Generating 2FA codes in your terminal

1. Introduction

2FA-Auth is a script written to run in BASH shell, and its goal is to help people to easily generate *"two-factor authentication (2fa)"* codes using GNU/Linux terminal/console.

2. Requirements

In order to make 2FA-Auth work perfectly well, you **MUST** install GIT, GnuPG (a.k.a. GPG), OATH Toolkit (OATHTOOL) in your system, and you need to create/import GnuPG keys in your profile.

Depending on the GNU/Linux distribution, there's a specific way to install these programs, but the process of create/import the GPG keys are the same for all distros. If you want, you can read these articles bellow (Tutorial for Beginners), once they teach you the basic of GnuPG (create, import, export, delete and revoke keys, as well as how to use your GPG keys):

How to install and use GnuPG on GNU/Linux – https://redd.it/creb29
GnuPG: how to export, import, delete and revoke your keys – https://redd.it/ct7yjr

– Explanation about the softwares required by 2FA-Auth

GIT is a program used to manage a project's source code, upload and download any code modification, and let many people work together (in parallel) in a project. For 2FA-Auth, you are going to use GIT to clone (download) the project into any directory on your computer.

GnuPG is a cryptography software which is used by 2FA-Auth to encrypt and decrypt 2FA tokens (alpha-numeric codes) used to generate "two-factor authentication" login codes.

GPG keys are your public and private key, files used to encrypt and decrypt any piece of information (in this project, it's used to encrypt and decrypt your 2FA tokens).

OATH Toolkit (OATHTOOL) is the software responsible for generate 2FA login codes based on the 2FA tokens that belong to the user.

3. <u>Usage</u>

First step: you need to download 2FA-Auth. So, open a terminal and type:

> $ git clone https://github.com/alencc1986/2FA-Auth.git

As soon as the cloning process is over, access 2FA-Auth directory and run it:

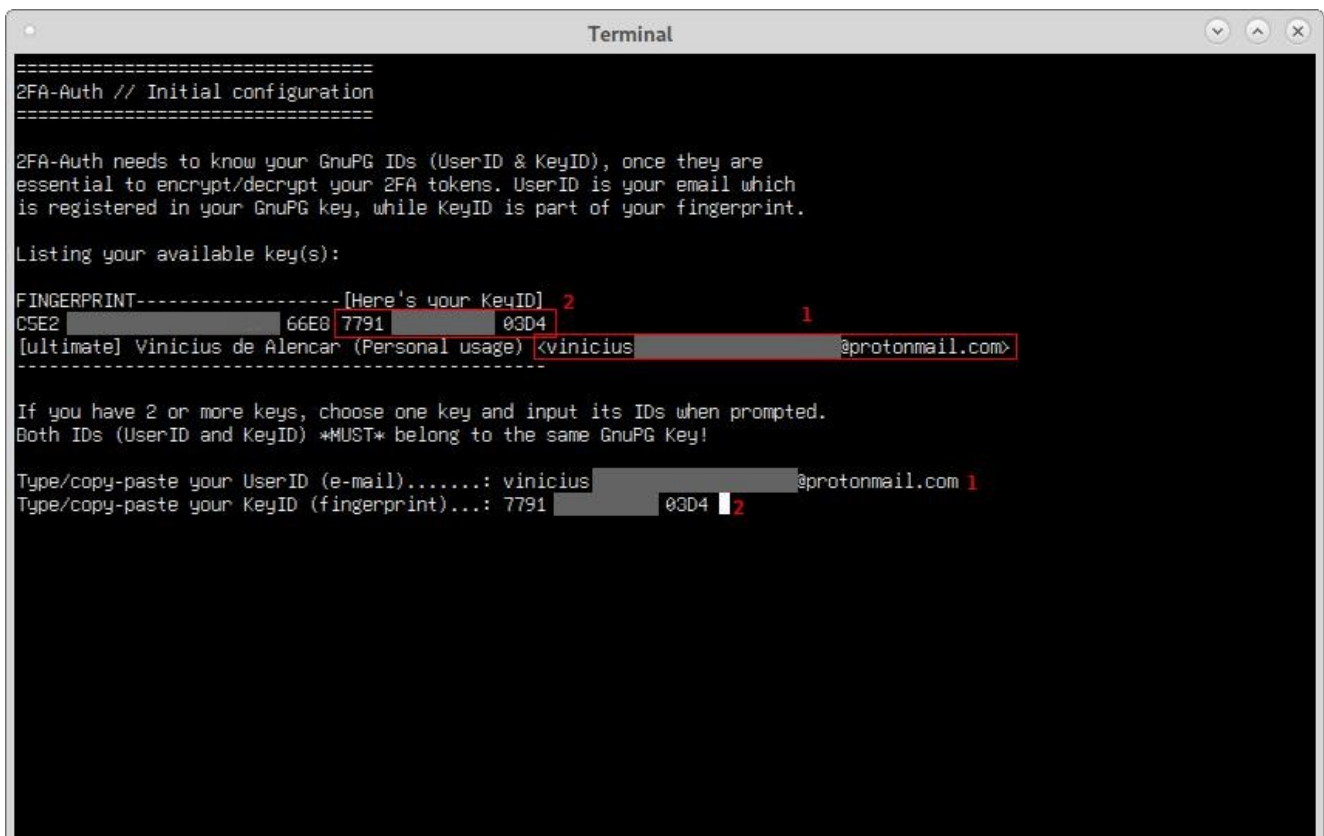> $ cd /path/to/2FA-Auth/
> $ ./2FA-auth.sh

Optionally, you can type "./2FA-Auth.sh help" in order to get a simple help menu.
Once it's your first time running the script, it's going to check these basic information:

– Are GnuPG and OATHTOOL installed?
– Did the user create or import his/her GPG keys?
– Is there a hidden directory called $HOME/.config/2fa-auth?

If the answer is "no" for questions 1 and 2, the script is going to stop and show a error message.
For the question 3, a hidden directory is created and used to storage your services tokens and GnuPG IDs (these IDs are storaged into a file called $HOME/.config/2fa-auth/2fa-info).

But, what are these GPG IDs?
They are not your GPG keys, but part of your key fingerprint (KeyID) and your e-mail used to create your key (UserID). Bellow, there's the initial configuration screen which shows all available keys in your system and asks you for your GPG IDs. If you have more the 1 key, just pick one and use it to encrypt/decrypt all your tokens.

As you can see, in this example there's just 1 key and the IDs are marked in red (number 1 is for GPG UserID and number 2 is for GPG KeyID). Once you inputted these information above, a main menu is going to appear, like bellow.
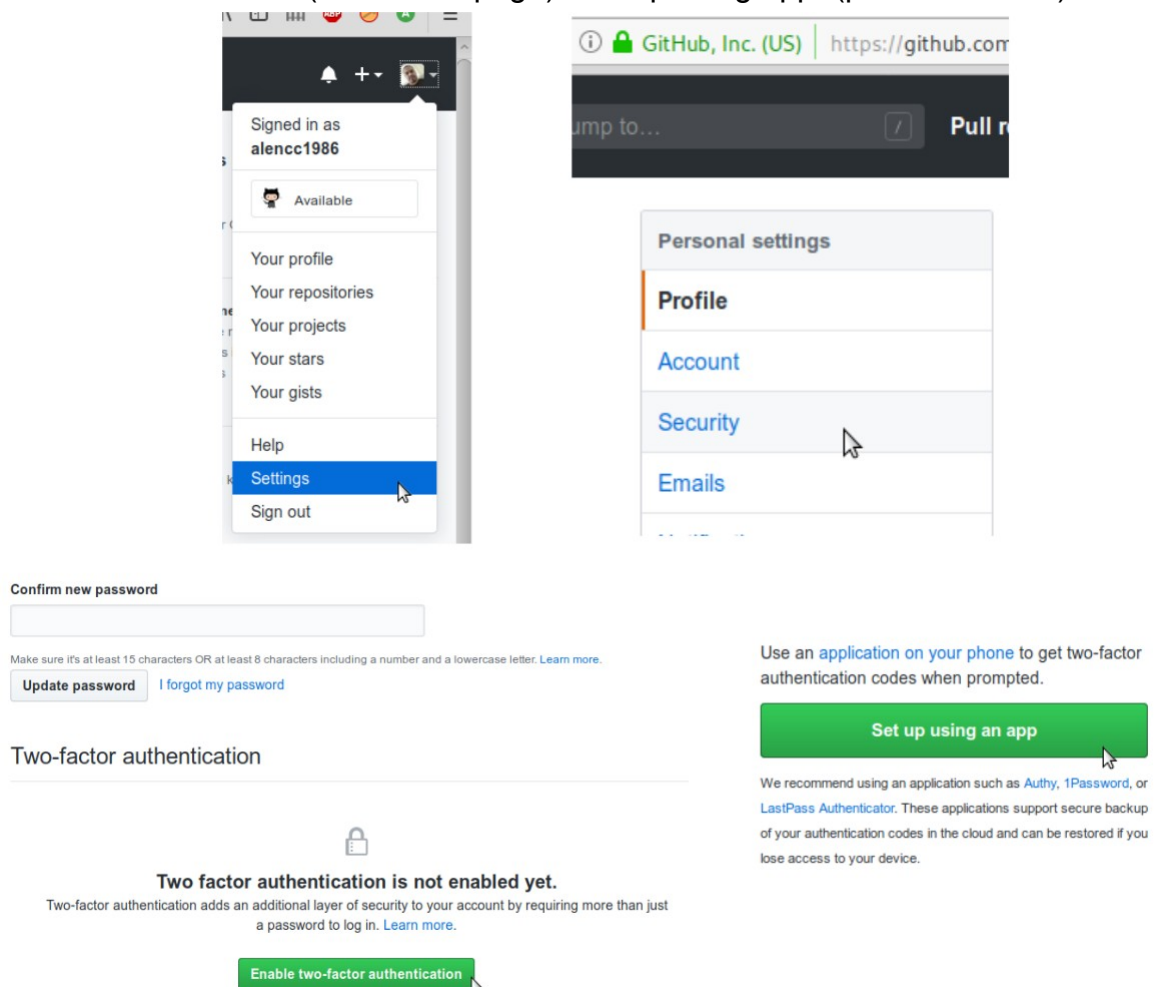

The first thing you need to do is to create (add) a token in your profile. Using GitHub® as example, access your account settings, go to "Security", then click on the button "Enable two-factor authentication" and (in the next page) "Set up using app" (pictures bellow).

The next screen is going to give you 16 recovery codes, which can be used if you lose access to your account and/or 2FA code generator isn't working anymore. Keep these codes in a safe place.

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

[⤓ Download]   [▤ Print]   [⧉ Copy]

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.

[Next]   [Cancel]

Pick one option (Download, Print or Copy), and then click on NEXT. On the next screen, you're going to see a QR-Code, like this bellow.

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.

123456

[Enable]   [Cancel]

This QR-code is used by mobile apps (like Google Authenticator®) to get the 2FA token and use it in order to generate 2FA authentication codes. But, it's not the case of 2FA-Auth. Above the QR-Code, there's a link which says "enter this text code".

Click on it and a message box (like this one) is going to appear.
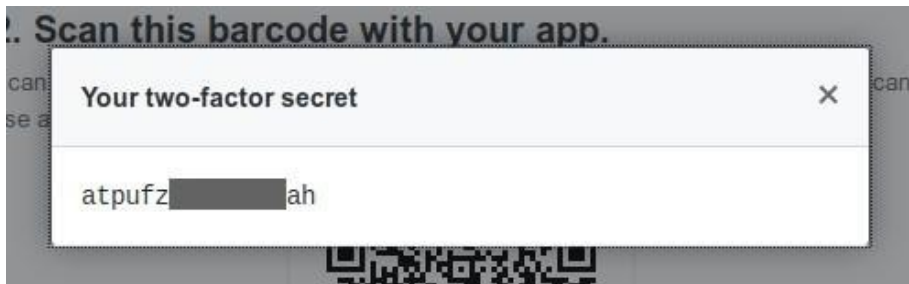
Select and copy (Ctrl + C) this token. Now, go back to your terminal, and in the 2FA-Auth's main menu, select the option "[1] Add a new 2FA token in your profile". On the next screen, type the service name (in this case "github" without quotes), press <ENTER>. In the next field, paste your GitHub's token (Ctrl + Shift + V) and press <ENTER>.

As you can see, your token has been add, but it's not finished yet. Press any key, and you'll be back to main menu. Select the option "[5] Generate 2FA codes in order to login on a site/service", and 2FA-Auth is going to generate your 2FA authentication code. For this step, 2FA -Auth will ask for a password, which is your GnuPG key's password. Why this? Because 2FA-Auth is decrypting your 2FA token to use it with OATHTOOL and generate your authentication code.

Copy your code, go back to GitHub's page and paste it in the box bellow the QR-Code (it's a box which says "Enter the six-digit code from the application"). As soon as you click in the button "Enable", you'll see a screen which confirms that your "two-factor authentication" is enabled.
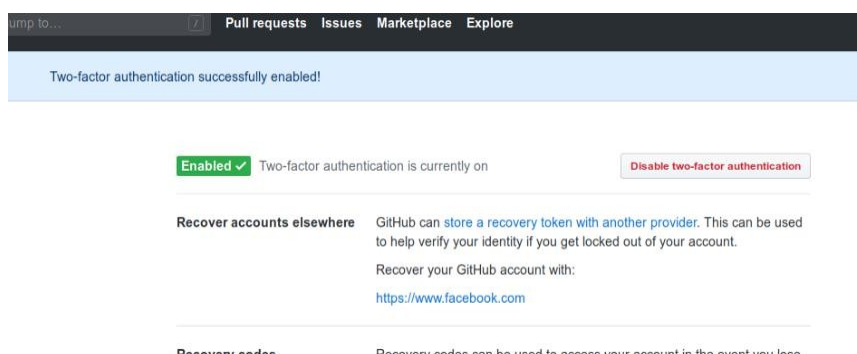
From now on, every single time you want to login using 2FA, open the browser, type your credentials (username and password), and, once the site or service asks for an authentication code (2FA code), open 2FA-Auth, select the option [5] in the main menu, and copy-paste your authentication code.

You can add more than one 2FA token, delete a old one that you're not using, list all available tokens, export them all (useful when you want to use the same token into different apps/scripts) and backup/restore your 2FA tokens and personal information (GnuPG UserID and KeyID).

Two things this project does **NOT** do: backup (export) your GPG keys and disable (over the Internet) your 2FA options in your service account.
If you want to backup (export) your GnuPG keys, read the article mentioned in the beginning of this manual (GnuPG: how to export, import, revoke and delete your keys). And if you want to cancel/disable 2FA option from your account (GitHub for example) go to your account, "Settings" > "Security" and than disable 2FA from your account. For GitHub, look for "Two-factor authentication" and point your mouse over the button "Enabled". You will see that it's going to turn red and the word "Enabled" becomes "Disable".

After click on this button, wait until the page reloads and check if 2FA was disabled from your account. Now, go back to your terminal, go to 2FA-Auth's main menu and select the option "[2] Delete one or all available tokens (be careful!)". And the message really means to be careful!!!

A list with all available 2FA tokens will appear. Select the token you want to remove and confirm with "y". If your want to delete all tokens, instead of use numbers to select an option, press "A" delete all tokens (and confirm with "y").

If you want to list your tokens, press "3" in the main menu.

If you're using different code generators, you need to use the same token for all apps or scripts that provide you a "two-factor authentication" code. So, in the main menu, select the option number "[4] Export all 2FA tokens to use them in another app/service" and wait until all your tokens get exported.

```
========================
2FA-Auth // Export token
========================

Exporting your tokens! Please, wait...

SUCCESS! Your tokens were exported to 2fa-tokens.txt!

Press any key to continue... ▮
```

A best practice is to keep a backup file (security copy) of your 2FA-Auth configuration. The option "6" in the main menu creates a backup file, while the option "7" restore your configuration.

```
========================
2FA-Auth // Config backup
========================

Saving your config in '/home/sysadmin/2fa-config-backup.tar'...
SUCCESS! Backup file created with your config!

Press any key to continue... ▮
```

```
========================
2FA-Auth // Config backup
========================

Restoring your config from '/home/sysadmin/2fa-config-backup.tar'...

Restoring your configuration files, MAYBE you can replace a working
token by a previous one which isn't working anymore. It's up to you
to decide: keep the 'old' config or restore/overwrite your files?

Would you like to continue and overwrite your config? [y/N] y
SUCCESS! Configuration restored! Restart 2FA-Auth!

Press any key to continue... ▮
```

Once you restore your 2FA -Auth configuration, like the message above says, restart 2FA-Auth (quit the script and run it again). Finally, the last option (option "I" in the main menu) is going to show you the information about 2FA-Auth and your GnuPG IDs.

```
========================
2FA-Auth // Information
========================

Version.............: v1.0-0
Description.........: Generate '2FA' codes in your terminal
Software license....: GNU GPL (General Public License) v3.0
Created by..........: Vinicius de Alencar (alencc1986)

Your GnuPG UserID...: vinicius          @protonmail.com
Your GnuPG KeyID....: 7791      93D4

Press any key to continue... ▮
```