OWASP TOP 10

RIESGOS DE SEGURIDAD EN APLICACIONES WEB

- Principios del diseño de software seguro
- Conceptos generales sobre el desarrollo de aplicaciones web
- OWASP Top 10, CWE y SANS Top 20
- Revisión de Guía de Desarrollo y Análisis de Vulnerabilidad del Top 10 de OWASP

VULNERABILIDADES

¿Qué son las vulnerabilidades?
¿Cómo encontrarlas?
¿Cómo arreglarlas?

CVE define una vulnerabilidad como:

"Una debilidad en la lógica computacional (por ejemplo, código) que se encuentra en los componentes de software y hardware que, cuando se explota, tiene un impacto negativo en la confidencialidad, integridad o disponibilidad. La mitigación de las vulnerabilidades en este contexto típicamente implica cambios de codificación, pero también podría incluir cambios de especificaciones o incluso depreciaciones de especificación (por ejemplo, eliminación de protocolos o la funcionalidad afectados en su totalidad) ".

https://cve.mitre.org/about/terminology.html

Common Vulnerabilities and Exposures

The Standard for Information Security Vulnerability Names

Estado de la seguridad

Vulnerabilities distribution by risk Threat rank (WASC)

Threat rank	N of Vulns	N of Sites	Vulns%	% Sites
Urgent	1353	75	4.87%	50.00%
Critical	12599	138	45.37%	92.00%
High	13673	93	49.23%	62.00%
Medium	139	36	0.50%	24.00%
Low	8	6	0.03%	4.00%

92% de las vulnerabilidades están en las aplicaciones, no en las redes. NIST

NIST
National Institute of Standards and Technology
U.S. Department of Commerce

MITRE

NIST National Institute of Standards and Technology U.S. Department of Commerce

CWE: Common weakness enumeration

Sirve como un lenguaje común, una vara de medir para herramientas de seguridad de software, y como una línea de base para la identificación de debilidades, la mitigación y los esfuerzos de prevención.

CWE/SANS Top 25
es una lista de los errores de programación más extendidos y críticos que pueden conducir a vulnerabilidades graves en el software.

+ de 1000 errores !!!

https://cwe.mitre.org/top25/

DESARROLLO SEGURO - OWASP TOP 10

Brief Listing of the Top 25

This is a brief listing of the Top 25 items, using the general ranking.

NOTE: 16 other weaknesses were considered for inclusion in the Top 25, but their general scores were not high enough. They are listed in a separate "On the Cusp" page.

Rank	Score	ID	Name
[1]	93.8	CWE-89	Improper Neutralization of Special Elements used in an SQL Command ('SQL Injection')
[2]	83.3	CWE-78	Improper Neutralization of Special Elements used in an OS Command ('OS Command Injection')
[3]	79.0	CWE-120	Buffer Copy without Checking Size of Input ('Classic Buffer Overflow')
[4]	77.7	CWE-79	Improper Neutralization of Input During Web Page Generation ('Cross-site Scripting')
[5]	76.9	CWE-306	Missing Authentication for Critical Function
[6]	76.8	CWE-862	Missing Authorization
[7]	75.0	CWE-798	Use of Hard-coded Credentials
[8]	75.0	CWE-311	Missing Encryption of Sensitive Data
[9]	74.0	CWE-434	Unrestricted Upload of File with Dangerous Type
[10]	73.8	CWE-807	Reliance on Untrusted Inputs in a Security Decision
[11]	73.1	CWE-250	Execution with Unnecessary Privileges
[12]	70.1	CWE-352	Cross-Site Request Forgery (CSRF)
[13]	69.3	CWE-22	Improper Limitation of a Pathname to a Restricted Directory ('Path Traversal')
[14]	68.5	CWE-494	Download of Code Without Integrity Check
[15]	67.8	CWE-863	Incorrect Authorization
[16]	66.0	CWE-829	Inclusion of Functionality from Untrusted Control Sphere
[17]	65.5	CWE-732	Incorrect Permission Assignment for Critical Resource
[18]	64.6	CWE-676	Use of Potentially Dangerous Function
[19]	64.1	CWE-327	Use of a Broken or Risky Cryptographic Algorithm
[20]	62.4	CWE-131	Incorrect Calculation of Buffer Size
[21]	61.5	CWE-307	Improper Restriction of Excessive Authentication Attempts

Common Weakness Enumeration

A Community-Developed List of Software Weakness Types

2011 CWE/SANS Top 25: Monster Mitigations

These mitigations will be effective in eliminating or reducing the severity of the Top 25. These mitigations will also a
even on the Top 25. If you adopt these mitigations, you are well on your way to making more secure software.

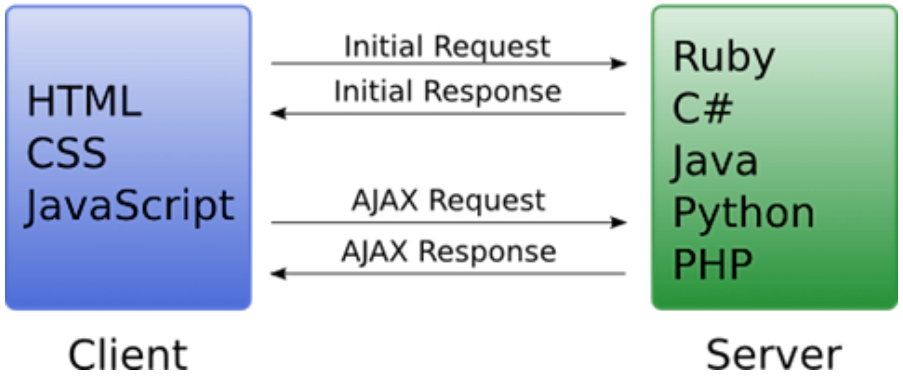
Monster Mitigation Index

ID	Name
M1	Establish and maintain control over all of your inputs.
M2	Establish and maintain control over all of your outputs.
M3	Lock down your environment.
M4	Assume that external components can be subverted, and your code can be read by anyone.
M5	Use industry-accepted security features instead of inventing your own.
GP1	*(general)* Use libraries and frameworks that make it easier to avoid introducing weaknesses.
GP2	*(general)* Integrate security into the entire software development lifecycle.
GP3	*(general)* Use a broad mix of methods to comprehensively find and prevent weaknesses.
GP4	*(general)* Allow locked-down clients to interact with your software.

DESARROLLO SEGURO – OWASP TOP 10

OWASP
Open Web Application Security Project

- Organización orientada a educar, capacitar a desarrolladores sobre las consecuencias de las vulnerabilidades en las aplicaciones web.
- Provee información sobre como evaluar los riesgos.



http://www.owasp.org

OWASP

Es seguro | https://www.owasp.org/index.php/Main_Page

Log in Request account

Search

Welcome to OWASP

the free and open software security community

- OWASP 2017 World Tour - Boston
- Dependency Check
- Proactive Controls
- ZAP Proxy
- Cheat Sheets
- Top 10
- OWTF
- ASVS
- SAMM
- Development Guide
- AppSensor
- Testing Guide
- ModSecurity Ruleset
- More...

About · Searching · Editing · New Article · OWASP Categories . CONTACT-US

Statistics · Recent Changes

Every vibrant technology marketplace needs an unbiased source of information on best practices as well as an active body advocating open standards. In the Application Security space, one of those groups is the Open Web Application Security Project (or OWASP for short).

The Open Web Application Security Project (OWASP) is a 501(c)(3) worldwide not-for-profit charitable organization focused on improving the security of software. Our mission is to make software security visible, so that individuals and organizations are able to make informed decisions. OWASP is in a unique position to provide impartial, practical information about AppSec to individuals, corporations, universities, government agencies and other organizations worldwide. Operating as a community of like-minded professionals, OWASP issues software tools and knowledge-based documentation on application security.

Everyone is free to participate in OWASP and all of our materials are available under a free and open software license. You'll find everything about OWASP here on or linked from our wiki and current information on our OWASP Blog. OWASP **does not endorse or recommend commercial products or services**, allowing our community to remain vendor neutral with the collective wisdom of the best minds in software security worldwide.

Citations

Who Trusts OWASP?

Citations of National & International Legislation, Standards, Guidelines, Committees and Industry Codes of Practice - Click Here

How can OWASP help your org?

Government

Guía de Seguridad en Aplicaciones para CISOs

Home
About OWASP
Acknowledgements
Advertising
AppSec Events
Books
Brand Resources
Chapters
Donate to OWASP
Downloads
Funding
Governance
Initiatives
Mailing Lists
Membership
Merchandise
News
Community portal
Presentations
Press
Projects
Video
Volunteer

Reference
Activities
Attacks
Code Snippets
Controls

Guía de Seguridad en Aplicaciones para CISOs

Application Security Guide For CISOs Version 1.0 (English) was published in November 2013.

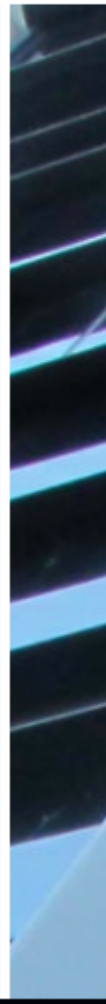
La Guía de Seguridad en Aplicaciones para CISOs versión 1.0 (Español) fue publicada en marzo de 2015.

Objetivos

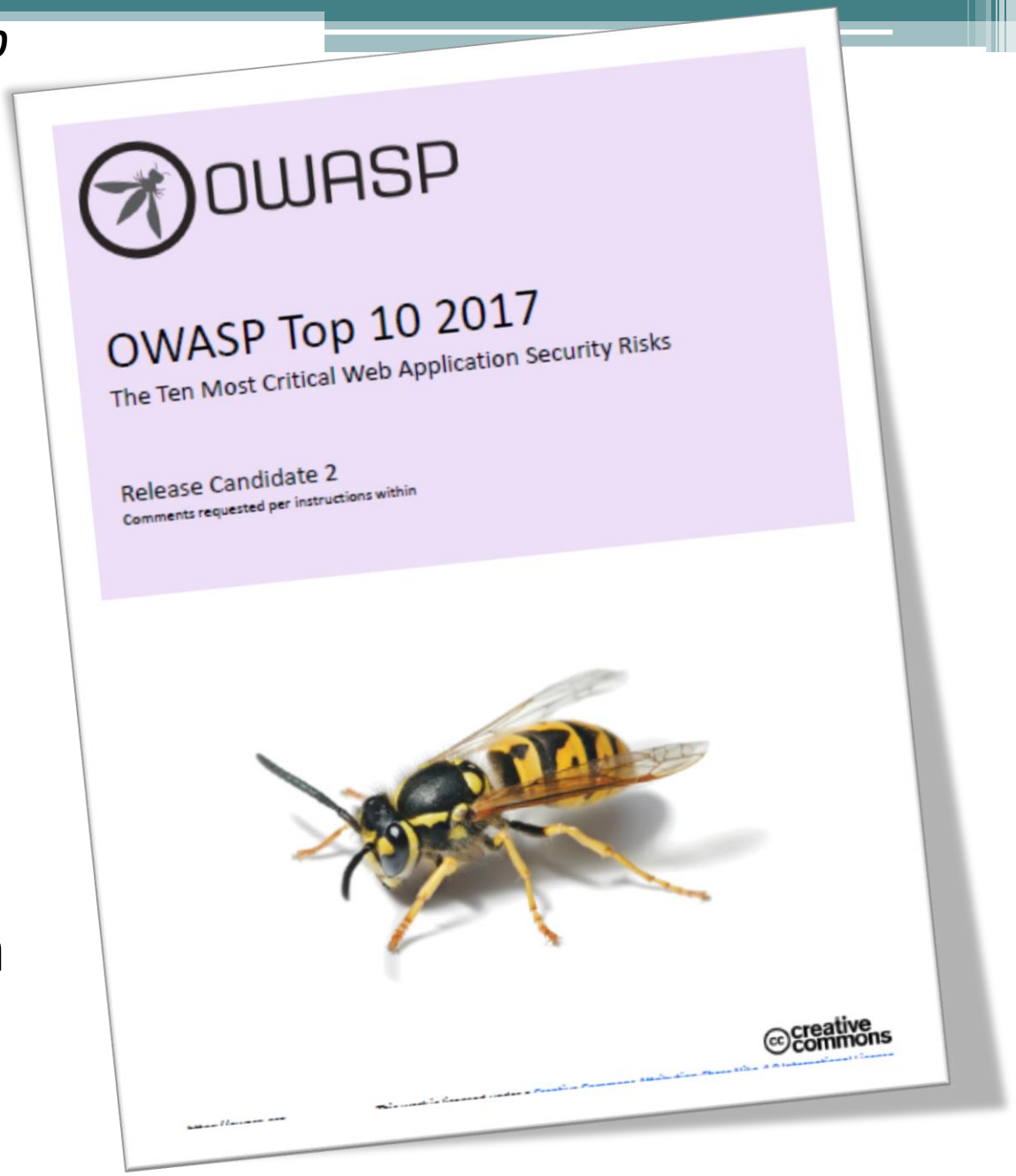
La guía ayuda a los CISOs a gestionar los riesgos de seguridad en las aplicaciones, teniendo en cuenta la exposición de las amenazas emergentes y los requisitos de cumplimiento. Sus objetivos son:

- Hacer visible a los CISOs la seguridad en las aplicaciones
- Asegurar la conformidad de las aplicaciones con normas de seguridad, privacidad y protección de datos
- Priorizar la corrección de las vulnerabilidades basándose en la exposición al riesgo para el negocio
- Proporcionar orientación para construir y administrar los procesos de seguridad de la aplicación
- Analizar las ciberamenazas e identificar las contramedidas
- Crear programas de capacitación de seguridad para las aplicaciones, desarrolladores y administradores
- Medir y gestionar los riesgos de seguridad en las aplicaciones y los procesos

Contenido



DESARROLLO SEGURO – OWASP TOP 10



10 vulnerabilidades mas críticas y mas comunes en aplicaciones web en todo el mundo.

DESARROLLO SEGURO - OWASP TOP 10

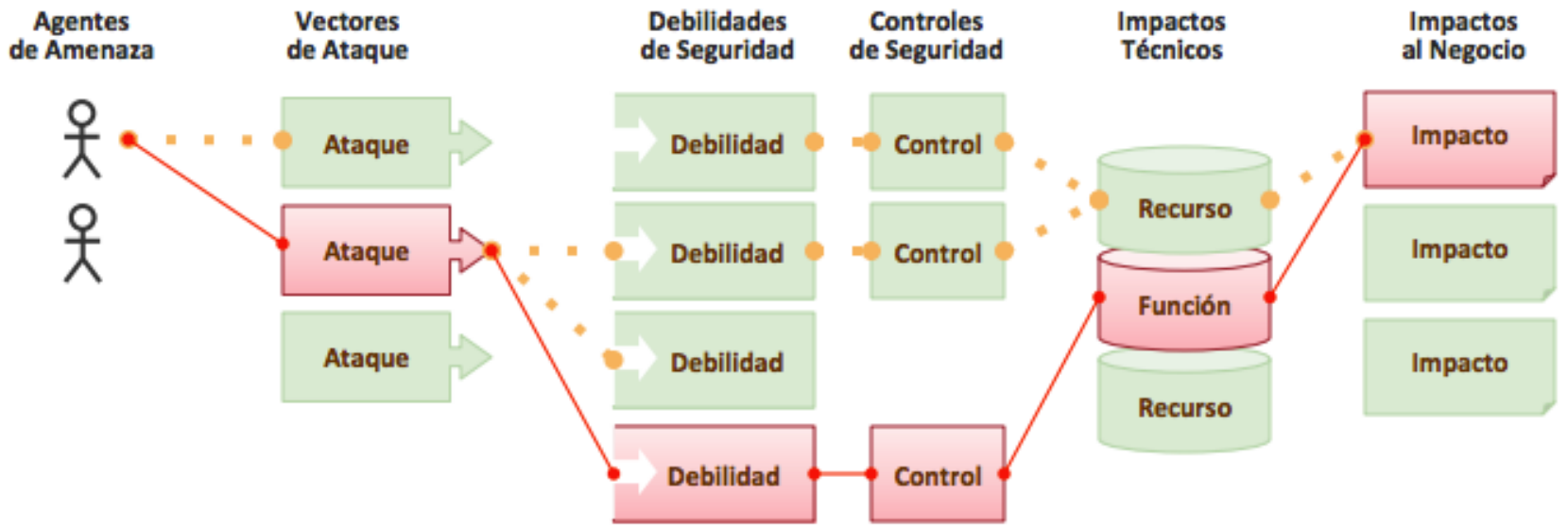
OWASP Top 10 2013	±	OWASP Top 10 2017
A1 – Injection	→	A1:2017 – Injection
A2 – Broken Authentication and Session Management	→	A2:2017 – Broken Authentication and Session Management
A3 – Cross-Site Scripting (XSS)	↘	A3:2013 – Sensitive Data Exposure
A4 – Insecure Direct Object References [Merged+A7]	U	A4:2017 – XML External Entity (XXE) [NEW]
A5 – Security Misconfiguration	↘	A5:2017 – Broken Access Control [Merged]
A6 – Sensitive Data Exposure	↗	A6:2017 – Security Misconfiguration
A7 – Missing Function Level Access Contr [Merged+A4]	U	A7:2017 – Cross-Site Scripting (XSS)
A8 – Cross-Site Request Forgery (CSRF)	☒	A8:2017 – Insecure Deserialization [NEW, Community]
A9 – Using Components with Known Vulnerabilities	→	A9:2017 – Using Components with Known Vulnerabilities
A10 – Unvalidated Redirects and Forwards	☒	A10:2017 – Insufficient Logging & Monitoring [NEW, Comm.]

DESARROLLO SEGURO - OWASP TOP 10

Enfoque de riesgo de Owasp

¿Qué son los riesgos de seguridad en aplicaciones?

Los atacantes pueden potencialmente usar rutas diferentes a través de la aplicación para hacer daño a su negocio u organización. Cada una de estas rutas representa un riesgo que puede, o no, ser lo suficientemente grave como para justificar la atención.



Owasp Risk Rating methodology

Threat Agents	Exploitability	Weakness Prevalence	Weakness Detectability	Technical Impacts	Business Impacts
App Specific	Easy	Widespread	Easy	Severe	App / Business Specific
	Average	Common	Average	Moderate	
	Difficult	Uncommon	Difficult	Minor	

DESARROLLO SEGURO - OWASP TOP 10

OWASP TOP 10 (2017)

RISK	Threat Agents	Attack Vectors: Exploitability	Security Weakness: Prevalence	Security Weakness: Detectability	Impacts: Technical	Impacts: Business	Score
A1:2017-Injection	App Specific	EASY 3	COMMON 2	EASY 3	SEVERE 3	App Specific	8.0
A2:2017-Authentication	App Specific	EASY 3	COMMON 2	AVERAGE 2	SEVERE 3	App Specific	7.0
A3:2017-Sens. Data Exposure	App Specific	AVERAGE 2	WIDESPREAD 3	AVERAGE 2	SEVERE 3	App Specific	7.0
A4:2017-XML External Entity (XXE)	App Specific	AVERAGE 2	COMMON 2	EASY 3	SEVERE 3	App Specific	7.0
A5:2017-Broken Access Control	App Specific	AVERAGE 2	COMMON 2	AVERAGE 2	SEVERE 3	App Specific	6.0
A6:2017-Security Misconfiguration	App Specific	EASY 3	WIDESPREAD 3	EASY 3	MODERATE 2	App Specific	6.0
A7:2017-Cross-Site Scripting (XSS)	App Specific	EASY 3	WIDESPREAD 3	EASY 3	MODERATE 2	App Specific	6.0
A8:2017-Insecure Deserialization	App Specific	DIFFICULT 1	COMMON 2	AVERAGE 2	SEVERE 3	App Specific	5.0
A9:2017-Vulnerable Components	App Specific	AVERAGE 2	WIDESPREAD 3	AVERAGE 2	MODERATE 2	App Specific	4.7
A10:2017-Insufficient Logging&Monitoring	App Specific	AVERAGE 2	WIDESPREAD 3	DIFFICULT 1	MODERATE 2	App Specific	4.0

Errores transversales al TOP 10

Mala validación de parámetros

- No se validan los input y output de variables.
- No se inicializan las variables.
- No se validan los tipos de datos.

Se pueden modificar las entradas

Mala validación de parámetros

```
Private Sub Button1_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles Button1.Click
    Timer1.Interval = TextBox2.Text * 1000
    Timer1.Start()
End Sub
```

```
function greet(username) {
    return "Hello, " + username;
}
var user = “World";
document.body.innerHTML = greet(user);
```

```
<table border="0" cellspacing="4">
<?php
include ("conect.php");
if (isset($_GET['del'])) {
    mysql_query("DELETE from articulos WHERE id=$_GET[del]",$conexion);
}
$ids = mysql_query("SELECT id,menu,titulo FROM articulos",$conexion);
while ($ide = mysql_fetch_array($ids)) {
    echo "<tr>";
    echo "<td>".$ide['id']."</td>";
    echo "<td>".$ide['menu']."</td>";
    echo "<td>".$ide['titulo']."</td>";
    echo "<td>"."<a href='editar.php?ed=$ide[id]'>Editar</a>"."</td>";
    echo "<td>"."<a href='listedit.php?del=$ide[id]'>Eliminar</a>"."</td>
    echo "</tr>";
}
?>
</table>
```

```
<?php
...
$sql = "SELECT * FROM forum_logs WHERE id =
$_GET["id"];
$result = mysql_query($sql);
...
?>
```

Mala validación de parámetros

- Como solucionarlo ???

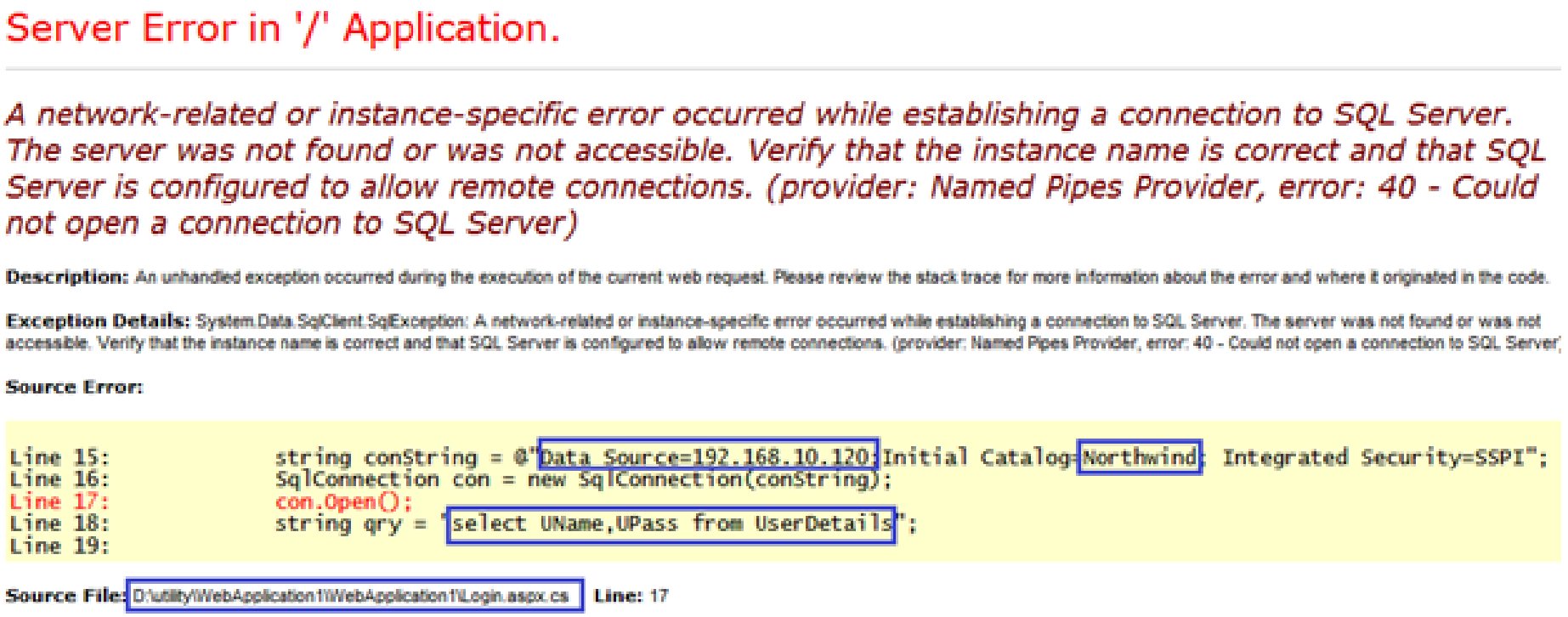
REALIZAR TODAS LAS VALIDACIONES DEL LADO DEL CLIENTE Y DEL LADO DEL SERVIDOR

Information disclosure

- Divulgar información a usuarios no autorizados o brindar información excesiva

Puede aprovechar la información para hacer daño

Information disclosure



Server Error in '/' Application.

A network-related or instance-specific error occurred while establishing a connection to SQL Server. The server was not found or was not accessible. Verify that the instance name is correct and that SQL Server is configured to allow remote connections. (provider: Named Pipes Provider, error: 40 - Could not open a connection to SQL Server)

Description: An unhandled exception occurred during the execution of the current web request. Please review the stack trace for more information about the error and where it originated in the code.

Exception Details: System.Data.SqlClient.SqlException: A network-related or instance-specific error occurred while establishing a connection to SQL Server. The server was not found or was not accessible. Verify that the instance name is correct and that SQL Server is configured to allow remote connections. (provider: Named Pipes Provider, error: 40 - Could not open a connection to SQL Server)

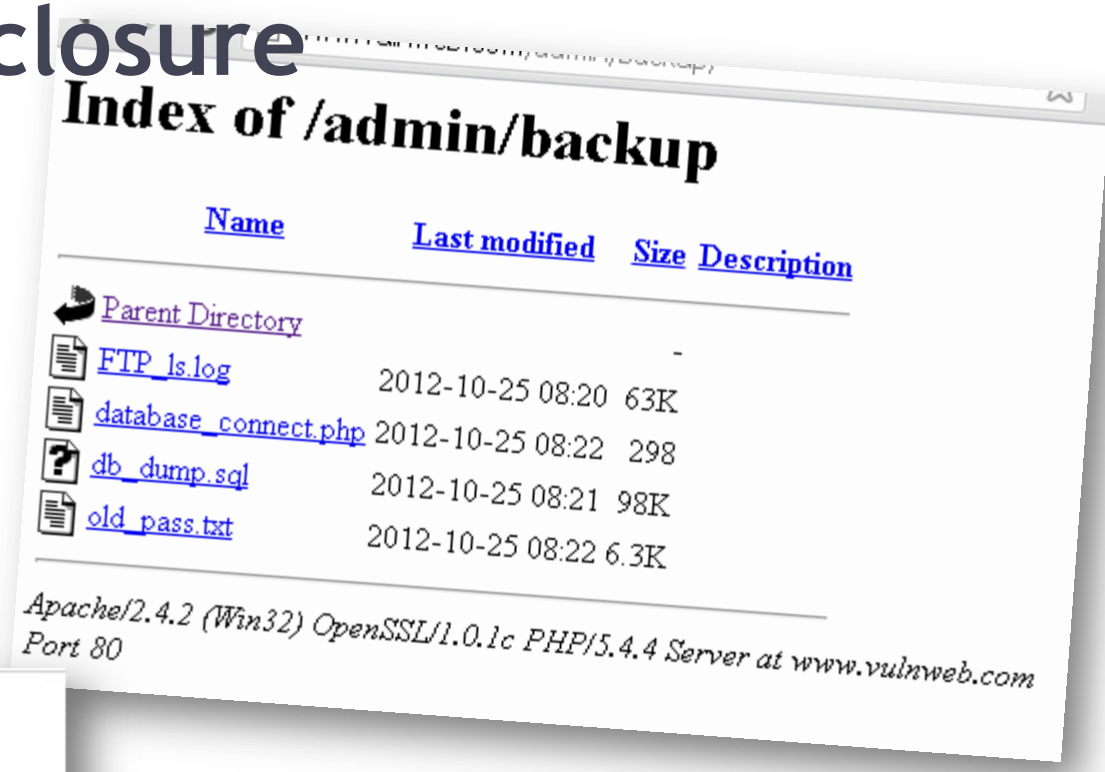
Source Error:

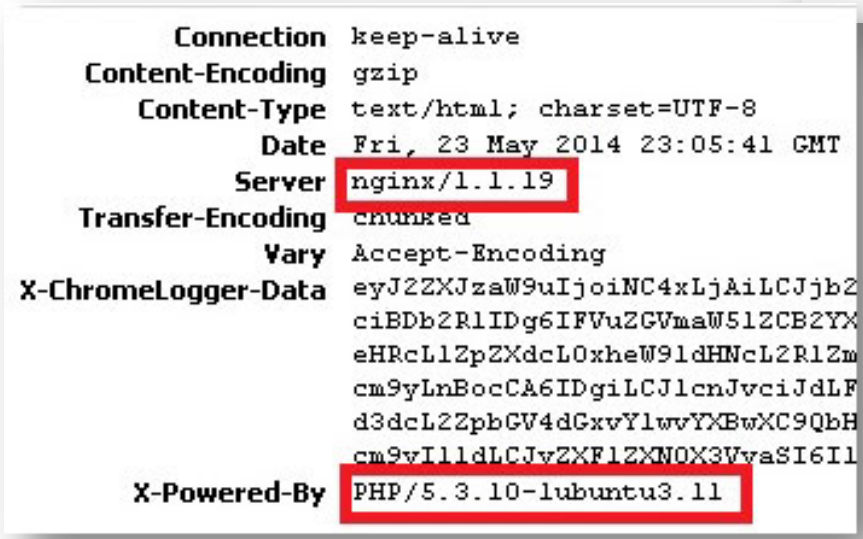
```
Line 15:            string conString = @"Data Source=192.168.10.120;Initial Catalog=Northwind; Integrated Security=SSPI";
Line 16:            SqlConnection con = new SqlConnection(conString);
Line 17:            con.Open();
Line 18:            string qry = "select UName,UPass from UserDetails";
Line 19:
```

Source File: D:\utility\WebApplication1\WebApplication1\Login.aspx.cs **Line:** 17

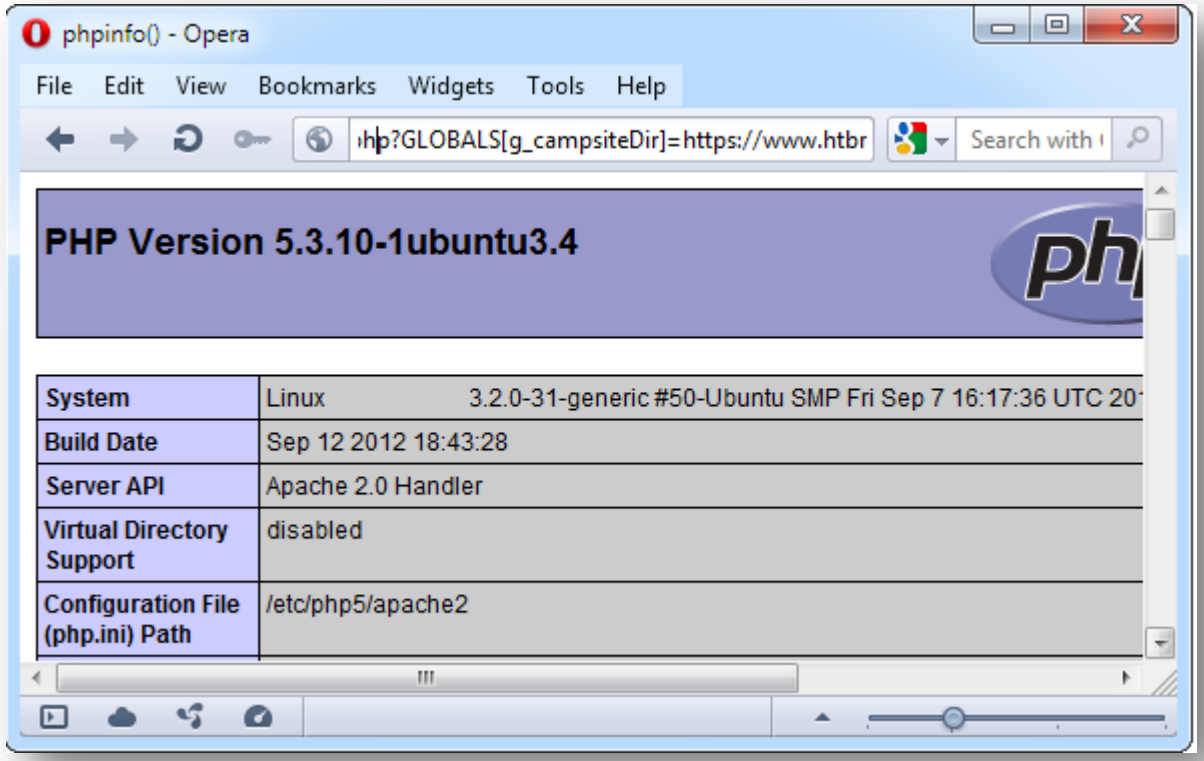
DESARROLLO SEGURO - OWASP TOP 10

Information disclosure

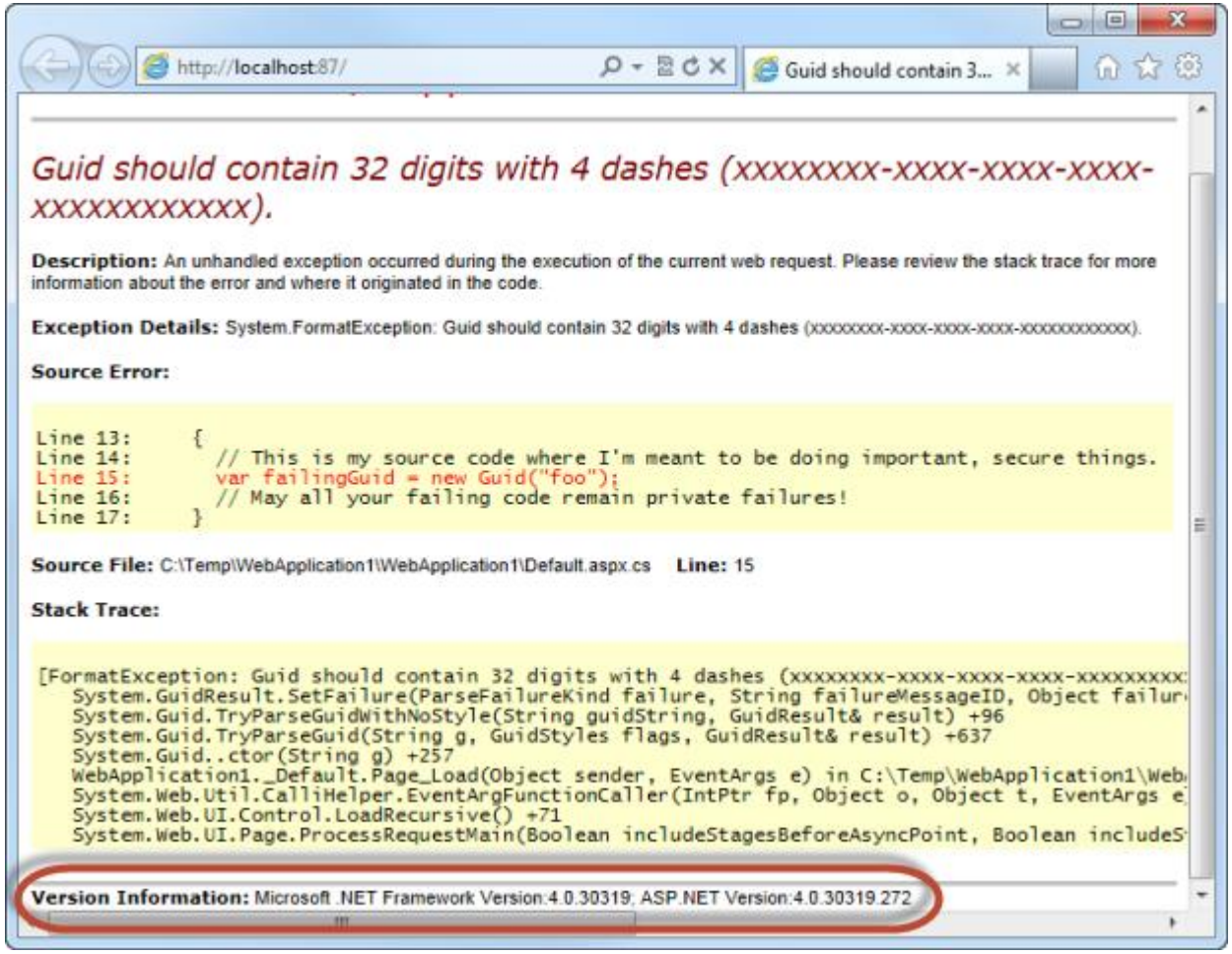




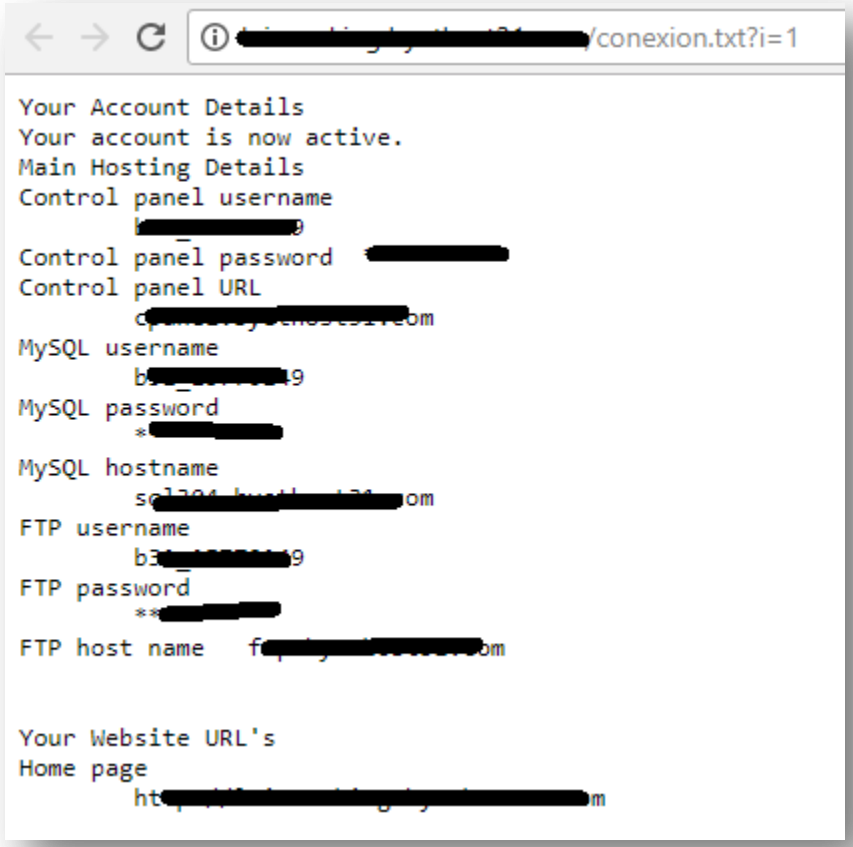
Information disclosure



Information disclosure



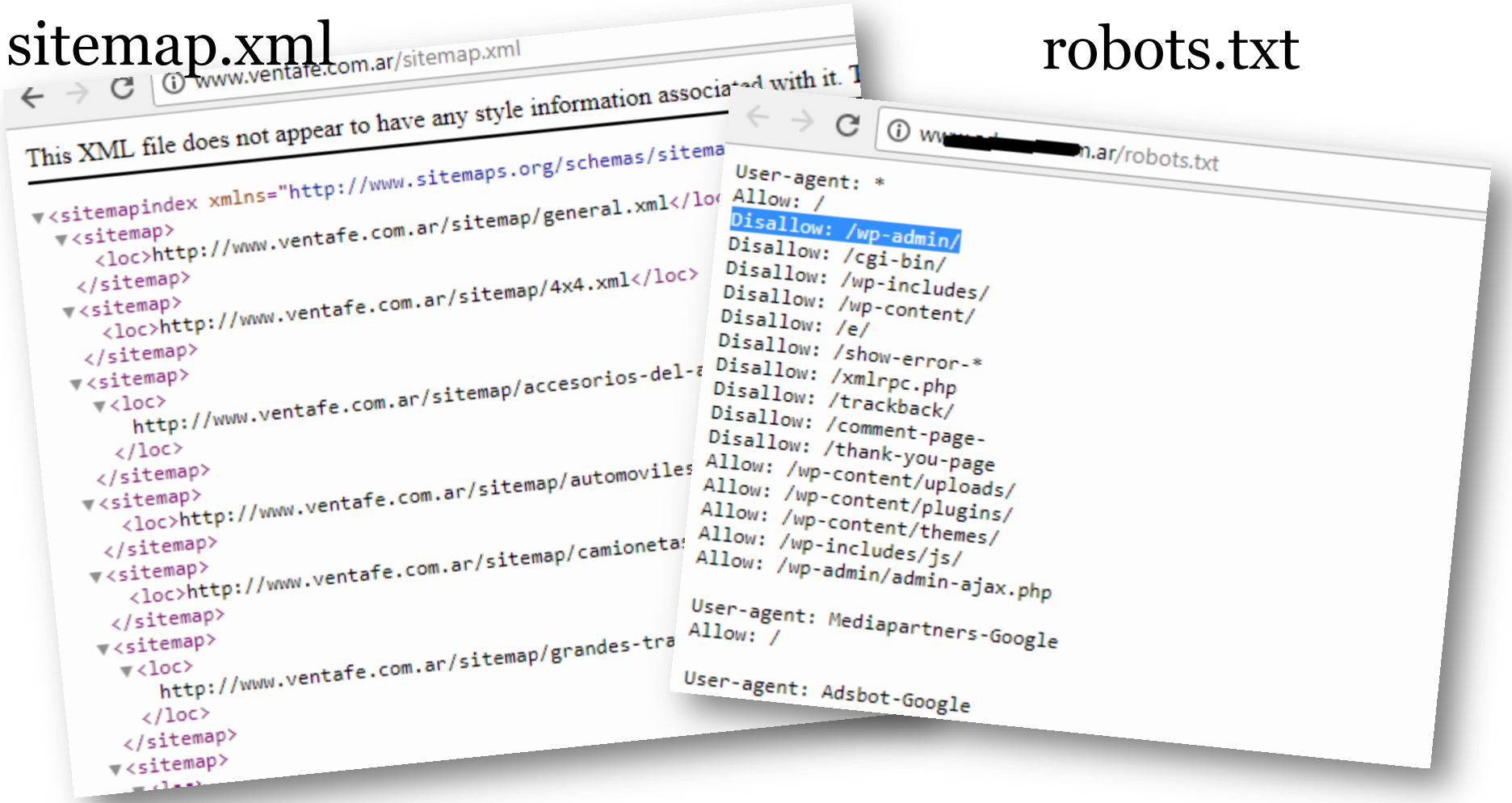
Information disclosure



Archivos con información

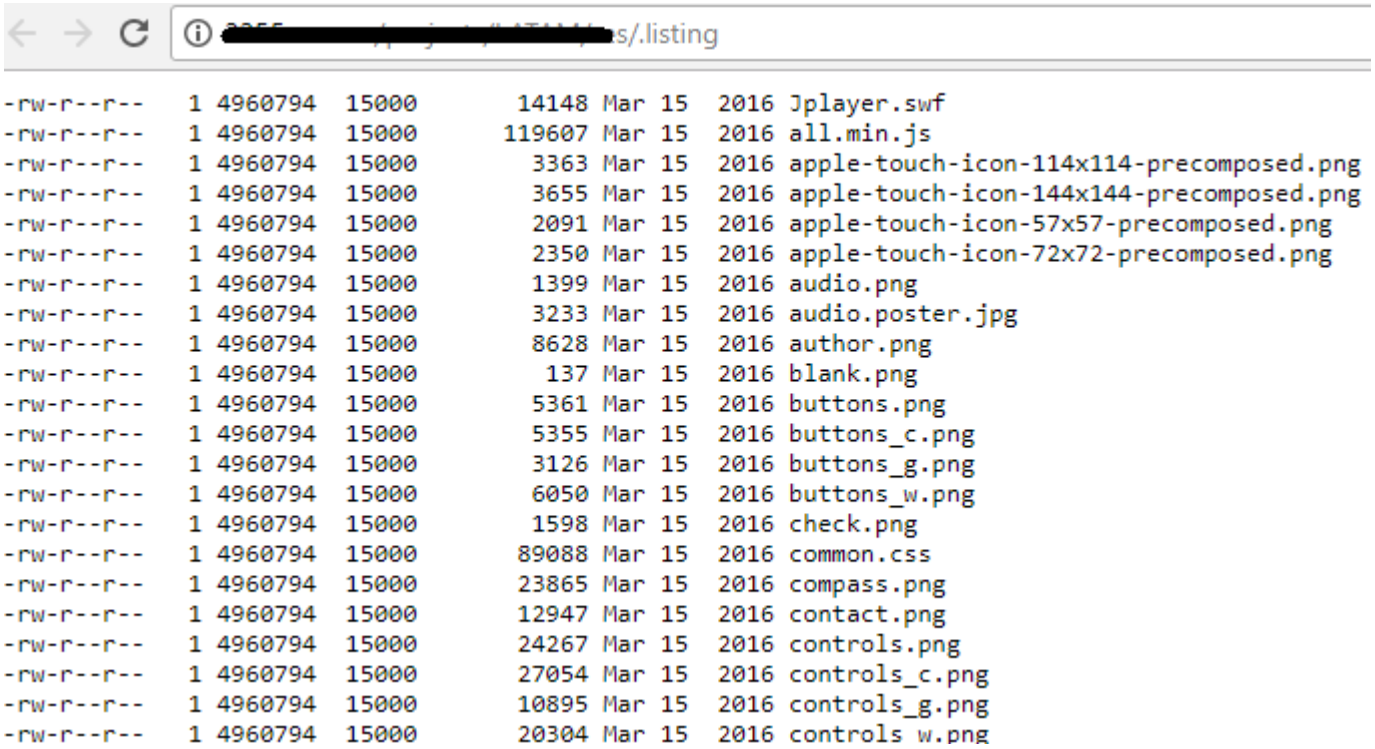
sitemap.xml

robots.txt



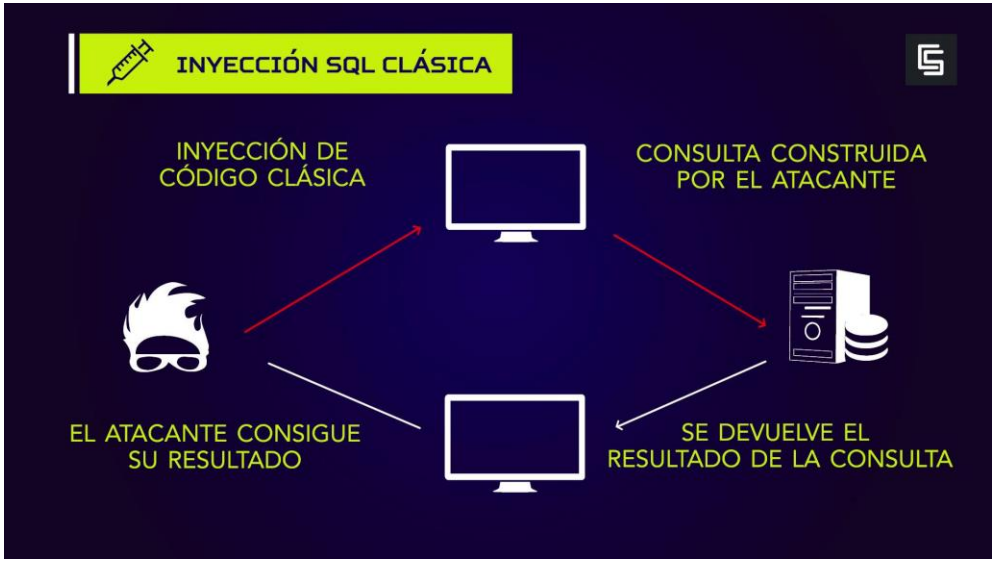
Archivos con información

- .listing

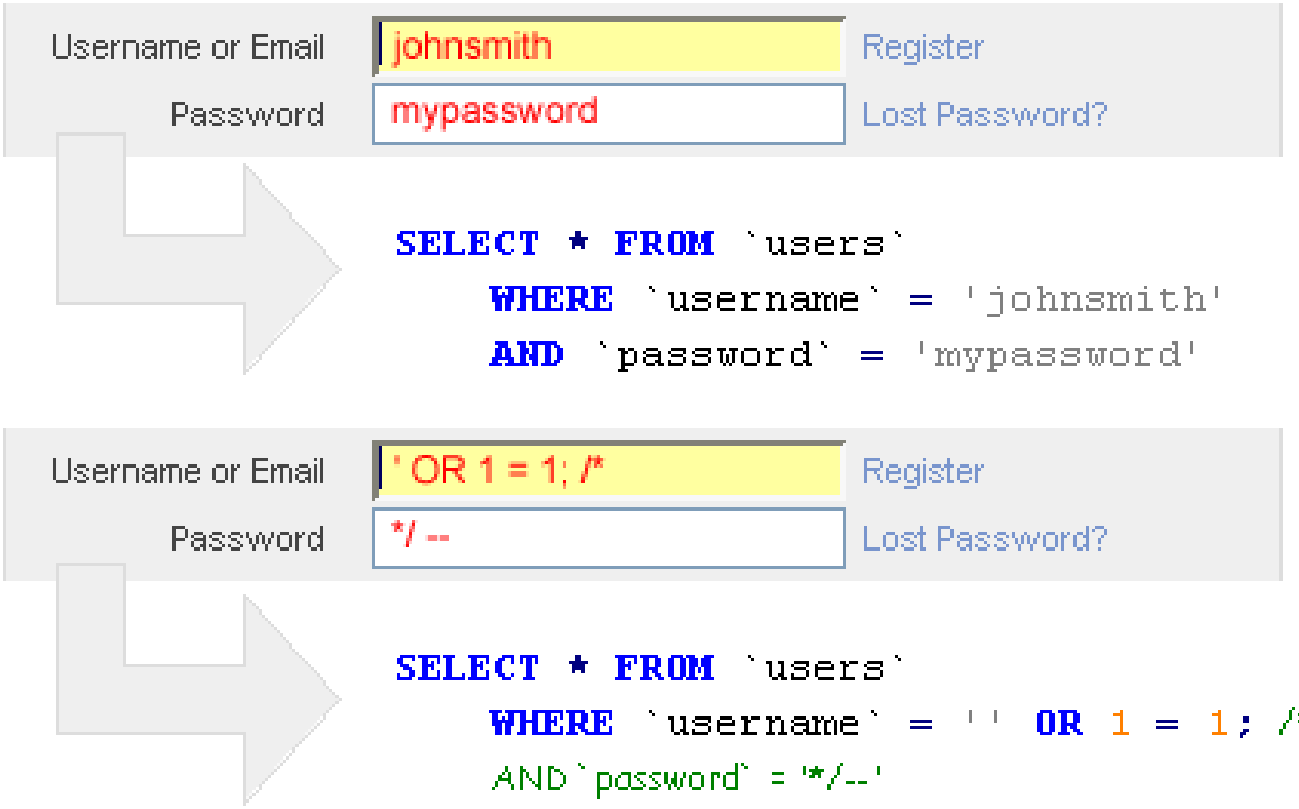


A1. Injección SQL

- Método de infiltración de código intruso que se vale de una vulnerabilidad presente en una aplicación en el nivel de la validación de las entradas.



A1. Injección SQL

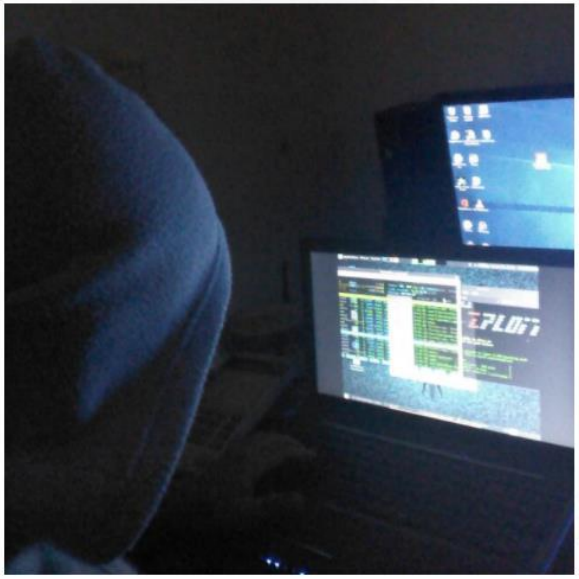


- Pueden ser de SQL, LDAP, Xpath, comandos de SO, parámetros

A1. Injección SQL

A1. Injección SQL

Demostración de inyección…



A1. Injección SQL

- **Prevención:**
- usar una API segura que evite el uso del intérprete por completo o proporciona una interfaz parametrizada, o migrar para usar ORMs o Frameworks.
- Validar las entradas
- Para cualquier consulta dinámica residual, escapar los caracteres especiales usando la sintaxis de escape específica para ese intérprete.
- Usar LIMIT y otros controles SQL dentro de las consultas para prevenir divulgación masiva de registros en caso de inyección SQL.

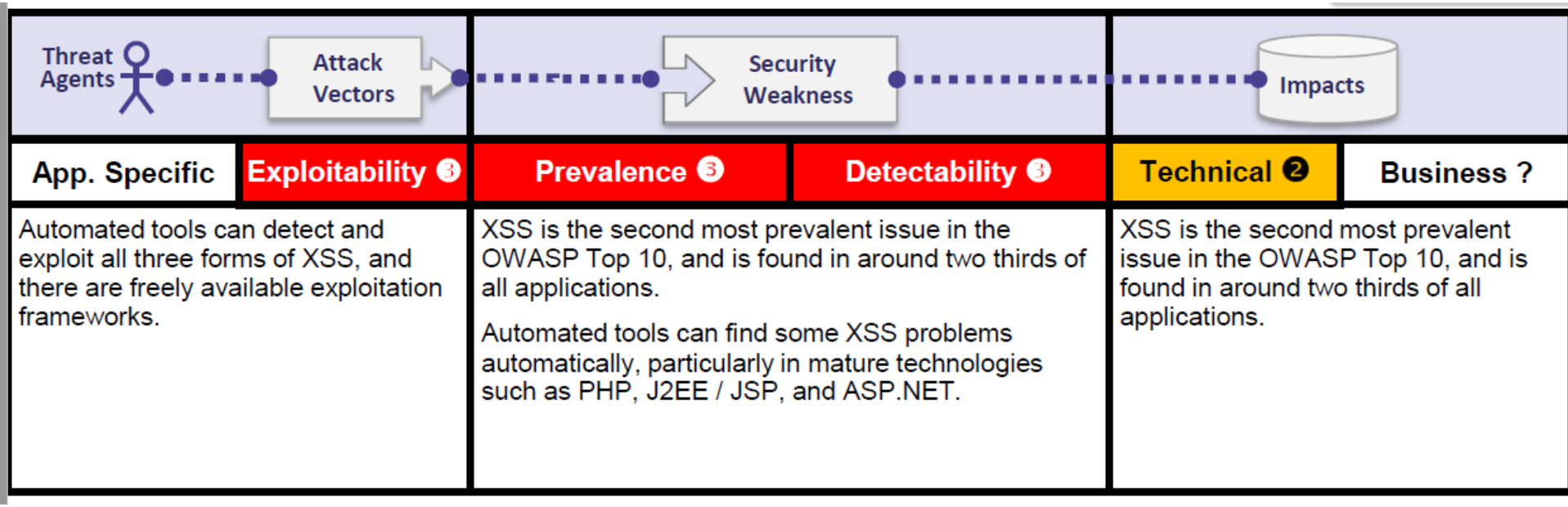
A1. Injección SQL

- **Prevención:**
- Utilizar usuarios personalizados con privilegios muy limitados.
- Emplear sentencias preparadas con variables vinculadas. Son proporcionadas por PDO, MySQLi y otras bibliotecas.
- Comprobar si la entrada proporcionada tiene el tipo de datos previsto.

XSS Cross-site scripting (A7)

Secuencias de comandos en sitios cruzados

- Vulnerabilidad que permite tomar el control del navegador del usuario



XSS Cross-site scripting (A7)

Secuencias de comandos en sitios cruzados

- ¿Qué permite?
  - Robo de identidades y sesiones
  - Manipulación de cookies
  - Modificación total o parcial de un sitio web
  - Redirección a sitios “dañinos”

  - Tipos de XSS
    - Reflejados
    - Almacenados

XSS Cross-site scripting (A7)

Secuencias de comandos en sitios cruzados

- Veamos un ejemplo…

```
</pre></font>
<form action="http://misitio.com/recibir.php" method="post">
    <p>Nombre de usuario:</p>
    <input type="text" name="usuario" value="">
    <p>Contrase&ntilde;a:</p>
    <input type="password" name="clave" value="">
    <p><input type="submit" name="enviar" value="Confirmar datos"></p>
</form>
```

XSS Cross-site scripting (A7)

Secuencias de comandos en sitios cruzados

¿Como prevenir XSS?

- En PHP disponemos de funciones como htmlspecialchars, htmlentities, strip_tags, filter_var, urlencode, htmlencode…
- En NET usar Microsoft Web protection library.
- Utilizar frameworks
- No almacenar passwords en cookies

A2. Autenticación

Broken authentication and sesion management

El atacante utiliza filtraciones o vulnerabilidades en las funciones de autenticación o gestión de las sesiones (ej. cuentas expuestas, contraseñas, identificadores de sesión) para suplantar otros usuarios.

A2. Autenticación

Broken authentication and sesion management

- No haber almacenado de forma correcta los credenciales de los usuarios en la base de datos.
- No poseer una política de contraseñas fuerte.
- Mala gestión de los mecanismos de modificación o recuperación de las contraseñas.
- Visualización de los identificadores de sesión en URL's.

```
http://example.com/sale/saleitems;sessionid=268544541&dest
```

- Mala gestión de la finalización de sesiones.
- Manejo de las contraseñas a través de la red sin cifrar
- Permitir ataques de fuerza bruta

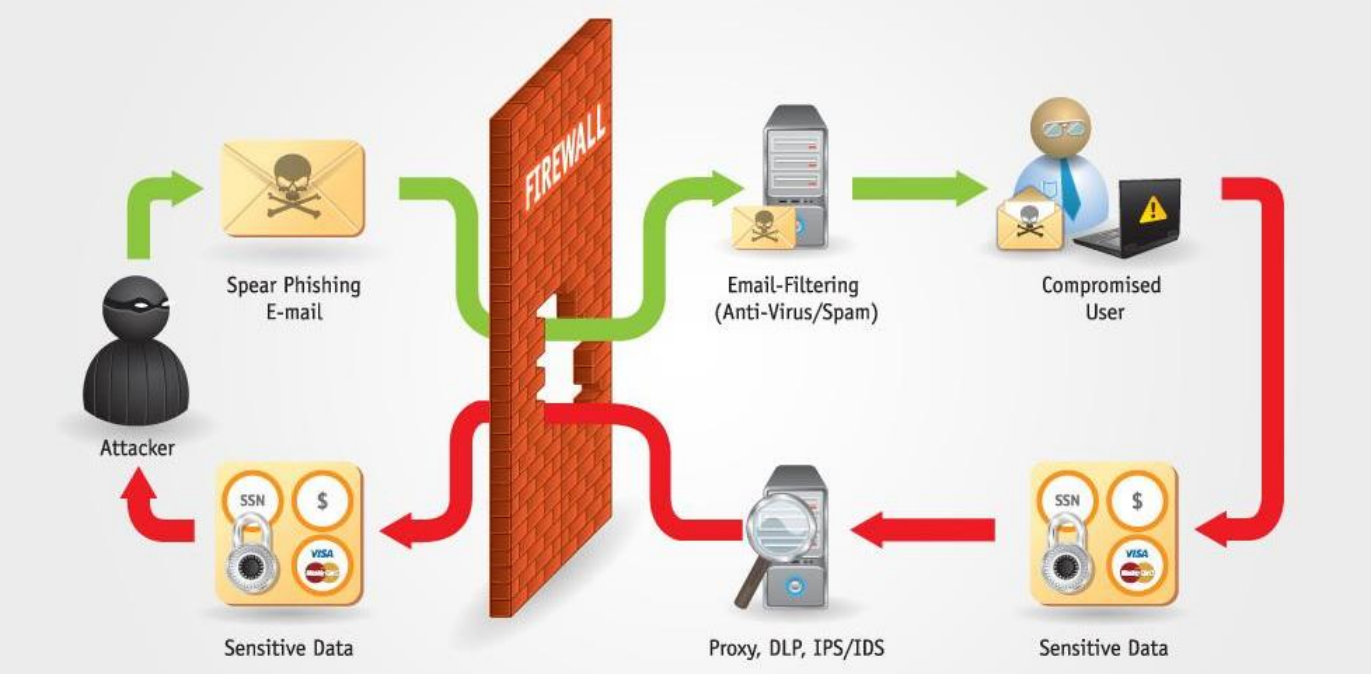
A2. Autenticación

Broken authentication and session management

- **Prevención:**
- No utilizar ningún tipo de credencial predeterminada, tales como “password1” o admin/admin.
- Utilizar una función moderna de hash unidireccional, tal como Argon2 o PBKDF2.
- Implementar comprobaciones de contraseñas débiles, como por ejemplo testear las contraseñas contra una lista del top de las 10.000 peores contraseñas.
- Alinear las políticas de contraseñas de acuerdo a la guía de NIST: 800-63 B's
- Donde sea posible, implementar la autenticación de múltiples
- Registrar logs de errores de autenticación para notificar a los administradores de ataques de fuerza bruta o de otros ataques detectados.

A3. Exposición de datos sensibles

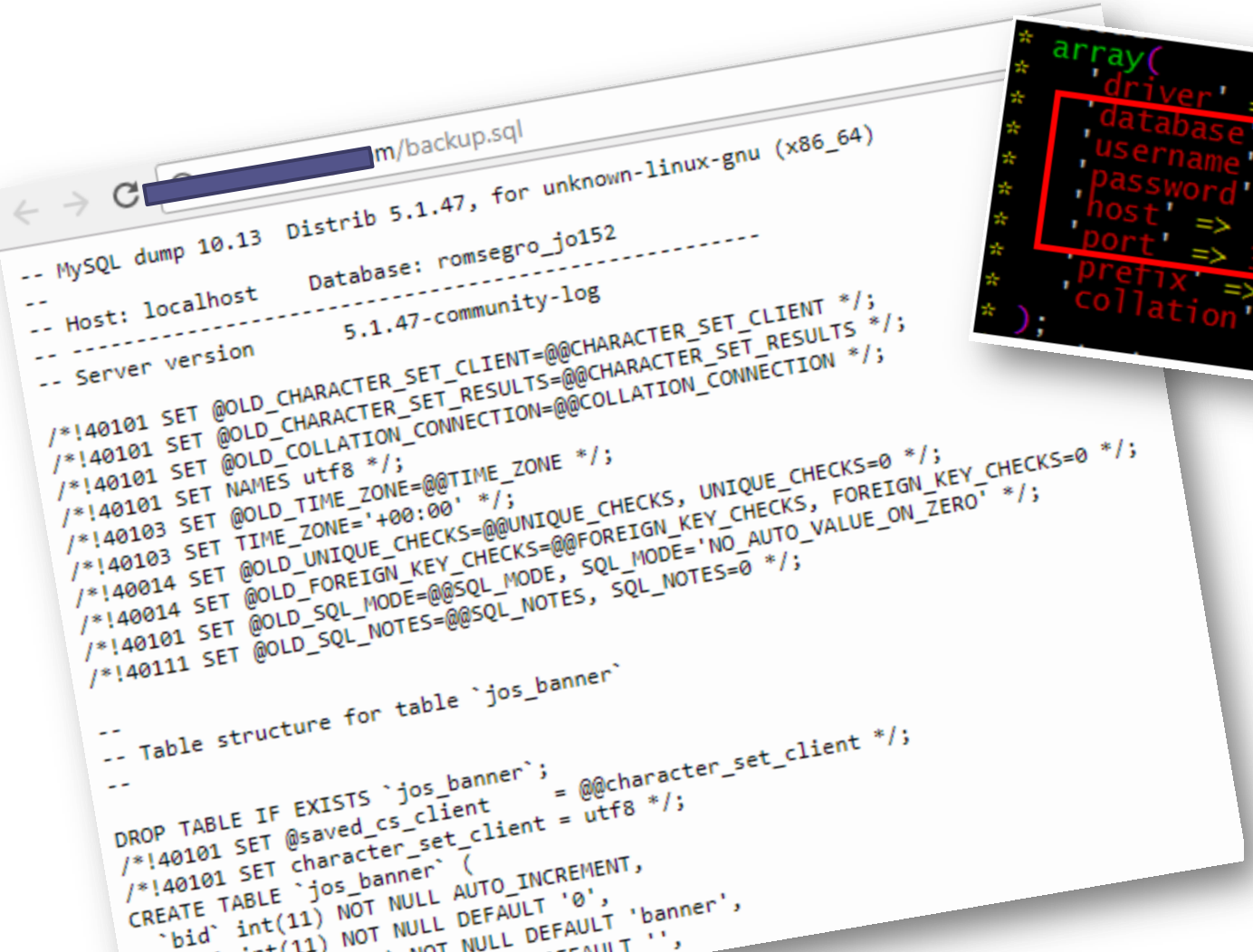
- ¿Soy Vulnerable? Lo primero que debe determinar es el conjunto de datos sensibles que requerirán protección extra. Por ejemplo, contraseñas, números de tarjetas de crédito, registros médicos, e información personal deberían protegerse.



A3. Exposición de datos sensibles

1. ¿Se almacenan en texto claro a largo plazo, incluyendo sus respaldos?
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 cabezales como directivas de seguridad del navegador cuando son enviados o provistos por el mismo?

A3. Exposición de datos sensibles



A3. Exposición de datos sensibles

- **Prevención:**

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. Datos que no se poseen no pueden ser robados.
3. Asegúrese de aplicar algoritmos de cifrado fuertes y estándar.
4. Asegúrese que las claves se almacenan con un algoritmo especialmente diseñado para protegerlas.
5. Deshabilite el autocompletar en los formularios que recolectan datos sensibles. Deshabilite también el cacheado de páginas que contengan datos sensibles.

Hardening de aplicaciones

- Cambiar los directorios de instalación por defecto
- Cambiar configuraciones por defecto.
- Usar contraseñas “fuertes”
- No utilizar protocolos inseguros: FTP, telnet, pop, snmt…
- Configurar directorios y permisos adecuados.
- No brindar información del servidor/aplicación (en errores).
- Cerrar los puertos que no se utilicen.
- Consultas parametrizadas.
- Prevenir la exploración de directorios.
- Validar !!!!

DESARROLLO SEGURO – OWASP TOP 10