ionCube Loader 10 User Guide

This document describes the available php.ini configuration options of the ionCube Loader that relate to processing of PHP encoded files, and also the ionCube24 service. It also describes which encoded files can be run by each ionCube Loader.

PERFORMANCE OF ENCODED FILES

We recommend that the encoded paths feature (see below) is used with encoded files in order to maximise performance.

ENCODED FILES

INI entry: `ioncube.loader.encoded_paths`

Purpose: Specify the locations of encoded files

The ionCube Loader will normally examine a PHP file before processing to test whether it is encoded, and will run the file itself if necessary. Although this checking is very efficient, restricting which files the Loader tests for being encoded may give extra performance. If set to a series of paths or files, only files in those locations are tested.

Entries should be separated by a `:` on Unix and `;` on Windows. A path may be prefixed with `+` or `-` to add or remove that path from the possible locations. `+` is assumed if no character is given.

Examples: (`...` means `ioncube.loader.encoded_paths`)

- Site with a single encoded module in `/var/www/html/modules/RSS`

```
... = "/var/www/html/modules/RSS"
```

- As above, with a site configuration file encoded too.

```
... = "/var/www/html/modules/RSS:/var/www/html/config/config.php"
```

- Encoded files may be anywhere except for `/var/www/html/framework`

```
... = "/:-/var/www/html/framework"
```

- Site with most modules encoded except for one

```
... = "/var/www/html/modules:-/var/www/html/modules/plain"
```

- As above, with an encoded config file in the plain directory

```
... = "/site/modules:-/site/modules/plain:/site/modules/plain/config.php"
```

Locations:

The `ioncube.loader.encoded_paths` property can be set in a `php.ini` file, in a `.htaccess` file (when using Apache), in a `.user.ini` file (when using CGI PHP 5.3+) or using `ini_set` within a PHP script. In ini files only the last value will be used for the encoded_paths property. If you wish to build up the value in several lines then, for PHP 5.1+, you can use the following syntax:

```
ioncube.loader.encoded_paths = "/path1"
ioncube.loader.encoded_paths = ${ioncube.loader.encoded_paths}":/path2"
; etc...
```

LIMITATIONS OF LOADERS AND ENCODED FILES

Encoded files can, in general, run on versions of PHP equal to or greater than the source language of the Encoder used to produce them. So a file produced by the Encoder for PHP 5.3 can be run by the Loaders for PHP 5.3, 5.4, 5.5 and PHP 5.6. This means that the Loaders are highly backwards compatible. However, there are the following limitations:

- The Loader for PHP 7.1 can only run files produced by the Encoder for PHP 7.1 included in the version 10 Encoder.
- The Loader for PHP 7.0 can only run files produced by the Encoder for PHP 5.6.
- The Loaders for PHP 5.5 and PHP 5.6 cannot run files produced by the PHP 4 Encoder.

IONCUBE24 : real-time intrusion protection and PHP error reporting

(Available for Linux 32 and 64 bit x86 servers)

ionCube24 (https://ioncube24.com) is an ionCube service that uses the ionCube Loader to provide both real-time protection against the exploit of website vulnerabilities and alerting of website errors.

Vulnerabilities in PHP applications are common, particularly in sites using Wordpress and other plugin based systems. Exploits result in website defacement or customer data being compromised, and ionCube24 provides a uniquely powerful defense.

PHP errors can cause intermittent or even persistent blank pages or errors for visitors until discovered, and without active monitoring this could go undetected indefinitely. ionCube24 active monitoring ensures you are always aware of problems in your website code.

ionCube24 is free to try, with the server side support built into the Linux Loaders as standard. With the Loader installed, ionCube24 can be activated at any time to give active intrusion protection and error reporting.

php.ini settings

ionCube24 has a powerful real-time web interface to configure, monitor and manage things, and there are also settings that can be used in a php.ini file as summarised below.

The setup process at https://ioncube24.com automatically gives the settings that you need to get started, but you may wish to make changes yourself once setup. The default values are given with each example.

Global settings

INI entry: `ic24.enable ; default 0`

Purpose: Enable or disable all ionCube24 features.

This defaults to 0 (off), and in this case no ionCube24 behaviour is activated.

Example:

```
ic24.enable = 1
```

INI entry: `ic24.api_access_key ; provided during setup`

Purpose: An authentication key for adminstration requests.

This value is provided when adding a server to ionCube24.

INI entry: `ic24.api_check_ip ; default 1`

Purpose: Specify whether the IP for admin requests should be validated

If set, ionCube24 refuses access to API functions unless the calling IP is a known ionCube IP address. This option should be left with the default setting unless web requests pass through a proxy and your site appears to be accessed from the IP of the proxy instead of ionCube. Note that access to API functions will still be authenticated by access key.

INI entry: `ic24.api_max_timeout ; default 7`

Purpose: Maximum timeout period when sending notifications to ionCube24.

The actual period is adaptive so that a brief increase in typical latency will favour a timeout followed by a retry rather than a longer than usual timeout.

INI entry: `ic24.home_dir ; no default`

Purpose: The home directory for ionCube24 related system files.

A location outside of the web root is recommended. It should be writable by the web server during startup.

Example:

```
ic24.home_dir = /var/www/ic24_home
```

INI entry: `ic24.update_domains_retry_interval ; default 30`

Purpose: The number of seconds to wait before retrying a fetch of the set of domains being managed.

Security related settings

INI entry: `ic24.sec.enable ; default "auto"`

Purpose: Enable the intrusion protection feature of ionCube24.

Accepted values:

- "auto" (default) - allow setting from the ionCube24 control panel.
- 1 : always enabled.
- 0 : disabled.

INI entry: `ic24.sec.initial_state ; default 1`

Purpose: The default for whether security should be enabled or disabled. The default is to enable protection. Any files on a protected domain will become blocked if they are changed, so setting this to 0 will avoid accidental blocking when using ionCube24 for the first time. Protection may be enabled and disabled using the ionCube24 control panel and also via the User API.

Accepted values:

- 1 : protection will be active when ionCube24 initialises.
- 0 : protection will be disabled.

INI entry: `ic24.sec.initial_action ; default "block"`

Purpose: The initial setting for how new and modified files should be treated when about to execute. The default is to block. The action is taken only if protection is enabled, and the setting may be changed via the ionCube24 control panel.

Accepted values:

- "block" : prevent execution of new or modified files
- "allow" : allow execution of new or modified files

Note that depending on the notification settings, a notification may still be generated when a new or modified file is about to execute even if it is not blocked.

INI entry: `ic24.sec.initial_notify ; default "always"`

Purpose: The initial setting for whether a notification is generated the first time an unacknowledged new or modified file is attempted to be executed. This setting can be changed via the ionCube24 control panel.

Accepted values:

- "always" : always notify of a new modification
- "once" : only the first detected modification is reported
- "never" : never notify of new and modified files

INI entry: `ic24.sec.exclusion_key ; provided during setup`

Purpose: A key that if present at the start of a file, will identify the file as trusted. This value is provided when adding a server to ionCube24.

INI entry: `ic24.sec.trusted_include_paths ; no default`

Purpose: List paths from where files can be included and automatically trusted.

Example:

```
ic24.sec.trusted_include_paths = "/var/cache:/var/cache2"
```

Directories can be excluded from the list by prefixing with a minus character -. e.g.

```
"/var/cache:-/var/cache/subdir"
```

This is useful if your site creates and/or modifies files by itself from time to time, e.g. in a cache

directory. Requests that *directly* access files on a trusted include path will be blocked but the file itself will not be blocked, so requests that use the file as intended will still work. See ioncube24.com for more details once signed up. As an alternative, if possible we recommend producing files that include the exclusion key.

INI entry: `ic24.sec.block_uploaded_files ; default 1`

Purpose: If set, block any uploaded files in ionCube24 that are processed using the standard PHP mechanism for uploaded files. This applies even if the file is subsequently included and where included files being automatically approved with the previous setting.

INI entry: `ic24.sec.block_stdin ; default 1`

Purpose: Refuse code that PHP sees via `stdin`. If disabled, code via `stdin` will run without security checking as there is no filepath. This setting should be left on as PHP would normally never receive a script via `stdin`.

PHP Error reporting settings

INI entry: `ic24.phperr.enable ; default "auto"`

Purpose: Enable reporting of PHP errors to ionCube24. When enabled, any non-ignored errors are reported to ionCube24 in realtime, triggering alerting so errors can be investigated as necessary.

Accepted values:

- "auto" (default) - allow setting from the ionCube24 control panel.
- 1 : always enabled.
- 0 : disabled.

Deprecated settings

Deprecated settings are subject to removal in a future release.

INI entry: `ic24.phperr.ignore ; default 0`

Purpose: Specify default error levels to always ignore for all domains.

Note that default and per-domain errors to ignore can also be set via the web interface, and are combined with this setting. Leaving this unset and using the web interface is recommended for maximum flexibility.

Example:

```
ic24.phperr.ignore = E_NOTICE | E_DEPRECATED
```