# Security and access control

DWES UD5.3

# 1.- Sessions

The **HTTP** protocol does not maintain information on the status of each request and they are treated as independent connections.

To solve this problem, there are different solutions at the application level that allow the server to have more connection data:

- Web browser (cookies)
- Web server (sessions)

In web applications, the two techniques are usually used together.

# 1.- Sessions

Using **cookies**, information can be stored until they expire. Although very useful, the use of cookies entails a series of drawbacks:

- Number of cookies that the browser can store
- Maximum cookie size
- Possible identity theft
- Cookies are stored on the client
- Traffic generated when sending cookies

To solve this, sessions are used on the server side.

# 1.- Sessions

PHP incorporates active session support by default.

Using the **phpinfo()** function you can consult the active configuration on the server regarding sessions.

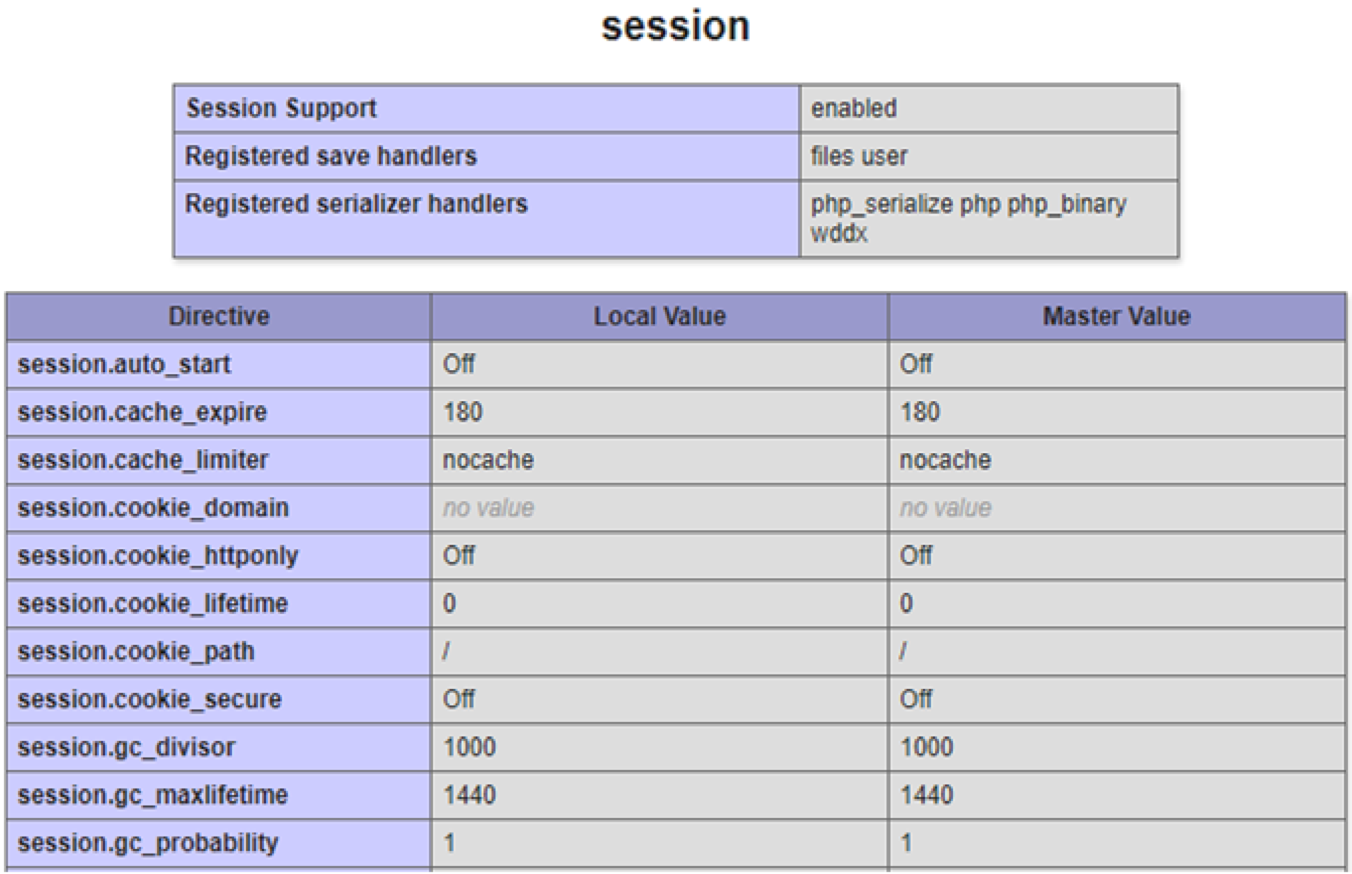
To change this configuration, you can modify the **php.ini** file and restart the web server:

http://php.net/manual/en/session.security.ini.php

If you do not have access to that file, you can change the policies at run time with the **corresponding functions:**

http://php.net/manual/es/session.configuration.php

# 1.- Sessions

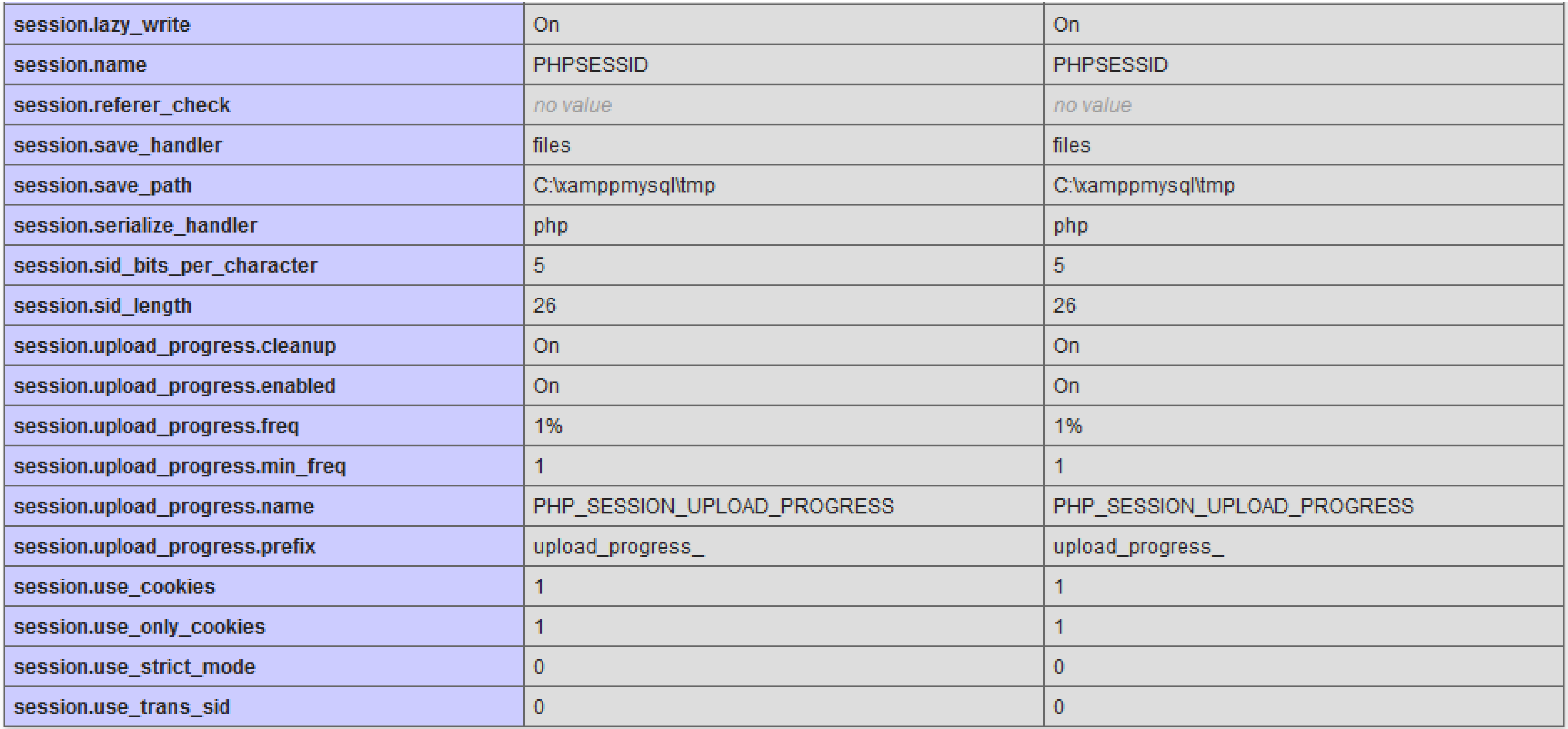


## session

| | |
|---|---|
| Session Support | enabled |
| Registered save handlers | files user |
| Registered serializer handlers | php_serialize php php_binary wddx |

| Directive | Local Value | Master Value |
|---|---|---|
| session.auto_start | Off | Off |
| session.cache_expire | 180 | 180 |
| session.cache_limiter | nocache | nocache |
| session.cookie_domain | *no value* | *no value* |
| session.cookie_httponly | Off | Off |
| session.cookie_lifetime | 0 | 0 |
| session.cookie_path | / | / |
| session.cookie_secure | Off | Off |
| session.gc_divisor | 1000 | 1000 |
| session.gc_maxlifetime | 1440 | 1440 |
| session.gc_probability | 1 | 1 |

# 1.- Sessions



| | | |
|---|---|---|
| session.lazy_write | On | On |
| session.name | PHPSESSID | PHPSESSID |
| session.referer_check | *no value* | *no value* |
| session.save_handler | files | files |
| session.save_path | C:\xamppmysql\tmp | C:\xamppmysql\tmp |
| session.serialize_handler | php | php |
| session.sid_bits_per_character | 5 | 5 |
| session.sid_length | 26 | 26 |
| session.upload_progress.cleanup | On | On |
| session.upload_progress.enabled | On | On |
| session.upload_progress.freq | 1% | 1% |
| session.upload_progress.min_freq | 1 | 1 |
| session.upload_progress.name | PHP_SESSION_UPLOAD_PROGRESS | PHP_SESSION_UPLOAD_PROGRESS |
| session.upload_progress.prefix | upload_progress_ | upload_progress_ |
| session.use_cookies | 1 | 1 |
| session.use_only_cookies | 1 | 1 |
| session.use_strict_mode | 0 | 0 |
| session.use_trans_sid | 0 | 0 |

# 1.- Sessions

For **security** or **configuration reasons**, some of the default values of the PHP session configuration can be changed.

- The session name ⇒ PHPSESSID.
- SID length.
- Session cookie lifetime.
- Session cache expiration.
- httponly (prevent unwanted behavior with JavaScript).

# 1.- Sessions

Each user's browser has its own session.

Sessions are distinguished by the session identifier **SID**.

User information is stored on the server and associated with the SID.
The SID is available in the user's client/browser and can be used in the application in one of the following ways:

- Propagate the SID in the URL
- Using a cookie (default method)

Both methods are automated with PHP.

# 2.- Propagating the SID in the URL

http://localhost/index.php**?PHPSESSID=4vjekic8fl7sqr0np45nfdrl6p**

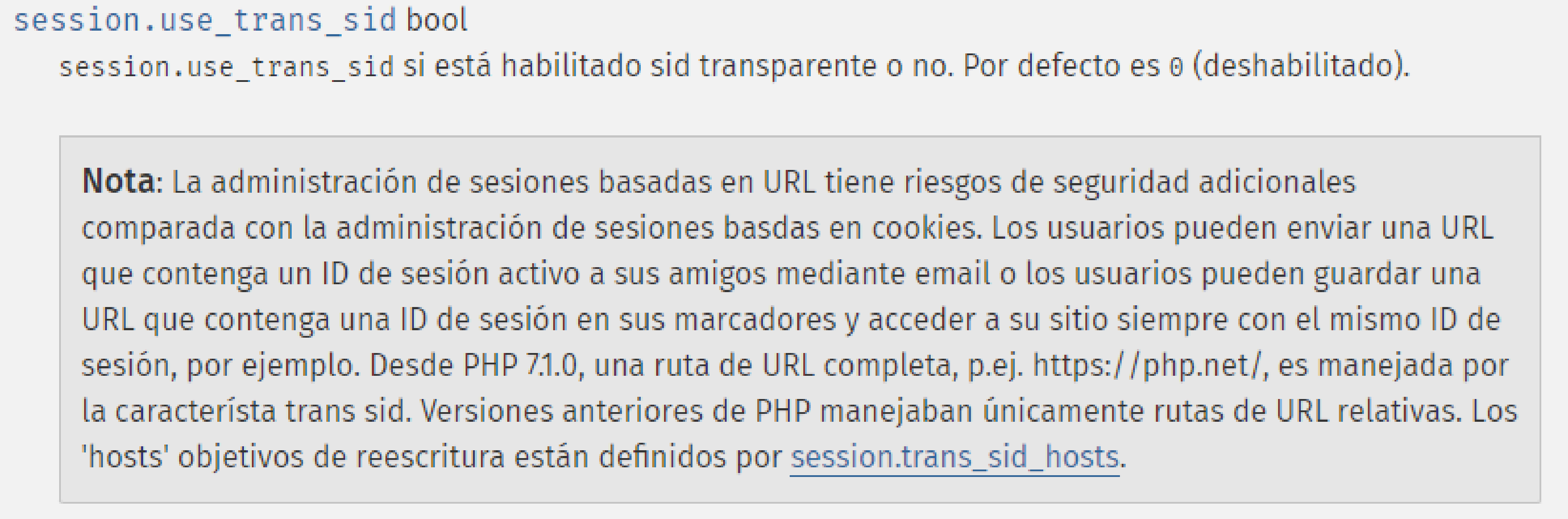
When cookies are not used, a global variable called **SID** is created in each session.

This variable should be added to all application link URLs.

This task can be done manually, programming it in the code, or automatically using the PHP option:

**session.use_trans_sid**

# 2.- Propagating the SID in the URL



`session.use_trans_sid` bool

`session.use_trans_sid` si está habilitado sid transparente o no. Por defecto es `0` (deshabilitado).

**Nota**: La administración de sesiones basadas en URL tiene riesgos de seguridad adicionales comparada con la administración de sesiones basdas en cookies. Los usuarios pueden enviar una URL que contenga un ID de sesión activo a sus amigos mediante email o los usuarios pueden guardar una URL que contenga una ID de sesión en sus marcadores y acceder a su sitio siempre con el mismo ID de sesión, por ejemplo. Desde PHP 7.1.0, una ruta de URL completa, p.ej. https://php.net/, es manejada por la característa trans sid. Versiones anteriores de PHP manejaban únicamente rutas de URL relativas. Los 'hosts' objetivos de reescritura están definidos por session.trans_sid_hosts.

# 3.- SID through cookies

When using sessions through cookies, the web server automatically saves a cookie on the client with the SID.

As already seen in Unit 5.2, cookies are sent whenever a request is made from the client.

In this way, the use of cookies to maintain the session is transparent for both the user and the programmer.

# 3.- SID through cookies

Both the use of cookies and the propagation of the SID in the URL have drawbacks.

The method with the most drawbacks is propagation in the URL:

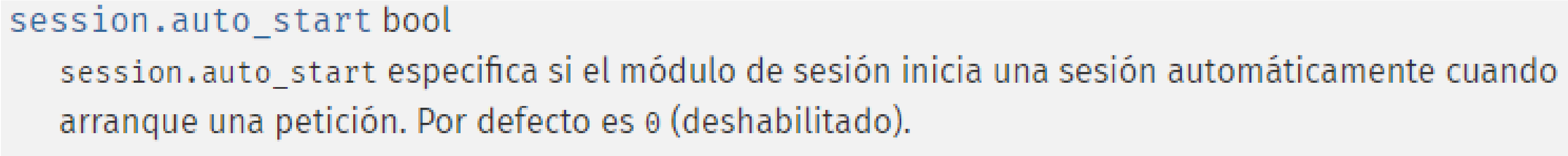
- Cannot maintain SID between different sessions.
- Sharing a URL shares the SID.

For these reasons, Apache's default session configuration is through the use of cookies.

This involves setting a cookie called PHPSESSID with a unique identification string as the value.

# 4.- Start of a session

Sessions can be started automatically by setting the **session.auto_start** parameter in the php.ini file.



session.auto_start bool

session.auto_start especifica si el módulo de sesión inicia una sesión automáticamente cuando arranque una petición. Por defecto es 0 (deshabilitado).

Or they can be started manually using the function:

**session_start()**

While a session is open, the superglobal variable **$_SESSION** can be used to store information or to retrieve that information.

# 4.- Start of a session

**session_start()**

Since using sessions requires the use of cookies, and these are sent in the headers, it is important that the function call occurs before information is displayed on the screen.

Or similarly, before the <!doctype html> line appears in the document.

The **session_start()** call must be made in all web application files that need session information.

# 4.- End of a session

Apache automatically manages session creation and destruction.

Using php.ini you can change the configuration.

Still, it may be necessary to log out at a certain time.

For example, if sessions are used to store login information, it will be necessary to log out if the user decides to log out.

**session_unset()**
Removes all created session variables but keeps the session identifier.

**session_destroy()**
Completely removes session information.

# 5.- Use of session variables

```
<?php
// The session is started or the previous existing session is recovered
session_start();

// Checks if the variable already exists
if (isset($_SESSION['visitas']))
        $_SESSION['visitas']++;
else
        $_SESSION['visitas'] = 0;
?>
<!doctype html>
<head>
    <meta charset="utf-8">
    <title>Ejemplo</title>
</head>
<body>
        Has visitado esta página <?=$_SESSION['visitas']?> veces
</body>
</html>
```

# 5.- Use of session variables

```
<?php
// The session is started or the previous existing session is recovered
session_start();

// On each visit a value is added to the "visits" array
$_SESSION['visitas'][] = mktime();
?>
<!doctype html>
<head>
        <meta charset="utf-8">
        <title>Ejemplo</title>
</head>
<body>
                Has visitado esta página <?=$_SESSION['visitas']?> veces
</body>
</html>
```

# 5.- Use of session variables

To change the default configuration (php.ini) use ini_set

```
<?php
    ini_set('session.name', 'miSesion');
    ini_set('session.cookie_httponly', 1);

    // The session is started or the previous existing session is recovered
    session_start();
...
?>
```

# Exercise - Modify discografia

Modify the Discography application to have the following:

- A registration page
- A login page (and logout option in the header)
- No page can be accessed if the user has not previously authenticated (the use of session is recommended)
- The latest searches are saved and displayed on the screen on the search page (the use of cookies is recommended)