ini

ENTITY ENTITY

ENTITY	ENTITY	ENTITY	ENTITY
session.save_path	""	`INI_ALL`	
session.name	"PHPSESSID"	`INI_ALL`	
session.save_handler	"files"	`INI_ALL`	
session.auto_start	"0"	`INI_PERDIR`	
session.gc_probability	"1"	`INI_ALL`	
session.gc_divisor	"100"	`INI_ALL`	
session.gc_maxlifetime	"1440"	`INI_ALL`	
session.serialize_handler	"php"	`INI_ALL`	
session.cookie_lifetime	"0"	`INI_ALL`	
session.cookie_path	"/"	`INI_ALL`	
session.cookie_domain	""	`INI_ALL`	
session.cookie_secure	"0"	`INI_ALL`	Prior to PHP 7.2.0, the default was `""`.
session.cookie_httponly	"0"	`INI_ALL`	Prior to PHP 7.2.0, the default was `""`.
session.cookie_partitioned	"0"	`INI_ALL`	Available as of PHP 8.5.0.
session.cookie_samesite	""	`INI_ALL`	Available as of PHP 7.3.0.
session.use_strict_mode	"0"	`INI_ALL`	
session.use_cookies	"1"	`INI_ALL`	
session.use_only_cookies	"1"	`INI_ALL`	Disabling this setting is deprecated as of PHP 8.4.0.
session.referer_check	""	`INI_ALL`	Setting a non-empty value is deprecated as of PHP 8.4.0.

ENTITY	ENTITY	ENTITY	ENTITY
session.cache_limiter	"nocache"	INI_ALL	
session.cache_expire	"180"	INI_ALL	
session.use_trans_sid	"0"	INI_ALL	Enabling this setting is deprecated as of PHP 8.4.0.
session.trans_sid_tags	"a=href,area=href,frame=src,form="	INI_ALL	Available as of PHP 7.1.0. Changing this setting is deprecated as of PHP 8.4.0.
session.trans_sid_hosts	$_SERVER['HTTP_HOST']	INI_ALL	Available as of PHP 7.1.0. Setting a non-empty value is deprecated as of PHP 8.4.0.
session.sid_length	"32"	INI_ALL	Available as of PHP 7.1.0. Changing this setting is deprecated as of PHP 8.4.0.
session.sid_bits_per_character	"4"	INI_ALL	Available as of PHP 7.1.0. Changing this setting is deprecated as of PHP 8.4.0.
session.upload_progress.enabled	"1"	INI_PERDIR	
session.upload_progress.cleanup	"1"	INI_PERDIR	
session.upload_progress.prefix	"upload_progress_"	INI_PERDIR	
session.upload_progress.name	"PHP_SESSION_UPLOAD_PROGRESS"	INI_PERDIR	
session.upload_progress.freq	"1%"	INI_PERDIR	
session.upload_progress.min_freq	"1"	INI_PERDIR	
session.lazy_write	"1"	INI_ALL	
session.hash_function	"0"	INI_ALL	Removed as of PHP 7.1.0.
session.hash_bits_per_character	"4"	INI_ALL	Removed as of PHP 7.1.0.
session.entropy_file	""	INI_ALL	Removed as of PHP 7.1.0.
session.entropy_length	"0"	INI_ALL	Removed as of PHP 7.1.0.

Session configuration options

ENTITY

The session management system supports a number of configuration options which you can place in your ENTITY file. We will give a short overview.

`session.save_handler` `string`
`session.save_handler` defines the name of the handler which is used for storing and retrieving data associated with a session. Defaults to `files`. Note that individual extensions may register their own `save_handler`s; registered handlers can be obtained on a per-installation basis by referring to `phpinfo`. See also `session_set_save_handler`.

`session.save_path` `string`
`session.save_path` defines the argument which is passed to the save handler. If you choose the default files handler, this is the path where the files are created. See also `session_save_path`.

There is an optional `N` argument to this directive that determines the number of directory levels your session files will be spread around in. For example, setting to `'5;/tmp'` may end up creating a session file and location like `/tmp/4/b/1/e/3/sess_4b1e384ad74619bd212e236e52a5a174If`. In order to use `N` you must create all of these directories before use. A small shell script exists in `ext/session` to do this, it's called `mod_files.sh`, with a Windows version called `mod_files.bat`. Also note that if `N` is used and greater than 0 then automatic garbage collection will not be performed, see a copy of ENTITY for further information. Also, if you use `N`, be sure to surround `session.save_path` in "quotes" because the separator (`;`) is also used for comments in ENTITY.

The file storage module creates files using mode 600 by default. This default can be changed with the optional `MODE` argument: `N;MODE;/path` where `MODE` is the octal representation of the mode. Setting `MODE` does not affect the process umask.

> If this is set to a world-readable directory, such as `/tmp` (the default), other users on the server may be able to hijack sessions by getting the list of files in that directory.

> When using the optional directory level argument `N`, as described above, note that using a value higher than 1 or 2 is inappropriate for most sites due to the large number of directories required: for example, a value of 3 implies that `(2 ** session.sid_bits_per_character) ** 3` directories exist on the filesystem, which can result in a lot of wasted space and inodes.
>
> Only use `N` greater than 2 if you are absolutely certain that your site is large enough to require it.

`session.name` `string`
`session.name` specifies the name of the session which is used as cookie name. It should only

contain alphanumeric characters. Defaults to `PHPSESSID`. See also `session_name`.

`session.auto_start` `bool`
`session.auto_start` specifies whether the session module starts a session automatically on request startup. Defaults to `0` (disabled).

`session.serialize_handler` `string`
`session.serialize_handler` defines the name of the handler which is used to serialize/deserialize data. PHP serialize format (name `php_serialize`), PHP internal formats (name `php` and `php_binary`) and WDDX are supported (name `wddx`). WDDX is only available, if PHP is compiled with WDDX support. `php_serialize` uses plain serialize/unserialize function internally and does not have limitations that `php` and `php_binary` have. Older serialize handlers cannot store numeric index nor string index contains special characters (`|` and `!`) in $_SESSION. Use `php_serialize` to avoid numeric index or special character errors at script shutdown. Defaults to `php`.

`session.gc_probability` `int`
`session.gc_probability` in conjunction with `session.gc_divisor` is used to manage probability that the gc (garbage collection) routine is started. Defaults to `1`. Must be greater than or equal to `0`. See session.gc_divisor for details.

`session.gc_divisor` `int`
`session.gc_divisor` coupled with `session.gc_probability` defines the probability that the gc (garbage collection) process is started on every session initialization. The probability is calculated by using gc_probability/gc_divisor, e.g. 1/100 means there is a 1% chance that the GC process starts on each request. `session.gc_divisor` defaults to `100`. Must be greater than `0`.

`session.gc_maxlifetime` `int`
`session.gc_maxlifetime` specifies the number of seconds after which data will be seen as 'garbage' and potentially cleaned up. Garbage collection may occur during session start (depending on session.gc_probability and session.gc_divisor). Defaults to `1440` (24 minutes).

> If different scripts have different values of `session.gc_maxlifetime` but share the same place for storing the session data then the script with the minimum value will be cleaning the data.
>
> In this case, use this directive together with session.save_path.

`session.referer_check` `string`
`session.referer_check` contains the substring you want to check each HTTP Referer for. If the Referer was sent by the client and the substring was not found, the embedded session id will be marked as invalid. Defaults to the empty string.

> Setting `session.referer_check` to a non-empty value is deprecated as of PHP 8.4.0.

`session.entropy_file` `string`
`session.entropy_file` gives a path to an external resource (file) which will be used as an additional entropy source in the session id creation process. Examples are `/dev/random` or `/dev/urandom` which are available on many Unix systems.

This feature is supported on Windows. Setting `session.entropy_length` to a non zero value will make PHP use the Windows Random API as entropy source.

> Removed in PHP 7.1.0.
>
> `session.entropy_file` defaults to `/dev/urandom` or `/dev/arandom` if it is available.

`session.entropy_length` `int`
`session.entropy_length` specifies the number of bytes which will be read from the file specified above. Defaults to `32`.

Removed in PHP 7.1.0.

`session.use_strict_mode` `bool`
`session.use_strict_mode` specifies whether the module will use strict session id mode. If this mode is enabled, the module does not accept uninitialized session IDs. If an uninitialized session ID is sent from the browser, a new session ID is sent to the browser. Applications are protected from session fixation via session adoption with strict mode. Defaults to `0` (disabled).

> Enabling `session.use_strict_mode` is mandatory for general session security. All sites are advised to enable this. See `session_create_id` example code for more details.

> If a custom session handler registered via `session_set_save_handler` does not implement SessionUpdateTimestampHandlerInterface::validateId, nor supplies the `validate_sid` callback, respectively, strict session ID mode is effectively disabled, regardless of the value of this directive. Particularly note that `SessionHandler` does *not* implement SessionHandler::validateId.

`session.use_cookies` `bool`
`session.use_cookies` specifies whether the module will use cookies to store the session id on the client side. Defaults to `1` (enabled).

`session.use_only_cookies` `bool`
`session.use_only_cookies` specifies whether the module will **only** use cookies to store the session id on the client side. Enabling this setting prevents attacks that involve passing session IDs in URLs. Defaults to `1` (enabled).

> Disabling `session.use_only_cookies` is deprecated as of PHP 8.4.0.

`session.cookie_lifetime` `int`
`session.cookie_lifetime` specifies the lifetime of the cookie in seconds which is sent to the browser. The value 0 means "until the browser is closed." Defaults to `0`. See also `session_get_cookie_params` and `session_set_cookie_params`.

> The expiration timestamp is set relative to the server time, which is not necessarily the same as the time in the client's browser.

`session.cookie_path` `string`
`session.cookie_path` specifies path to set in the session cookie. Defaults to `/`. See also `session_get_cookie_params` and `session_set_cookie_params`.

`session.cookie_domain` `string`
`session.cookie_domain` specifies the domain to set in the session cookie. Default is none at all meaning the host name of the server which generated the cookie according to cookies specification. See also `session_get_cookie_params` and `session_set_cookie_params`.

`session.cookie_secure` `bool`
`session.cookie_secure` specifies whether cookies should only be sent over secure connections. With this option set to `on`, sessions only work with HTTPS connections. If it is `off`, then sessions work with both HTTP and HTTPS connections. Defaults to `off`. See also `session_get_cookie_params` and `session_set_cookie_params`.

`session.cookie_httponly` `bool`
Marks the cookie as accessible only through the HTTP protocol. This means that the cookie won't be accessible by scripting languages, such as JavaScript. This setting can effectively help to reduce identity theft through XSS attacks (although it is not supported by all browsers).

`session.cookie_partitioned` `bool`
Marks the cookie as partitioned (CHIPS), which means that the cookie will be isolated to a first-party context. This setting can help mitigate cross-site tracking (although it is not supported by all browsers).

If enabled, session.cookie_secure must also be enabled, otherwise the session cookie will not be sent and a warning is raised.

`session.cookie_samesite` `string`
Allows servers to assert that a cookie ought not to be sent along with cross-site requests. This assertion allows user agents to mitigate the risk of cross-origin information leakage, and provides some protection against cross-site request forgery attacks. Note that this is not supported by all browsers. An empty value means that no SameSite cookie attribute will be set. `Lax` and `Strict` mean that the cookie will not be sent cross-domain for POST requests; `Lax` will send the cookie for

cross-domain GET requests, while `Strict` will not.

`session.cache_limiter` `string`
`session.cache_limiter` specifies the cache control method used for session pages. It may be one of the following values: `nocache`, `private`, `private_no_expire`, or `public`. Defaults to `nocache`. See also the `session_cache_limiter` documentation for information about what these values mean.

`session.cache_expire` `int`
`session.cache_expire` specifies time-to-live for cached session pages in minutes, this has no effect for nocache limiter. Defaults to `180`. See also `session_cache_expire`.

`session.use_trans_sid` `bool`
`session.use_trans_sid` whether transparent sid support is enabled or not. Defaults to `0` (disabled).

> Enabling `session.use_trans_sid` is deprecated as of PHP 8.4.0.

> URL based session management has additional security risks compared to cookie based session management. Users may send a URL that contains an active session ID to their friends by email or users may save a URL that contains a session ID to their bookmarks and access your site with the same session ID always, for example.
>
> Since PHP 7.1.0, full URL path, e.g. https://php.net/, is handled by trans sid feature. Previous PHP handled relative URL path only. Rewrite target hosts are defined by session.trans_sid_hosts.

`session.trans_sid_tags` `string`
`session.trans_sid_tags` specifies which HTML tags are rewritten to include session id when transparent sid support is enabled. Defaults to `a=href,area=href,frame=src,form=`

`form` is special tag. `ENTITYinput hidden="session_id" name="session_name"ENTITY` is added as form variable.

> Changing `session.trans_sid_tags` from its default value is deprecated as of PHP 8.4.0.

> Before PHP 7.1.0, url_rewriter.tags was used for this purpose. Since PHP 7.1.0, `fieldset` is no longer considered as special tag.

`session.trans_sid_hosts` `string`
`session.trans_sid_hosts` specifies which hosts are rewritten to include session id when transparent sid support is enabled. Defaults to `$_SERVER['HTTP_HOST']` Multiple hosts can be specified by ",", no space is allowed between hosts. e.g. `php.net,wiki.php.net,bugs.php.net`

Setting `session.trans_sid_hosts` to a non-empty value is deprecated as of PHP 8.4.0.

`session.sid_length` `int`
`session.sid_length` allows you to specify the length of session ID string. Session ID length can be between 22 to 256.

The default is 32. A longer session ID is harder to guess.

Changing `session.sid_length` from its default value is deprecated as of PHP 8.4.0.

Available as of PHP 7.1.0.

`session.sid_bits_per_character` `int`
`session.sid_bits_per_character` allows you to specify the number of bits in encoded session ID character. The possible values are '4' (0-9, a-f), '5' (0-9, a-v), and '6' (0-9, a-z, A-Z, "-", ",").

The default is 4. More bits result in a stronger session ID.

Changing `session.sid_bits_per_character` from its default value is deprecated as of PHP 8.4.0.

Available as of PHP 7.1.0.

`session.hash_function` `mixed`
`session.hash_function` allows you to specify the hash algorithm used to generate the session IDs. '0' means MD5 (128 bits) and '1' means SHA-1 (160 bits).

It is also possible to specify any of the algorithms provided by the hash extension (if it is available), like `sha512` or `whirlpool`. A complete list of supported algorithms can be obtained with the `hash_algos` function.

Removed in PHP 7.1.0.

`session.hash_bits_per_character` `int`
`session.hash_bits_per_character` allows you to define how many bits are stored in each character when converting the binary hash data to something readable. The possible values are '4' (0-9, a-f), '5' (0-9, a-v), and '6' (0-9, a-z, A-Z, "-", ",").

> Removed in PHP 7.1.0.

`session.upload_progress.enabled` `bool`
Enables upload progress tracking, populating the `$_SESSION` variable. Defaults to 1, enabled.

> Upload progress writes the client-supplied file name and field name to the session store while the upload in progress, to allow polling the process from a separate request. In applications that do not use upload progress bars, consider disabling this option; file names and field names are not validated and may contain PHP code, making any local file inclusion vulnerability trivial to exploit.

`session.upload_progress.cleanup` `bool`
Cleanup the progress information as soon as all POST data has been read (i.e. upload completed). Defaults to 1, enabled.

> It is highly recommended to keep this feature enabled.

`session.upload_progress.prefix` `string`
A prefix used for the upload progress key in the `$_SESSION`. This key will be concatenated with the value of `$_POST[ini_get("session.upload_progress.name")]` to provide a unique index.

Defaults to "upload_progress_".

`session.upload_progress.name` `string`
The name of the key to be used in `$_SESSION` storing the progress information. See also session.upload_progress.prefix.

If `$_POST[ini_get("session.upload_progress.name")]` is not passed or available, upload progressing will not be recorded.

Defaults to "PHP_SESSION_UPLOAD_PROGRESS".

`session.upload_progress.freq` `mixed`
Defines how often the upload progress information should be updated. This can be defined in bytes (i.e. "update progress information after every 100 bytes"), or in percentages (i.e. "update progress information after receiving every 1% of the whole filesize").

Defaults to "1%".

`session.upload_progress.min_freq` `int`
The minimum delay between updates, in seconds. Defaults to "1" (one second).

`session.lazy_write` `bool`

`session.lazy_write`, when set to 1, means that session data is only rewritten if it changes. Defaults to 1, enabled.

Upload progress will not be registered unless session.upload_progress.enabled is enabled, and the $_POST[ini_get("session.upload_progress.name")] variable is set. See Session Upload Progress for more details on this functionality.

□□□ #2
□ □□□ □□□ 2026-10-31 18:01:27 UTC
□ □□□ □□□ 2026-10-31 18:01:29 UTC