

Seguridad en la Red Informática Mundial

Top-Ten Vulnerabilidades según OWASP

Semana 13 clases 25 y 26

Mtra. María Noemí Araiza Ramírez

Top-Ten Vulnerabilidades según OWASP



Objetivos:

¿Qué es OWASP Top 10?

¿Qué objetivos y motivaciones tiene?

¿Qué ha cambiado de 2013 a 2017?

¿Qué son los Riesgos de Seguridad de Aplicaciones?

¿Conocer cómo me afectan o cuál es mi riesgo?

¿Conocer los 10 Riesgos más importantes?

OWASP Top 10 2013	±	OWASP Top 10 2017
A1 – Inyección	→	A1:2017 – Inyección
A2 – Pérdida de Autenticación y Gestión de Sesiones	→	A2:2017 – Pérdida de Autenticación y Gestión de Sesiones
A3 – Secuencia de Comandos en Sitios Cruzados (XSS)	↘	A3:2017 – Exposición de Datos Sensibles
A4 – Referencia Directa Insegura a Objetos [Unido+A7]	U	A4:2017 – Entidad Externa de XML (XXE) [NUEVO]
A5 – Configuración de Seguridad Incorrecta	↘	A5:2017 – Pérdida de Control de Acceso [Unido]
A6 – Exposición de Datos Sensibles	↗	A6:2017 – Configuración de Seguridad Incorrecta
A7 – Ausencia de Control de Acceso a las Funciones [Unido+A4]	U	A7:2017 – Secuencia de Comandos en Sitios Cruzados (XSS)
A8 – Falsificación de Peticiones en Sitios Cruzados (CSRF)	✗	A8:2017 – Deserialización Insegura [NUEVO, Comunidad]
A9 – Uso de Componentes con Vulnerabilidades Conocidas	→	A9:2017 – Uso de Componentes con Vulnerabilidades Conocidas
A10 – Redirecciones y reenvíos no validados	✗	A10:2017 – Registro y Monitoreo Insuficientes [NUEVO, Comunidad]

Top-Ten Vulnerabilidades según OWASP



¿Cuál es mi Riesgo?

Top 10 2013

Agente de Amenaza	Vectores de Ataque	Prevalencia de Debilidades	Detectabilidad de Debilidades	Impacto Técnico	Impacto al Negocio
Específico de la aplicación	Fácil	Difundido	Fácil	Severo	Específico de la aplicación /negocio
	Promedio	Común	Promedio	Moderado	
	Difícil	Poco Común	Difícil	Menor	

Agente de Amenaza	Explotabilidad	Prevalencia de Vulnerabilidad	Detección de Vulnerabilidad	Impacto Técnico	Impacto de Negocio
Específico de la Aplicación	Fácil 3	Difundido 3	Fácil 3	Severo 3	Específico del Negocio
	Promedio 2	Común 2	Promedio 2	Moderado 2	
	Difícil 1	Poco Común 1	Difícil 1	Mínimo 1	

Top 10 2017

A6

Exposición de datos sensibles

Recordemos que está ubicada en la tabla de amenazas de 2013, para 2017 esta amenaza pasa a ser la amenaza 3.

A3
:2017

Exposición de Datos Sensibles

A6

Exposición de datos sensibles

Muchas aplicaciones web no protegen adecuadamente datos sensibles tales como números de tarjetas de crédito o credenciales de autenticación.

Los atacantes pueden robar o modificar tales datos para llevar a cabo fraudes, robos de identidad u otros delitos.

Los datos sensibles requieren de métodos de protección adicionales tales como el cifrado de datos, así como también de precauciones especiales en un intercambio de datos con el navegador.

Top-Ten Vulnerabilidades según OWASP



A6

Exposición de datos sensibles

A3
:2017

Exposición de Datos Sensibles

Muchas aplicaciones web y APIs no protegen adecuadamente datos sensibles, tales como información financiera, de salud o Información Personalmente Identificable (PII).

Los atacantes pueden robar o modificar estos datos protegidos inadecuadamente para llevar a cabo fraudes con tarjetas de crédito, robos de identidad u otros delitos.

Los datos sensibles requieren métodos de protección adicionales, como el cifrado en almacenamiento y tránsito.

A6

Exposición de datos sensibles


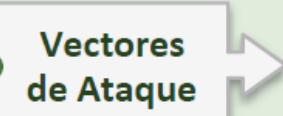
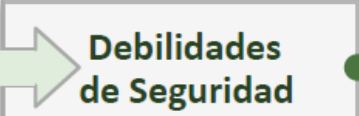


Este es un problema de autorización, que resulta muy sencillo de explotar.

Como resultado, el usuario es capaz de acceder a un objeto del sistema al que no está autorizado, teniendo como impacto la exposición de datos privados hacia el resto de usuarios de la aplicación.

Ocurre cuando un desarrollador expone al exterior una referencia a un objeto de implementación interno, tal como un archivo, directorio, o base de datos.

Top-Ten Vulnerabilidades según OWASP

A6 Exposición de datos sensibles

 Agentes de Amenaza	 Vectores de Ataque	 Debilidades de Seguridad		 Impactos Técnicos	 Impactos al negocio
Específico de la Aplicación	Explotabilidad DIFÍCIL	Prevalencia NO COMÚN	Detección PROMEDIO	Impacto SEVERO	Específico de la Aplicación/Negocio
<p>Considere quién puede obtener acceso a sus datos sensibles y cualquier respaldo de éstos. Esto incluye los datos almacenados, en tránsito, e inclusive en el navegador del cliente. Incluye tanto amenazas internas y externas.</p>	<p>Los atacantes típicamente no quiebran la criptografía de forma directa, sino algo más como robar claves, realizar ataques “man in the middle”, robar datos en texto claro del servidor, mientras se encuentran en tránsito, o del navegador del usuario.</p>	<p>La debilidad más común es simplemente no cifrar datos sensibles. Cuando se emplea cifrado, es común detectar generación y gestión débiles de claves, el uso de algoritmos débiles, y particularmente técnicas débiles de hashing de contraseñas. Las debilidades a nivel del navegador son muy comunes y fáciles de detectar, pero difíciles de explotar a gran escala. Atacantes externos encuentran dificultades detectando debilidades en a nivel de servidor dado el acceso limitado y que son usualmente difíciles de explotar.</p>		<p>Los fallos frecuentemente comprometen todos los datos que deberían estar protegidos. Típicamente, esta información incluye datos sensibles como ser registros médicos, credenciales, datos personales, tarjetas de crédito, etc.</p>	<p>Considere el valor de negocio de la pérdida de datos y el impacto a su reputación. ¿Cuál su responsabilidad legal si estos datos son expuestos? También considere el daño a la reputación.</p>

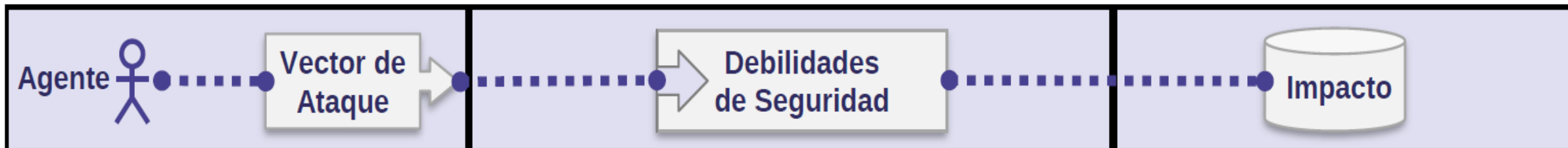
Top-Ten Vulnerabilidades según OWASP

A6

Exposición de datos sensibles

A3
:2017

Exposición de Datos Sensibles

					
App. Específica	Explotabilidad: 2	Prevalencia 3	Detectabilidad: 2	Técnico: 3	¿Negocio?
<p>En lugar de atacar la criptografía, los atacantes roban claves, ejecutan ataques <i>Man in the Middle</i> o roban datos en texto plano del servidor, en tránsito, o desde el cliente. Se requiere un ataque manual pero pueden utilizarse bases de datos con <i>hashes</i> que han sido hechas públicas para obtener las contraseñas originales utilizando GPUs.</p>		<p>En los últimos años, este ha sido el ataque de mayor impacto. El error más común es simplemente no cifrar los datos sensibles. Cuando se emplea criptografía, es común la generación y gestión de claves, algoritmos, cifradores y protocolos débiles. En particular algoritmos débiles de <i>hashing</i> para el almacenamiento de contraseñas. Para los datos en tránsito las debilidades son fáciles de detectar, mientras que para los datos almacenados es muy difícil. Ambos tienen una explotabilidad muy variable.</p>		<p>Los fallos con frecuencia comprometen datos que deberían estar protegidos. Típicamente, esta información incluye Información Personal Sensible (PII) como registros de salud, datos personales, credenciales y tarjetas de crédito, que a menudo requieren mayor protección, según lo definido por las leyes o reglamentos como el PIBR de la UE o las leyes locales de privacidad.</p>	

A6

Exposición de datos sensibles

¿Soy vulnerable?

Lo primero que debe determinar es el conjunto de datos sensibles que requerirán protección extra. Por ejemplo, contraseñas, números de tarjetas de crédito, registros médicos, e información personal deberían protegerse. Para estos datos:

1. ¿Se almacenan en texto claro a largo plazo, incluyendo sus respaldos?

A6

Exposición de datos sensibles

¿Soy vulnerable?

2. ¿Se transmite en texto claro, interna o externamente? El tráfico por Internet es especialmente peligroso.
3. ¿Se utiliza algún algoritmo criptográfico débil/antiguo?
4. ¿Se generan claves criptográficas débiles, o falta una adecuada rotación o gestión de claves?
5. ¿Se utilizan tanto cabecales como directivas de seguridad del navegador cuando son enviados o provistos por el mismo?

A6

Exposición de datos sensibles

¿La aplicación es vulnerable? (Top 10 2017)

Lo primero es determinar las necesidades de protección de los datos en tránsito y en almacenamiento. Por ejemplo, contraseñas, números de tarjetas de crédito, registros médicos, información personal y datos sensibles del negocio requieren protección adicional, especialmente si se encuentran en el ámbito de aplicación de leyes de privacidad, como por ejemplo el Reglamento General de Protección de Datos (RGPD) o regulaciones financieras, como PCI Data Security Standard (PCI DSS).

A6

Exposición de datos sensibles

¿La aplicación es vulnerable? (Top 10 2017)

Para todos estos datos:

- ¿Se transmite datos en texto claro? Esto se refiere a protocolos como HTTP, SMTP, TELNET, FTP. El tráfico en Internet es especialmente peligroso. Verifique también todo el tráfico interno, por ejemplo, entre los balanceadores de carga, servidores web o sistemas de backend.
- ¿Se utilizan algoritmos criptográficos obsoletos o débiles, ya sea por defecto o en código heredado? Por ejemplo MD5, SHA1, etc.

A6

Exposición de datos sensibles

¿La aplicación es vulnerable? (Top 10 2017)

- ¿Se utilizan claves criptográficas predeterminadas, se generan o reutilizan claves criptográficas débiles, o falta una gestión o rotación adecuada de las claves?
- Por defecto, ¿se aplica cifrado? ¿se han establecido las directivas de seguridad o encabezados para el navegador web?
- ¿El User-Agent del usuario (aplicación o cliente de correo), verifica que el certificado enviado por el servidor sea válido?.

A6

Exposición de datos sensibles

¿Cómo se puede evitar?

Los riesgos completos de utilizar cifrado de forma no segura, uso de SSL, y protección de datos escapan al alcance del Top 10.

Dicho esto, para los datos sensibles, se deben realizar como mínimo lo siguiente:

1. Considere las amenazas de las cuáles protegerá los datos (ej: atacante interno, usuario externo), asegúrese de cifrar los datos sensibles almacenados o en tráfico de manera de defenderse de estas amenazas.
 2. No almacene datos sensibles innecesariamente. Descártelos apenas sea posible. Datos que no se poseen no pueden ser robados.
-

A6

Exposición de datos sensibles

¿Cómo se puede evitar?

3. Asegúrese de aplicar algoritmos de cifrado fuertes y estándar así como claves fuertes y gestiónelas de forma segura. Considere utilizar módulos criptográficos validados como FIPS 140
4. Asegúrese que las claves se almacenan con un algoritmo especialmente diseñado para protegerlas como ser bcrypt, PBKDF2 o scrypt.
5. Deshabilite el autocompletar en los formularios que recolectan datos sensibles. Deshabilite también el cacheado de páginas que contengan datos sensibles.

A6

Exposición de datos sensibles

¿Cómo se previene? (Top 10 2017)

Como mínimo, siga las siguientes recomendaciones y consulte las referencias:

- Clasifique los datos procesados, almacenados o transmitidos por el sistema. Identifique qué información es sensible de acuerdo a las regulaciones, leyes o requisitos del negocio y del país.
 - Aplique los controles adecuados para cada clasificación.
 - Cifre todos los datos sensibles cuando sean almacenados.
-

A6

Exposición de datos sensibles

¿Cómo se previene? (Top 10 2017)

- No almacene datos sensibles innecesariamente. Descártelos tan pronto como sea posible o utilice un sistema de tokenización que cumpla con PCI DSS. Los datos que no se almacenan no pueden ser robados.
- Cifre todos los datos en tránsito utilizando protocolos seguros como TLS con cifradores que utilicen Perfect Forward Secrecy (PFS), priorizando los algoritmos en el servidor. Aplique el cifrado utilizando directivas como HTTP Strict Transport Security (HSTS).

A6

Exposición de datos sensibles

¿Cómo se previene? (Top 10 2017)

- Utilice únicamente algoritmos y protocolos estándares y fuertes e implemente una gestión adecuada de claves. No cree sus propios algoritmos de cifrado.
 - Deshabilite el almacenamiento en cache de datos sensibles.
 - Almacene contraseñas utilizando funciones de hashing adaptables con un factor de trabajo (retraso) además de SALT, como Argon2, scrypt, bcrypt o PBKDF2.
 - Verifique la efectividad de sus configuraciones y parámetros de forma independiente.
-

A6

Exposición de datos sensibles

Ejemplos de escenarios de ataques

Escenario 1: Una aplicación cifra los números de tarjetas de crédito en una base de datos utilizando cifrado automático de la base de datos.

Esto significa que también se descifra estos datos automáticamente cuando se recuperan, permitiendo por medio de una debilidad de inyección de SQL recuperar números de tarjetas en texto claro.

El sistema debería cifrar dichos número usando una clave pública, y permitir solamente a las aplicaciones de backend descifrarlo con la clave privada.

A6

Exposición de datos sensibles

Ejemplos de escenarios de ataques

Escenario 2: Un sitio simplemente no utiliza SSL para todas sus páginas que requieren autenticación.

El atacante monitorea el tráfico en la red (como ser una red inalámbrica abierta), y obtiene la cookie de sesión del usuario.

El atacante reenvía la cookie y secuestra la sesión, accediendo los datos privados del usuario.

A6

Exposición de datos sensibles

Ejemplos de escenarios de ataques

Escenario 3: La base de datos de claves usa hashes sin salt para almacenar las claves.

Una falla en una subida de archivo permite a un atacante obtener el archivo de claves.

Todas las claves pueden ser expuestas mediante una tabla rainbow de hashes precalculados.

A6

Exposición de datos sensibles

Ejemplos de escenarios de ataques (Top 10 2017)

Escenario 1: una aplicación cifra números de tarjetas de crédito en una base de datos utilizando su cifrado automático.

Sin embargo, estos datos son automáticamente descifrados al ser consultados, permitiendo que, si existe un error de inyección SQL se obtengan los números de tarjetas de crédito en texto plano.

A6

Exposición de datos sensibles

Ejemplos de escenarios de ataques (Top 10 2017)

Escenario 2: un sitio web no utiliza o fuerza el uso de TLS para todas las páginas, o utiliza cifradores débiles.

Un atacante monitorea el tráfico de la red (por ejemplo en una red Wi-Fi insegura), degrada la conexión de HTTPS a HTTP e intercepta los datos, robando las cookies de sesión del usuario.

El atacante reutiliza estas cookies y secuestra la sesión del usuario (ya autenticado), accediendo o modificando datos privados. También podría alterar los datos enviados.

A6

Exposición de datos sensibles

Ejemplos de escenarios de ataques (Top 10 2017)

Escenario 3: se utilizan hashes simples o hashes sin SALT para almacenar las contraseñas de los usuarios en una base de datos.

Una falla en la carga de archivos permite a un atacante obtener las contraseñas.

Utilizando una Rainbow Table de valores precalculados, se pueden recuperar las contraseñas originales.



Inexistente Control de Acceso a nivel de funcionalidades

Recordemos que está ubicada en la tabla de amenazas de 2013, sin embargo esta amenaza se unió a la amenaza 4 y pasa a ser la amenaza 5 en 2017.



Pérdida de Control de Acceso

Top-Ten Vulnerabilidades según OWASP



A7

Inexistente Control de Acceso a nivel de funcionalidades

La mayoría de aplicaciones web verifican los derechos de acceso a nivel de función antes de hacer visible en la misma interfaz de usuario.

A pesar de esto, las aplicaciones necesitan verificar el control de acceso en el servidor cuando se accede a cada función.

Si las solicitudes de acceso no se verifican, los atacantes podrán realizar peticiones sin la autorización apropiada.

Top-Ten Vulnerabilidades según OWASP



A7

Inexistente Control de Acceso a nivel de funcionalidades

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 defectos para acceder, de forma no autorizada, a funcionalidades y/o datos, cuentas de otros usuarios, ver archivos sensibles, modificar datos, cambiar derechos de acceso y permisos, etc.

Top-Ten Vulnerabilidades según OWASP



A7

Inexistente Control de Acceso a nivel de funcionalidades






Hablamos de este tipo de ataque cuando en nuestro servicio hay al menos alguna URL o dato que nos permite saltarnos las verificaciones oportunas.

Quizá el mejor ejemplo sean los ataques mediante buscadores (Google, Bing), cuando el interesado es capaz de acceder a una web por algún camino en el que sin necesidad de loguearse ya aparece como tal (quizá el haber enlazado a una página de reseteo de contraseña no dinámica).

Top-Ten Vulnerabilidades según OWASP

A7

**Inexistente Control de Acceso
a nivel de funcionalidades**

 Agentes de Amenaza	 Vectores de Ataque	 Debilidades de Seguridad		 Impactos Técnicos	 Impactos al negocio
Específico de la Aplicación	Explotabilidad FÁCIL	Prevalencia COMÚN	Detección PROMEDIO	Impacto MODERADO	Específico de la aplicación/negocio
<p>Cualquiera con acceso a la red puede enviar una petición a su aplicación. ¿Un usuario anónimo podría acceder a una funcionalidad privada o un usuario normal acceder a una función que requiere privilegios?</p>	<p>El atacante, que es un usuario legítimo en el sistema, simplemente cambia la URL o un parámetro a una función con privilegios. ¿Se le concede acceso? Usuarios anónimos podrían acceder a funcionalidades privadas que no estén protegidas.</p>	<p>Las aplicaciones no siempre protegen las funcionalidades adecuadamente. En ocasiones la protección a nivel de funcionalidad se administra por medio de una configuración, y el sistema está mal configurado. Otras veces los programadores deben incluir un adecuado chequeo por código, y se olvidan.</p> <p>La detección de este tipo de vulnerabilidad es sencillo. La parte más compleja es identificar qué páginas (URLs) o funcionalidades atacables existen.</p>		<p>Estas vulnerabilidades permiten el acceso no autorizado de los atacantes a funciones del sistema.</p> <p>Las funciones administrativas son un objetivo clave de este tipo de ataques.</p>	<p>Considere el valor para su negocio de las funciones expuestas y los datos que éstas procesan. Además, considere el impacto a su reputación si esta vulnerabilidad se hiciera pública.</p>

Top-Ten Vulnerabilidades según OWASP

A7

Inexistente Control de Acceso
a nivel de funcionalidades

A5
:2017

Pérdida de Control de Acceso

App. Específica	Explotabilidad: 2	Prevalencia: 2	Detectabilidad: 2	Técnico: 3	¿Negocio?
<p>La explotación del control de acceso es una habilidad esencial de los atacantes. Las herramientas SAST y DAST pueden detectar la ausencia de controles de acceso pero, en el caso de estar presentes, no pueden verificar si son correctos. Es detectable utilizando medios manuales o de forma automática en algunos <i>frameworks</i> que carecen de controles de acceso.</p>		<p>Las debilidades del control de acceso son comunes debido a la falta de detección automática y a la falta de pruebas funcionales efectivas por parte de los desarrolladores de aplicaciones.</p> <p>La detección de fallas en el control de acceso no suele ser cubierto por pruebas automatizadas, tanto estáticas como dinámicas.</p>		<p>El impacto técnico incluye atacantes anónimos actuando como usuarios o administradores; usuarios que utilizan funciones privilegiadas o crean, acceden, actualizan o eliminan cualquier registro.</p> <p>El impacto al negocio depende de las necesidades de la aplicación y de los datos.</p>	

A7

Inexistente Control de Acceso a nivel de funcionalidades

¿Soy vulnerable?

La mejor manera de determinar si una aplicación falla en restringir adecuadamente el acceso a nivel de funcionalidades es verificar cada funcionalidad de la aplicación:

1. ¿La interfaz de usuario (UI) muestra la navegación hacia funcionalidades no autorizadas?
 2. ¿Existe autenticación del lado del servidor, o se han perdido las comprobaciones de autorización?
 3. ¿Los controles del lado del servidor se basan exclusivamente en la información proporcionada por el atacante?
-

A7

Inexistente Control de Acceso a nivel de funcionalidades

¿Soy vulnerable?

Usando un proxy, navegue su aplicación con un rol privilegiado. Luego visite reiteradamente páginas restringidas usando un rol con menos privilegios. Si el servidor responde a ambos por igual, probablemente es vulnerable. Algunas pruebas de Proxies apoyan directamente este tipo de análisis.

También puede revisar la implementación del control de acceso en el código. Intente seguir una solicitud unitaria y con privilegios a través del código y verifique el patrón de autorización. Luego busque en el código para detectar donde no se está siguiendo ese patrón.

Las herramientas automatizadas no suelen encontrar estos problemas.

A7

**Inexistente Control de Acceso
a nivel de funcionalidades**

¿La aplicación es vulnerable? (Top 10 2017)

Las restricciones de control de acceso implican que los usuarios no pueden actuar fuera de los permisos previstos.

Típicamente, las fallas conducen a la divulgación, modificación o destrucción de información no autorizada de los datos, o a realizar una función de negocio fuera de los límites del usuario.

A7

Inexistente Control de Acceso a nivel de funcionalidades

¿La aplicación es vulnerable? (Top 10 2017)

Las vulnerabilidades comunes de control de acceso incluyen:

- Pasar por alto las comprobaciones de control de acceso modificando la URL, el estado interno de la aplicación o HTML, utilizando una herramienta de ataque o una conexión vía API.
 - Permitir que la clave primaria se cambie a la de otro usuario, pudiendo ver o editar la cuenta de otra persona.
 - Elevación de privilegios. Actuar como un usuario sin iniciar sesión, o actuar como un administrador habiendo iniciado sesión como usuario estándar.
-

Top-Ten Vulnerabilidades según OWASP



A7

**Inexistente Control de Acceso
a nivel de funcionalidades**

¿La aplicación es vulnerable? (Top 10 2017)

- Manipulación de metadatos, como reproducir un token de control de acceso JWT (JSON Web Token), manipular una cookie o un campo oculto para elevar los privilegios, o abusar de la invalidación de tokens JWT.
- La configuración incorrecta de CORS permite el acceso no autorizado a una API.

Top-Ten Vulnerabilidades según OWASP



A7

**Inexistente Control de Acceso
a nivel de funcionalidades**

¿La aplicación es vulnerable? (Top 10 2017)

- Forzar la navegación a páginas autenticadas como un usuario no autenticado o a páginas privilegiadas como usuario estándar.
- Acceder a una API sin control de acceso mediante el uso de verbos POST, PUT y DELETE.

A7

Inexistente Control de Acceso a nivel de funcionalidades

¿Cómo se puede evitar?

La aplicación debería tener un módulo de autorización consistente y fácil de analizar, invocado desde todas las funciones de negocio. Frecuentemente, esa protección es provista por uno o más componentes externos al código de la aplicación.

1. El proceso para gestión de accesos y permisos debería ser actualizable y auditable fácilmente. No lo implemente directamente en el código sin utilizar parametrizaciones.

A7

Inexistente Control de Acceso a nivel de funcionalidades

¿Cómo se puede evitar?

2. La implementación del mecanismo debería negar todo acceso por defecto, requiriendo el establecimiento explícito de permisos a roles específicos para acceder a cada funcionalidad.
3. Si la funcionalidad forma parte de un workflow, verifique y asegúrese que las condiciones del flujo se encuentren en el estado apropiado para permitir el acceso.

NOTA: La mayoría de las aplicaciones web no despliegan links o botones para funciones no autorizadas, pero en la práctica el “control de acceso de la capa de presentación” no provee protección. Ud. Debería implementar chequeos en los controladores y/o lógicas de negocios.

A7

Inexistente Control de Acceso a nivel de funcionalidades

¿Cómo se previene? (Top 2017)

El control de acceso sólo es efectivo si es aplicado del lado del servidor o en Server-less API, donde el atacante no puede modificar la verificación de control de acceso o los metadatos.

- Con la excepción de los recursos públicos, la política debe ser denegar de forma predeterminada.
 - Implemente los mecanismos de control de acceso una vez y reutilícelo en toda la aplicación, incluyendo minimizar el control de acceso HTTP (CORS).
-

A7

Inexistente Control de Acceso a nivel de funcionalidades

¿Cómo se previene? (Top 10 2017)

- Los controles de acceso al modelo deben imponer la propiedad (dueño) de los registros, en lugar de aceptar que el usuario puede crear, leer, actualizar o eliminar cualquier registro.
 - Los modelos de dominio deben hacer cumplir los requisitos exclusivos de los límites de negocio de las aplicaciones.
 - Deshabilite el listado de directorios del servidor web y asegúrese que los metadatos/fuentes de archivos (por ejemplo de GIT) y copia de seguridad no estén presentes en las carpetas públicas.
-

A7

Inexistente Control de Acceso a nivel de funcionalidades

¿Cómo se previene? (Top 10 2017)

- Registre errores de control de acceso y alerte a los administradores cuando corresponda (por ej. fallas reiteradas).
 - Limite la tasa de acceso a las APIs para minimizar el daño de herramientas de ataque automatizadas.
 - Los tokens JWT deben ser invalidados luego de la finalización de la sesión por parte del usuario.
 - Los desarrolladores y el personal de QA deben incluir pruebas de control de acceso en sus pruebas unitarias y de integración.
-

A7

**Inexistente Control de Acceso
a nivel de funcionalidades**

Ejemplos de escenarios de ataques

Escenario #1: El atacante simplemente fuerza la navegación hacia las URLs objetivo. La siguiente URLs requiere autenticación. Los derechos de administrador también son requeridos para el acceso a la página “admin_getappInfo”.

<http://example.com/app/getappInfo>

http://example.com/app/admin_getappInfo

Si un usuario no autenticado puede acceder a ambas páginas, eso es una vulnerabilidad. Si un usuario autenticado, no administrador, puede acceder a “admin_getappInfo”, también es una vulnerabilidad, y podría llevar al atacante a más páginas de administración protegidas inadecuadamente.

A7

**Inexistente Control de Acceso
a nivel de funcionalidades**

Ejemplos de escenarios de ataques

Escenario 2:

Una página proporciona un parámetro de “acción” para especificar la función que ha sido invocada, y diferentes acciones requieren diferentes roles.

Si estos roles no se verifican al invocar la acción, es una vulnerabilidad.

A7

Inexistente Control de Acceso a nivel de funcionalidades

Ejemplos de escenarios de ataques (Top 10 2017)

Escenario 1: la aplicación utiliza datos no validados en una llamada SQL para acceder a información de una cuenta:

```
pstmt.setString(1, request.getParameter("acct"));  
ResultSet results = pstmt.executeQuery();
```

Un atacante simplemente puede modificar el parámetro “acct” en el navegador y enviar el número de cuenta que desee. Si no se verifica correctamente, el atacante puede acceder a la cuenta de cualquier usuario:

```
http://example.com/app/accountInfo?acct=notmyacct
```

A7

**Inexistente Control de Acceso
a nivel de funcionalidades**

Ejemplos de escenarios de ataques (Top 10 2017)

Escenario 2: un atacante simplemente fuerza las búsquedas en las URL. Los privilegios de administrador son necesarios para acceder a la página de administración:

<http://example.com/app/getappInfo>

http://example.com/app/admin_getappInfo

Si un usuario no autenticado puede acceder a cualquier página o, si un usuario no-administrador puede acceder a la página de administración, esto es una falla.

Referencias

<https://www.owasp.org>