<https://docs.travis-ci.com/user/encryption-keys/>

Encryption keys

**We have separate documentation on**[**encrypting files**](https://docs.travis-ci.com/user/encrypting-files/)**.**

A repository’s .travis.yml file can have “encrypted values”, such as [environment variables](https://docs.travis-ci.com/user/environment-variables/), notification settings, and deploy api keys. These encrypted values can be added by anyone, but are only readable by Travis CI. The repository owner does not keep any secret key material.

**Please note that encrypted environment variables are not available for**[**pull requests from forks**](https://docs.travis-ci.com/user/pull-requests#Pull-Requests-and-Security-Restrictions)**.**

Encryption scheme [#](https://docs.travis-ci.com/user/encryption-keys/#Encryption-scheme)

Travis CI uses asymmetric cryptography. For each registered repository, Travis CI generates an RSA keypair. Travis CI keeps the private key private, but makes the repository’s public key available to everyone. For example, the GitHub repository foo/bar has its public key available at https://api.travis-ci.org/repos/foo/bar/key. Anyone can run travis encrypt for any repository, which encrypts the arguments using the repository’s public key. Therefore, foo/bar’s encrypted values can be decrypted by Travis CI, using foo/bar’s private key, but the values cannot be decrypted by anyone else (not even the encrypter, or “owner” of the foo/bar repository!).

Usage [#](https://docs.travis-ci.com/user/encryption-keys/#Usage)

The easiest way to encrypt something with the public key is to use Travis CLI. This tool is written in Ruby and published as a gem. First, you need to install the gem:

gem install travis

Bash

Then, you can use encrypt command to encrypt data (This example assumes you are running the command in your project directory. If not, add -r owner/project):

travis encrypt SOMEVAR="secretvalue"

Bash

This will output a string looking something like:

secure: ".... encrypted data ...."

YAML

Now you can place it in the .travis.yml file.

You can also skip the above, and add it automatically by running:

travis encrypt SOMEVAR="secretvalue" --add

Bash

Please note that the name of the environment variable and its value are both encoded in the string produced by “travis encrypt.” You must add the entry to your .travis.yml with key “secure” (underneath the “env” key). This makes the environment variable SOMEVAR with value “secretvalue” available to your program.

You may add multiple entries to your .travis.yml with key “secure.” They will all be available to your program.

Encrypted values can be used in [secure environment variables in the build matrix](https://docs.travis-ci.com/user/environment-variables#Defining-Variables-in-.travis.yml) and [notifications](https://docs.travis-ci.com/user/notifications).

**Note on escaping certain symbols**[**#**](https://docs.travis-ci.com/user/encryption-keys/#Note-on-escaping-certain-symbols)

When you use travis encrypt to encrypt sensitive data, it is important to note that it will be processed as a bash statement. This means that secret you are encrypting should not cause errors when bash parses it. Having incomplete data will cause bash to dump the error statement to the log, which contains portions of your sensitive data.

Thus, you need to escape [special characters](http://www.tldp.org/LDP/abs/html/special-chars.html) such as braces, parentheses, backslashes, and pipe symbols. For example, when you want to assign the string 6&a(5!1Ab\ to FOO, you need to execute:

travis encrypt "FOO=6\\&a\\(5\\!1Ab\\\\"

Bash

travis encrypts the string FOO=6\&a\(5\!1Ab\\, which then bash uses to evaluate in the build environment.

Equivalently, you can do

travis encrypt 'FOO=6\&a\(5\!1AB\\'

Bash

**Notifications Example**[**#**](https://docs.travis-ci.com/user/encryption-keys/#Notifications-Example)

We want to add campfire notifications to our .travis.yml file, but we don’t want to publicly expose our API token.

The entry should be in this format:

notifications:

campfire:

rooms: [subdomain]:[api token]@[room id]

YAML

For us, that is somedomain:abcxyz@14.

We encrypt this string

travis encrypt somedomain:abcxyz@14

Bash

Which produces something like this

Please add the following to your .travis.yml file:

secure: "ABC5OwLpwB7L6Ca...."

We add to our .travis.yml file

notifications:

campfire:

rooms:

secure: "ABC5OwLpwB7L6Ca...."

YAML

And we’re done.

**Detailed Discussion**[**#**](https://docs.travis-ci.com/user/encryption-keys/#Detailed-Discussion)

The secure var system takes values of the form { 'secure' => 'encrypted string' } in the (parsed YAML) configuration and replaces it with the decrypted string.

So

notifications:

campfire:

rooms:

secure: "encrypted string"

YAML

becomes

notifications:

campfire:

rooms: "decrypted string"

YAML

while

notifications:

campfire:

rooms:

- secure: "encrypted string"

YAML

becomes

notifications:

campfire:

rooms:

- "decrypted string"

YAML

In the case of secure env vars

env:

- secure: "encrypted string"

YAML

becomes

env:

- "decrypted string"

YAML

Fetching the public key for your repository [#](https://docs.travis-ci.com/user/encryption-keys/#Fetching-the-public-key-for-your-repository)

You can fetch the public key with Travis API, using /repos/:owner/:name/key or /repos/:id/key endpoints, for example:

https://api.travis-ci.org/repos/travis-ci/travis-ci/key

You can also use the travis tool for retrieving said key:

travis pubkey

Bash

Or, if you’re not in your project directory:

travis pubkey -r owner/project

Bash

Note, travis uses travis.slug in your project to determine the endpoints if it exists (check by using git config --local travis.slug), if you rename your repo or move your repo to another user/organization, you might need to change it.