



## Generating 2FA codes in your terminal

### Table of contents

1. Introduction
2. Requirements
3. Usage
4. Information

=====

#### =====

### 1. Introduction

#### =====

**2FA-Auth** is a project written to run in GNU/Linux terminal with BASH as default interpreter. Its goal is to help all users to easily create "Two-Factor Authentication (2FA)" codes without using your cellphone/tablet.

#### =====

### 2. Requirements

#### =====

2FA-Auth requires 2 programs: **GnuPG** (a.k.a. **GPG**) and **OATH Toolkit**. A simple explanation about these programs:

- **GnuPG(\*)** encrypt and decrypt your tokens file;
- **OATH Toolkit** is responsible for get your tokens and generate 2FA authentication codes.

(\*) **GnuPG** is a cryptography tool that uses keys and subkeys to sign, encrypt, decrypt, certificate and/or authenticate every single piece of information.

### 3. Usage

All steps must be done on your terminal. So, let's start!  
Open your terminal and type:

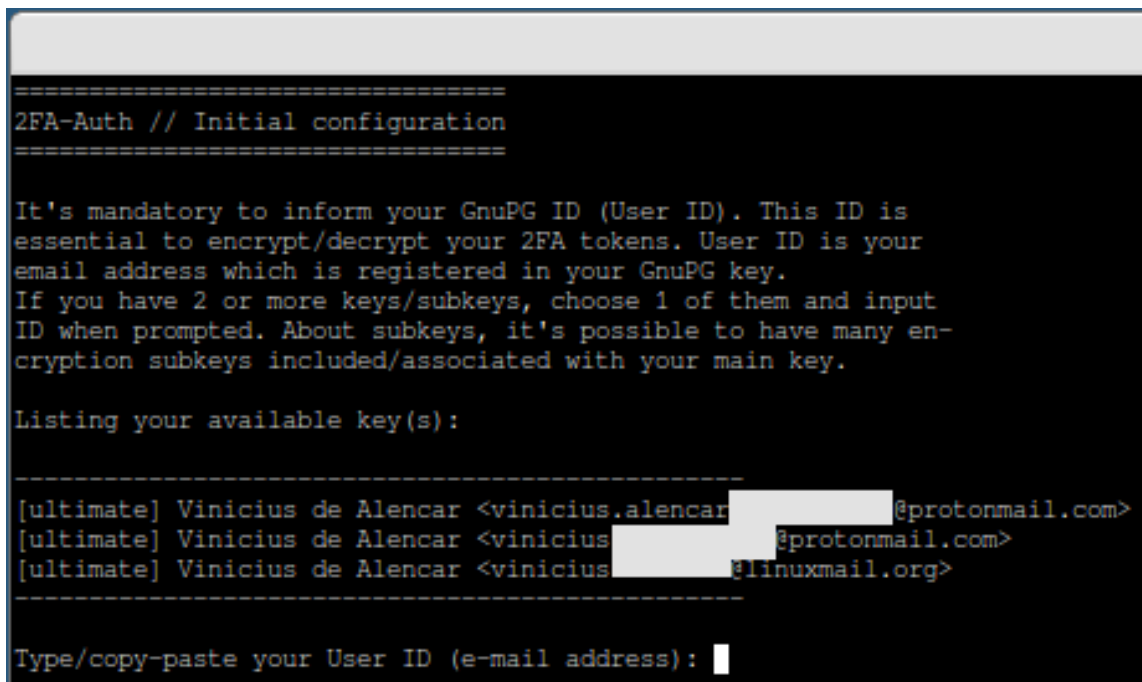
```
$ git clone https://github.com/alenccl1986/2FA-Auth.git
# With the step, you clone (download) 2FA-Auth project.

$ cd /path/to/2FA-Auth
$ ./2FA-Auth.sh
# Alternative cmd: you can type /path/to/2FA-Auth/2FA-Auth.sh
```

When you run 2FA-Auth for the first time, it's going to verify if the dependencies (GnuPG and OATH Toolkit) are installed. If not, 2FA-Auth tries to automatically install them. Maybe, 2FA-Auth cannot install them (this happens when 2FA-Auth doesn't support your distribution's package manager). In this case, you need to install them manually.

Once these programs are installed, the project tries to find your GnuPG keys (it's needed to have your GPG keys created or imported in your profile; otherwise, 2FA-Auth will stop running and will show an error message). With the GPG keys created/imported in your profile, 2FA-Auth lists all available keys.

So, the first screen you'll see is this one (the initial configuration screen):



```
=====
2FA-Auth // Initial configuration
=====

It's mandatory to inform your GnuPG ID (User ID). This ID is
essential to encrypt/decrypt your 2FA tokens. User ID is your
email address which is registered in your GnuPG key.
If you have 2 or more keys/subkeys, choose 1 of them and input
ID when prompted. About subkeys, it's possible to have many en-
cryption subkeys included/associated with your main key.

Listing your available key(s):

-----
[ultimate] Vinicius de Alencar <vinicius.alencar[REDACTED]@protonmail.com>
[ultimate] Vinicius de Alencar <vinicius[REDACTED]@protonmail.com>
[ultimate] Vinicius de Alencar <vinicius[REDACTED]@linuxmail.org>
-----

Type/copy-paste your User ID (e-mail address): █
```

In this screen, you can see that it's necessary to input (type or copy-paste) your ID. If you have more than your key, choose one of them.

Note: this piece of information is saved in the file "\$HOME/.config/2fa-auth/2fa-auth.info", and every time that you run 2FA-Auth, it reads this file.

The next screen is the main menu, as you can see below:

```
=====
2FA-Auth // Main menu
=====

-----
| 2FA-Auth has 2 terminal parameters |
| 'changekey' -- change GnuPG key/encryption |
| 'gencode'   -- generate auth codes in a fast |
|               way without use the main menu |
|-----|

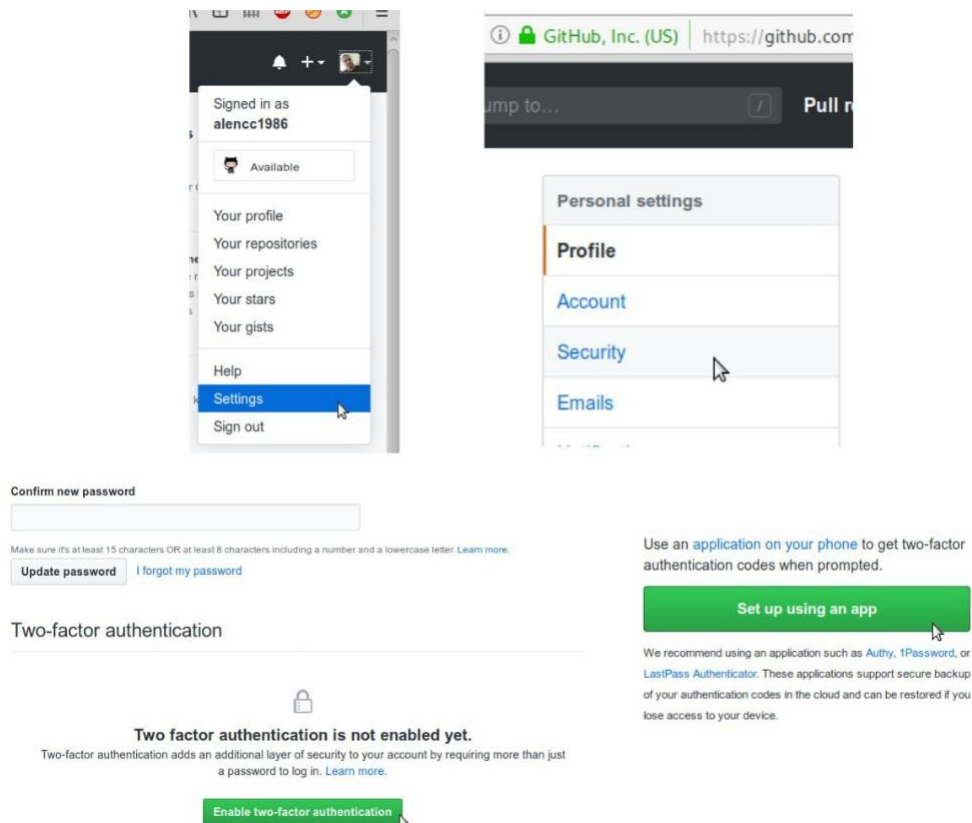
[1] Add new 2FA auth tokens
[2] Delete 2FA auth tokens
[3] List all 2FA auth tokens
[4] Rename 2FA auth tokens
[5] Export all 2FA auth tokens
[6] Generate 2FA auth codes
[7] Backup your tokens/config
[8] Restore your tokens/config

[C] Change GnuPG encryption key
[I] Information
[Q] Quit

Option: 
```

Use numbers and/or the letters “C”, “I” or “Q” to select one option. To generate 2FA codes, you need to add 2FA tokens. First, you need to configure your account to work with “Two-Factor Authentication”. Using GitHub as example, follow these steps:

In your profile, select “Settings”, go to “Security”, click on “Enable two-factor authentication” and then “Set up using an app”.



When activating 2FA authentication, it's normal to get "Recovery Codes". They are used if you are facing trouble with "two-factor authentication" code.

### 1. Recovery codes

Recovery codes are used to access your account in the event you cannot receive two-factor authentication codes.

Download, print, or copy your recovery codes before continuing two-factor authentication setup below.

• 97454-4	• 99758-8
• e84f6-f	• 0f2d6-c
• a04dc-5	• 35595-9
• d3a99-4	• 8a009-c
• 0e244-0	• efde4-7
• 99294-9	• e17e0-1
• c0669-1	• 70baf-7
• a0112-5	• 2201b-d

DownloadPrintCopy


Treat your recovery codes with the same level of attention as you would your password! We recommend saving them with a password manager such as [Lastpass](#), [1Password](#), or [Keeper](#).

NextCancel

Download/copy them, and click on "Next". On the next screen, you are going to see a QR-Code that is used to scan with your cellphone/tablet's camera.

### 2. Scan this barcode with your app.

Scan the image above with the two-factor authentication app on your phone. If you can't use a barcode, [enter this text code](#) instead.

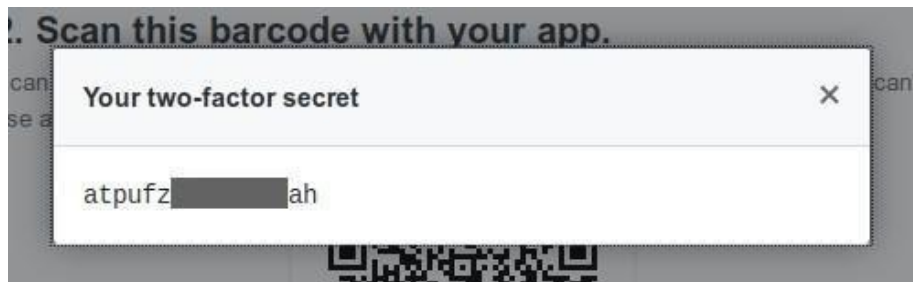


**Enter the six-digit code from the application**

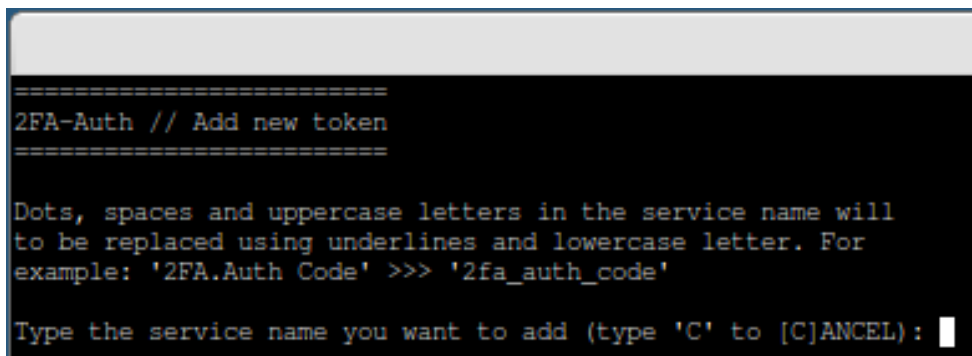
After scanning the barcode image, the app will display a six-digit code that you can enter below.

EnableCancel

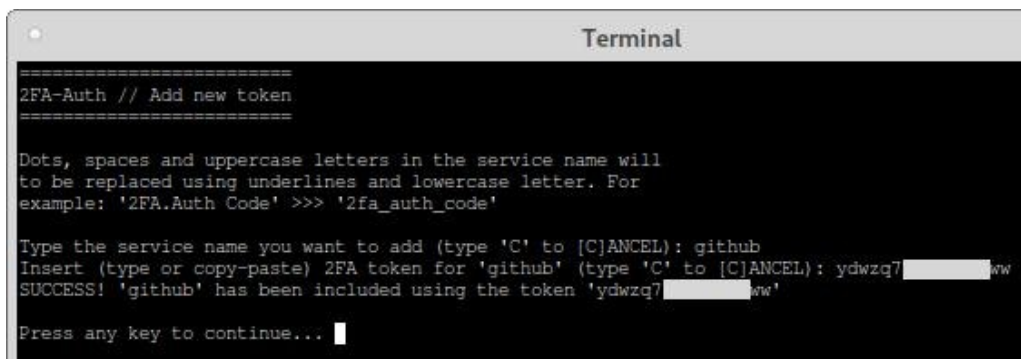
Well, this is not the case, once it's not possible to scan with 2FA-Auth. Just click on the link "enter this text code". This screen will provide an alpha-numeric code (this is your 2FA token):



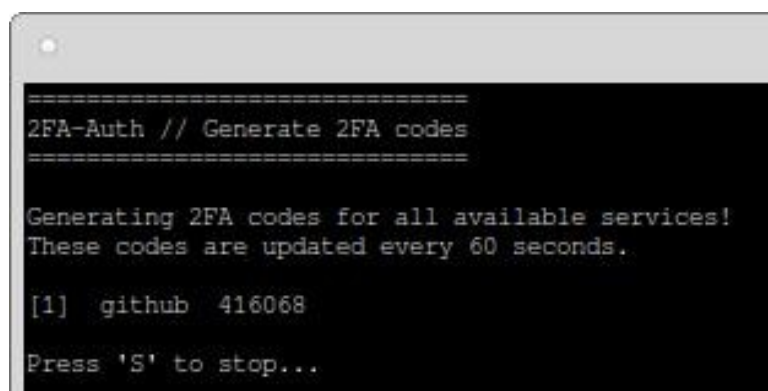
Copy this token, go to 2FA-Auth and select the option [1] to add this token in your profile.



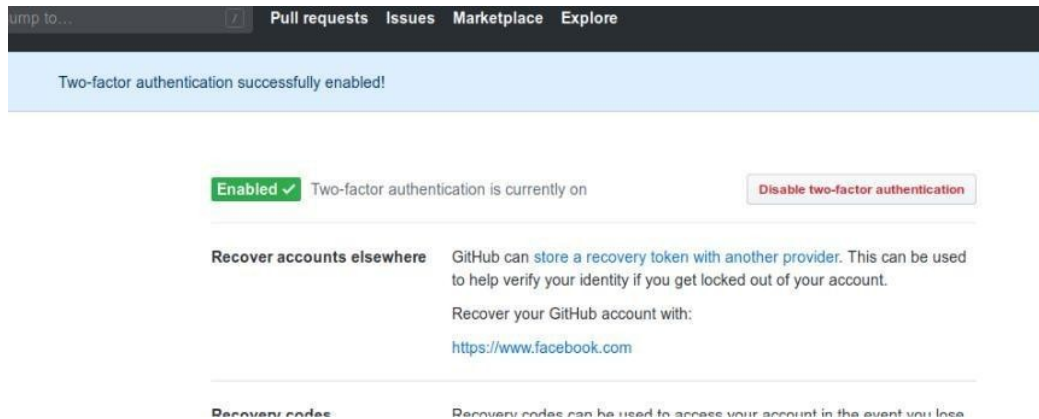
Type the account name and press <ENTER>.  
Type or copy your 2FA token. Again, press <ENTER>.



Now, once your token was added, you need to generate the 2FA code. Press any key, select option [5] to generate your authentication code.



Get this code and input it into the site field that says “Enter the six-digit code from the application” (in the case of GitHub, this field is named like this; in other sites/services, find a similar field). Detail: you need to input this code as soon as possible, because 2FA codes change every 60 seconds. Once the site accepts your code, you’re going to see this screen:



Now, your account and 2FA-Auth have 2FA authentication set. Use it every time that your account asks for 2FA code. If you want to disable 2FA from your account, go to your profile settings and select “Disable”. See the image bellow:

## Two-factor authentication

Enabled

## Two-factor authentication

Disable

Back to 2FA-Auth, in the main menu, select the option [2], and select the service you want to delete. Confirm it with “y”.

```

Terminal
=====
2FA-Auth // Delete token
=====
Select a service to be deleted (type 'A' to delete [A]LL or 'C' to [C]ANCEL):
[1] github
Option: 1
Are you sure you want to delete 'github' token? [y/N] y
SUCCESS! 'github' token deleted!
Press any key to continue... █

```

Now, you have disabled 2FA authentication and deleted the token for your account.

The other options are:

- List all available tokens;
- Export your tokens; (\*)
- Backup and restore your files with your GnuPG IDs and 2FA tokens; (\*\*)
- Change your GnuPG key (use it when you create/import new GPG key and want to change you tokens file encryption).

(\*) Exporting your 2FA tokens is useful when you want to authenticate using more than 1 app/script which generates 2FA codes. For example, you want to use 2FA-Auth and Google® Authenticator®.

(\*\*) 2FA-Auth doesn't backup your GnuPG Keys.

There's an easy way to generate 2FA authentication codes without use 2FA-Auth's main menu. Once you have included your 2FA tokens, open a terminal and type:

```
$ /path/to/2FA-Auth/2FA-Auth.sh gencode
```

Also, another parameter is 'changekey'. Use this parameter (or option 'C' in main menu) when/if you create or import new GnuPG key and want to replace your tokens encryption.

#### ===== **4. Information** =====

"2FA-Auth" was created by Vinicius de Alencar (<https://github.com/alencc1986>) once he wanted an easy way to use 2FA authentication (with OATH Toolkit) on GNU/Linux.

This idea came after he saw a tutorial using 2 independent scripts that were used to encrypt/decrypt the tokens and use them to generate 2FA authentication tokens. So, why not create a project which can be more user-friendly than 2 independent scripts? Or manually generate 2FA codes in your terminal with OATH Toolkit?

Copyleft - Vinicius de Alencar (alencc1986) - 2020