

Seguridad en aplicaciones WEB

Master de redes de datos Seguridad y Privacidad en redes

Nicolás Macia - nmacia@cert.unlp.edu.ar

Paula Venosa - pvenosa@cert.unlp.edu.ar

Einar Lanfranco - elanfranco@cert.unlp.edu.ar



Seguridad en aplicaciones WEB

- Tendencia en los ataques
- Problemas en una aplicación web atenta contra:
 - La reputación de la Organización que aloja dicha aplicación.
 - La disponibilidad del servicio que da a sus usuarios.
 - La confidencialidad y la integridad de los datos.
 - El uso responsable del servidor.
- OWASP (Open Web Application Security Project):
 - Es un proyecto conformado por una comunidad mundial libre y abierta centrada en mejorar la seguridad del software.
 - Tiene como fin ayudar a las organizaciones a desarrollar y mantener aplicaciones confiables.



Seguridad en aplicaciones WEB

¿Qué ofrece OWASP? - <http://www.owasp.org>

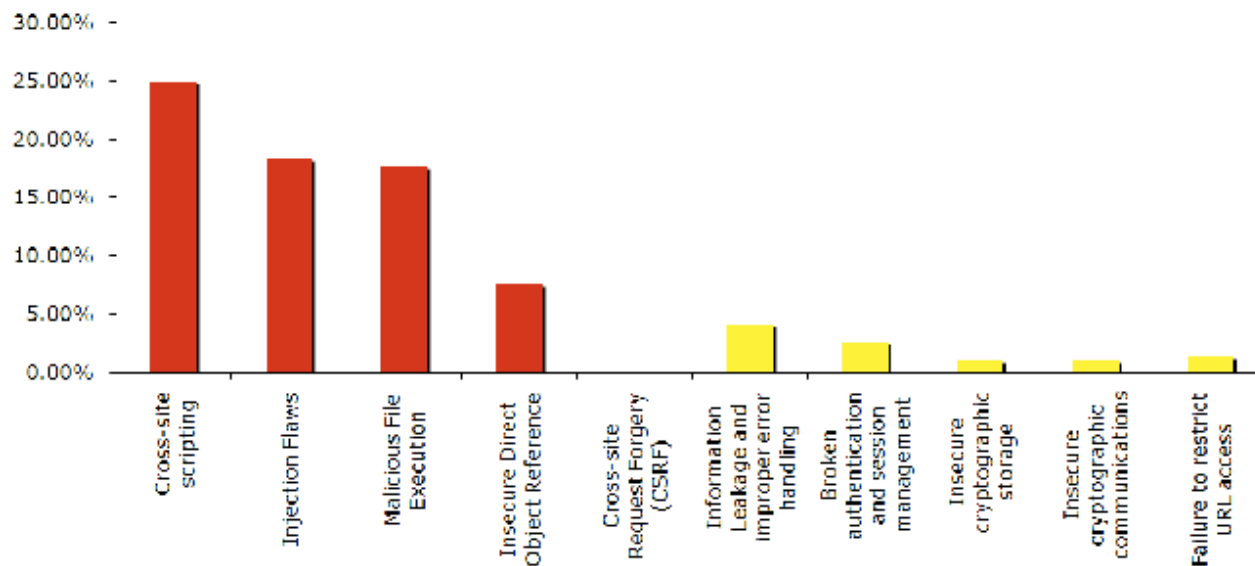
- Top Ten
- Guia de desarrollo: como construir aplicaciones WEB seguras. Mejores prácticas. Recomendaciones para el programador.
- Guia de Testing de aplicaciones: como realizar tests de seguridad en las aplicaciones web
- Guia de revisión de código
- Guia de referencia rápida sobre prácticas de codificación segura.
- Herramientas: webscarab (proxy) y webgoat (learning), zed attack proxy, etc.
- APIs de seguridad para diferentes lenguajes.
- Mucho mas....



Seguridad en aplicaciones WEB

OWASP Top Ten 2007:

- Incluye los problemas de seguridad más críticos presentes en las aplicaciones WEB, según las vulnerabilidades reportadas por MITRE en 2006



Seguridad en aplicaciones WEB

OWASP Top Ten 2007:

- Cross site scripting
- Injection flaws
- Malicious file execution
- Insecure Direct Object Reference
- Cross Site Request Forgery (CSRF)
- Information Leakage and Improper Error Handling
- Broken Authentication and Session Management
- Insecure Cryptographic Storage
- Insecure Communications
- Failure to Restrict URL Access



Seguridad en aplicaciones WEB

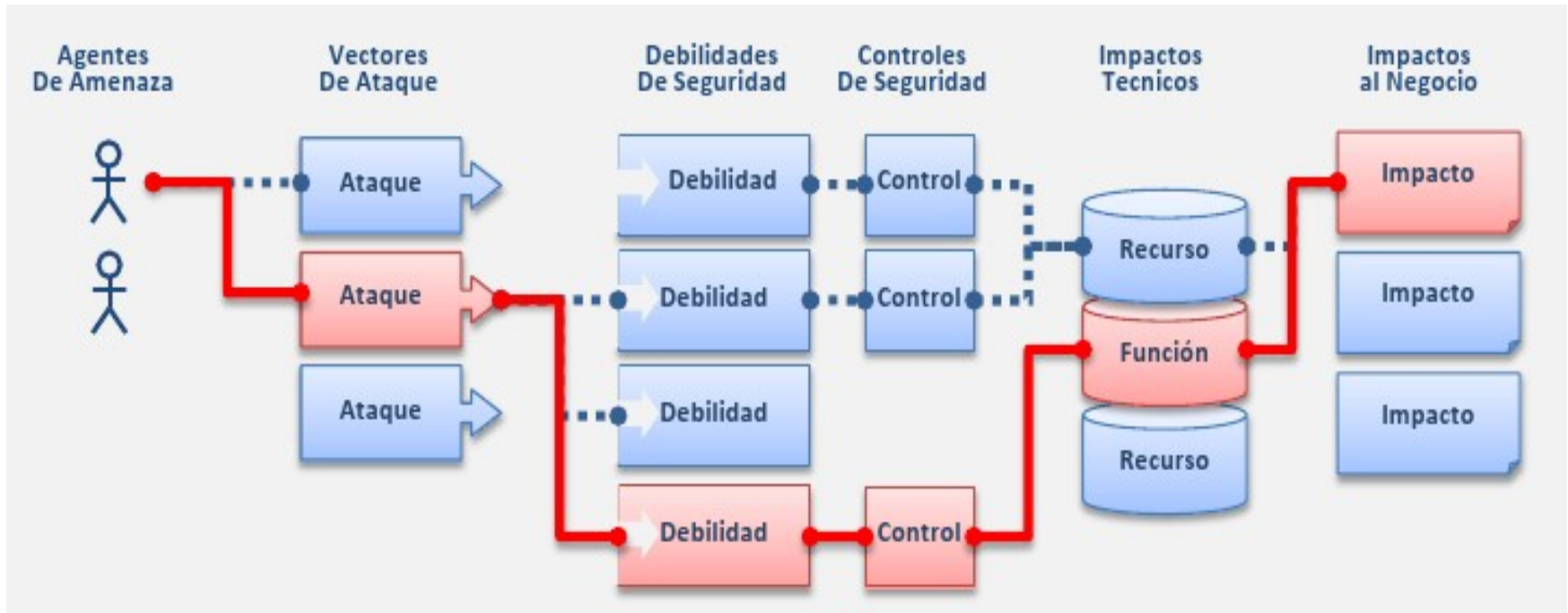
OWASP Top Ten 2010:

El **Top 10 de OWASP 2010** se centra en riesgos.

Cambia la metodología de clasificación. Se estima el riesgo, en lugar de basarnos solamente en la frecuencia de la ocurrencia del problema.

Overall Risk Severity				
Impact	HIGH	Medium	High	Critical
	MEDIUM	Low	Medium	High
	LOW	Note	Low	Medium
		LOW	MEDIUM	HIGH
	Likelihood			

Seguridad en aplicaciones WEB



¿Cuál es mi riesgo?

http://www.owasp.org/index.php/OWASP_Risk_Rating_Methodology

Seguridad en aplicaciones WEB

OWASP Top Ten 2010:

- A1: Injection
- A2: Cross-Site Scripting (XSS)
- A3: Broken Authentication and Session Management
- A4: Insecure Direct Object References
- A5: Cross-Site Request Forgery (CSRF)
- A6: Security Misconfiguration
- A7: Insecure Cryptographic Storage
- A8: Failure to Restrict URL Access
- A9: Insufficient Transport Layer Protection
- A10: Unvalidated Redirects and Forwards



Seguridad en aplicaciones WEB

OWASP Top Ten 2010

A1: Injection

A2: Cross-Site Scripting (XSS)

A3: Broken Authentication and Session Management

A4: Insecure Direct Object References

A5: Cross Site Request Forgery (CSRF)

A6: Security Misconfiguration

A7: Failure to Restrict URL Access

A8: Insecure Cryptographic Storage

A9: Insufficient Transport Layer Protection

A10: Unvalidated Redirects and Forwards



OWASP

The Open Web Application Security Project
<http://www.owasp.org>

http://www.owasp.org/index.php/Top_10

Seguridad en aplicaciones WEB

OWASP Top 10 – 2007 (Anterior)	OWASP Top 10 – 2010 (Nuevo)
A2 – Fallas de Inyección	↑ A1 – Inyección
A1 – Secuencia de Comandos en Sitios Cruzados (XSS)	↓ A2 – Secuencia de Comandos en Sitios Cruzados (XSS)
A7 – Pérdida de Autenticación y Gestión de Sesiones	↑ A3 – Pérdida de Autenticación y Gestión de Sesiones
A4 – Referencia Directa Insegura a Objetos	= A4 – Referencia Directa Insegura a Objetos
A5 – Falsificación de Petición en Sitios Cruzados (CSRF)	= A5 – Falsificación de Petición en Sitios Cruzados (CSRF)
<era T10 2004 A10 – Gestión Insegura de la Configuración>	+ A6 – Configuración Defectuosa de Seguridad (NUEVO)
A8 – Almacenamiento Criptográfico Inseguro	↑ A7 – Almacenamiento Criptográfico Inseguro
A10 – Falla de restricción de acceso a URL	↑ A8 – Falla de restricción de acceso a URL
A9 – Comunicaciones Inseguras	= A9 – Protección Insuficiente en la Capa de Transporte
<no disponible en T10 2007>	+ A10 – Redirecciones y Destinos No Validados (NUEVO)
A3 – Ejecución de Ficheros Malintencionados	- <eliminado del T10 2010>
A6 – Revelación de Información y Gestión Incorrecta de Errores	- <eliminado del T10 2010>

OWASP Top Ten

A1- Injection

¿Cómo atacar?

- Incluir comandos mal intencionados en los datos de una aplicación y enviarlos a un interprete.

¿Qué ocurre?

- Los interpretes los toman como parámetros válidos y los interpretan

¿Qué interpretes se pueden atacar?

- SQL, OS Shell, LDAP, XPath, Hibernate, etc...

Generalmente se usan para:

- Ejecución de comandos vía shell
- Consultas SQL sobre la base de datos



A1- Injection

- Las fallas de inyección, particularmente SQL, son comunes en aplicaciones WEB.
- El Impacto por lo general severo:
 - Acceso a la base que usa la aplicación, (datos y/o esquema).
 - Acceso a ejecución según los permisos con los que corre la aplicación.



OWASP Top Ten

A1- Injection

Por ejemplo, si tenemos una página que lista un directorio pasado por parámetro de la siguiente manera:

<http://www.pepe.com/get.pl?usuario=pepe>

- Incluyendo parámetros maliciosos que permitan modificar lo que originalmente haría el comando, se podría listar otros directorios. Por ejemplo, incluyendo el path ../ como parte del nombre del archivo que se pide, se puede realizar lo siguiente:

<http://www.pepe.com/get.pl?usuario=pepe/../juan>

- Si el interprete es un shell, se podrían pasar otros comandos separados por “;”

<http://www.pepe.com/get.pl?usuario=pepe;cp%20/etc/passwd%20/tmp>



OWASP Top Ten

A1- Injection

- SQL injection

En general funcionan agregando parámetros que permitan alterar los resultados de una consulta sql.

- Sirve para:
 - Obtener acceso a la aplicación
 - Obtener información adicional
 - Causar daños.
- La integridad, confidencialidad y disponibilidad de la información se ve comprometida.



OWASP Top Ten

A1- Injection

- SQL injection: Obtener acceso a una aplicación:
 - Suponiendo que la consulta de autenticación de una pagina que pide **id** y **pass** es:

```
“select * from users where id=’ ” . $id . ” ’ and pass=’ ” . $pass . ” ’ ”;  
select * from users where id=’admin’ and pass=’admin’;
```

- ¿Que sucede si usamos id y pass con **1' or '1=1**

```
“select * from users where id=’ ” . “1' or '1=1” . ” ’ and pass=’ ” . “1' or '1=1” . ” ’ ”;
```

lo cual se resuelve en:

```
“select * from users where id=’1' or '1=1' and pass=’1' or '1=1' ”;
```



OWASP Top Ten

A1- Injection

- SQL injection: Para obtener acceso a una aplicación WEB, dependiendo del motor de base de datos, otras estructuras que se pueden usar son:
 - ' or 1=1--
 - " or 1=1--
 - or 1=1--
 - ' or 'a'='a
 - " or "a"="a
 - ') or ('a'='a
- VEAMOS UN EJEMPLO



OWASP Top Ten

A2- Cross site scripting

XSS permite a un atacante ejecutar scripts en el browser de la víctima.

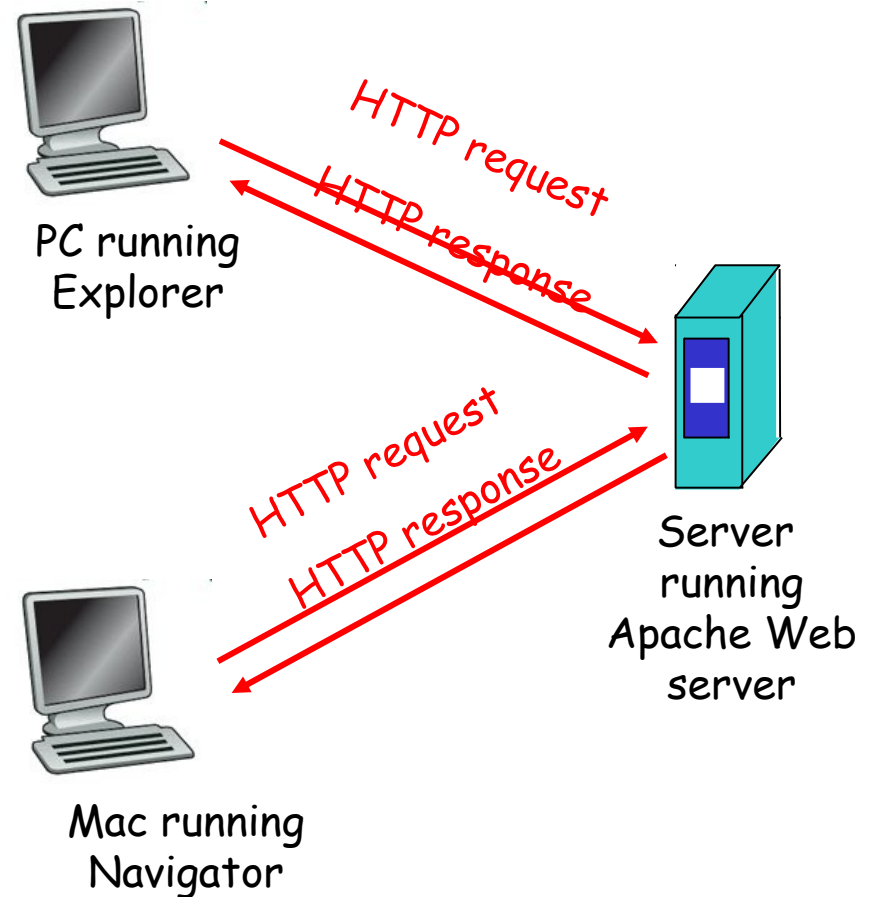
- Es importante distinguir que intervinien:
 - El atacante
 - La víctima
 - El sitio web vulnerable.



HTTP Generalidades

HTTP: hypertext transfer protocol

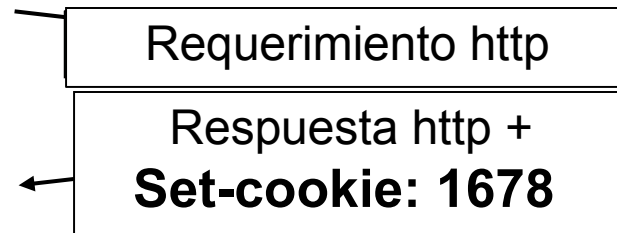
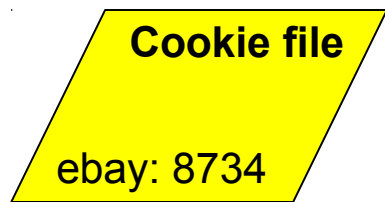
- Protocolo de la capa aplicación de la Web
- Modelo cliente/servidor
 - *cliente*: browser que requiere, recibe, “despliega” objetos Web
 - *servidor*: Servidor Web envía objetos en respuesta a requerimientos
- HTTP 1.0: RFC 1945
- HTTP 1.1: RFC 2616



Cookies: conservando el “estado” (cont.)

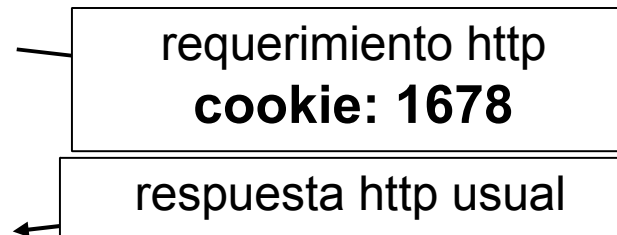
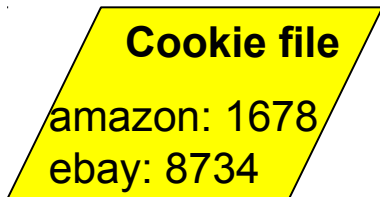
cliente

servidor



Servidor crea
ID 1678
para usuario

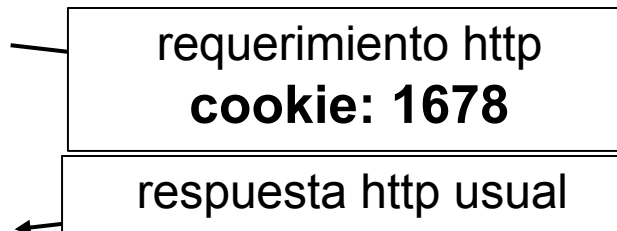
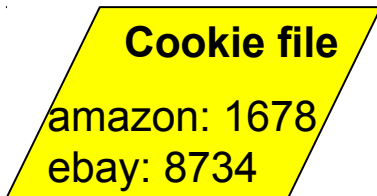
Entrada en
base de
datos



Acción
específica
de la cookie

acces
o

Un semana más:



Acción
específica
de la cookie

acces
o



OWASP Top Ten

A2- Cross site scripting

- Riesgos existentes

Con este ataque se pueden realizar:

- Robo de sesiones de usuarios.
 - Inserción de contenido hostil en sitio web vulnerable.
 - Ataques de phishing.
-
- En la mayoría de los casos, las fallas de XSS ocurren cuando una aplicación web toma datos de un usuario y los retorna sin validarlos ni codificarlos.



OWASP Top Ten - XSS

Tipos de XSS: Existen 3 tipos de fallas de cross site scripting:

- Type-0, también conocido como XSS DOM based o XSS local.
- Type-1, también conocido como XSS reflejado o no persistente.
- Type-2, también conocido como XSS almacenado o persistente.



XSS Type-2 – Almacenado

- Con esta vulnerabilidad es posible robar el acceso de los usuarios y violar la integridad y confidencialidad de los datos personales de estos.
- VEAMOS UN EJEMPLO



XSS Type-1 – Reflejado

- Con esta vulnerabilidad es posible robar información de acceso sensible (usuarios, passwords, cookies). Esto acarrea el posible compromiso de la integridad y confidencialidad de los datos personales.

–VEAMOS UN EJEMPLO



XSS Type-0 – DOM based o local

- En esta variante de XSS el script malicioso no es inyectado utilizando un sitio vulnerable, en forma reflejada o almacenada.
- La diferencia entre el tipo-0 y los otros tipos de XSS es que el código se inyecta a través de la URL pero no se incluye en el código fuente de la página. Está embebido en la URL provista al usuario víctima.
- El XSS tipo-0 ejecuta código remotamente en la máquina de un usuario con los permisos de este usuario.



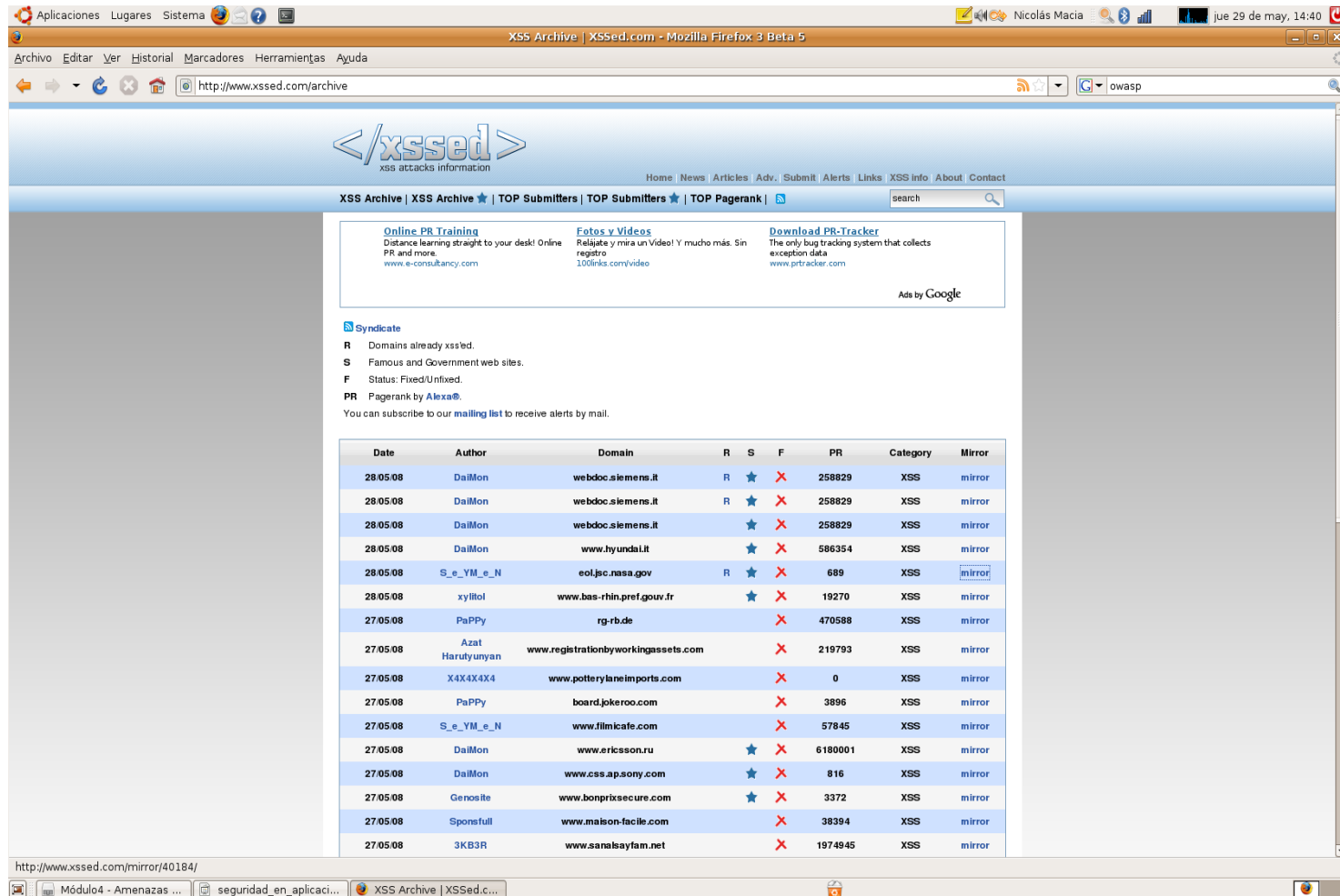
XSS Type-0 – DOM based o local

- Ej1: `http://vulnerable.com/index.html#name=<script>alert();</script>`
 - La almohadilla (#) le dice al navegador que todo lo que va después de ella es un fragmento, y no parte de la petición. Los navegadores no mandan los fragmentos al servidor, así que el servidor solamente vería: `http://vulnerable.com/index.html`, logrando que ni el servidor vea la inyección.
- Ej2: Suponga que la víctima tiene un blog en `http://victima.blog.com`
Este blog usa cookies por lo que siempre que ingresa, ya se encuentra logeado.
 - Para dejar comentarios en el blog, la victima hace una peticion del siguiente tipo:
`http://victima.blog.com/comment.php?nick=admin&comment=comentario`
 - El atacante envía la siguiente URL maliciosa a la víctima:
`http://www.google.com/index.html#name=<script>>window.location='http://victima.blog.com/comment.php?nick=admin&comment=mi_comentario'</script>`



OWASP Top Ten: XSS

Ejemplos de la vida real: <http://www.xssed.com/>



The screenshot shows the XSS Archive website in a Mozilla Firefox 3 Beta 5 browser. The page displays a list of XSS attacks with columns for Date, Author, Domain, R, S, F, PR, Category, and Mirror. The attacks are sorted by PR (PageRank).

Date	Author	Domain	R	S	F	PR	Category	Mirror
28/05/08	DaIMon	webdoc.siemens.it	R	★	×	258829	XSS	mirror
28/05/08	DaIMon	webdoc.siemens.it	R	★	×	258829	XSS	mirror
28/05/08	DaIMon	webdoc.siemens.it		★	×	258829	XSS	mirror
28/05/08	DaIMon	www.hyundai.it		★	×	586354	XSS	mirror
28/05/08	S_e_YM_e_N	eol.jsc.nasa.gov	R	★	×	689	XSS	mirror
28/05/08	xyllol	www.bas-rhin.pref.gouv.fr		★	×	19270	XSS	mirror
27/05/08	PaPPy	rg-rb.de			×	470588	XSS	mirror
27/05/08	Azat Harutyunyan	www.registrationbyworkingassets.com			×	219793	XSS	mirror
27/05/08	X4X4X4X4	www.potterylineimports.com			×	0	XSS	mirror
27/05/08	PaPPy	board.jkeroo.com			×	3896	XSS	mirror
27/05/08	S_e_YM_e_N	www.filmicafe.com			×	57845	XSS	mirror
27/05/08	DaIMon	www.ericsson.ru		★	×	6180001	XSS	mirror
27/05/08	DaIMon	www.css.ap.sony.com		★	×	816	XSS	mirror
27/05/08	Genosite	www.bonprixsecure.com		★	×	3372	XSS	mirror
27/05/08	Sponsfull	www.maison-facile.com			×	38394	XSS	mirror
27/05/08	3KB3R	www.sanalsayfam.net			×	1974945	XSS	mirror

OWASP Top Ten: XSS

Ejemplos de la vida real: <http://www.xssed.com/>

The screenshot illustrates a successful Cross-Site Scripting (XSS) attack on the eBay Motors website. The browser window shows the search page with a script injected into the search bar: `<><script>alert(1)</script>`. Two alert boxes are displayed: one with the text "XSS BY AZAT" and another with the number "1". The browser's address bar shows the URL `http://www.xssed.com/mirror/39969/`. The page title is "motors.desc.shop.ebay.com XSS Vulnerability | XSSed.com - Mozilla Firefox 3 Beta 5".

OWASP Top Ten

- A3 - Broken Authentication and Session Management

Fallas en la protección de credenciales y tokens de sesión que permiten que el atacante obtenga acceso a cuentas de usuario, cuentas de administrador o recursos no autorizados.



OWASP Top Ten

A3 - Broken Authentication and Session Management

Escenario #1: Aplicación de reserva de vuelos que soporta re-escritura de direcciones URL poniendo los identificadores de sesión en la propia dirección:

`http://example.com/sale/sitems;jsessionid=2P0OC2JM0OQSNDLPS2JV?dest=Hawaii`

Un usuario autenticado en el sitio quiere mostrar la venta a sus amigos. Envía por correo electrónico el enlace anterior, sin ser consciente de que está proporcionando su identificador de sesión. Cuando sus amigos utilicen el anterior enlace utilizarán su sesión y su tarjeta de crédito.

Escenario #2: No se establecen correctamente los tiempos de desconexión en la aplicación. Un usuario utiliza un ordenador público para acceder al sitio. En lugar de utilizar la función de “Cerrar sesión”, cierra la pestaña del navegador y se marcha. Un atacante utiliza el mismo navegador al cabo de una hora, y ese navegador todavía se encuentra autenticado.

VEAMOS UN EJEMPLO



OWASP Top Ten

- A4 - Insecure Direct Object Reference

Una referencia directa a un objeto ocurre cuando un desarrollador expone una referencia a la implementación interna de un objeto (archivo, directorio, registro de base de datos, una clave, URL o parámetro de un formulario). Un atacante puede manipular la referencia para acceder a objetos sin autorización, al menos que exista un control.

- Acceder a un archivo del filesystem del servidor
- Acceder a un archivo de definicion de conexión sql.
- VEAMOS UN EJEMPLO



OWASP Top Ten

- **A5 - Cross Site Request Forgery (CSRF)**

También conocido como XSRF, CSRF, y Cross Site Reference Forgery, consiste en explotar la confianza que el sitio tiene en el usuario.

Si las acciones de un sitio son urls fijas, como por ejemplo:

`http://my_banco/transferir?cantidad=1000&cuentadestino=123323`

es posible que un tercero ejecute la misma a nombre de la víctima.

Tipicamente un atacante puede embeber código malicioso HTML o Javascript en un mail o un sitio web para requerir una tarea específica a través de una url y ejecutarla sin el conocimiento del usuario, directamente u utilizando una falla de Cross-site Scripting.



OWASP Top Ten

- A5 - Cross Site Request Forgery (CSRF)

¿CSRF = XSS?

Cross-Site Scripting explota la confianza que el cliente tiene en el Sitio Web o la aplicación. Los usuarios generalmente confían que el contenido mostrado en sus navegadores es lo que el sitio Web visitado quiere realmente presentar al usuario. El sitio Web asume que si una acción requerida fue ejecutada, es lo que el usuario quiso ejecutar intencionalmente. CSRF explota la confianza que el sitio tiene en el usuario.

VEAMOS UN EJEMPLO DE LA VIDA REAL



OWASP Top Ten

A6 - Security Misconfiguration

Tener en cuenta todo el entorno:

- Desde el sistema operativo hasta el servidor de aplicaciones
- Incluir las librerías utilizadas!!
- Por ejemplo, todas las credenciales deberían cambiar en el ambiente de Producción.

¿El código fuente es secreto?

- Piense en todos los lugares donde se encuentra su código fuente
- Una seguridad eficaz **NO** debería requerir que su código fuente sea secreto



OWASP Top Ten

A6 - Security Misconfiguration

Impacto Típico

- Instalación de código malicioso debido a un parche faltante en el OS o servidor
- Falla de XSS debido a un parche faltante en el framework de la aplicación
- Acceso no autorizado a cuentas por defecto, funcionalidad de la aplicación, etc, debido a una defectuosa configuración del servidor



OWASP Top Ten

A6 - Security Misconfiguration

Escenario #1: La aplicación está basada en un ambiente de trabajo como Struts o Spring. Se han presentado defectos de XSS en algunos de los componentes que utiliza la aplicación. Se ha liberado una actualización que sirve para corregir esos defectos. Hasta que no se realicen dichas actualizaciones, los atacantes podrán encontrar y explotar los fallos, ahora conocidos, de la aplicación.

Escenario #2: La consola de administración del servidor de aplicaciones está instalada y no ha sido removida. Las cuentas predeterminadas no han sido cambiadas. Los atacantes descubren que las páginas de administración están activas, se registran con las claves predeterminadas y toman posesión de los servicios.

VEAMOS UN EJEMPLO



OWASP Top Ten 2010

- A7 - Insecure Cryptographic Storage:

Las aplicaciones que encriptan frecuentemente utilizan diseños pobres de criptografía, algoritmos de cifrado inadecuados, errores serios en el uso de criptografía fuerte.

- Vulnerabilidades:

- No encriptar datos sensibles
- Usar algoritmos de cifrado caseros.
- Usar algoritmos fuertes de manera no segura.
- Usar algoritmos cuya debilidad ya fue probada
- Harcodear claves y guardarlas en medios no protegidos

VEAMOS UN EJEMPLO



Seguridad en aplicaciones WEB

- A8 - Failure to Restrict URL Access
 - Frecuentemente la única protección para las URLs es que los links a las páginas no sean presentados a los usuarios no autorizados.
 - Esto implica seguridad por oscuridad y puede producir fallas en la protección de funciones sensibles y datos de la aplicación.

Formas de ataque:

- URLs ocultas, destinadas a ser usadas por usuarios privilegiados, se encuentran accesibles para todos los usuarios que conozcan o descubran su existencia.



Seguridad en aplicaciones WEB

- A9 - Insecure Communications

Ocurre cuando los mecanismos de encriptación fallan o cuando la aplicación es forzada a salir del modo encriptación

Cuando ello ocurre el tráfico puede ser sniffado y la información sensible está expuesta.

Es importante:

- Que todo el tráfico autenticado se transmita utilizando SSL.
- Que la información sensible (credenciales, información privada, información financiera) se transmita usando protocolos encriptados.



Seguridad en aplicaciones WEB

A10 - Redirecciones y Destinos No Validados

Las redirecciones en aplicaciones web son muy comunes

- Frecuentemente incluyen parámetros suministrados por el usuario en la URL destino
- Si no son validados, el atacante puede enviar a la víctima a un sitio de su elección
- Internamente las aplicaciones envían el pedido a una nueva página en la misma aplicación



Seguridad en aplicaciones WEB

A10 - Redirecciones y Destinos No Validados

Escenario #1:

La aplicación tiene una página llamada “redirect.jsp” que recibe un único parámetro llamado “url”. El atacante compone una URL maliciosa que redirige a los usuarios a una aplicación que realiza el phishing e instala código malicioso.

<http://www.example.com/redirect.jsp?url=evil.com>

Escenario #2:

La aplicación utiliza destinos para redirigir las peticiones entre distintas partes de la aplicación. Para facilitar esto, algunas páginas utilizan un parámetro para indicar dónde será dirigido el usuario si la transacción es correcta. En este caso, el atacante compone una URL que evadirá el control de acceso de la aplicación y llevará al atacante a una función de administración a la que en una situación habitual no debería tener acceso.

<http://www.example.com/boring.jsp?fwd=admin.jsp>



Seguridad en aplicaciones WEB

A10 - Redirecciones y Destinos No Validados

Impacto Típico

- Redireccionar a una víctima hacia un sitio de phishing o malware
- El pedido del atacante es ejecutado, pasando por alto los controles de seguridad.



Formas de protección

Generales

- Validar todas las entradas.
- Codificar todas las salidas. PHP → htmlspecialchars()
- Logging y monitoreo de los accesos y errores
- Uso de frameworks de desarrollo.
- Utilización de herramientas provistas por los servidores e intérpretes. Ej: mecanismos de escape de caracteres
- Procedimentar instalaciones/configuraciones
- Automatizar actualizaciones

https://www.owasp.org/index.php/Cheat_Sheets



Formas de protección (cont)

A1- Injection

- Validar los datos de entrada para asegurarnos que no tienen código malicioso.
- Estructurar los datos de entrada de manera que sean tratados como datos y no como potenciales ejecutables.
- Utilizar Stored procedures y prepared statements.
- Uso adecuado de privilegios: Que los servidores corran con los permisos mínimos necesarios y que el usuario de conexión a la base de datos no sea privilegiado.

https://owasp.org/index.php/Injection_Prevention_Cheat_Sheet

http://www.owasp.org/index.php/SQL_Injection_Prevention_Cheat_Sheet



Formas de protección

A2 - XSS

- Usuario
 - No seguir links provistos. Tipearlos en forma propia.
- Desarrollador
 - Validar las entradas
 - No basarse en uso de filtros via blacklist.
 - Validar vía Whitelist
 - Codificar las salidas

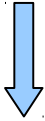
[http://www.owasp.org/index.php/XSS_\(Cross_Site_Scripting\)_Prevention_Cheat_Sheet](http://www.owasp.org/index.php/XSS_(Cross_Site_Scripting)_Prevention_Cheat_Sheet)



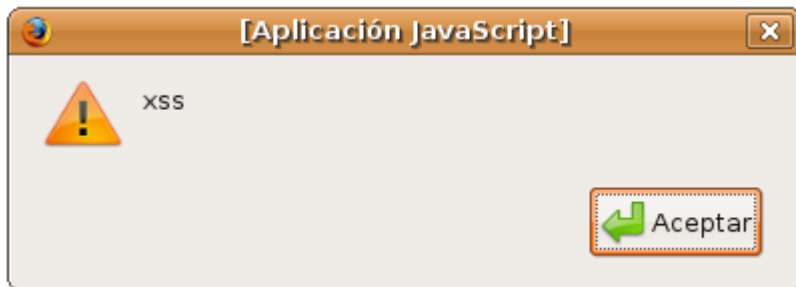
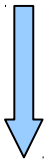
XSS - php

[http://host/test.php?user=<script>alert\('xss'\)</script>](http://host/test.php?user=<script>alert('xss')</script>)

```
<?
echo $_GET['user'];
?>
```



`<script>alert('xss')</script>`



```
<?
echo htmlspecialchars ($_GET['user']);
?>
```



`"<script>alert('xss')</script>"`



`<script>alert('xss')</script>`

htmlspecialchars: Convierte caracteres especiales a entidades HTML



Formas de protección (cont)

A3 - Broken Authentication and Session Management

- No permitir que el proceso de login se inicie en una página no encriptada.
- Usar períodos de timeout que produzcan logouts automáticos.
- No exponer identificadores de sesión o una porción válida de las credenciales en URLs o logs.
- No aceptar identificadores de sesión nuevos, preseteados o inválidos desde la URL o el requerimiento (“Session fixation attack”).

http://www.owasp.org/index.php/Authentication_Cheat_Sheet



Formas de protección (cont)

A4 - Insecure Direct Object Reference

- Eliminar la referencia directa a objetos Reemplazarla con un valor temporal de mapeo (ej. 1, 2, 3)
- Validar la referencia directa al objeto:
 - Verificar que el valor del parámetro se encuentra adecuadamente formateado
 - Verificar que el usuario se encuentra autorizado a acceder el objeto determinado



Formas de protección (cont)

A5 - Cross Site Request Forgery (CSRF)

- Evitar períodos demasiado largo para las sesiones de usuarios.
- Requerir que el usuario se loguee antes de ejecutar una acción que involucre datos sensibles o transacciones financieras.
- Utilizar componentes aleatorias en las URLs.

[https://www.owasp.org/index.php/Cross-Site_Request_Forgery_\(CSRF\)_Prevention_Cheat_Sheet](https://www.owasp.org/index.php/Cross-Site_Request_Forgery_(CSRF)_Prevention_Cheat_Sheet)



Formas de protección (cont)

A6 – Defectuosa Configuración de Seguridad

- Verificar la gestión de configuración de sus sistemas
- Uso de guías de securización
- Mantener actualizadas todas las plataformas
- Aplicar parches en todos los componentes
- Esto incluye librerías de software, no solo OS y servidor de aplicaciones
- Analizar los efectos de estos cambios (en un entorno de prueba)



Formas de protección (cont)

A7- Insecure Cryptographic Storage

- No crear algoritmos criptográficos, usar los conocidos.
- Generar las claves offline y guardar las claves privadas con mucho cuidado.
- Asegurarse de que la infraestructura (por ejemplo la BD) donde se guardan las claves está asegurada
- Asegurarse que los datos encriptados en disco no son fácilmente descriptables.



Formas de protección (cont)

A8 - Failure to Restrict URL Access

- No asumir que los usuarios no son capaces de descubrir URLs o APIs con privilegios especiales: protegerlas adecuadamente.
- Bloquear el acceso a todos los tipos de archivos que la aplicación nunca usará.

A9- Insecure Communications

- Usar SSL en todas las conexiones para transmitir datos de autenticación e información sensible.



Formas de protección (cont)

A10 - Redirecciones y Destinos No Validados

- Intentar evitar el uso de redirecciones y destinos
- No utilizar parámetros provistos por usuarios para definir la URL destino
- Si se deben utilizar dichos parámetros, entonces:
 - Validar cada parámetro para asegurarse que es válido y se encuentra autorizado para el usuario actual
 - Utilizar mapeos del lado del servidor para 'traducir' la opción provista al usuario en la verdadera página de destino



Algunos Tips

- **No se detenga en el Top 10.** Vea también Guía de Testeo OWASP y la Guía de Revisión de Código OWASP.
- **Cambio constante.**
- **Piense positivamente.**
- **Utilice herramientas inteligentemente.**
- **Software Development Life Cycle (SDLC) Seguro.**



OWASP Top Ten 2007

Malicious File Execution:

Permite a los atacantes incluir código y datos hostiles, lo cual resulta en ataques devastadores que llegan a comprometer totalmente al servidor.

Consiste en:

- Ejecución de código remota
- Instalación remota de rootkit y compromiso completo del sistema
- Puede ser usado para realizar ataques de phishing



OWASP Top Ten 2007

Malicious File Execution:

- Afecta a todas las aplicaciones web que acepten nombres de archivos o archivos por parte del usuario. PHP es particularmente vulnerable a ataques de remote file include.
- VEAMOS UN EJEMPLO



OWASP Top Ten 2007

- Information Leakage and Improper Error Handling

Las aplicaciones pueden dejar escapar información relacionada a su configuración, su trabajo interno o violar privacidad debido a problemas de la aplicación misma.

Dicha información puede ser aprovechada por los atacantes para llevar a cabo el ataque.

Lo que ocurre por ejemplo:

- Cuando se produce un error se muestra demasiada información.
- También ocurre cuando existen malas configuraciones, por ejemplo servidor web Apache



Formas de protección (cont)

Malicious File Execution

- No usar una entrada provista por el usuario como nombre de archivo para acceder a un recurso del servidor. En caso de ser utilizadas chequearlas adecuadamente.
- Configurar el firewall para no prevenir conexiones nuevas salientes a internet o a otros servidores
- Propias de PHP: deshabilitar `allow_url_fopen` y `allow_url_include` en el `php.ini`.



Formas de protección (cont)

Information Leakage and Improper Error Handling

- Asegurarse que todo el equipo de desarrollo de software utiliza el mismo criterio para manejar excepciones.
- Deshabilitar o limitar el manejo de errores para no mostrar información de debug a los usuarios.
- Configurar adecuadamente todos los componentes de la aplicación (Servidor WEB, BD) en lo que se refiere a mensajes de error.
- php.ini → display_errors = off
- Web server → configurar paginas para los errores 400, 404 y 500



Seguridad en Aplicaciones WEB

FIN

