Conferencias FIST

Barcelona, 18 de Marzo

Inseguridad de los sistemas de autenticación en aplicaciones web

Vicente Aguilera Díaz
vaguilera@isecauditors.com

internet security auditors

Contenido

0. Introducción
1. Deficiencias y Ataques al sistema de autenticación
2. Medidas de protección
3. Referencias

0. Introducción

Al hablar de autenticación existen 2 contextos:

- autenticación de usuarios
  proceso mediante el cual alguien prueba su identidad.

- autenticación de datos
  proceso mediante el cual se prueba la integridad de los datos.

Nos centraremos en la **autenticación de usuarios**

0. Introducción

Seleccionar un buen mecanismo de autenticación no es trivial:

- basada en usuario/contraseña
- basada en tokens
- basada en certificados digitales
- basada en mecanismos biométricos
- ...

El mecanimo más extendido: **usuario/contraseña**.

0. Introducción

En adelante, al hablar del sistema de autenticación, nos centraremos en:

autenticación de usuarios basada en formularios

1. Deficiencias y Ataques

1.1 Fugas de información

1.2 Debilidad de los campos del formulario de autenticación

1.3 Deficiencias de las funcionalidades “extras”

1.4 Deficiencias en el sistema de gestión de sesiones

1.5 Validaciones deficientes de los datos de E/S

1.6 Deficiencias en la recogida de datos

1.7 Deficiencias de configuración

1.8 Deficiencias en las relaciones de confianza

1. Deficiencias y Ataques

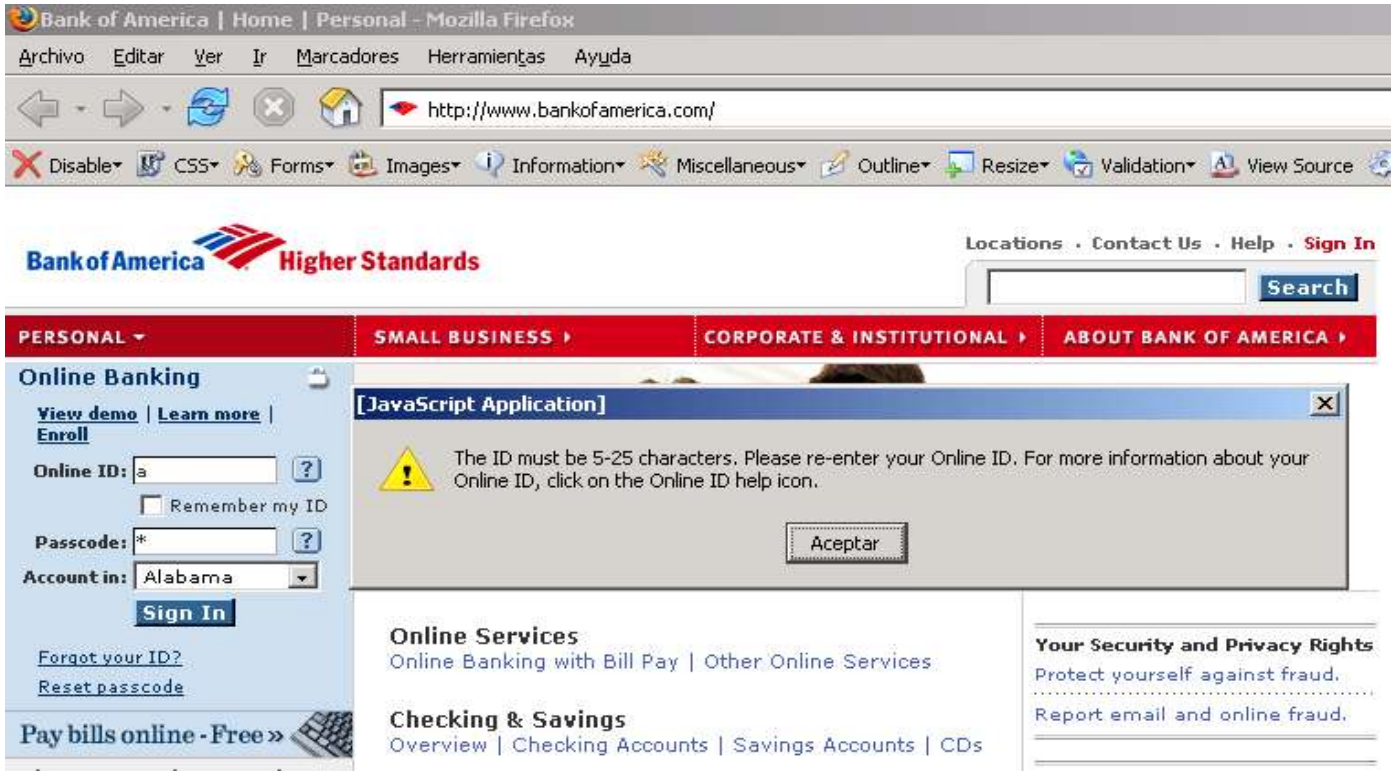
1.1 Fugas de información

- En el código que recibe el cliente
- Mensajes que devuelve la aplicación
- Mensajes en foros y grupos de noticias
- Información facilitada por la empresa de desarrollo
- Webs personales
- Etc.

Veamos un ejemplo...

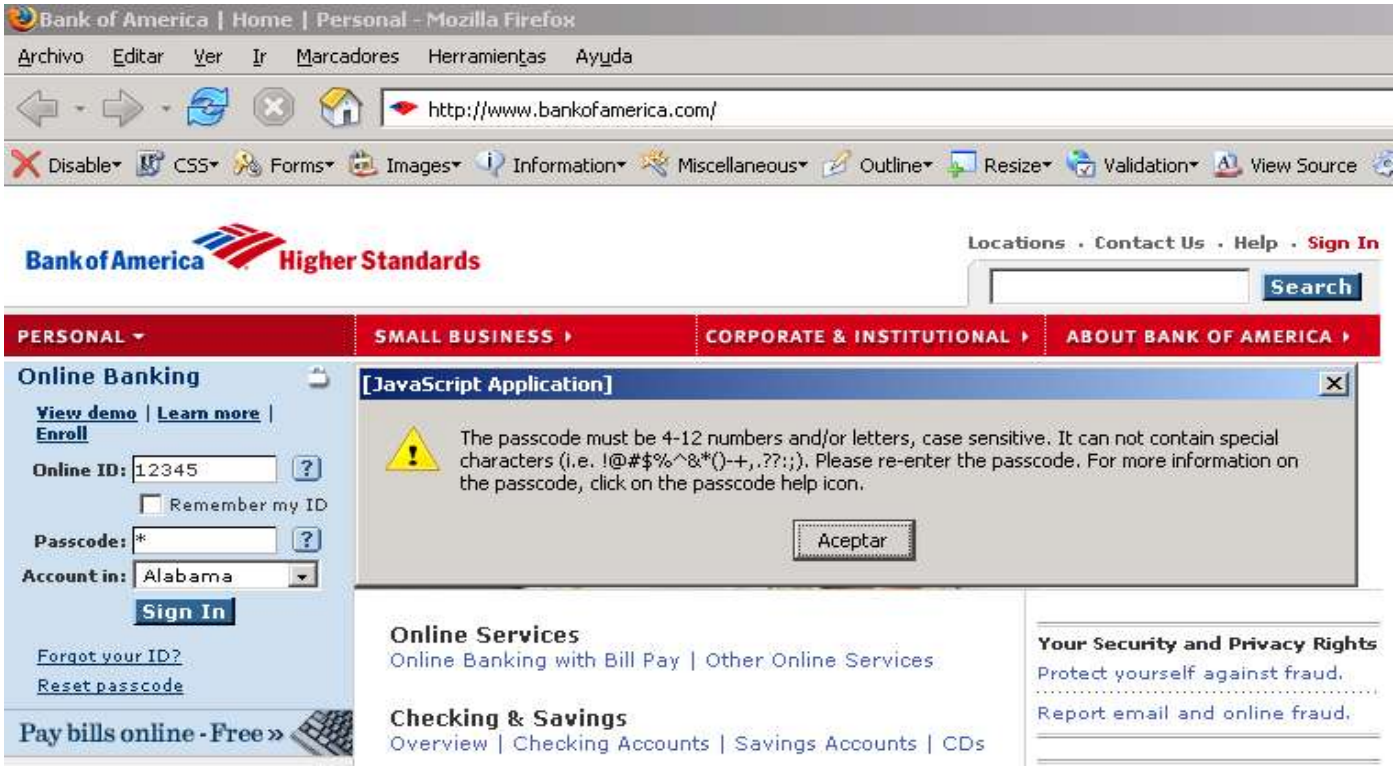
1. Deficiencias y Ataques

1.1 Fugas de información



1. Deficiencias y Ataques

1.1 Fugas de información



1. Deficiencias y Ataques

1.1 Fugas de información

Posibilita:

- Enumeración de usuarios

Permite obtener identificadores válidos de usuario ▸ DoS

- Password cracking

Por fuerza bruta
Basado en diccionarios

- Obtención de información sensible

Usuarios/Contraseñas
Paths de la aplicación
Plataforma y versión de desarrollo
Etc.

1. Deficiencias y Ataques

1.2 Debilidad de los campos del formulario de autenticación

Campos débiles:

- Longitud excesivamente corta
- Rango de caracteres permitido excesivamente limitado
- Atributo “autocomplete=on” por defecto
- Sin límite de caracteres de entrada

Posibilita:

- Enumeración de usuarios
- Password cracking

1. Deficiencias y Ataques

1.3 Deficiencias de las funcionalidades “extras”

Debilidades en los procesos de:

- Registro

  Alta de usuarios

- Modificación de datos

  Modificación de password o datos de registro

- Recuperación de contraseñas

  Implementación del sistema ¿Olvidó su contraseña?

- Otros

1. Deficiencias y Ataques

1.3 Deficiencias de las funcionalidades “extras”

Caso de ejemplo – Proceso de recuperación de contraseñas

- Identificación en persona

  Difícil de llevar a cabo
- Identificación vía fax

  No se dispone de información con la que comparar
- E-mail

  Contraseñas que no caducan, correo compartido, envío “no seguro”, ...
- Pregunta respuesta

  Número de preguntas? Dificultad a la hora de seleccionar preguntas
- Llamada telefónica / SMS
- Métodos híbridos

1. Deficiencias y Ataques

1.3 Deficiencias de las funcionalidades “extras”

Posibilita:

- Enumeración de usuarios
- Password Cracking
- Creación de cuentas válidas ‣ Fugas de información
- Obtención de contraseñas

1. Deficiencias y Ataques

1.4 Deficiencias en el sistema de gestión de sesiones

- Envío de ID de sesión por un canal “inseguro”
- Algoritmo débil en la generación de ID
- Longitud excesivamente corta del ID
- Generación de ID previa a la autenticación
- Validaciones deficientes del ID recibido
- Tiempo de vida ilimitado para el ID

Posibilita ataques de:

- Intercepción
- Predicción
- Fuerza bruta
- Fijación de ID
- Manipulación de ID

1. Deficiencias y Ataques

1.4 Deficiencias en el sistema de gestión de sesiones

Caso de ejemplo – Fijación de ID

1. Generación de ID

El atacante obtiene un ID (se autentica contra la aplicación) o utiliza uno aleatorio. En algunos casos requiere mantener la sesión “viva”

2. Fijación de ID

El atacante necesita introducir el ID generado en el navegador del usuario víctima

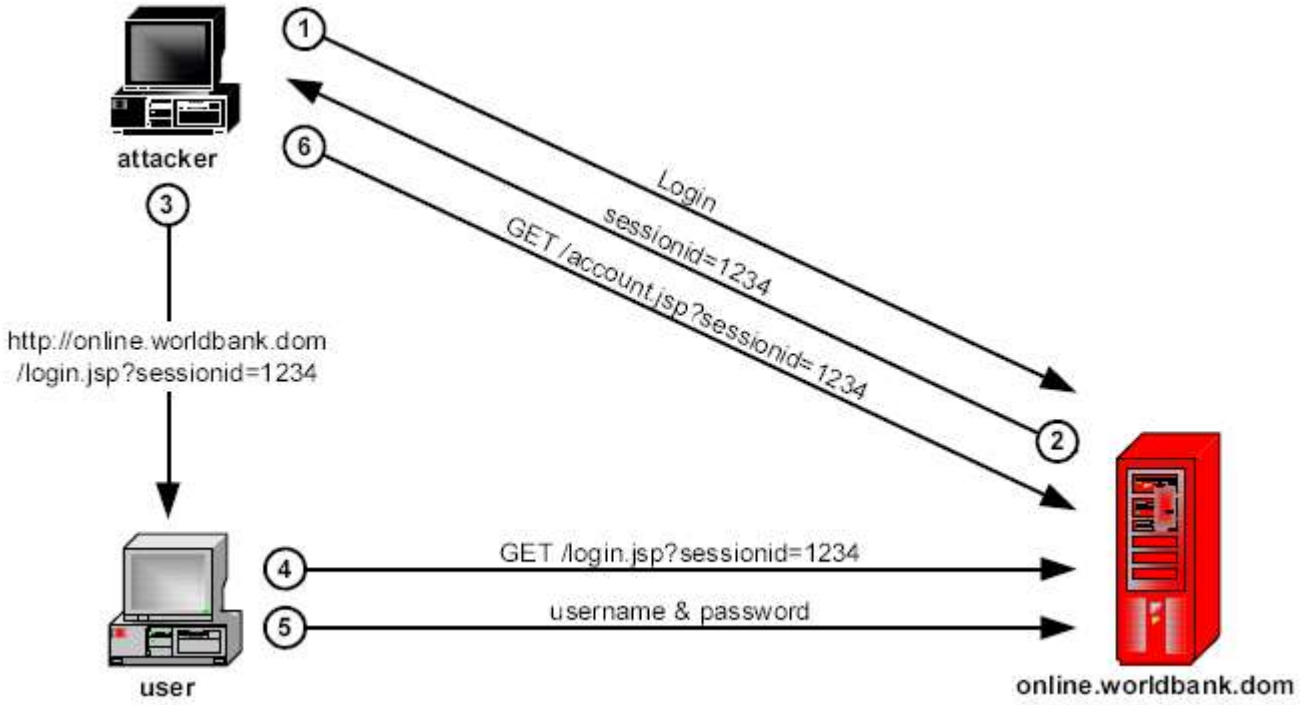
3. Robo de sesión

El atacante espera a que la víctima inicie sesión con el ID fijado a continuación entra en su sesión

1. Deficiencias y Ataques

1.4 Deficiencias en el sistema de gestión de sesiones

Caso de ejemplo – Fijación de ID

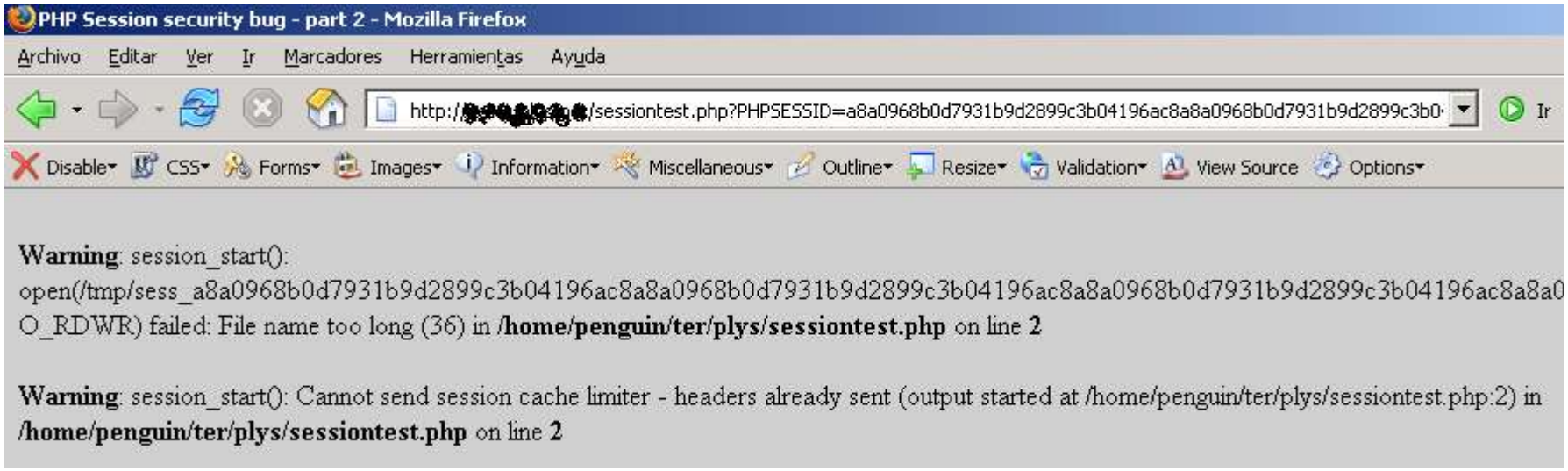


1. Deficiencias y Ataques

1.4 Deficiencias en el sistema de gestión de sesiones

Caso de ejemplo – Manipulación de ID

Introduciendo un PHPSESSID excesivamente largo...



1. Deficiencias y Ataques

1.5 Validaciones deficientes de los datos de E/S

Filtro inexistente/deficiente de validación de los datos de E/S de la aplicación

Posibilita:

- Inyección de código SQL (SQLInjection)

Evasión de la autenticación. Acceso a la base de datos y al sistema operativo.

- Inyección de código script (XSS, XST)

Phishing. Web Defacement. Obtención de información sensible.

- Manipulación de parámetros

Evasión de la autenticación. Escalar privilegios.

- ...

1. Deficiencias y Ataques

1.5 Validaciones deficientes de los datos de E/S

Caso de ejemplo – Inyección de código SQL

- Evadir la autenticación

Hacer que las condiciones especificadas en la cláusula WHERE se cumplan

- Obtener información de la base de datos

Mediante la provocación de errores (p.e. en la conversión de tipos)
Inclusión de sentencias SELECT ... FROM ... WHERE ... LIKE '...%'

Si el servidor web no devuelve una página de error...

Buscar alternativas que permitan deducir el resultado de la sentencia (p.e. en SQLServer incluir WAITFOR DELAY 'HH:MM:SS')

1. Deficiencias y Ataques

1.6 Debilidad en la recogida de datos

- Uso de un canal inseguro
- No redirección tras la autenticación

Posibilita:

- Sniffing
- Phishing
- Suplantación de personalidad
- Obtención de información sensible

1. Deficiencias y Ataques

1.6 Debilidad en la recogida de datos

Caso de ejemplo – Uso de un canal inseguro

Si la página que contiene el formulario de autenticación se accede por HTTP (aunque el envío se realize posteriormente vía HTTPs):

1. El usuario puede desconfiar (al pensar que su información se transmite en claro)

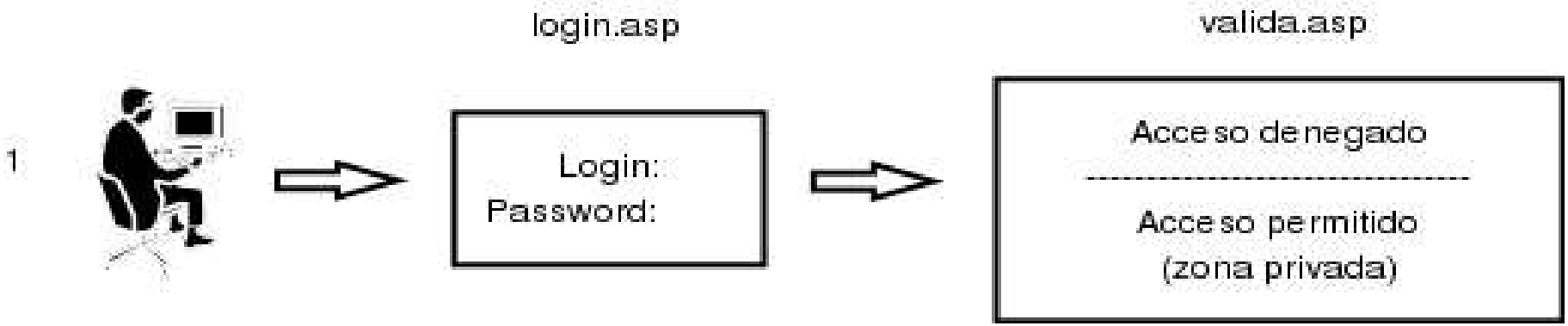
2. Un phiser puede capturar las credenciales antes de enviarlas a la página de validación

1. Deficiencias y Ataques

1.6 Debilidad en la recogida de datos

Caso de ejemplo – No redirección tras la autenticación

Si el usuario cierra la sesión pero no el navegador, otro usuario podría hacer "back" repetidamente y al llegar a la primera página privada refrescarla y se enviarían los datos enviados por POST

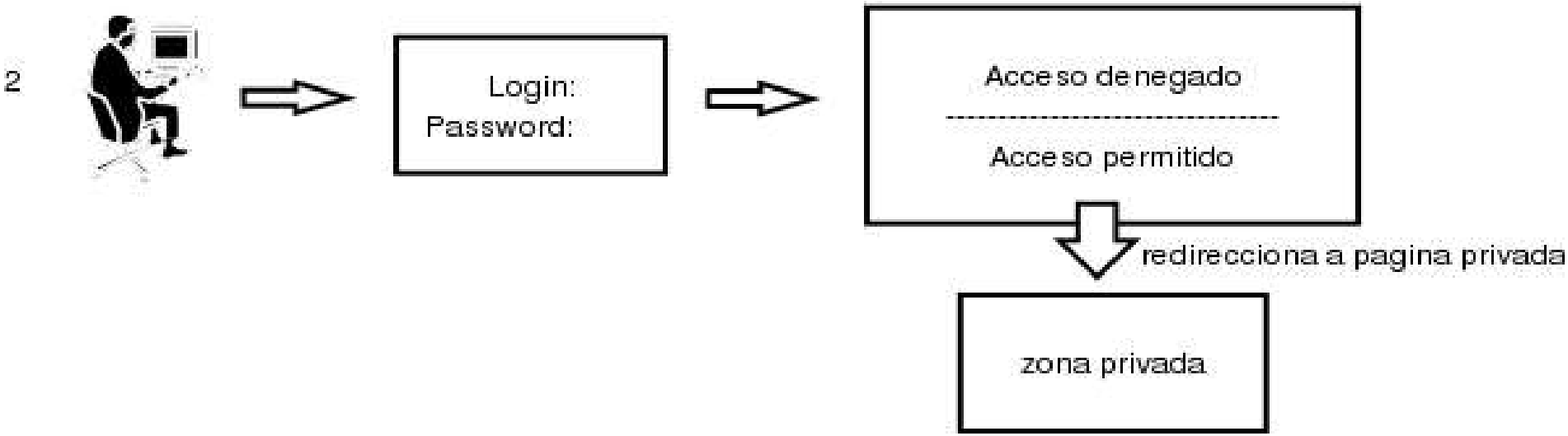


1. Deficiencias y Ataques

1.6 Debilidad en la recogida de datos

Caso de ejemplo – Con redirección tras la autenticación

En este caso no se enviarían los datos del POST



1. Deficiencias y Ataques

1.7 Deficiencias de configuración

- Instalaciones por defecto
- Cuentas por defecto

Posibilita:

- Conocimiento de credenciales válidas

Por ejemplo, usuario/password de la consola de administración de WebSphere: wcsadmin/wcsadmin

- Evasión del sistema de autenticación

Utilizando recursos instalados por defecto por la plataforma y que son conocidos por el atacante

1. Deficiencias y Ataques

1.7 Deficiencias de configuración

Caso de ejemplo – Creación de credenciales

Supongamos una aplicación desarrollada con WebSphere Commerce Suite (WCS) con opciones por defecto, cuyo formulario de autenticación sólo permita introducir ID/password.

Un atacante podría realizar una petición al servlet “UserRegistrationAdd” y crear un usuario con el cual acceder a la aplicación:

https://victima.com/webapp/wcs/stores/servlet/UserRegistrationAdd?URL=LogonForm&logonId=atacanteID&logonPassword=atacantePWD&logonPasswordVerify=atacantePWD

1. Deficiencias y Ataques

1.8 Deficiencias en las relaciones de confianza

Relaciones:

- Aplicación/Base de datos/Servidor Web/Sistema Operativo

Una vulnerabilidad en cualquiera de ellos puede afectar al resto

- Entre servicios

Por ejemplo, la compartición de cuentas de usuario

- Usuarios y su entorno

Existe la posibilidad de atacar directamente al usuario y/o su entorno (físico/personal) aplicando técnicas de ingeniería social

1. Deficiencias y Ataques

1.8 Deficiencias en las relaciones de confianza

Posibilita:

- Reutilización de credenciales

Si se dispone de una cuenta en un servidor FTP, puede ser utilizada para acceder a otros servicios (p.e. Correo)

- Ataques colaterales

Si en la aplicación no se detectan deficiencias, quizás sea factible explotar alguna vulnerabilidad de la máquina en la que reside o en otras en las que confía

- Ingeniería social

Aprovecharse de la ingenuidad de los usuarios y/o personal relacionado con ellos para obtener sus credenciales
Visitar el entorno del usuario

2. Medidas de Protección

2.1 Impedir ataques automáticos (enumeración usuarios, password cracking, etc.)

Solución 1: Utilizar one-time-logins y/o one-time-passwords

Ejemplo:

Supongamos que un usuario tiene como password: j2as!OP.4w
En lugar de solicitar el password, la aplicación podría preguntar:
¿1º, 4º, 2º, 6º, 3º, 8º, 9º y 10º caracter del password?
Y hacer que la posición que solicita fuera aleatoria

2. Medidas de Protección

2.1 Impedir ataques automáticos (enumeración usuarios, password cracking, etc.)

Solución 2: Utilizar CAPTCHA

Programa que puede generar tests que:

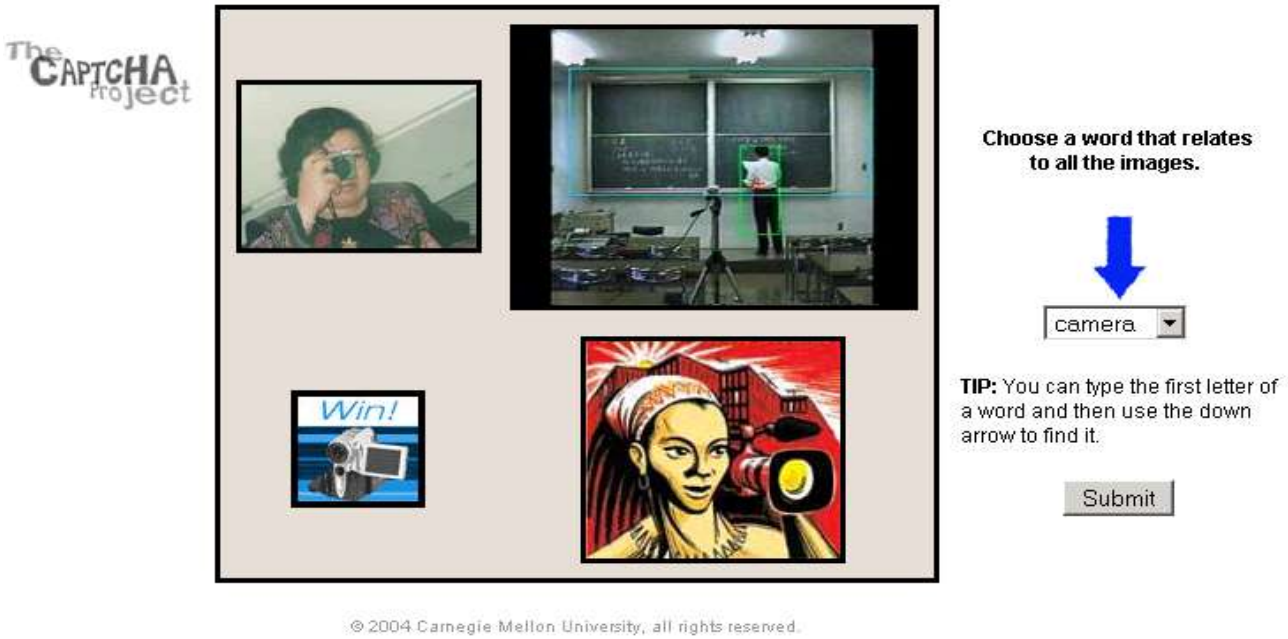
- los humanos pueden pasar
- las máquinas no pueden pasar

Nota: en la Universidad de California (en Berkeley) han desarrollado un programa para romper captcha-gimpy con un 83% de aciertos. Un grupo de Cambridge ha logrado alcanzar el 93%.

2. Medidas de Protección

2.1 Impedir ataques automáticos (enumeración usuarios, password cracking, etc.)

Solución 2: Utilizar CAPTCHA-pix



2. Medidas de Protección

2.1 Impedir ataques automáticos (enumeración usuarios, password cracking, etc.)

Solución 2: Utilizar CAPTCHA-text

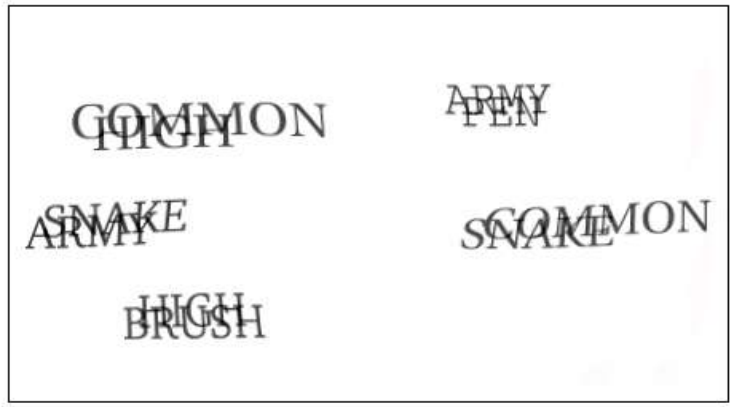
Type a word or string of characters appearing in the picture.

Enter

2. Medidas de Protección

2.1 Impedir ataques automáticos (enumeración usuarios, password cracking, etc.)

Solución 2: Utilizar CAPTCHA-gimpy



In the spaces below, type three (3) different English words appearing in the picture above.

Submit

Click Reload or Refresh in your browser to get new images.

2. Medidas de Protección

2.1 Impedir ataques automáticos
(enumeración usuarios, password cracking, etc.)

Solución 3: Bloqueo de cuenta tras un nº de intentos fallidos

Por ejemplo, tras 5 intentos fallidos bloquear la cuenta.

2. Medidas de Protección

2.2 Implementación sistema "¿Olvidó su contraseña?"

Recomendaciones:

- Siempre se ha de resetear la contraseña

En caso de que alguién obtuviera nuestra contraseña empleando este proceso lo detectaríamos rápidamente. No podríamos entrar!

- Si se envían correos con datos sensibles, han de caducar

Por ejemplo, en 24h. El objetivo es evitar que esos datos puedan ser utilizados por alguien que tuviera acceso a nuestro correo. A ser posible, los e-mails deberían enviarse encriptados.

- Si se opta por las preguntas y respuestas

Utilizar varias preguntas de seguridad. No han de ser predecibles y debe definirlas la aplicación. No dejar la responsabilidad de seleccionar las preguntas en manos del usuario.

2. Medidas de Protección

2.2 Implementación sistema “¿Olvidó su contraseña?”

Recomendaciones:

- Enviar la contraseña por vía telefónica (voz, SMS)

(caso NETCODE en ASB Bank)

- Utilizar métodos híbridos

Por ejemplo:
combinar preguntas/respuestas con llamada telefónica.

2. Medidas de Protección

2.3 Recomendaciones genéricas

Utilizar HTTPs para acceder al formulario de autenticación y durante el envío de datos sensibles.

Si se utilizan cookies, utilizar la variable SECURE para asegurar que sólo viajaran por un canal encriptado.

No permitir que los mensajes de la aplicación posibiliten deducir información (a través de mensajes de error o mensajes “controlados”)

No permitir contraseñas débiles en el sistema

2. Medidas de Protección

2.3 Recomendaciones genéricas

No reutilizar cuentas en distintos servicios

Realizar validaciones de todos los datos de E/S. El filtro debe ser en negativo “Por defecto rechazo todo. Sólo acepto caracteres dentro del rango ...”

Eliminar del servidor web todos aquellos recursos que no sean estrictamente necesarios (manuales, ficheros de ejemplo, etc.) y no realizar instalaciones “por defecto”.

3. Referencias

CAPTCHA - www.captcha.net
ESP Game - www.espgame.org
PASSMARK - www.passmarksecurity.com
NETCODE - www.asbbank.co.nz/netcode/
Sesiones - www.acros.si/papers/session_fixation.pdf
OWASP - www.owasp.org

Dudas, preguntas, comentarios, ...

?