Barrenero Documentation

Release latest

José Antonio Perdiguero López

Feb 14, 2022

Installation

CHAPTER 1

Overview

A set of services and tools for effective mining cryptocurrencies.

This projects aims to create a platform to develop and use cryptocurrency miners such as Ether, Storj… The main goal is to provide a flexible and robust set of services and tools for effective mining cryptocurrencies and performs real time checks over these miners.

Barrenero consists of following services:

Miner Tools and scripts for mining cryptocurrencies. Code can be found in different repositories: ether, storj.

API REST API for interacting with Barrenero. Code can be found in this repository.

Telegram Telegram bot for Barrenero that serves information and provides interactive methods through Barrenero API. Code can be found in this repository.

Telegraf Extension for Barrenero that harvests information and send it using Telegraf. Code can be found in this repository.

Code repository can be found in GitHub.

1.1 Help us Donating

This project is free and open sourced, you can use it, spread the word, contribute to the codebase and help us donating:

Ether 0x04BE4C8b74d2205b5fE2a31Ca18C670765feac7c

ADA addr1qxe963ree0zmdxtqypl7uvhtuvuzlxq6vrzm0lsrfsu53lffcradg5rrhf6q2wsuae4l4hrm8trlk278awztt82j8slsqz8uz5

Bitcoin 1Jtj2m65DN2UsUzxXhr355x38T6pPGhqiA

PayPal barrenerobot@gmail.com

1.2 Requirements

- Python 3.5 or newer. Download from official python site.
- Docker. Install following docker doc.

1.2.1 Installation

1. Install services:

```
sudo ./make install
```

2. Configure barrenero services as explained *here*.

1.2.2 Update

1. Update services:

```
sudo ./make update
```

1.2.3 Configuration

To properly configure Barrenero you must configure each service that composes it.

Nvidia Overclock

Self-explained configuration parameters in *nvdia.cfg* file.

Miner

Self-explained configuration parameters in *miner/<currency>.cfg* file.

API

Configuration explanation can be found *here*.

Telegram

Self-explained configuration parameters in *telegram/setup.cfg* file.

Telegraf

Self-explained configuration parameters in *telegraf/setup.cfg* file.

1.2.4 Contribute

Barrenero project is open to contributions, so if you find anything that can be improved, a bug that can be fixed or simply adds a new functionality, feel free to create a merge request or an issue.

1.2.5 Donate

This project is free and open sourced, you can use it, spread the word, contribute to the codebase and help us donating:

Ether	0x566d41b925ed1d9f643748d652f4e66593cba9c9
Bitcoin	1Jtj2m65DN2UsUzxXhr355x38T6pPGhqiA
PayPal	barrenerobot@gmail.com

1.2.6 Miner

Overview

This service aims to create a platform that provides an easy way of adding miners for different cryptocurrencies, isolating each miner into a docker container, easy to build, update and independent of the system.

Miners currently supported:

- Ether) based on *ethminer*.
- Storj).

1.2.7 API

Configuration

To properly configure Barrenero API you must define the following keys in *.env* file:

Django Secret Key

Put the Django secret key in *DJANGO_SECRET_KEY* variable.

More info here.

API superuser password

To create an API superuser password that allows users to do actions such restarting services you must define a password and encrypt it using Django tools:

```
from django.contrib.auth.hashers import make_password

password = make_password('foo_password')
```

You should put the result in *DJANGO_API_SUPERUSER* variable.

Etherscan token

Put your Etherscan API token in *DJANGO_ETHERSCAN_TOKEN* variable.

More info here.

Ethplorer token

Put your Ethplorer API token in *DJANGO_ETHPLORER_TOKEN* variable.

More info here.

Resources

These are the resources exposed by Barrenero API.

Register User

Register a new user providing username, password, account and API password.

Request

URL

/api/v1/auth/register

Parameters

Username Name to register user

Password Password to register user

Account Ethereum wallet account, starting with *0x*.

API Password Password used to identify as a API superuser, that gives access to methods such restarting services.

Response

```
{
    "username": string,
    "account": string,
    "is_api_superuser": bool,
    "token": string
}
```

Login User

Retrieve user token and user given username and password.

Request

URL

/api/v1/auth/user

Parameters

Username Name to register user

Password Password to register user

Response

```
{
  "username": string,
  "account": string,
  "is_api_superuser": bool,
  "token": string
}
```

Barrenero Status

Retrieve graphic cards and services status.

Request

URL

/api/v1/status/

Headers

Authorization Token <auth_token>

Response

```
{
  "graphics": [
    {
      "id": int,
      "power": float,
      "fan": int,
      "gpu_usage": int,
      "mem_usage": int,
      "gpu_clock": int,
      "mem_clock": int
```

(continues on next page)

(continued from previous page)

```
    }
  ],
  "services": [
    {
      "name": "Ether",
      "status": "active/inactive"
    },
    {
      "name": "Storj",
      "status": "active/inactive"
    }
  ]
}
```

Restart Service

Restart a Barrenero service giving the name.

Request

URL

/api/v1/restart/

Headers

Authorization Token <auth_token>

Parameters

name Barrenero service name to restart

Response

```
{
  "name": "Ether",
  "status": "restarted"
}
```

Ether Miner Status

Retrieve Ether miner status.

Request

URL

/api/v1/ether/

Headers

Authorization Token <auth_token>

Response

```
{
  "status": "active/inactive",
  "hashrate": [
    {
      "graphic_card": int,
      "hashrate": float
    }
  ],
  "nanopool": {
    "balance": {
      "confirmed": float,
      "unconfirmed": float
    },
    "hashrate": {
      "current": float,
      "one_hour": float,
      "three_hours": float,
      "six_hours": float,
      "twelve_hours": float,
      "twenty_four_hours": float
    },
    "workers": [
      {
        "id": string,
        "hashrate": float
      }
    ],
    "last_payment": {
      "date": string,
      "hash": string,
      "value": float,
      "confirmed": bool
    }
  }
}
```

Storj Miner Status

Retrieve Storj nodes status.

Request

URL

/api/v1/storj/

Headers

Authorization Token <auth_token>

Response

```
[
  {
    "id": string,
    "status": string,
    "config_path": string,
    "uptime": string,
    "restarts": int,
    "peers": int,
    "offers": int,
    "data_received": int,
    "delta": int,
    "port": int,
    "shared": string,
    "shared_percent": int,
    "response_time": float,
    "reputation": int,
    "version": string
  }
]
```

Ethereum Wallet

Wallet status, including balance for each token and last transactions.

Request

URL

/api/v1/wallet/

Headers

Authorization Token <auth_token>

Response

```
{
  "tokens": {
    "TOKEN_SYMBOL": {
      "name": string,
      "symbol": string,
      "balance": float,
      "price_usd": float,
      "balance_usd": float
    }
  },
  "transactions": [
    {
      "token": {
        "name": string,
        "symbol": string
      },
      "hash": string,
      "source": string,
      "destination": string,
      "value": float,
      "timestamp": string
    }
  ]
}
```

Overview

API code can be found in this repository.

This service defines a lightweight REST API on top of Barrenero Miner, providing an easy and simple way to interact with all miners. This API exposes methods for:

- Query current machine status, such as active services, GPU stats…
- Query Ether miner and pool status.
- Restart Ether miner service.
- Query Storj miner status.
- Restart Storj miner service.
- Query Ethereum wallet value and last transactions.

1.2.8 Telegram

Overview

Telegram code can be found in this repository.

Telegram bot for Barrenero that serves information and provides interactive methods through Barrenero API.

This bot provides a real time interaction with Barrenero through its API, allowing a simple way to register an user in the API and link it to a Telegram chat. Once the registration is done, it’s possible to query for Barrenero status, restart services and performs any action allowed in the API.

1.2.9 Telegram

Overview

Telegraf code can be found in this repository.

Extension for Barrenero that harvests information and send it using Telegraf.

This extension provides an automatic way of harvesting Barrenero status through its API and send it through Telegraf.

CHAPTER 2

Indices and tables

- genindex
- modindex
- search