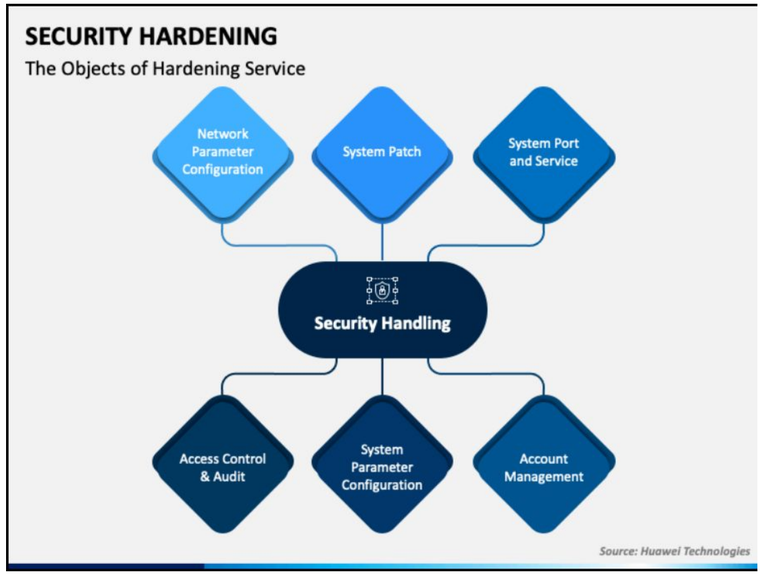
- **Diseño Inseguro**
  - Engloba los diferentes riesgos asociados a los defectos de diseño y de arquitectura web.
  - Estas vulnerabilidades son difíciles de subsanar una vez se haya realizado el desarrollo.
  - Implementar la seguridad en el SDLC (Systems Development Life Cycle).
  - Integrar el lenguaje y los controles de seguridad en las historias de los usuarios.
  - Escribir test de integración para validar que todos los flujos críticos son resistentes frente al modelo de amenazas
  - Segregar las capas de niveles en función de las necesidades de exposición y protección



https://www.javatpoint.com/software-engineering-software-development-life-cycle

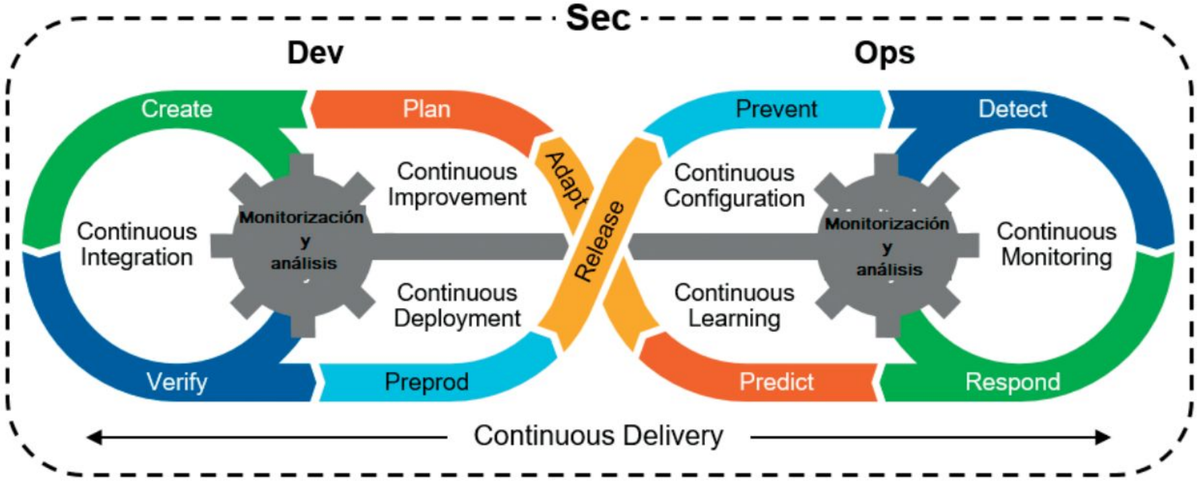
OWASP Top 10

- **Configuración de la seguridad incorrecta**
  - Poner en marcha un proceso de **hardening**, **endurecimiento** o **bastionado**.
  - Realizar segmentación entre los diferentes componentes de la arquitectura web
  - Aplicar directivas de seguridad que apuesten por una defensa en profundidad o Deep Security de los componentes.
  - Aplicar las buenas prácticas de seguridad relacionada con los diferentes elementos de la arquitectura (https://www.cisecurity.org)



https://conocimientolibre.mx/hardening/security-hardening-slide3/

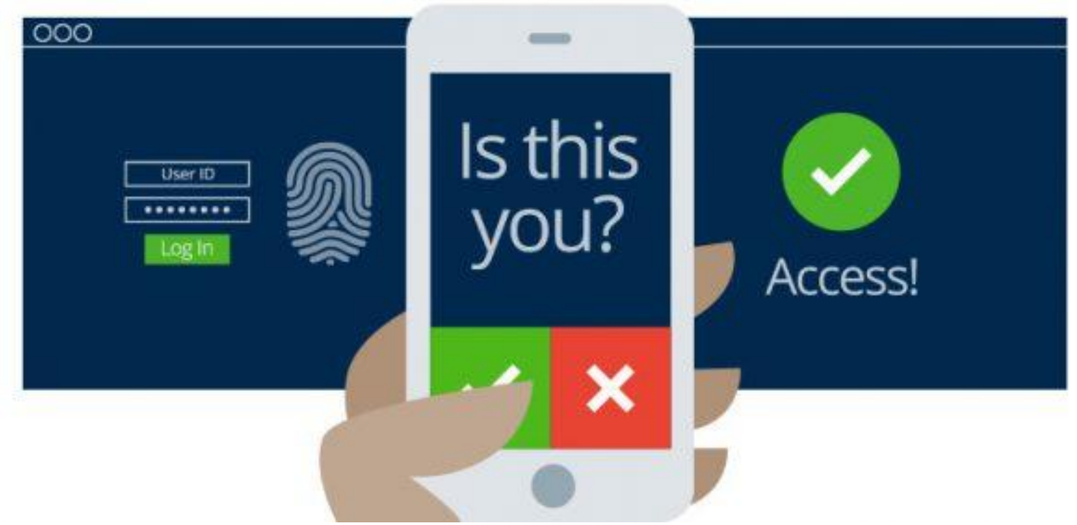
- **Componentes vulnerables y obsoletos**
  - Apostar por **DevSecOps**, un enfoque de gestión centrado en monitorizar, analizar y aplicar medidas de seguridad.
  - Eliminar los componentes, archivos y características no utilizados.
  - El software es vulnerable si no tiene soporte o está desactualizado
  - Inventariar continuamente las versiones de los componentes, tanto del lado del servidor como del lado del cliente
  - Emplear herramientas de análisis de componentes para automatizar el proceso
  - Obtener componentes únicamente de fuentes oficiales.



https://www.eset.com/fileadmin/ESET/ES/Landings/2019/Premios_periodismo/SIC132_reportaje_SecDevOps_Jos%C3%A9_Manuel_Vera.pdf

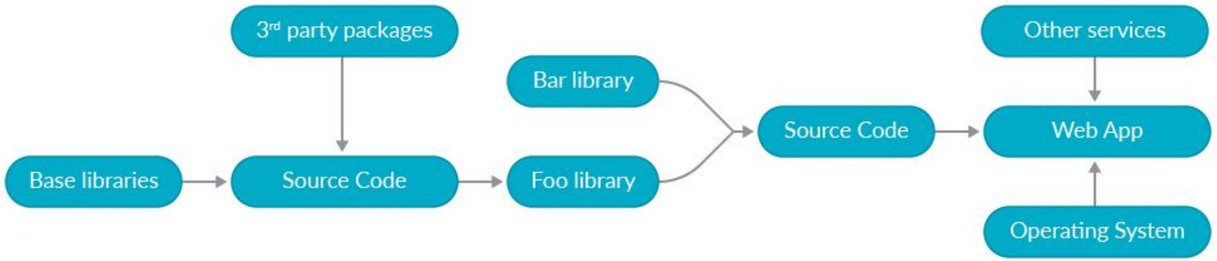
OWASP Top 10

- **Identificación y autentificación**
  - Implementar la **autenticación multifactor** para evitar ataques automatizados de fuerza bruta y la reutilización de credenciales robadas.
  - Procesos inseguros de recuperación de credenciales.
  - Escasas medidas contra ataques de fuerza bruta.
  - Incorrecta renovación de los identificadores de sesión para cada autenticación válida
  - No implementar **credenciales** por defecto



https://dominiogeek.com/diferencia-identificacion-autenticacion-autorizacion/

- **Software y en la integridad de los datos**
  - Garantizar que los datos serializados que carecen de firma o de encriptación se envían solo a clientes confiables. De cara a evitar su manipulación.
  - **Plugins**, **bibliotecas**, repositorios y redes de entrega de contenido no fiables.
  - Ataques maliciosos a las **supply chains** de software.



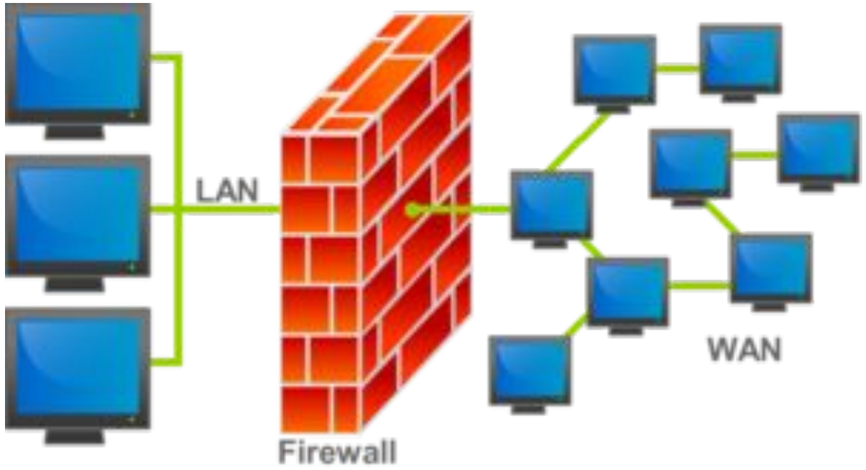
https://sysdig.com/blog/software-supply-chain-security/

OWASP Top 10

- **Registro y la supervisión de la seguridad**
  - Los eventos auditables, como los inicios de sesión o las transacciones de alto valor no se registran.
  - Las advertencias y los errores no generan mensajes de registro o son inadecuados.
  - Los registros de las aplicaciones y las API no se supervisan.
  - Las pruebas de penetración no activan las alertas de seguridad.

https://www.strongdm.com/blog/audit-log-review-management

- **Falsificación de solicitudes del lado del servidor**
  - Se producen cuando un atacante tiene la posibilidad de forzar al servidor a realizar conexiones hacia objetivos que no estaban previstos inicialmente
  - Evadir cortafuegos.
  - Forzar conexiones a elementos de la red interna.
  - Interactuar con recursos que inicialmente eran restringidos.



https://es.wikipedia.org/wiki/Cortafuegos_(inform%C3%A1tica)

Evitar Ataques Arquitectura WEB

1. Information Gathering
2. Configuration and Deployment Management Testing
3. Identity Management Testing
4. Authentication Testing
5. Authorization Testing
6. Session Management Testing
7. Input Validation Testing
8. Testing for Error Handling
9. Testing for Weak Cryptography
10. Business Logic Testing
11. Client-Side Testing

https://owasp.org/www-project-web-security-testing-guide/stable/

THE BRIDGE

Lorem ipsum Dolor sit amet, consectetur Adipiscing Elit. Etiam eget quam lacus.