Programación web segura

Puesta en producción Segura

Curso de Especialización en Ciberseguridad

Prof. Daniel Fuentes Brenes

Índice

- Objetivos
- Introducción
- Tecnologías Web
- Conceptos generales de seguridad de aplicaciones web
- Controles de seguridad
- Clasificación de vulnerabilidades y ataques
- Recomendaciones de seguridad

Índice

- **Objetivos**
- Introducción
- Tecnologías Web
- Conceptos generales de seguridad de aplicaciones web
- Controles de seguridad
- Clasificación de vulnerabilidades y ataques
- Recomendaciones de seguridad

Objetivos

- Conocer el amplio rango de debilidades de las aplicaciones web.

- Ser consciente de las recomendaciones de seguridad básicas para el desarrollo de aplicaciones web.

Índice

- Objetivos
- **Introducción**
- Tecnologías Web
- Conceptos generales de seguridad de aplicaciones web
- Controles de seguridad
- Clasificación de vulnerabilidades y ataques
- Recomendaciones de seguridad

Introducción

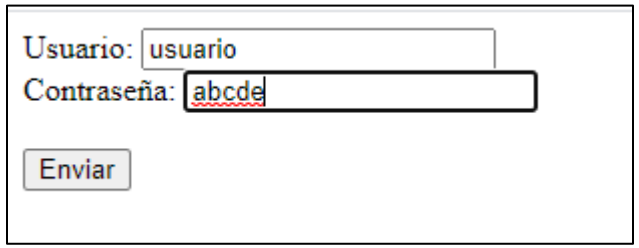
index.php

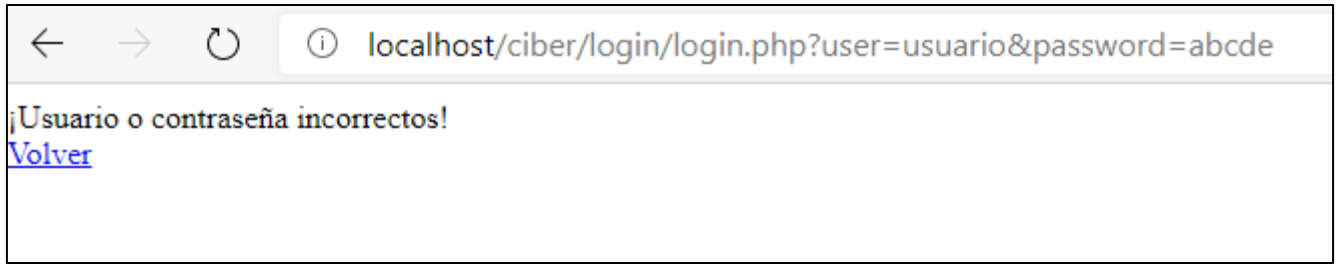
```php
<html>
<head>
    <title>Login</title>
</head>
<body>
<div id="login">
    <form action= "login.php" method="GET">
        <label>Usuario: </label>
        <input type="text" name="user"/><br>
        <label>Contraseña: </label>
        <input type="text" name="password"/><br><br>
        <input type="submit" value="Enviar"/>
    </form>
</div>
</body>
</html>
```

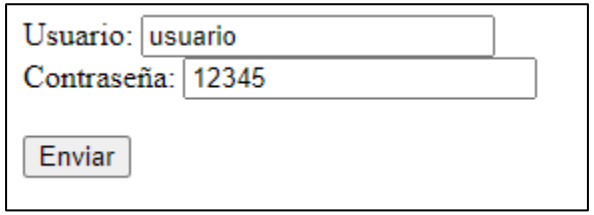
login.php

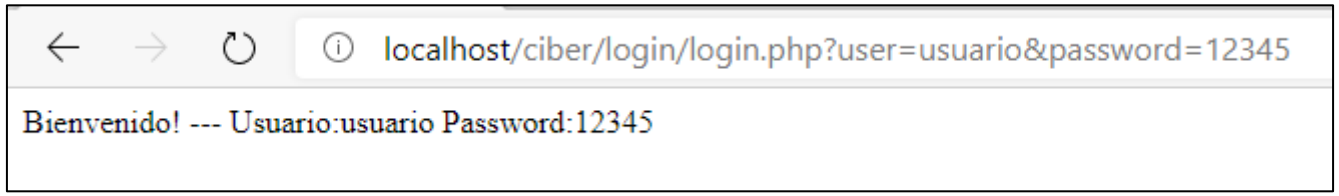
```php
<?php
$user = $_GET['user'];
$password = $_GET['password'];

if (($user == "usuario") AND ($password == "12345")) {
    echo "Bienvenido! --- Usuario:".$user." Password:".$password;
} else {
    echo "¡Usuario o contraseña incorrectos!";
    echo '<br><a href="index.php">Volver</a>';
}
?>
```

Introducción

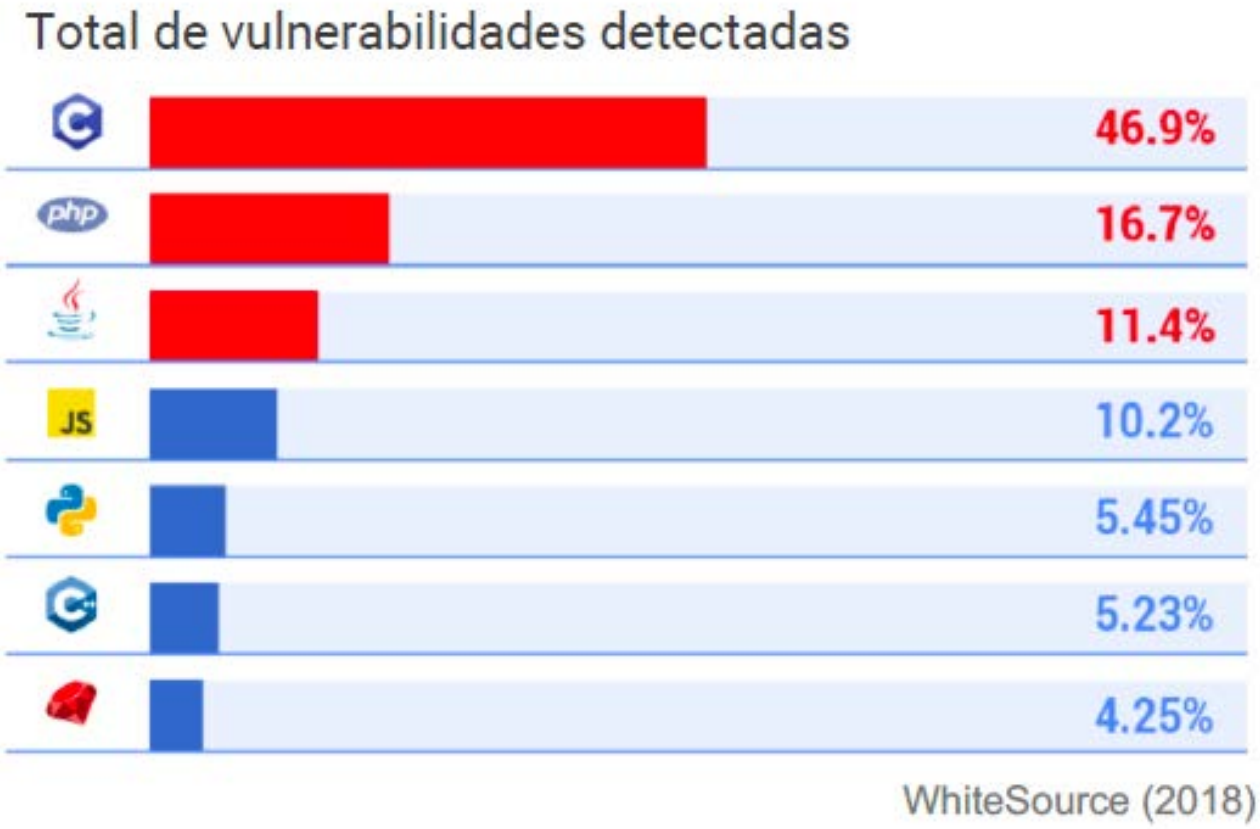




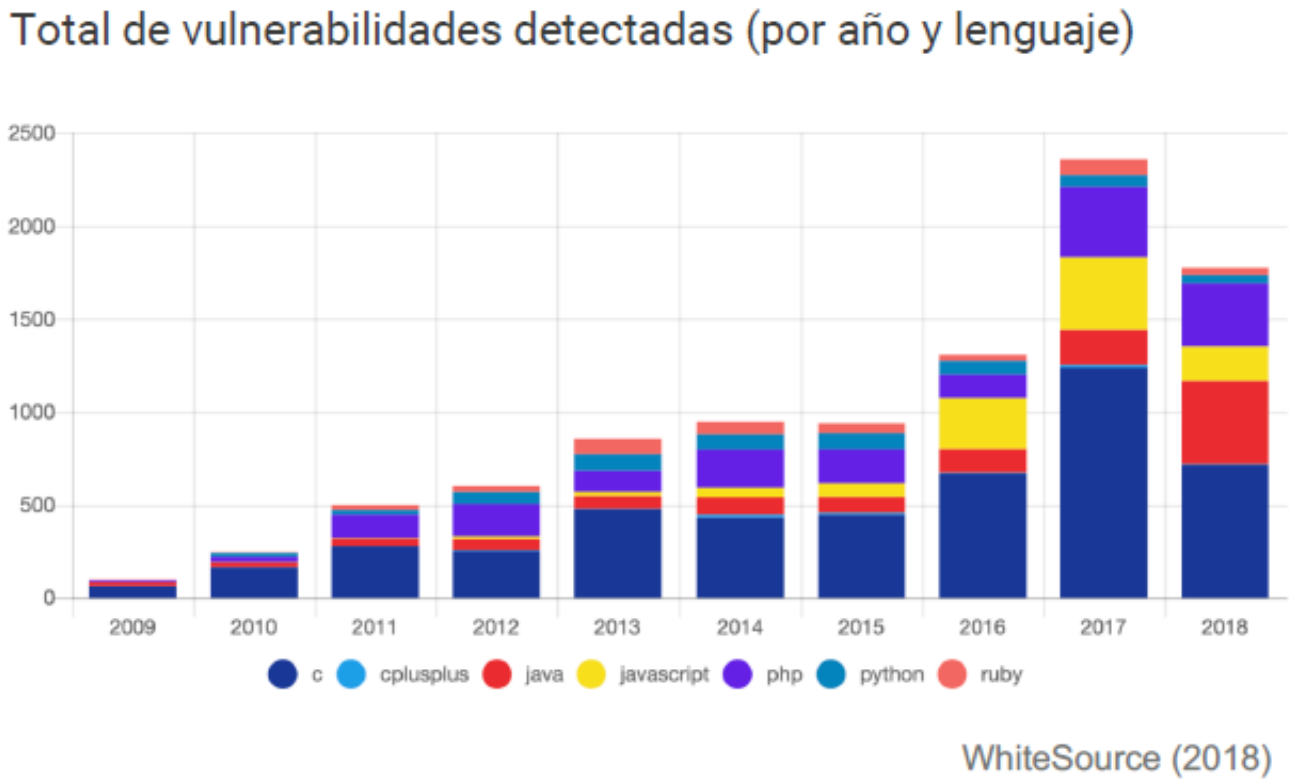




- **Lenguajes de programación**



- **Lenguajes de programación**



Causas

- Uso de Internet a nivel mundial.
- Aumento de desarrollo de aplicaciones para la web. Aplicaciones de escritorio $\rightarrow$ App web y móvil.
- Datos accesibles desde cualquier lado.
- No se ha prestado mucha atención a la seguridad.
- Más del 60% del total de los ataques observados en Internet se realizan a aplicaciones Web.
- Más del 80% de las vulnerabilidades descubiertas están en aplicaciones web.

Causas

- Los diferentes agentes que intervienen en el desarrollo y explotación de aplicaciones web no son conscientes de los peligros que les acechan.
- Se infravalora la seguridad.
- Hay que conocer bien los riesgos para poder minimizarlos mediante técnicas de seguridad.

Índice

- Objetivos
- Introducción
- **Tecnologías Web**
- Conceptos generales de seguridad de aplicaciones web
- Controles de seguridad
- Clasificación de vulnerabilidades y ataques
- Recomendaciones de seguridad

¿Qué es una aplicación web?

- Una aplicación cliente servidor que interactúa con usuarios u otras aplicaciones utilizando el protocolo HTTP.
- Los usuarios suelen interaccionar con los navegadores. Las aplicaciones, con agentes web.

¿De qué se componen las aplicaciones web?

- Tres capas:
- 1)De presentación
  - Es la responsable de presentar datos al usuario o sistema final. El servidor devuelve datos y el navegador los visualiza de una forma entendible por el usuario.
  - Este, as su vez, interacciona y envía al servidor valores de parámetros, que el servidor pasa a la siguiente capa.
  - En esta capa están los servidores y los navegadores.

¿De qué se componen las aplicaciones web?

- Tres capas:
- 2)De aplicación
  - Es el motor de la aplicación web. Implementa la lógica de negocio, procesa la entrada del usuario, toma decisiones, obtiene datos y los presenta a la capa de presentación para enviarlos al usuario.
  - Incluye tecnologías como CGI, Java, servicios .NET, PHP,…

¿De qué se componen las aplicaciones web?

- Tres capas:
- 3) De datos
  - Se emplea para almacenar aquellos datos necesitados por las aplicaciones y actúa de repositorio, tanto temporal como permanente de datos.
  - Bases de datos y repositorios XML.

Protocolo Hypertext Transfer Protocol (HTTP)

- Protocolo de comunicaciones usado para acceder a la World Wide Web y que emplean las aplicaciones web hoy en día.
- Se basa en un modelo de mensajes en el que un cliente envía una petición y el servidor devuelve un mensaje.
- Todos los mensajes disponen de una o más cabeceras y un cuerpo del mensaje.

Protocolo Hypertext Transfer Protocol (HTTP)

- Petición HTTP

```
GET /auth/488/YourDetails.ashx?uid=129 HTTP/1.1
Accept: application/x-ms-application, image/jpeg, application/xaml+xml,
image/gif, image/pjpeg, application/x-ms-xbap, application/x-shockwave-
flash, */*
Referer: https://mdsec.net/auth/488/Home.ashx
Accept-Language: en-GB
User-Agent: Mozilla/4.0 (compatible; MSIE 8.0; Windows NT 6.1; WOW64;
Trident/4.0; SLCC2; .NET CLR 2.0.50727; .NET CLR 3.5.30729; .NET CLR
3.0.30729; .NET4.0C; InfoPath.3; .NET4.0E; FDM; .NET CLR 1.1.4322)
Accept-Encoding: gzip, deflate
Host: mdsec.net
Connection: Keep-Alive
Cookie: SessionId=5B70C71F3FD4968935CDB6682E545476
```

Protocolo Hypertext Transfer Protocol (HTTP)

- Respuesta HTTP

```
HTTP/1.1 200 OK
Date: Tue, 19 Apr 2011 09:23:32 GMT
Server: Microsoft-IIS/6.0
X-Powered-By: ASP.NET
Set-Cookie: tracking=tI8rk7joMx44S2Uu85nSWc
X-AspNet-Version: 2.0.50727
Cache-Control: no-cache
Pragma: no-cache
Expires: Thu, 01 Jan 1970 00:00:00 GMT
Content-Type: text/html; charset=utf-8
Content-Length: 1067

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://
www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd"><html xmlns="http://
www.w3.org/1999/xhtml" ><head><title>Your details</title>
...
```

Protocolo Hypertext Transfer Protocol (HTTP)

- HTTP vs HTTPS

  - HTTP: Utiliza TCP como mecanismo de transporte sin encriptación. Fácilmente interceptable por atacantes.

  - HTTPS : El mismo protocolo en la capa de aplicación, pero en el que se emplea un mecanismo de transporte seguro, el Secure Sockets Layer (SSL). Reduce el riesgo de intercepción.

Protocolo Hypertext Transfer Protocol (HTTP)

- Métodos HTTP
  - GET: Diseñado para recuperar recursos.
    - Se puede emplear para enviar parámetros al recurso pedido mediante la cadena del URL.
    - Dichos URLs se pueden mostrar y almacenar.
    - Capacidad limitada.
  - POST: Diseñado para realizar acciones (formularios).
    - Introduce los parámetros en la solicitud HTTP para el servidor. Por ello, no quedan visibles para el usuario.
    - La capacidad del método POST es ilimitada.
    - Al ir hacia atrás en un navegador que ha hecho una petición por POST no se genera automáticamente la petición.

Protocolo Hypertext Transfer Protocol (HTTP)

- Cookies

  - Permiten al servidor enviar ítems de datos al cliente, el cual almacena y reenvía en su momento al servidor.
  - Un servidor crea y asigna una cookie mediante la cabecera de la respuesta:
  - Set-Cookie: usuario=2325423k2342j2342
  - Pares nombre/valor, con tiempo de expiración.

Protocolo Hypertext Transfer Protocol (HTTP)

- Cookies

  - Permiten al servidor enviar ítems de datos al cliente, el cual almacena y reenvía en su momento al servidor.
  - Un servidor crea y asigna una cookie mediante la cabecera de la respuesta:
  - Set-Cookie: usuario=2325423k2342j2342
  - Pares nombre/valor, con tiempo de expiración.

Protocolo Hypertext Transfer Protocol (HTTP)

Proxy HTTP

- Servidor que media el acceso entre un navegador y el servidor web de destino.
- Cuando un cliente usa un proxy es este quien hace todas las peticiones al servidor.
- La mayoría de los proxies también realizan funciones de caché, autenticación y control de acceso.

Tecnologías para realizar funcionalidades

Del lado del servidor

- Actualmente las aplicaciones web emplean tanto recursos estáticos como dinámicos, creándose la respuesta del servidor sobre la marcha en estos últimos, recibiendo el usuario contenido personalizado.

- Este contenido dinámico se genera mediante la ejecución de guiones o código en el servidor. Ejemplo: <?php … ?>

Tecnologías para realizar funcionalidades

Del lado del servidor

- Normalmente, las peticiones del cliente no se reducen a un URL sino que se envían varios parámetros.
- También se puede considerar información de la cabecera de la petición (por ejemplo, el tipo de navegador).

Tecnologías para realizar funcionalidades

Del lado del servidor

- Las aplicaciones web pueden usar un buen número de tecnologías en el servidor:
  - Lenguajes interpretados: PHP, VBScript, Perl, Ruby on Rails,…
  - Plataforma de desarrollo Web: ASP.NET, Java.
  - Servidores Web: Apache, IIS, Tomcat,…
  - Bases de datos: SQL y NoSQL.
  - Otros componentes en el back-end: sistemas de ficheros, servicios SOAP,…

Tecnologías para realizar funcionalidades

Del lado del cliente

- Lenguajes de marcas: HTML, XML, XHTML. Lenguajes de presentación: CSS, XSLT.
- Leguajes de programación: JavaScript, JQuery, AJAX, VBScript, etc.
- Formatos de transferencia: XML, JSON.

Índice

- Objetivos
- Introducción
- Tecnologías Web
- **Conceptos generales de seguridad de aplicaciones web**
- Controles de seguridad
- Clasificación de vulnerabilidades y ataques
- Recomendaciones de seguridad

¿Activo, riesgo, amenaza y vulnerabilidad?

- **Activo** = Personas, propiedades e información.
- **Amenaza** = Lo que puede explotar una vulnerabilidad, de forma intencional o accidental, obtener, dañar o destruir un activo.
- **Vulnerabilidad** = Debilidad a nivel de seguridad de una aplicación que puede ser explotada a partir de amenazas y así obtener el activo.
- **Riesgo** = La potencial pérdida, daño o destrucción de un activo como resultado de una amenaza que explota una vulnerabilidad.

¿Activo, riesgo, amenaza y vulnerabilidad?

- **Activo** = Lo que estamos intentando proteger.
- **Amenaza** = Contra quien estamos intentándolo proteger.
- **Vulnerabilidad** = Debilidad en nuestros esfuerzos de protección.
- **Riesgo** = Intersección entre activo, amenaza y vulnerabilidad.
- Un atacante (amenaza) puede explotar una vulnerabilidad (fallo de seguridad) y permitir acceder a un recurso (activo) $\rightarrow$ Riesgo.

Objetivos de un ataque:

- Robar datos.
- Chantaje.
- Trampolín para otros ataques.
- Vandalismo.
- Demostrar vulnerabilidad o satisfacer la curiosidad.
- Debilitar la reputación de la compañía.

Índice

- Objetivos
- Introducción
- Tecnologías Web
- Conceptos generales de seguridad de aplicaciones web
- **Controles de seguridad**
- Clasificación de vulnerabilidades y ataques
- Recomendaciones de seguridad

Controles de seguridad

Los **controles de seguridad primarios** que cualquier aplicación (web) debe incorporar son:

- “CIA Triad” → Confidenciality, Integrity and Availability
- Control de acceso
- “Nonrepudiation controls”

Controles de seguridad

Confidencialidad: protección de la información ante la divulgación no autorizada.

- Información sensible: cuentas bancarias, información personal, números de tarjetas de crédito, secretos y documentos de estado, etc.
- Las técnicas criptográficas ofrecen dicha confidencialidad y protegen los datos (Advanced Encription Standard (AES)).
- Transmisión mediante el protolocol SSL/TLS.
- Permisos de usuario y listas de acceso restringido para acceder a información sensible.

Controles de seguridad

Integridad: protección ante modificaciones no autorizadas de la información.

- Solución: criptografía → “hasing data” MD5

Disponibilidad: las entidades autorizadas podrán acceder a la información cuando se necesite y de forma continuada.

- Ejemplos: ataque de denegación de servicios o desastres naturales.
- Sistemas tolerantes a fallos, redundancias, copias de seguridad, etc.

Controles de seguridad

Control de acceso (autenticación y autorización): identificar qué entidad está intentando acceder a un recurso y controlar dicho uso por medio de niveles de derechos.

- Tiene que ver con la integridad. Contraseñas estáticas.
- Contraseñas de un solo uso. Autenticación biométrica.
- Basada en claves públicas y privadas.

Controles de seguridad

“Nonrepudation controls”: permiten evitar que los usuarios puedan negar que han realizado acciones.

- Firma de un documento informando al usuario. Autentificación + técnicas criptográficas.
- Registro de quién realiza qué transacciones (accountability).

Índice

- Objetivos
- Introducción
- Tecnologías Web
- Conceptos generales de seguridad de aplicaciones web
- Controles de seguridad
- **Clasificación de vulnerabilidades y ataques**
- Recomendaciones de seguridad

Ataques web

Hacking web

- Búsqueda y explotación de vulnerabilidades de seguridad en sistemas o redes.

- Tipos de hackers

	White Hat	Black Hat	Grey Hat
Fines maliciosos	**No**	**Sí**	**No**
Contratado por compañía	**Sí**	**No**	**No**
Permiso del dueño de la aplicación	**Sí**	**No**	**No**

Ataques web

Tipos de ataques

- Inyecciones de código
- Ficheros
- Robo de sesiones
- Accesos ilegales
- Otros

Ataques web

Tipos de ataques

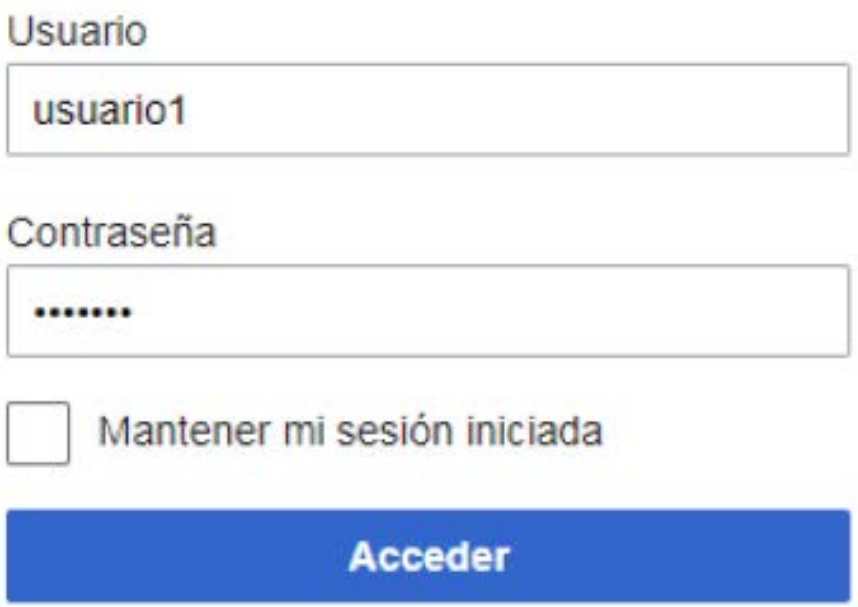
- Inyecciones de código
- Ficheros
- Robo de sesiones
- Accesos ilegales
- Otros

Ataques web

Inyecciones de código: **Inyección SQL**

- Manipulación de una consulta SQL para lograr un resultado diferente al objetivo con el que fue diseñada.
- Se centra en sitios web que construyen sentencias SQL a partir de entradas dadas por el usuario, normalmente para generar páginas dinámicas.
- Si un atacante es capaz de alterar la construcción de una sentencia SQL, se ejecutará con los mismos permisos que se ejecutaría la sentencia en condiciones normales, pudiendo ganar el control absoluto de la base de datos o ejecutar programas en el sistema.

Inyecciones de código: **Inyección SQL**

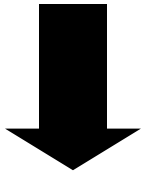


Inyecciones de código: **Inyección SQL**

```
Select * from users where usuario =
‘ +USUARIO+ ’ and password = ‘
```

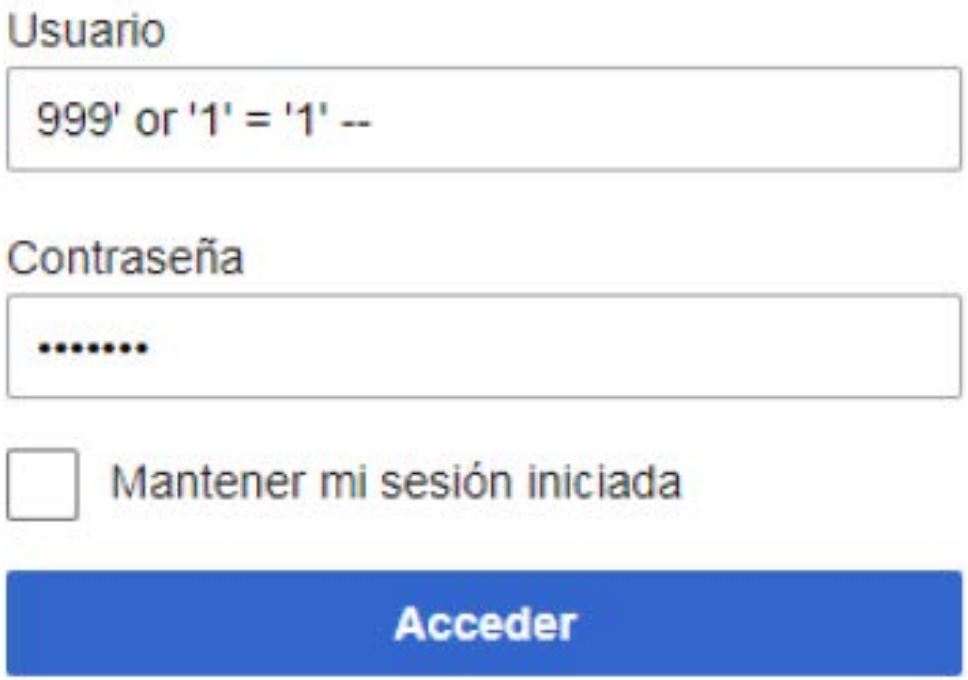
USUARIO = “usuario1”

PASSWORD = “password1”



```
Select * from users where usuario =
‘usuario1’ and password =
‘password1’
```

Inyecciones de código: **Inyección SQL**

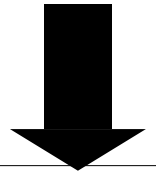


Inyecciones de código: **Inyección SQL**

```
Select * from users where usuario =
‘ +USUARIO+ ’ and password = ‘
```

USUARIO = “999” or ‘1’ = ‘1’ --”

PASSWORD = “password1”



```
Select * from users where usuario =
‘999’ or ‘1’ = ‘1’ --’ and password =
‘password1’
```

Inyecciones de código: **Inyección SQL**

- ¿Cómo evitar la Inyección SQL? **SANEAR SIEMPRE LOS DATOS**

Ataques web

Inyecciones de código: **Cross-site scripting (XSS)**

- Vulnerabilidad que permite la ejecución de script en el lado del cliente al visualizar una página web.
- Normalmente en HTML/JavaScript, pero también en otro lenguajes como VBScript, ActiveX, Java, Flash o cualquier tecnología basada en el cliente.
- Cuando un atacante hace que el navegador ejecute un código, este se ejecutará dentro de la zona de seguridad del sitio web que lo aloja.
- Tipos:
  - XSS Almacenado
  - XSS Reflejado

Ataques web

Inyecciones de código: **Cross-site scripting (XSS)**

- XSS Almacenado

Ataques web

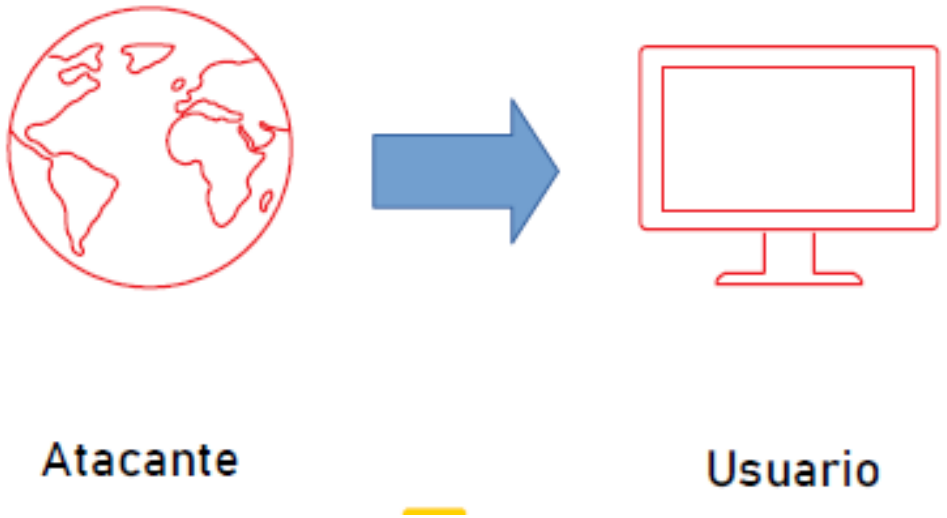
Inyecciones de código: **Cross-site scripting (XSS)**

- XSS Almacenado

Ataques web

Inyecciones de código: **Cross-site scripting (XSS)**

- XSS Reflejado



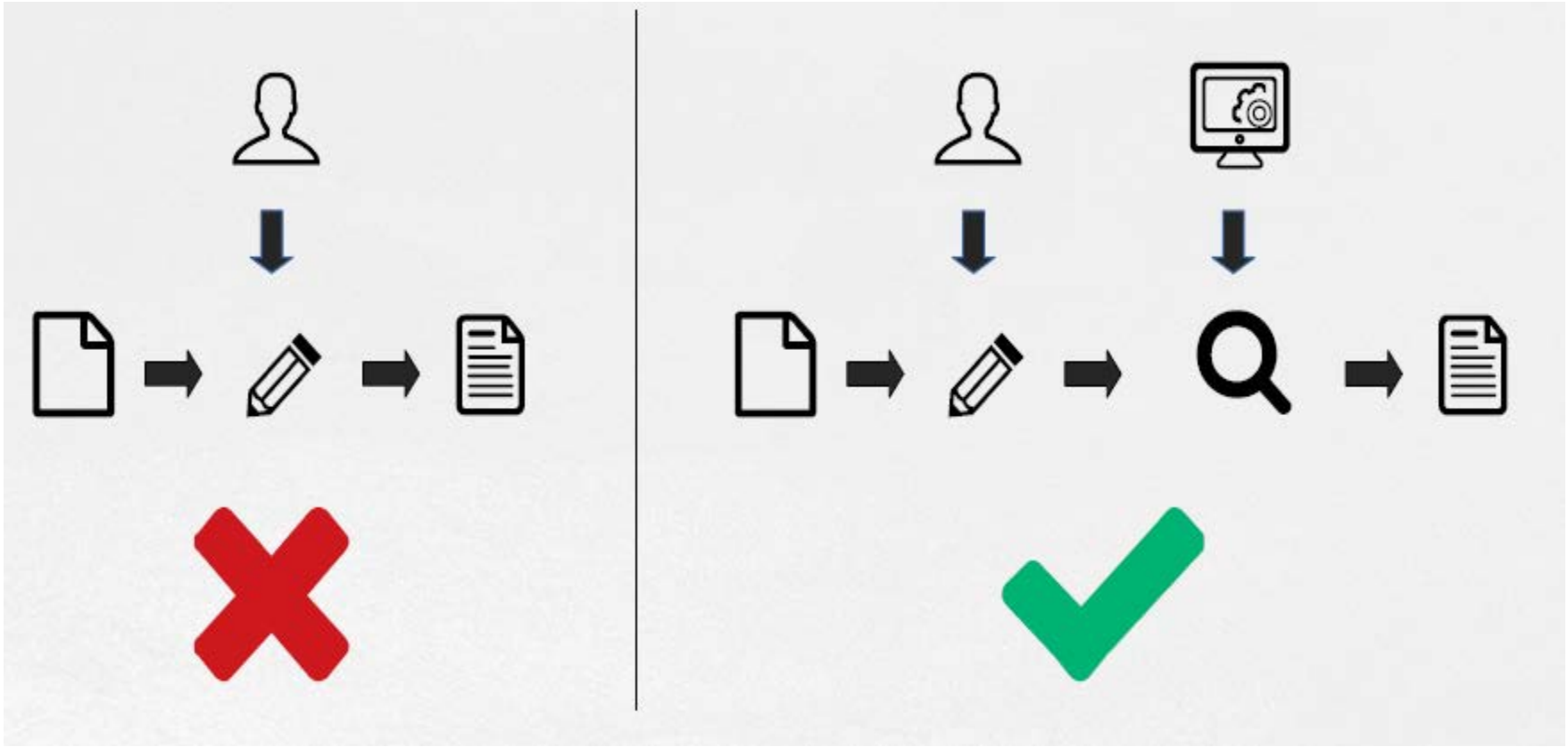
Ataques web

Inyecciones de código: **Cross-site scripting (XSS)**

- XSS Reflejado

Inyecciones de código: **Cross-site scripting (XSS)**

- ¿Cómo evitar Cross-site scripting (XSS)? **SANEAR SIEMPRE LOS DATOS**



Ataques web

Tipos de ataques

- Inyecciones de código
- Ficheros
- Robo de sesiones
- Accesos ilegales
- Otros

Ficheros: **Unrestricted File Upload**

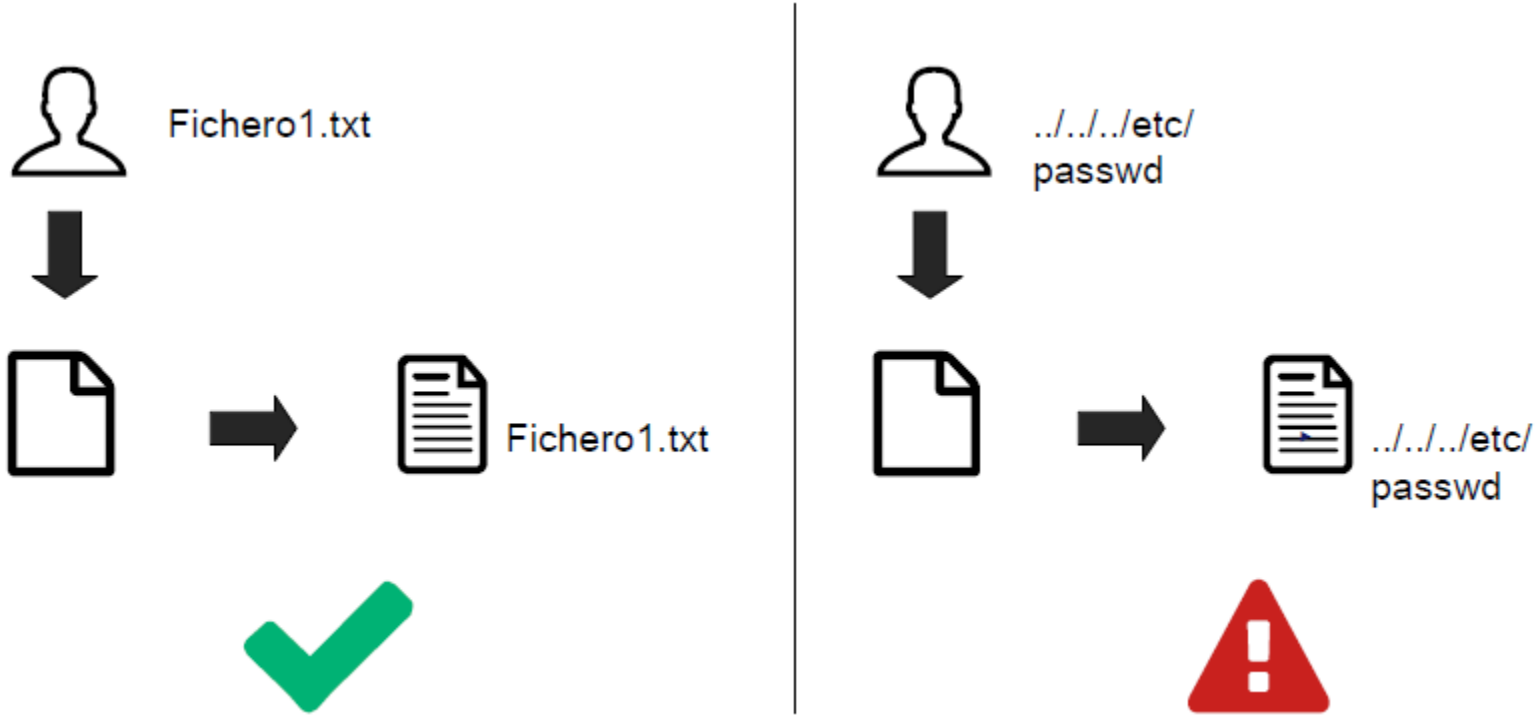
- Formulario de subida de ficheros, sin restricciones en tamaño, extensión ni tipo de fichero.
- Tipos de ataques:
  - Fichero malicioso ejecutable en servidor (.jsp, .php, etc.).
  - Fichero de gran tamaño.
  - Fichero con nombre/ruta sensible (sobreescribe fichero de sistema).

Ficheros: **Unrestricted File Upload**

- ¿Cómo evitar Unrestricted File Upload?
  - Restringir la extensión y tipo de fichero.
  - Restringir tamaño máximo de fichero.
  - Sanear nombre de fichero.
  - Descargar fichero y no abrir desde el servidor.

Ficheros: **Local File Inclusion**

- Manipulación de la ruta a un fichero del servidor que es cargado mediante la aplicación para acceder a otro de forma fraudulenta.



Ficheros: **Local File Inclusion**

- ¿Cómo evitar Local File Inclusion?
  - Restringir el nombre y la ruta a acceder.

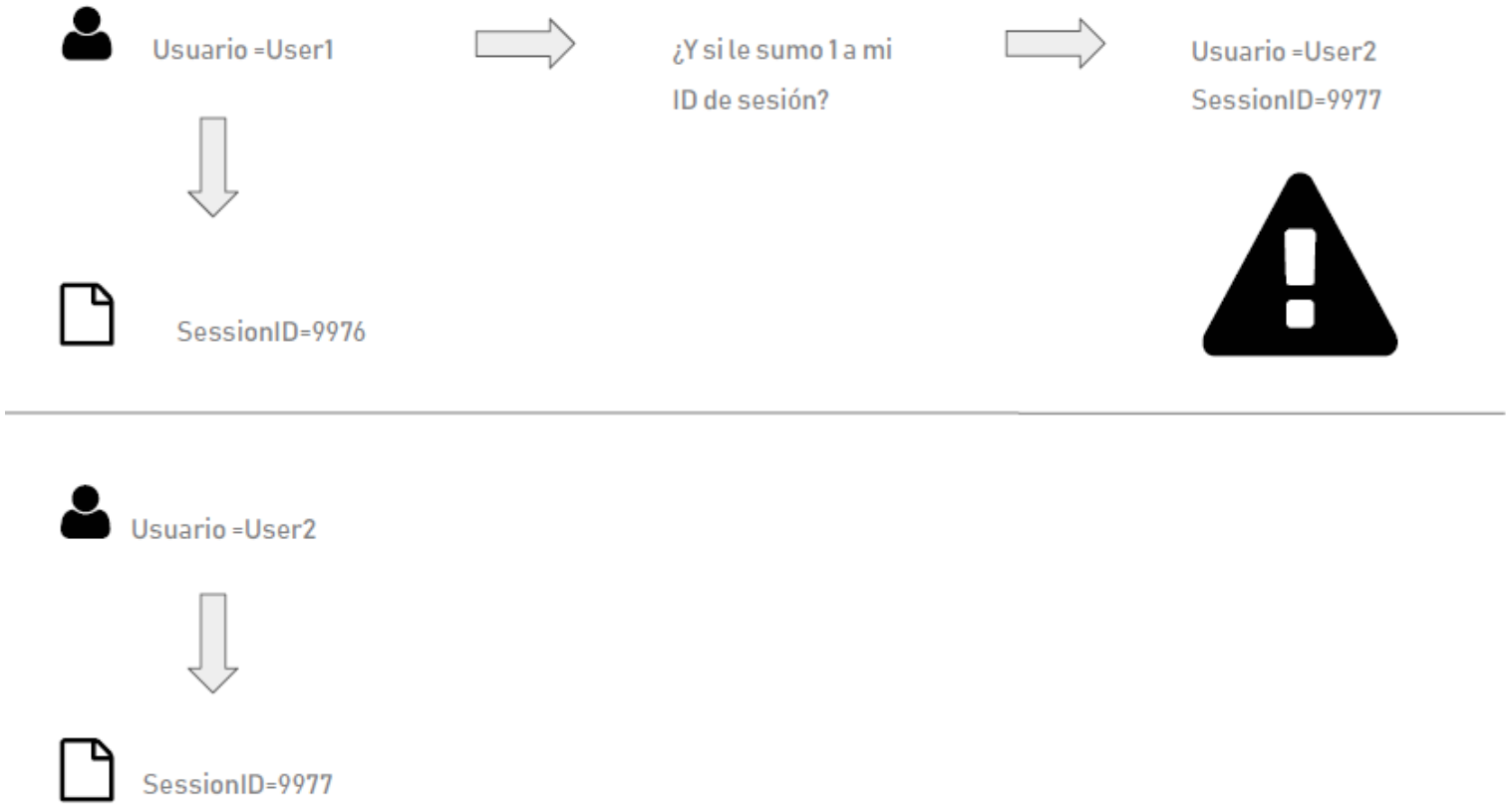
Ataques web

Tipos de ataques

- Inyecciones de código
- Ficheros
- Robo de sesiones
- Accesos ilegales
- Otros

Robo de sesiones: **Session Prediction**

- Deducción, a partir de un id de sesión propio, del id de sesión de otro usuario, con el fin de acceder a la aplicación como dicho usuario.



Robo de sesiones: **Session Prediction**

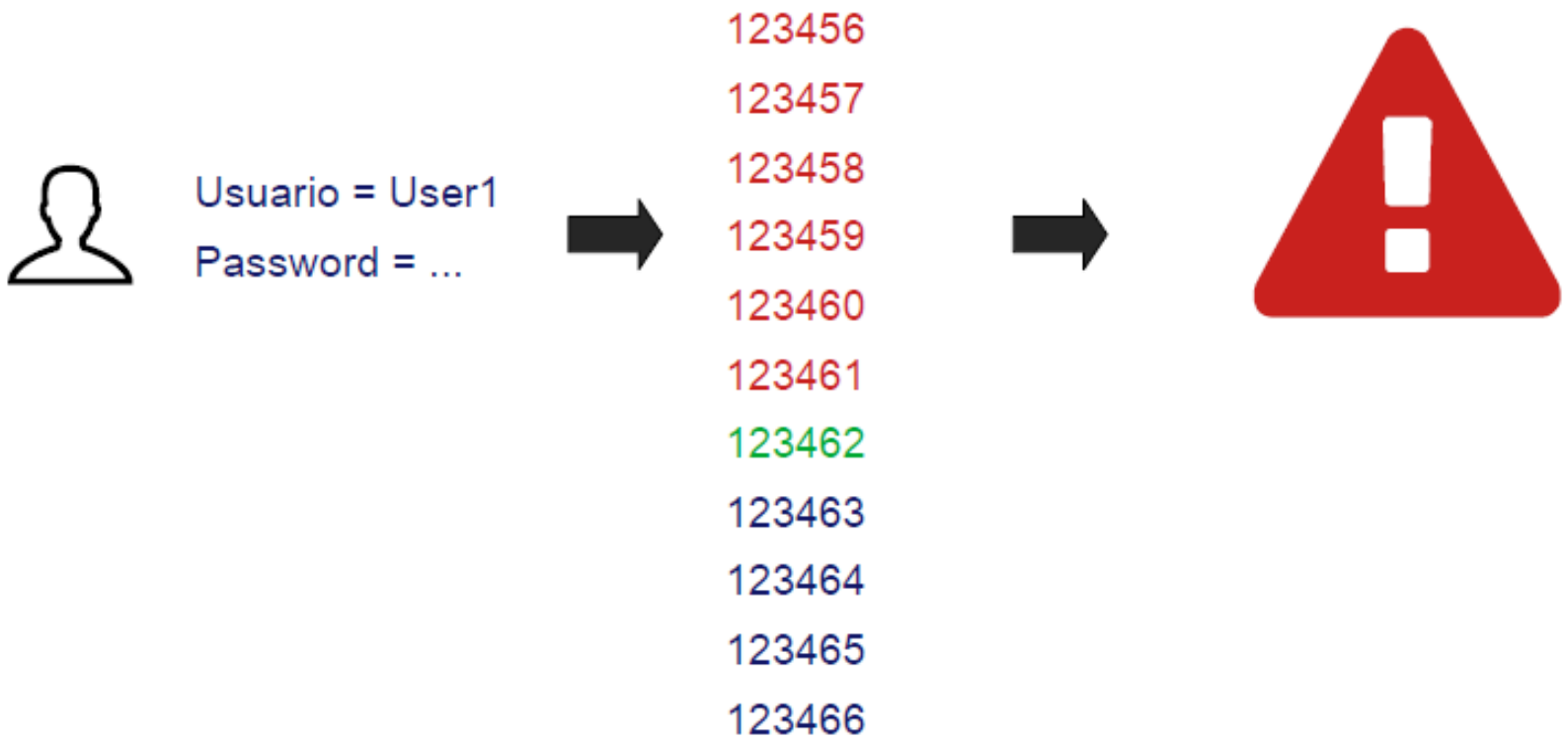
- ¿Cómo evitar Session Prediction?
  - ID de sesión creado en el servidor.
  - Único
  - Aleatorio.
  - Encriptado.
  - Tiempo de expiración.
  - Bonus: solicitar contraseña de usuario actual para poder cambiarla por una nueva.

Ataques web

Robo de sesiones: **Fuerza Bruta**

- Intento de averiguar un dato mediante la prueba consecutiva de todas las posibles combinaciones generando miles de intentos.
- Ataque de prueba y error utilizado para adivinar el nombre de usuario, contraseña, número de tarjeta de crédito o clave criptográfica, entre otras.
- Tipos de fuerza bruta:
  - Fuerza bruta
  - Ataques de diccionario.
  - Fechas.

Robo de sesiones: **Fuerza Bruta**



Robo de sesiones: **Fuerza Bruta**

- ¿Cómo evitar Fuerza Bruta?
  - Contador de reintentos.
  - Captcha
  - Bloqueo de usuario.

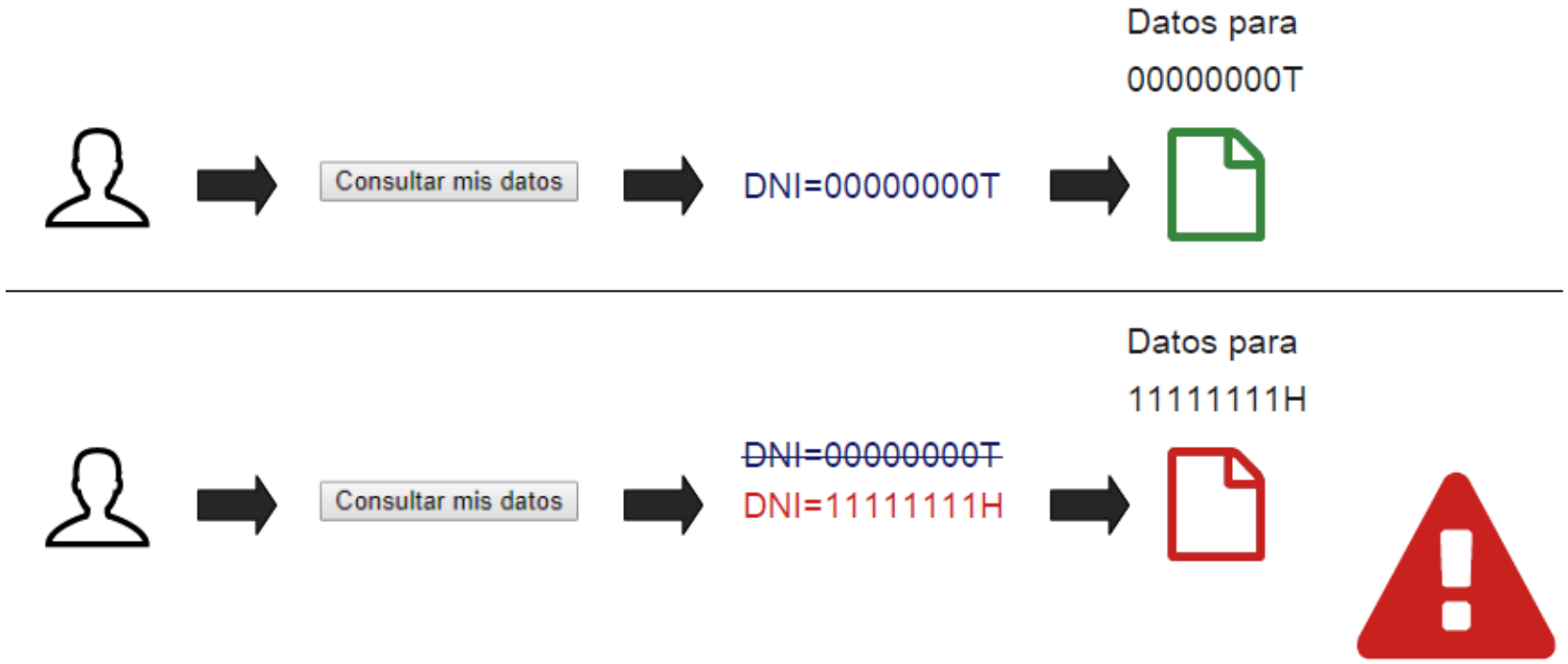
Ataques web

Tipos de ataques

- Inyecciones de código
- Ficheros
- Robo de sesiones
- Accesos ilegales
- Otros

Accesos ilegales: **Parameter Tampering**

- Manipulación del valor de algún parámetro que se intercambia entre cliente y servidor con fines malintencionados.

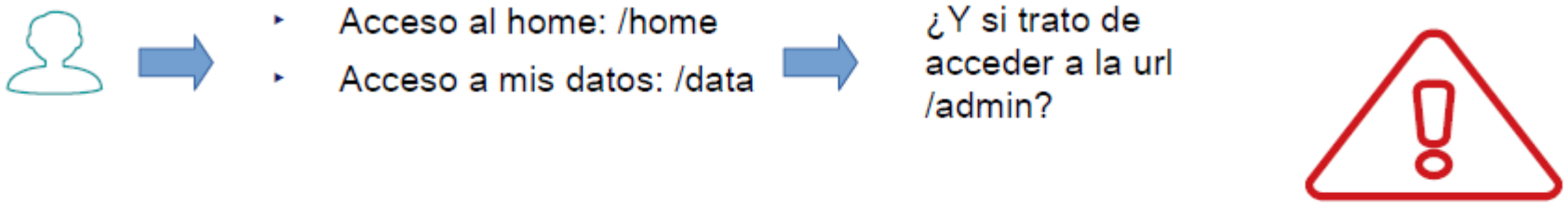


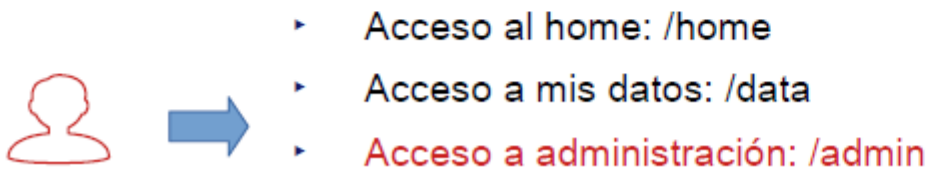
Accesos ilegales: **Parameter Tampering**

- ¿Cómo evitar Parameter Tampering?
  - Validaciones en el servidor.
  - No enviar datos innecesarios.

Accesos ilegales: **Control inseguro de roles**

- Tratamiento inseguro de roles de usuario basándose simplemente en la ocultación de las funcionalidades para las que no se tiene permiso.





Accesos ilegales: **Control inseguro de roles**

- ¿Cómo evitar Control inseguro de roles ?
  - Comprobaciones de permisos en el servidor.
  - Validar en el servidor cada acción del usuario.

Ataques web

Tipos de ataques

- Inyecciones de código
- Ficheros
- Robo de sesiones
- Accesos ilegales
- Otros

Ataques web

Otros ataques: **Buffer overflow**

- Ataques que alteran el flujo de una aplicación mediante la sobrescritura de partes de memoria.
- Objetivo: realizar un desbordamiento y, por ejemplo, llevar a cabo un ataque por denegación de servicio).
- Actualmente no son muy comunes porque el atacante necesita analizar el código fuente de la aplicación o los binarios y modificarlo.

Ataques web

Otros ataques: **Cadenas con formato**.

- Se altera el flujo de la aplicación mediante el uso de cadenas de caracteres que contienen formatos específicos.

- Por ejemplo, las cadenas de funciones como fprintf de C/C++:
  - fprintf(fichero,"%s %d\n", nombre, numero);

Ataques web

Otros ataques: **Inyección LDAP**.

- Lightweight Directory Access Protocol (LDAP) es un protocolo estándar para consultar y mantener servicios de directorios (conjunto de objetos organizados de una forma jerárquica).
- El ataque consistiría en modificar las cadenas de consultas al directorio LDAP.
- Ejemplo búsqueda LDAP:

ldapsearch -D "cn=exampleuser,example.com" -w secret -p 389 -h server.example.com -b "dc=example,dc=com" -s sub "(sn=smith)"

Ataques web

Otros ataques: **OWASP.**

- Las 10 amenazas principales de la seguridad Web según el Open Web Application Security Project (OWASP).

- http://www.owasp.org
- https://owasp.org/www-project-top-ten/
- https://wiki.owasp.org/?title=Special:Redirect/file/OWASP-Top-10-2017-es.pdf (español)

Ataques web

Otros ataques: **OWASP.**

OWASP Top 10 2017
A1:2017 – Inyección
A2:2017 – Pérdida de Autenticación y Gestión de Sesiones
A3:2017 – Exposición de Datos Sensibles
A4:2017 – Entidad Externa de XML (XXE) [NUEVO]
A5:2017 – Pérdida de Control de Acceso [Unido]
A6:2017 – Configuración de Seguridad Incorrecta
A7:2017 – Secuencia de Comandos en Sitios Cruzados (XSS)
A8:2017 – Deserialización Insegura [NUEVO, Comunidad]
A9:2017 – Uso de Componentes con Vulnerabilidades Conocidas
A10:2017 – Registro y Monitoreo Insuficientes [NUEVO, Comunidad]

Índice

- Objetivos
- Introducción
- Tecnologías Web
- Conceptos generales de seguridad de aplicaciones web
- Controles de seguridad
- Clasificación de vulnerabilidades y ataques
- **Recomendaciones de seguridad**

Recomendaciones

A la hora de diseñar aplicaciones basadas en la Web hay que tener los siguientes principios de seguridad presentes:

1. Validación de la entrada y la salida.

- Todas las entradas y salidas deben ser validadas para asegurarnos que son las esperadas.
- Filtrar sólo lo esperado y rechazar lo contrario (si esperamos un NIF, todo lo que no sea una cadena de 8 números más una letra se descarta).

Recomendaciones

2. Si se produce un fallo, cierre seguro.

- Cualquier mecanismo de seguridad debe ser diseñado de tal forma que si falla, lo haga cerrando todo correctamente.
- Además, debe evitar aceptar peticiones de seguridad posteriores.
- Por ejemplo, si la autenticación de un usuario falla, entonces posteriores peticiones no deben devolver valores negativos o null.

3. Medidas de seguridad simples.

Cuanto más complejo sea el sistema de seguridad, más difícil será de entender y mantener. Es más probable que haya fallos de seguridad. Y además, será un reto para sobrepasarlo.

Recomendaciones

3. Medidas de seguridad simples.

Cuanto más complejo sea el sistema de seguridad, más difícil será de entender y mantener. Es más probable que haya fallos de seguridad. Y además, será un reto para sobrepasarlo.

4. Utilizar y reutilizar componentes seguros.

Seguro que el problema de seguridad que tenemos ya ha sido tratado por alguien y se ha encontrado una solución robusta.

Es mejor aprovechar los esfuerzos de investigación y desarrollo y utilizar esos módulos que se suponen que resuelven el problema.

Recomendaciones

5. Defensa en cascada.

- Confiar en que un componente haga su función al 100% es totalmente irreal.
- Los buenos sistemas no predicen lo inesperado, sino que reaccionan a ello.
- Si un componente tiene un fallo de seguridad, un segundo debe capturarlo

Recomendaciones

7. Separación de privilegios.

- Compartimentar usuarios, procesos y datos ayuda a contener problemas si aparecen.
- Identificar qué privilegios se deben asignar y hacerlo estrictamente.

8. La seguridad basada en la ocultación no funciona.

- Si ocultamos algo no lo estamos protegiendo. Es cuestión de tiempo que a alguien se le ocurra buscarlo o lo encuentre por casualidad. Es mejor protegerlo.