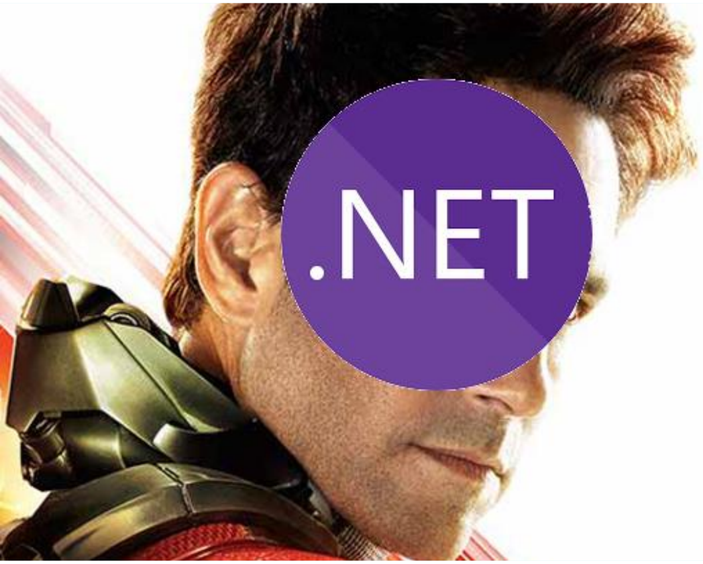
.NET-Man and the OWASP

Los diez riesgos más críticos en Aplicaciones Web

14/02/2020

<XantarDev /> Technical Community

Quiénes somos



Darío Cerredelo

Profesional del desarrollo de software especializado en .NET y QA.

https://twitter.com/dariocp

David Gonzalo

Entusiasta del desarrollo y especialista en tecnologías Microsoft.

https://twitter.com/dagope

Agenda

01 **Introducción**

OWASP Top 10

02 **A1:2017**

Inyección

03 **A2:2017**

Pérdida de Autenticación

04 **A3:2017**

Exposición de Datos Sensibles

05 **A4:2017**

Entidades Externas XML (XXE)

06 **A5:2017**

Pérdida de Control de Acceso

Agenda

07 **A6:2017**

Configuración de Seguridad Incorrecta

08 **A7:2017**

Cross-Site Scripting (XSS)

09 **A8:2017**

Deserialización Insegura

10 **A9:2017**

Uso de Componentes con Vulnerabilidades Conocidas

11 **A10:2017**

Registro y Monitoreo Insuficientes

12 **Ruegos y preguntas**

OWASP

Sobre OWASP

El Proyecto Abierto de Seguridad en Aplicaciones Web (OWASP por sus siglas en inglés) es una **comunidad abierta** dedicada a facilitar que las organizaciones desarrollen, adquieran y mantengan aplicaciones y APIs en las que se pueda confiar.

¿En qué trabaja?

- Herramientas y **estándares** de seguridad en aplicaciones.
- Libros completos de revisiones de seguridad en aplicaciones, desarrollo de código fuente seguro y revisiones de seguridad en código fuente.
- Presentaciones y videos.
- Controles de seguridad estándar y bibliotecas.
- Investigaciones de vanguardia.
- Conferencias alrededor del mundo.

OWASP Top 10

¿En qué se basa?

- Envío de datos de más de 40 empresas de seguridad.
- Encuesta 500 profesionales del sector.
- Vulnerabilidades recopiladas de cientos de organizaciones.
- 100.000 aplicaciones y APIs.

Categorización

Las 10 principales categorías fueron seleccionadas y priorizadas de acuerdo con estos datos de **prevalencia**, combinados con estimaciones consensuadas de **explotabilidad**, **detectabilidad** e **impacto**.

Objetivo principal

Educar a los desarrolladores, diseñadores, arquitectos, gerentes y organizaciones **sobre las consecuencias de las debilidades** más comunes y más importantes de la seguridad de las aplicaciones web.

A1 - Inyección

A1:2017

Inyección

Los errores de inyección se producen cuando se envían datos no confiables, como parte de un comando o consulta, permitiendo ejecutar comandos involuntarios o accesos a datos sin la debida autorización.

¿La aplicación es vulnerable?

- Los datos suministrados por el usuario no son validados, filtrados o sanitizados por la aplicación.
- Se invocan consultas dinámicas o no parametrizadas, sin codificar los parámetros.

Cómo se previene

- Utilizar una API segura, que proporcione una interfaz parametrizada (ORM).
- Realizar validaciones de entradas de datos en el servidor.
- Escapar caracteres especiales en cualquier consulta dinámica residual.

Riesgo

Explotación **3 Fácil**

Casi cualquier fuente de datos puede ser un vector de inyección: variables de entorno, parámetros, etc.

Prevalencia **2 Común**

Estos defectos son muy comunes, particularmente en código heredado.

Detección **3 Fácil**

Los errores de inyección son fáciles de descubrir al examinar el código o mediante escáneres.

Impacto **3 Severo**

Una inyección puede causar divulgación, pérdida o corrupción de información.

Demo

Inyección SQL

A2 - Pérdida de Autenticación

A2:2017

Pérdida de Autenticación

La autenticación y gestión de sesiones son implementadas incorrectamente, permitiendo a los atacantes comprometer usuarios y contraseñas, tokens de sesión, o explotar otros errores de implementación para asumir la identidad de otros usuarios.

¿La aplicación es vulnerable?

- Permite ataques automatizados mediante lista, fuerza bruta o credenciales por defecto.
- Almacena las contraseñas en texto claro o cifradas con métodos débiles.
- Expone ID de sesión en las URL, no lo invalida o no lo rota luego de un tiempo o cierre de sesión.

Cómo se previene

- Autenticación multifactor para evitar ataques por fuerza bruta o reúso de credenciales.
- Cambiar credenciales por defecto e implementar controles de contraseñas débiles.
- El ID de sesión no debe incluirse en la URL y debe invalidarse correctamente.

Riesgo

Explotación — **3 Fácil**

Los atacantes tienen acceso a millones de credenciales por fugas de información, así como a credenciales por defecto.

Prevalencia — **2 Común**

Los errores son comunes debido al diseño y la implementación de la mayoría de los controles de acceso.

Detección — **2 Media**

Pueden utilizarse medios manuales y explotarlos utilizando herramientas automatizadas con listas de contraseñas.

Impacto — **3 Severo**

Solo tienen que obtener el acceso a unas pocas cuentas o a una cuenta de administrador para comprometer el sistema.

A3 - Exposición de Datos Sensibles

A3:2017

Exposición de Datos Sensibles

Muchas aplicaciones no protegen adecuadamente los datos sensibles. Los atacantes pueden robar o modificar estos datos protegidos inadecuadamente para llevar a cabo fraudes. Los datos sensibles requieren métodos de protección adicionales, como el cifrado en almacenamiento y tránsito.

¿La aplicación es vulnerable?

- Se transmite datos en texto claro. Esto se refiere a protocolos como HTTP, SMTP, TELNET, FTP.
- Se utilizan algoritmos criptográficos obsoletos o débiles. Por ejemplo MD5, SHA1, etc.
- No se establecen las directivas de seguridad o cabeceras para el navegador web.

Cómo se previene

- Identificar qué información es sensible y aplicar los controles adecuados.
- No almacenar datos sensibles innecesariamente, y cifrarlos cuando sean almacenados.
- Cifrar todos los datos en tránsito utilizando protocolos seguros como TLS.

Riesgo

Explotación **2 Media**

Los atacantes roban claves, ejecutan ataques MiTM, o roban datos del servidor, en tránsito o desde el cliente.

Prevalencia **3 Difundida**

En los últimos años, ha sido el ataque de mayor impacto. El error más común es no cifrar los datos sensibles.

Detección **2 Media**

En los datos en tránsito las debilidades son fáciles de detectar, mientras que para los almacenados es muy difícil.

Impacto **3 Severo**

Los fallos con frecuencia comprometen datos que deberían estar protegidos.

Demo

Muestra de tráfico HTTP

A4 - Entidades Externas XML (XXE)

A4:2017

Entidades Externas XML (XXE)

Muchos procesadores XML antiguos o mal configurados evalúan referencias a entidades externas en documentos XML. Las entidades externas pueden utilizarse para revelar archivos internos en servidores no actualizados, escanear puertos de la LAN, ejecutar código de forma remota y realizar ataques de denegación de servicio (DoS).

¿La aplicación es vulnerable?

- Se acepta XML directamente, carga XML desde fuentes no confiables o inserta datos no confiables.
- Los procesadores XML utilizados en los servicios web SOAP poseen habilitadas las DTDs.

Cómo se previene

- Actualizar los procesadores y bibliotecas XML que utilice la aplicación o el sistema subyacente.
- Deshabilitar las entidades externas de XML y el procesamiento DTD en los analizadores sintácticos.
- De ser posible, utilizar formatos de datos menos complejos, como JSON.

Riesgo

Explotación 2 Media

Pueden explotarse procesadores XML vulnerables incluyendo contenido hostil en un documento XML.

Prevalencia 2 Común

Procesadores XML antiguos permiten especificar una entidad externa, que se referencia y evalúa al procesarse.

Detección 3 Fácil

Existen herramientas SAST y DAST que detectan y explotan estos problemas.

Impacto 3 Severo

Estos errores se pueden utilizar para extraer datos, ejecutar una solicitud remota desde el servidor, etc.

Demo

Acceso a archivos del sistema mediante XXE

A5 - Pérdida de Control de Acceso

A5:2017

Pérdida de Control de Acceso

Las restricciones sobre lo que los usuarios autenticados pueden hacer no se aplican correctamente. Los atacantes pueden explotar estos errores para acceder, de forma no autorizada, a funcionalidades y/o datos, cuentas de otros usuarios, archivos sensibles, etc.

¿La aplicación es vulnerable?

- Se pasan por alto las comprobaciones de la URL, el estado de la aplicación o el HTML.
- Se fuerza la navegación a páginas autenticadas o a páginas privilegiadas como usuario estándar.

Cómo se previene

- Denegar de forma predeterminada, con la excepción de los recursos públicos.
- Implementar los mecanismos de control de acceso una vez y reutilizarlos en toda la aplicación.
- Registrar los errores de control de acceso y alertar a los administradores cuando corresponda.

Riesgo

Explotación 2 Media

Es detectable manualmente, o automáticamente en algunos frameworks o herramientas SAST y DAST.

Prevalencia 2 Común

Comunes debido a la falta de detección automática y a la falta de pruebas funcionales efectivas.

Detección 2 Media

La detección no suele ser cubierta por pruebas automatizadas, tanto estáticas como dinámicas.

Impacto 3 Severo

Atacantes actuando como usuarios o administradores; usuarios que utilizan funciones privilegiadas, etc.

Demo

Acceso no autorizado a secciones de una web

A6 - Configuración de Seguridad Incorrecta

A6:2017

Configuración de Seguridad Incorrecta

La configuración de seguridad incorrecta es un problema muy común y se debe en parte a establecer la configuración de forma manual o por omisión. Por ejemplo: cabeceras HTTP mal configuradas, mensajes de error con contenido sensible, falta de parches y actualizaciones, etc.

¿La aplicación es vulnerable?

- Falta configuración adecuada en cualquier parte del stack tecnológico.
- Se encuentran instaladas o habilitadas características innecesarias.
- Las cuentas predeterminadas y sus contraseñas siguen activas y sin cambios.

Cómo se previene

- Usar una plataforma minimalista sin funcionalidades innecesarias, componentes o documentación.
- Seguir un procedimiento para revisar y actualizar las configuraciones apropiadas.
- Utilizar un proceso automatizado para verificar los ajustes y configuración en todos los entornos.

Riesgo

Explotación **3 Fácil**

Vulnerabilidades sin parchear, acceso a cuentas por defecto, páginas no utilizadas, archivos desprotegidos, etc.

Prevalencia **3 Difundida**

Configuraciones incorrectas de seguridad pueden ocurrir en cualquier nivel del stack tecnológico.

Detección **3 Fácil**

Los escáneres automatizados son útiles para detectar configuraciones erróneas, opciones heredadas, etc.

Impacto **2 Moderado**

Los defectos frecuentemente dan a los atacantes acceso no autorizado a algunos datos o funciones del sistema.

A7 - Cross-Site Scripting (XSS)

A7:2017

Cross-Site Scripting (XSS)

Una aplicación toma datos no confiables y los envía al navegador web sin una validación y codificación apropiada. Permite ejecutar comandos en el navegador de la víctima y el atacante puede secuestrar una sesión, modificar los sitios web o redireccionar al usuario hacia un sitio malicioso.

¿La aplicación es vulnerable?

- XSS Reflejado: se utilizan datos sin validar, codificados como parte del HTML o JavaScript de salida.
- XSS Persistente: se almacenan datos sin validar, utilizados por otro usuario o un administrador.

Cómo se previene

- Mantener los datos no confiables separados del contenido activo del navegador.
- Codificar los datos de los campos de salida HTML.

Riesgo

Explotación **3 Fácil**

Existen herramientas automatizadas que permiten detectar y explotar XSS.

Prevalencia **3 Difundida**

XSS es la segunda vulnerabilidad más frecuente en OWASP Top 10 y se encuentra en dos tercios de las aplicaciones.

Detección **3 Fácil**

Las herramientas automatizadas pueden detectar algunos problemas XSS de forma automática.

Impacto **2 Moderado**

El impacto de XSS es moderado para el caso de XSS Reflejado, y severo para XSS Persistente.

Demo

Captura del ID de sesión mediante XSS y suplantación de identidad

A8 - Deserialización Insegura

A8:2017

Deserialización Insegura

Estos errores ocurren cuando una aplicación recibe objetos serializados dañinos y estos pueden ser manipulados o borrados por el atacante para realizar ataques de repetición, inyecciones o elevar sus privilegios de ejecución. En el peor de los casos, la deserialización insegura puede conducir a la ejecución remota de código en el servidor.

¿La aplicación es vulnerable?

- Deserializa objetos hostiles, manipulados por un atacante.
- Permite ataques relacionados con la estructura de datos y objetos.
- Permite ataques de manipulación de datos.

Cómo se previene

- No aceptar objetos serializados de fuentes no confiables.
- Implementar verificaciones de integridad, como firmas digitales, en cualquier objeto serializado.

Riesgo

Explotación — **1 Difícil**

Los exploits distribuidos raramente funcionan sin cambios o ajustes en su código fuente.

Prevalencia — **2 Común**

Algunas herramientas pueden descubrir defectos de deserialización.

Detección — **2 Media**

Se necesitan más herramientas para ayudar a identificar y abordar estos defectos de deserialización.

Impacto — **3 Severo**

Los errores de deserialización pueden llevar a la ejecución remota de código.

A9 - Componentes con Vulnerabilidades Conocidas

A9:2017

Uso de Componentes con Vulnerabilidades Conocidas

Los componentes como bibliotecas, frameworks y otros módulos se ejecutan con los mismos privilegios que la aplicación. Si se explota un componente vulnerable, el ataque puede provocar una pérdida de datos o tomar el control del servidor.

¿La aplicación es vulnerable?

- Se desconocen las versiones de todos los componentes que utiliza, incluidas dependencias.
- El software es vulnerable, no posee soporte o se encuentra desactualizado.
- No se asegura la configuración de los componentes correctamente.

Cómo se previene

- Remover dependencias, funcionalidades, componentes y archivos y documentación innecesaria.
- Asegurar la existencia de un plan para monitorizar, evaluar y aplicar actualizaciones o cambios de configuraciones durante el ciclo de vida de las aplicaciones de la organización.

Riesgo

Explotación 2 Media

Es sencillo obtener exploits para vulnerabilidades conocidas, pero otras requieren de esfuerzo.

Prevalencia 3 Difundida

Estos defectos están muy difundidos. Se pierde el control de componentes utilizados y no se actualizan.

Detección 2 Media

Esta debilidad es detectable mediante el uso de analizadores o la inspección de cabeceras.

Impacto 2 Moderado

Muchos ataques han sido realizados explotando vulnerabilidades conocidas en componentes comunes.

A10 - Registro y Monitoreo Insuficientes

A10:2017

Registro y Monitoreo Insuficientes

El registro y monitoreo insuficiente, junto a la falta de respuesta ante incidentes permiten a los atacantes mantener el ataque en el tiempo, pivotear a otros sistemas y manipular, extraer o destruir datos.

¿La aplicación es vulnerable?

- Los registros no son monitoreados para detectar actividades sospechosas.
- Las pruebas de penetración y escaneos no generan alertas.
- La aplicación no logra detectar, escalar o alertar sobre ataques en tiempo real.

Cómo se previene

- Asegurar que los errores de inicio de sesión y control de acceso se pueden registrar.
- Establecer la monitorización y alerta para que las actividades sospechosas sean detectadas.
- Adoptar un plan de respuesta o recuperación de incidentes.

Riesgo

Explotación — **2 Media**

El registro y monitoreo insuficientes es la base de casi todos los mayores incidentes de seguridad.

Prevalencia — **3 Difundida**

Puede identificarse un nivel insuficiente examinando los registros después de las pruebas de penetración.

Detección — **1 Difícil**

Las acciones deben registrarse lo suficiente como para comprender los daños que podrían haber causado.

Impacto — **2 Moderado**

Permitir que el sondeo de vulnerabilidades continúe puede aumentar la probabilidad de una explotación exitosa.

¡Gracias!

¿Alguna pregunta?