



OWASP Top 10 2013

Los diez riesgos más importantes
en aplicaciones web

Felipe Zipitría

OWASP/

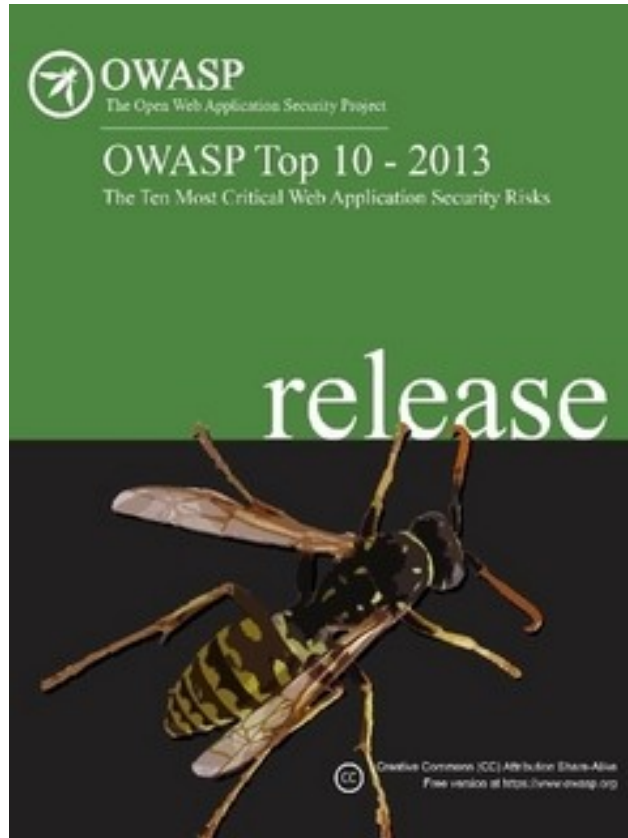
GSI- Facultad de Ingeniería
felipe.zipitria@owasp.org

Copyright © The OWASP Foundation
Permission is granted to copy, distribute and/or modify this document
under the terms of the OWASP License.

The OWASP Foundation

<http://www.owasp.org/>

Introducción al Top 10



- Es un documento EDUCATIVO.
- Es GRATUITO.
- DESCRIBE los riesgos más críticos en aplicaciones Web
 - Descripción del mismo
 - Escenario de ejemplo de un ataque
 - Pautas para verificar si nuestra aplicación es vulnerable
 - Recomendaciones para prevenir dicho riesgo

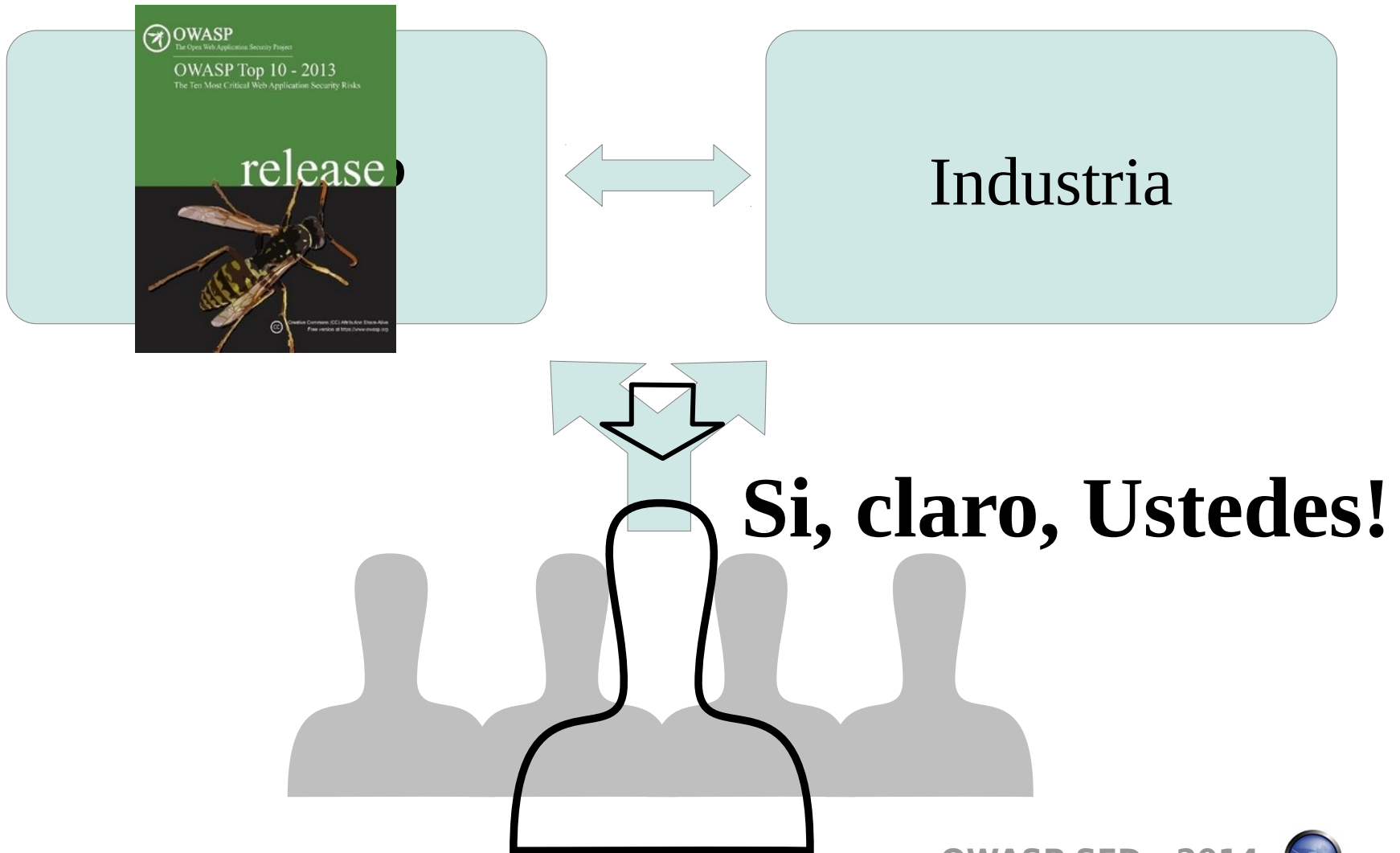


¿De dónde obtiene la información?

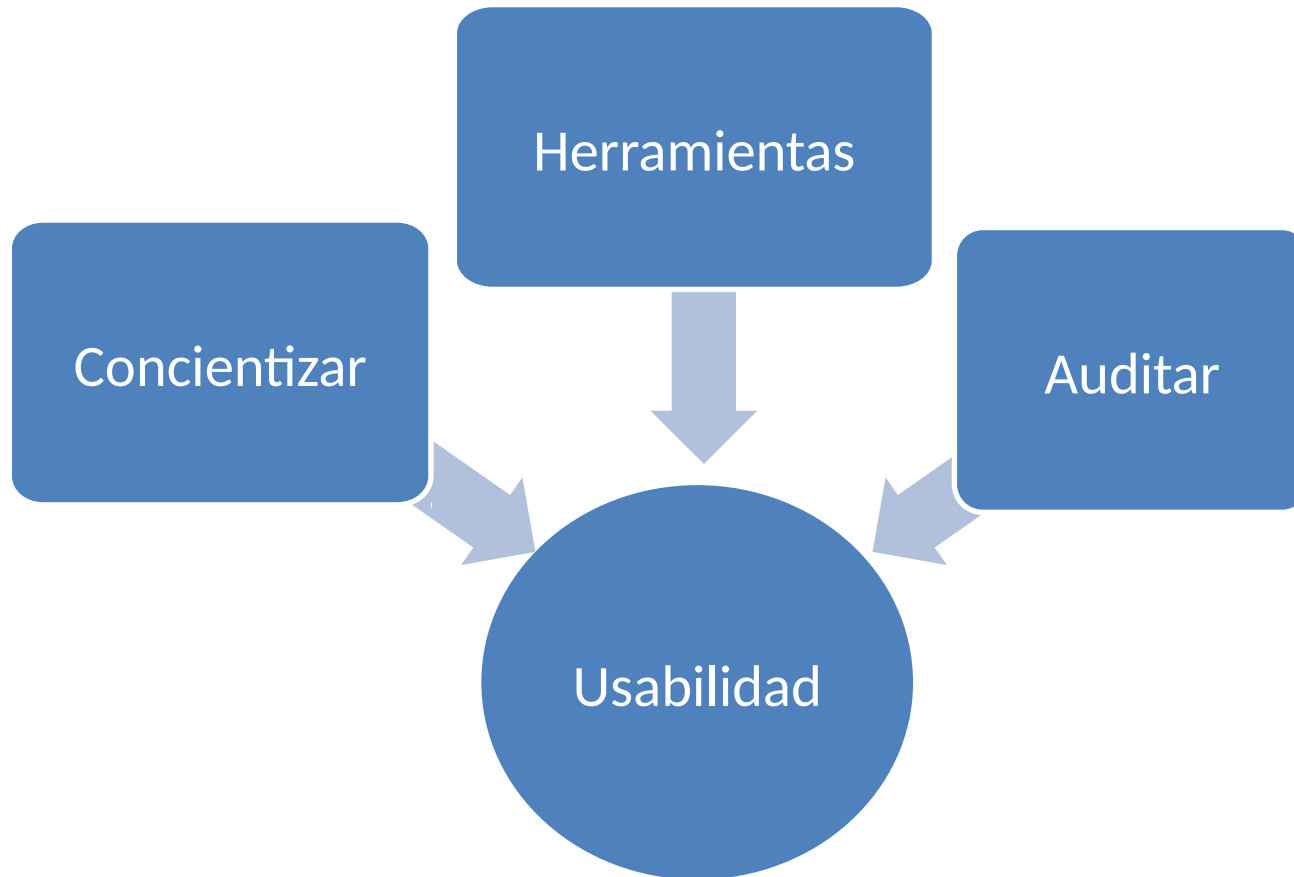
- 8 conjuntos de datos de 7 firmas que se especializan en seguridad de aplicaciones.
- Los datos se extienden a más de 500,000 vulnerabilidades a lo largo de cientos de organizaciones y miles de aplicaciones.
- Los puntos del Top 10 se seleccionan y priorizan de acuerdo a que tanto predominio tienen, en combinación con estimaciones del consenso de su explotabilidad, detectabilidad, e impacto estimado.



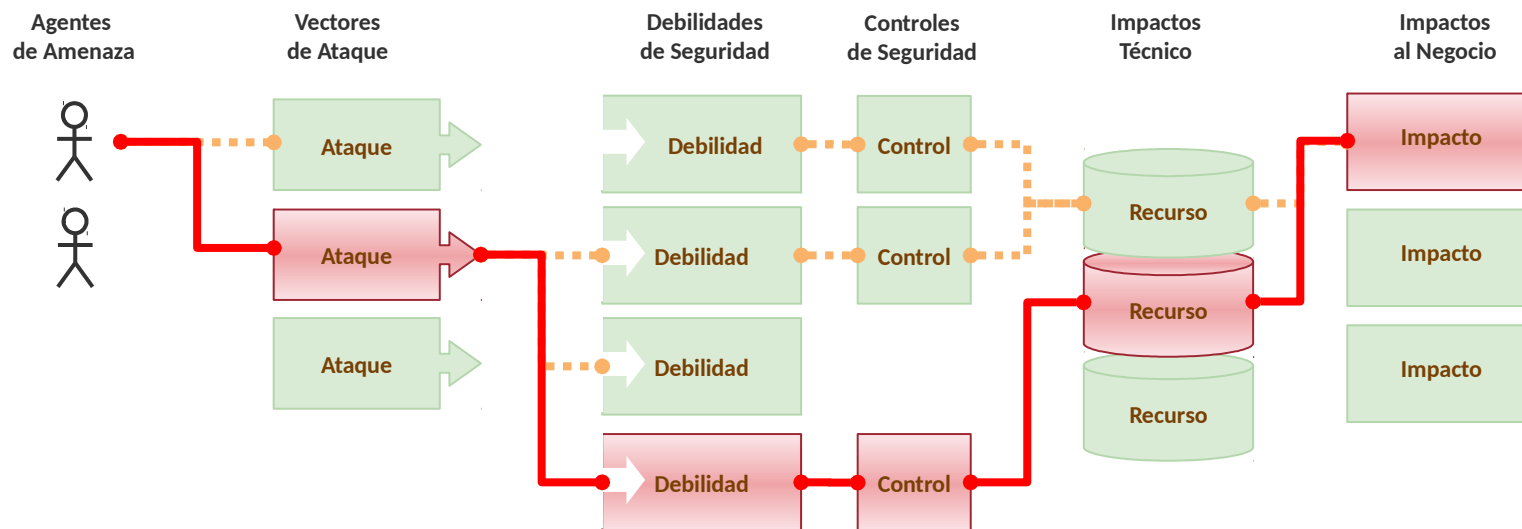
Top Ten OWASP



Un acercamiento por fases a la Seguridad de Aplicaciones



Metodología para catalogar el Riesgo



Agente de Amenaza	Vector de Ataque	Predominio de Debilidad	Detectabilidad de Debilidad	Impacto Técnico	Impacto al Negocio
?	1 Facil	Amplia	Facil	Severo	?
	2 Mediano	Comun	Mediano	Moderado	
	3 Dificil	Poco Comun	Dificil	Menor	

1

2

2

1

1.66

1

1.66 calificación de riesgo ponderado

Ejemplo de A1 Inyección






¿Que ha cambiado en esta versión?

Riesgos agregados y eliminados

- *A6 Exponer datos sensibles*
 - [2010-A7](#) Almacenamiento criptográfico inseguro y [2010-A9](#) Protección insuficiente en capa de transporte se fusionaron
- *A7 Falla de control de acceso*
 - renombrada/ampliada de [2010-A8](#) Falla de restricción de acceso a URL



Comparación del Top 10 2013 con 2010

OWASP 2010	OWASP 2013
A1 Inyección	= A1 Inyección
A2 Secuencia de comandos en sitios cruzados (XSS)	↑ A2 Pérdida de autenticación y gestión de sesiones (era 2010-A3)
A3 Pérdida de autenticación y gestión de sesiones	↓ A3 Secuencia de comandos en sitios cruzados (XSS)
A4 Referencia directa insegura a objetos	= A4 Referencia directa insegura a objetos
A5 Falsificación de pedido en sitios cruzados (CSRF)	↑ A5 Defectuosa Configuración de seguridad
A6 – Defectuosa Configuración de seguridad	A6 Exponer datos sensibles (2010-A7 Almacenamiento criptográfico inseguro y 2010-A9 Protección insuficiente en capa de transporte se fusionaron para formar 2013-A6) 
A7 Almacenamiento criptográfico inseguro	A7 Ausencia de control de acceso a Funciones (renombrada/ampliada de 2010-A8 Falla de restricción de acceso a URL) 
A8 Falla de restricción de acceso a URL	↓ A8 Falsificación de pedido en sitios cruzados (CSRF)
A9 Protección insuficiente en la capa de red	A9 Uso de componentes con vulnerabilidades conocidas (nuevo pero era parte de 2010-A6 – Configuración defectuosa de seguridad) 
A10 Redirecciones y destinos no validados	= A10 Redirecciones y destinos no validados

A1 - Inyección

Inyección significa

- Incluir comandos mal intencionados en los datos de una aplicación los cuales son enviados a un intérprete

Los intérpretes...

- Toman los datos, y los interpretan como comandos válidos
- SQL, OS Shell, LDAP, Xpath, Hibernate, etc.

Impacto típico

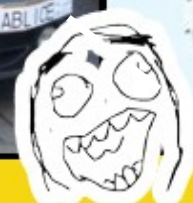
- Por lo general es grave. Todos los datos de una base pueden ser leídos o modificados
- También puede comprometerse el sistema operativo base, cuentas de usuario, o esquemas de base de datos



Ejemplos prácticos

¿Que querrá decir?

It All Starts with a '



Never, Ever underestimate
The Power of '



Sigue ahi..

- 2004: A6
- 2007: A2
- 2010: A1
- 2013: A1!

It All Starts with a '

It's not fun when you're next!



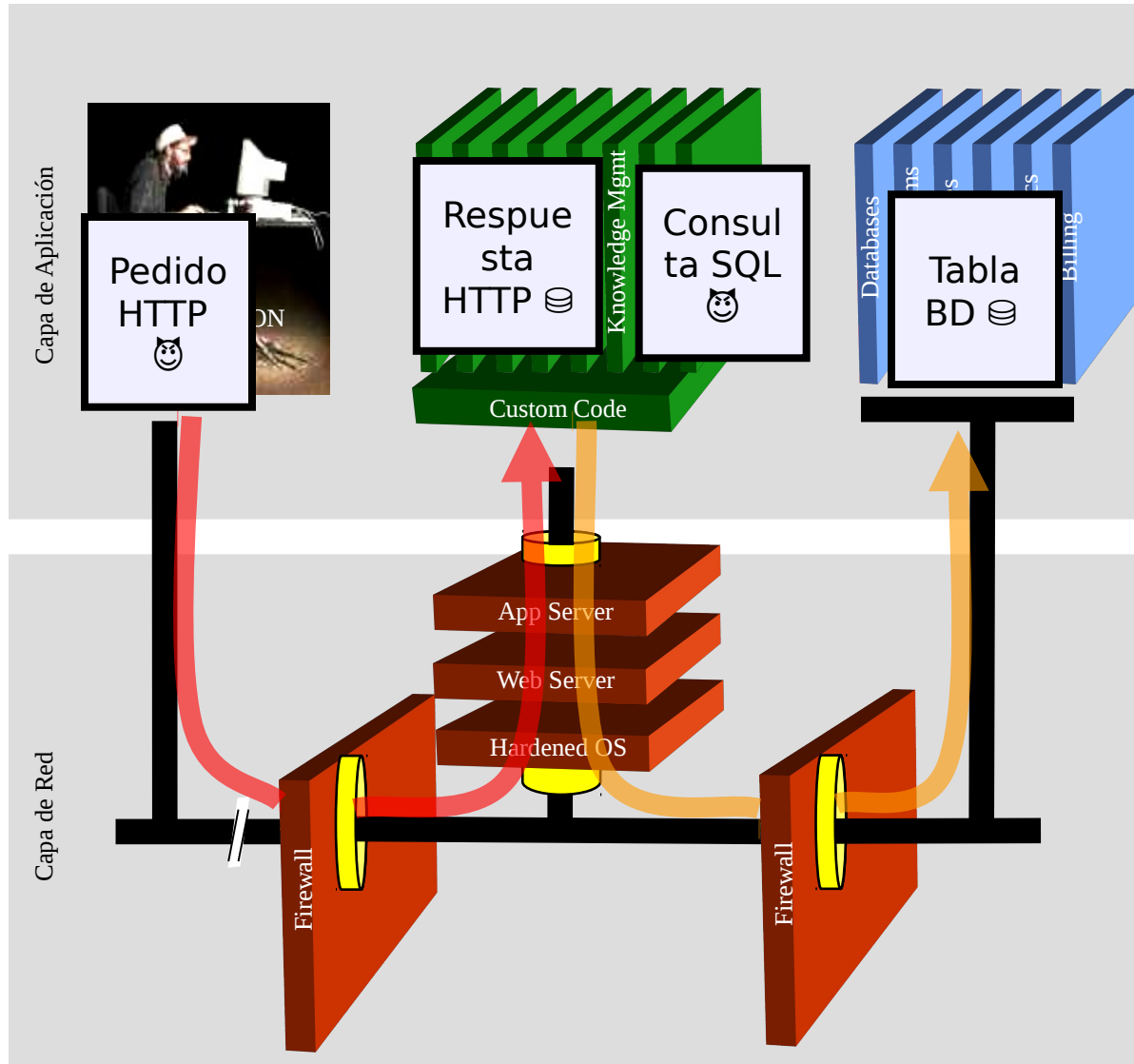
SQL Injection

Never, Ever underestimate
The Power of '

 mantra



Inyección SQL - Demostración



A screenshot of a web application login form. The 'Account:' field contains the text 'OR 1=1 --'. The 'SKU:' field is empty. A 'Submit' button is visible below the fields.

1. Aplicación presenta un formulario web al atacante
2. Atacante envía un ataque en los datos del formulario
3. Aplicación dirige el ataque a la base de datos en una consulta SQL
4. Base de datos ejecuta el ataque y envía los resultados cifrados nuevamente a la aplicación
5. Aplicación descifra los datos normalmente y envía los resultados al atacante



A1 - Como evitar Fallas de Inyección

■ Recomendaciones

- Evitar el intérprete completamente
- Utilizar una interfaz que soporte variables parametrizadas (Ej., declaraciones preparadas, o procedimientos almacenados),
 - Las variables parametrizadas permiten al intérprete distinguir entre código y datos
- Decodificar y convertir todas las entradas del usuario a su forma mas simple antes de enviarlas al interprete
- Siempre efectuar una validación 'positiva' de todas las entradas realizadas por el usuario
- Seguir el principio de mínimo privilegio en las conexiones con bases de datos para reducir el impacto de una falla

■ Referencias

- ▶ Para mayor información:

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

OWASP SFD 2014



A2 - Pérdida de Autenticación y Gestión de Sesiones

HTTP es un protocolo *sin estado*

- Significa que las credenciales tienen que viajar en cada pedido HTTP

Falla en la gestión de sesiones

- SESSIONID es utilizado para mantener la sesión
- Cualquier factor que comprometa dicho atributo, para el atacante es lo similar a poseer el usuario y la clave

Cuidar la puertas laterales

- Cambio/recordar contraseña, pregunta secreta, etc.

Impacto típico

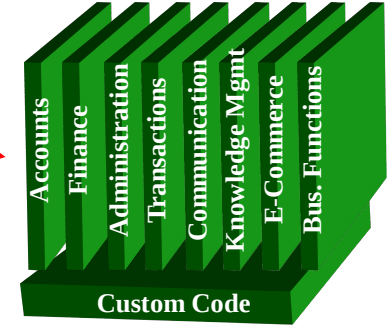
- Cuentas de usuario comprometidas o sesiones robadas



Perdida de Autenticación - Demostración

1

Usuario envía credenciales



2

Sitio utiliza reescritura de URLs
(ej., escribe sesión en URL)

3

Usuario hace clic en un enlace hacia
<http://www.hacker.com> en un foro

4

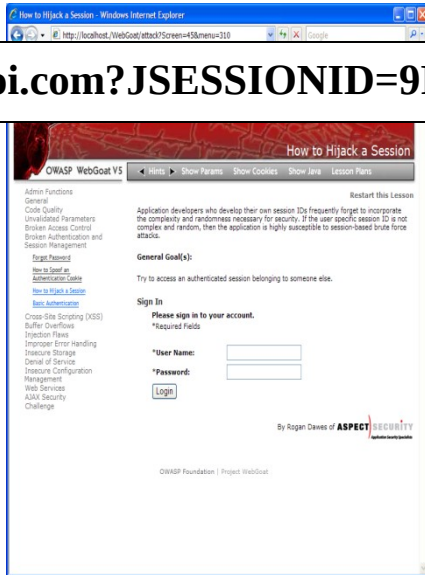


5

Hacker utiliza la
JSESSIONID y toma
posesión de la cuenta del
usuario

Hacker verifica encabezados de referencia en los
logs de www.hacker.com
y encuentra la JSESSIONID del usuario

www.boi.com?JSESSIONID=9FA1DB9EA...



A2 - Como evitar la Perdida de Autenticación y Gestión de Sesiones

■ Verificar la arquitectura

- ▶ Autenticación debería ser simple, centralizada y estandarizada
- ▶ Utilizar el gestor de sesiones estándar provisto por el servidor de aplicaciones – no inventar uno propio!
- ▶ Estar seguro que SSL protege tanto las credenciales como las sesiones de usuario todo el tiempo

■ Verificar la implementación

- ▶ No utilizar solamente análisis automático
- ▶ Verificar el certificado SSL
- ▶ Examinar todas las funciones relacionadas a autenticación
- ▶ Verificar que “cierre de sesión” efectivamente destruya la sesión
- ▶ Utilizar OWASP's WebScarab para testear la implementación



A3 - Secuencia de Comandos en Sitios Cruzados (XSS)

Ocurre cada vez que

- Datos no validados son enviados al navegador de una víctima

Los datos no validados pueden

- Encontrarse almacenados en una BD (persistente)
- Ser reflejados desde una entrada Web (formulario, campo oculto, URL, etc.)

Impacto típico

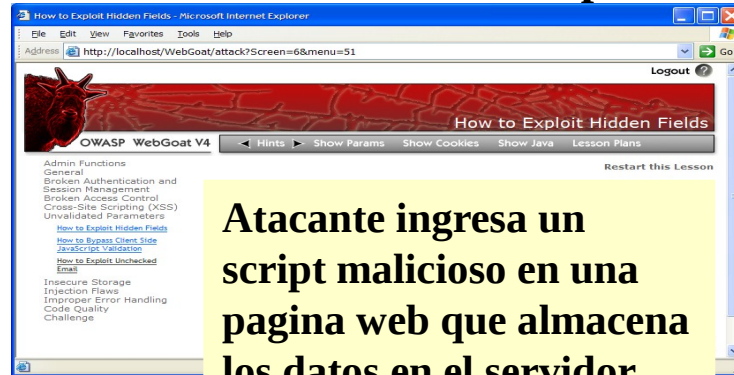
- Robar los datos de sesión de un usuario
- Más grave: instalar un proxy XSS para controlar el navegador del usuario



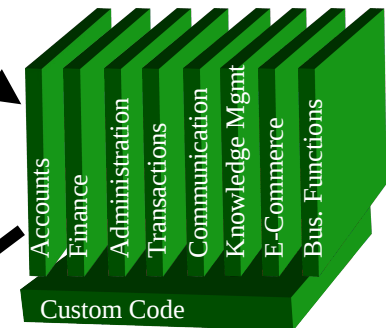
XSS - Demostración

1

Atacante establece una trampa – actualizar perfil

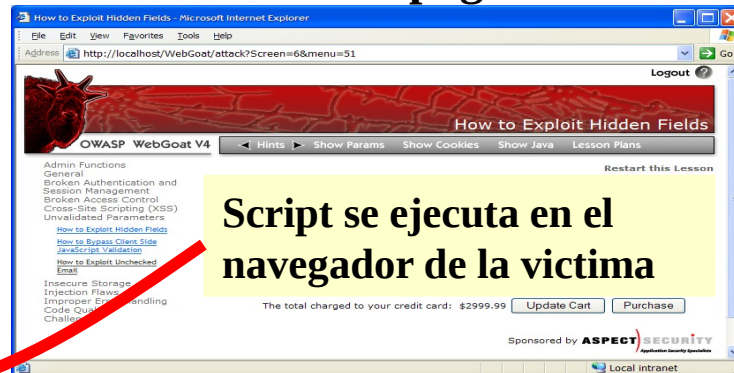


Aplicación con vulnerabilidad XSS almacenado



2

Victima visualiza la página – accede al perfil



3

Script silenciosamente envía la sesión de la victima al atacante



A3 - Como evitar Fallas de XSS

■ Recomendaciones

- Eliminar la Falla
 - No incluir entradas suministradas por el usuario en la página de salida
- Defenderse de la Falla
 - Recomendación Principal: Codificar todos los datos de entrada en la página de salida (Utilizar OWASP's ESAPI para dicha tarea): <http://www.owasp.org/index.php/ESAPI>
 - Siempre efectuar una validación 'positiva' de todas las entradas realizadas por el usuario
 - Definir políticas de **C**ontent **S**ecurity **P**olicy (W3C) (HTML5)

■ Referencias

- ▶ [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)



A4 - Referencia Directa insegura a objetos

¿Cómo sucede?

- un desarrollador expone una referencia a un objeto de implementación interno, tal como un archivo, directorio, o base de datos

Los atacantes abusan de esto porque...

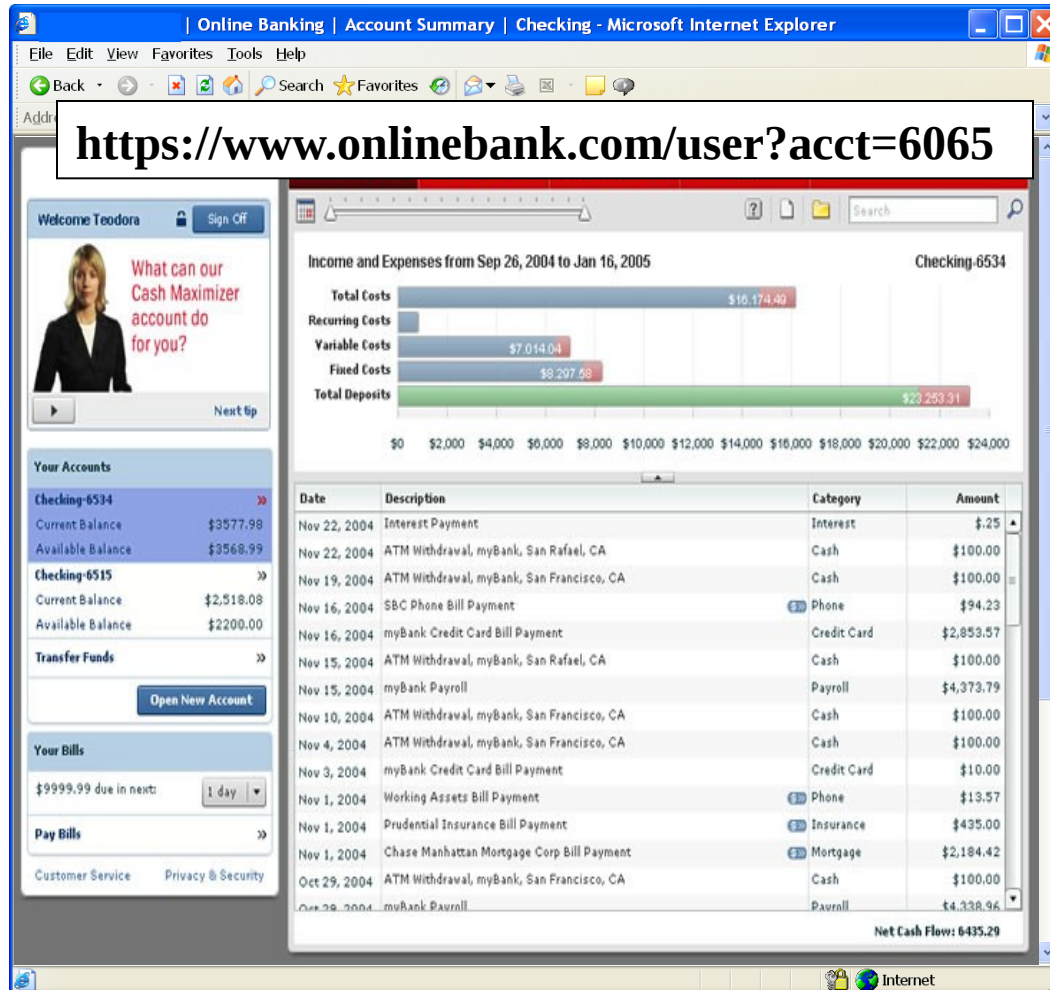
- Para referencias directas a recursos protegidos, la aplicación que no protege el acceso
- Si la referencia es indirecta, no se protege el mapeo para el usuario actual y por ello accede al recurso mapeado
- ¡Puede ser tan simple como modificar un parámetro!

Impacto típico

- Acceso a recursos que no autorizados



Referencia Directa Insegura a Objetos - Ejemplo



- Atacante identifica su número de cuenta 6065
?acct=6065
- Lo modifica a un número parecido
?acct=6066
- Atacante visualiza los datos de la cuenta de la víctima



A4 - Como evitar Referencias Directas Inseguras a Objetos

- Eliminar la referencia directa a objetos
 - Reemplazarla con un valor temporal de mapeo (ej. 1, 2, 3)
 - ESAPI proporciona soporte para mapeos numéricos y aleatorios
 - **IntegerAccessReferenceMap** & **RandomAccessReferenceMap**

<http://app?file=Report123.xls>

<http://app?file=1>

<http://app?id=9182374>

<http://app?id=7d3J93>



Report123.xls

Acct:9182374

- 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
 - Restricciones en los parámetros funcionan muy bien!
 - Verificar que el modo de acceso al objeto solicitado se encuentra autorizado (ej., lectura, escritura, modificación)



A5 - Configuración Defectuosa de Seguridad

Las aplicaciones web dependen de cimientos seguros

- Desde el sistema operativo hasta el servidor de aplicaciones
- No olvidarse de todas las librerías utilizadas!!

Es su código fuente un secreto?

- Piense en todos los lugares donde se encuentra su código fuente
- Una seguridad eficaz no requiere que su código fuente sea secreto

La CS debe ser extendida a todas las partes de la aplicación

- Por ejemplo, todas las credenciales deberían cambiar en el ambiente de producción

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



A5 - Defectuosa Configuración de Seguridad

Las aplicaciones Web dependen de bases sólidas

Esto incluye el SO, Servidor Web/Aplicación, SGBD, aplicaciones, y *todas las librerías de código* (ver nuevo A9)

Características innecesarias

¿Están habilitadas o instaladas algunas características innecesarias (p. ej. puertos, servicios, páginas, cuentas, privilegios)?

Gestión de errores

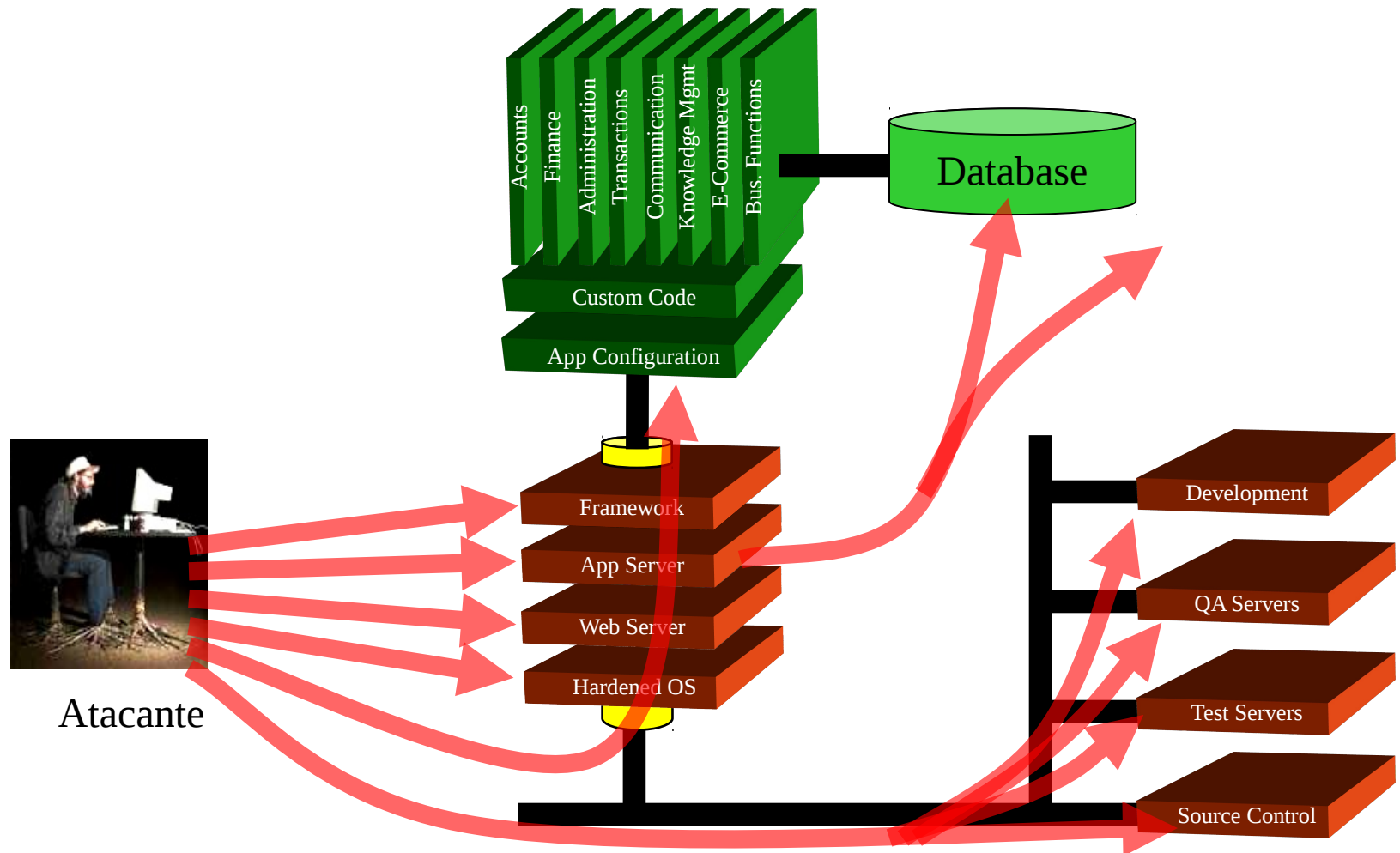
¿Su manejo de errores revela rastros de las capas de aplicación u otros mensajes de error demasiado informativos?

Framework de desarrollo

¿Están las configuraciones de seguridad en su framework de desarrollo (p. ej. Struts, Spring, ASP.NET) y librerías sin configurar a valores seguros?



Configuración Defectuosa de Seguridad - Ejemplo

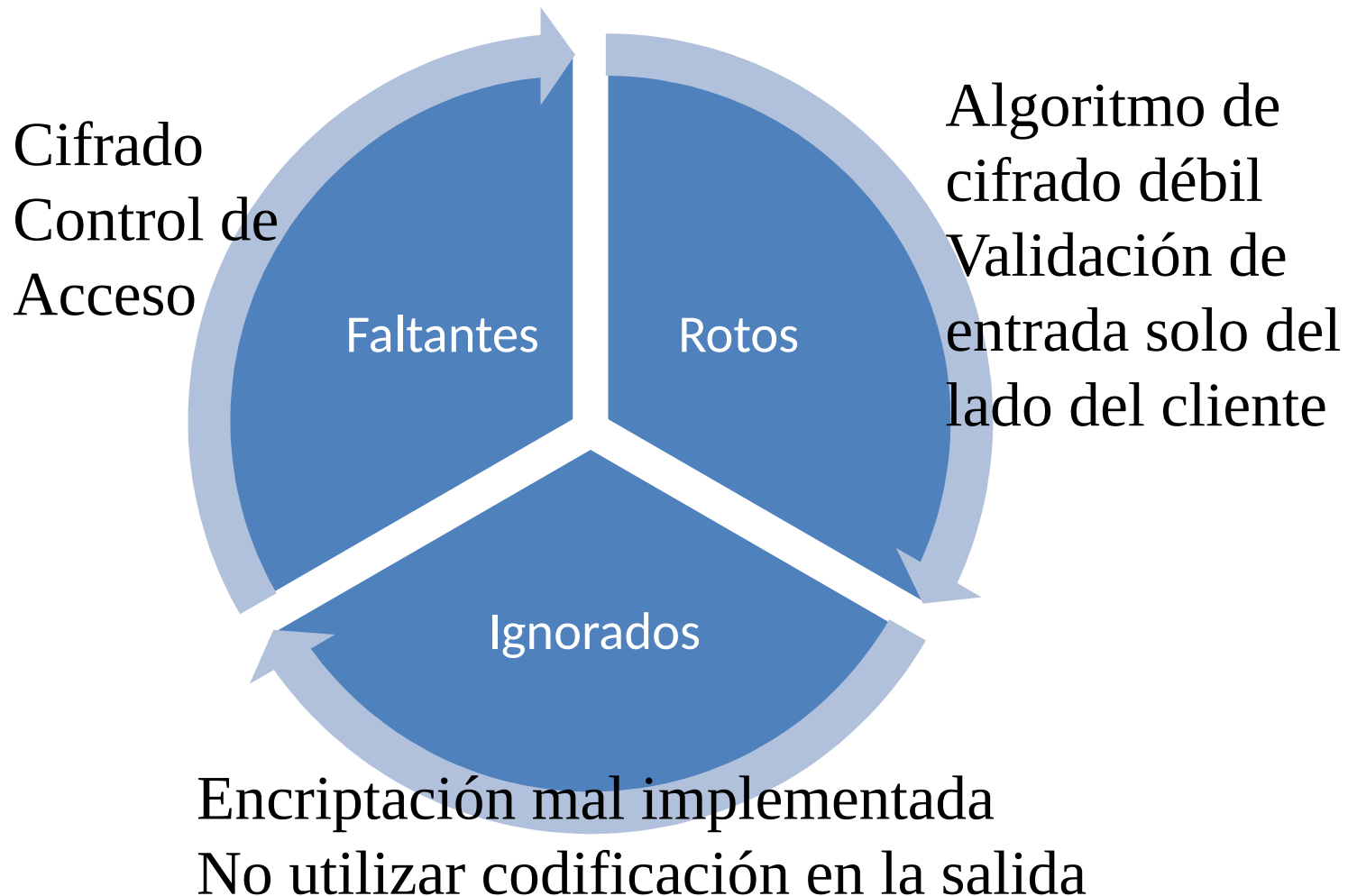


A5 - Como evitar una Defectuosa Configuración de Seguridad

- Verificar la gestión de configuración de sus sistemas
 - ▶ Uso de guías de para asegurar instalaciones base
 - Automatizar tareas es MUY UTIL aquí
 - ▶ Mantener actualizadas todas las plataformas
 - ▶ Aplicar parches en todos los componentes
 - Esto incluye librerías de software, no solo SO y servidor de aplicaciones
 - ▶ Los entornos de Desarrollo, QA y Producción deben ser configurados idénticamente (con diferentes contraseñas usadas en cada entorno).
- Puede “volcar” la configuración de la aplicación?
 - ▶ Desarrolle reportes en sus procesos
 - ▶ La regla es simple: Si no se puede verificar, no es seguro
- Verificar la implementación
 - ▶ Un simple escaneo puede encontrar problemas de configuración genéricos y parches faltantes



El problema de las vulnerabilidades



Cada vulnerabilidad surge de...



Resumen: ¿Cómo resuelvo los problemas de seguridad en mis aplicaciones?

■ Desarrollar código seguro

- Seguir las mejores prácticas en la Guía de OWASP: “Como construir aplicaciones web seguras”
 - <http://www.owasp.org/index.php/Guide>
- Utilizar el Estándar OWASP de **Verificación de Seguridad en Aplicaciones (ASVS)** como una guía para determinar los requerimientos para que una aplicación sea segura
 - <http://www.owasp.org/index.php/ASVS>
- Utilizar componentes estándares de seguridad que se adapten a la organización o entorno (OWASP's ESAPI)

■ Revisar las aplicaciones

- Formar un equipo de expertos para revisar las aplicaciones
- Revisar las aplicaciones siguiendo las guías OWASP
 - Guía de Revisión de Código OWASP:
http://www.owasp.org/index.php/Code_Review_Guide
 - Guía de Testeos OWASP:
http://www.owasp.org/index.php/Testing_Guide



Resumen

- Documentación de OWASP
- OWASP Top 10 2013
- ¿Y ahora quién podrá (defenderme)?
 - ▶ Ustedes!

OWASP
The Open Web Application Security Project
OWASP Development
Guide v2.0.1
(Guía de Desarrollo)

beta



OWASP
The Open Web Application Security Project
OWASP Testing Guide v3.0
(Guía de Pruebas)

beta



OWASP
The Open Web Application Security Project
Backend Security Project

beta



Referencias

- OWASP Top 10 2013
- OWASP Uruguay
- OpenClipart
- Mantralooks (OWASP Mantra)
- **C**oncientizar
- **H**erramientas
- **A**uditar
- **U**sabilidad

