Introduction

Over the past 9 years GitHub has quickly become the largest host of source code in the world, managing nearly 57 million repositories and 100 million pull requests for over 26 million users. (GitHub et. al. 2017) As the open source movement continues to gain momentum so to does the number of individual contributors to these repositories. In some cases these repositories had over 10,000 contributors. (GitHub et. al. 2016)

At the same time, the emergence of cryptographic networks and assets, such as Ethereum, has created new protocols for sending and managing value. Ethereum is a cryptographic network for running distributed programs; allowing users of Ethereum to send peer-to-peer (P2P) transactions and interact with smart contracts deployed on the global network.

Ethereum smart contracts are software applications written in a high-level scripting language and compiled into byte code to be run on a version of the Ethereum Virtual Machine (EVM). The EVM interprets the byte code instruction set and translates the program into machine code to be executed. (Buterin et. al. 2017)

GitToken combines the work flows of Git version control system leveraged by GitHub's web-based source code management platform and the Ethereum network to provide a set of open-source software tools and programs to enable any GitHub user to issue their own ERC20 tokens to incentivize and reward contributors, and monitor the fundamentals of their projects by integrating token generation with git contributions.

Contributions are mapped to GitHub web hook events, and include but are not limited to, creating issues, committing code, merging branches, forking repositories, and reviewing code. GitToken provides a Docker image to configuring and deploy a server for listening to GitHub web hook events.

Contributors receive tokens through interacting with an organization that has configured a GitToken server instance and has setup a GitHub web hook. When a contributor creates a new event, her/his GitHub login username is provided in the web hook request. Each username is mapped to an Ethereum address in the GitToken contract.

Contributors verify their GitHub identity by authenticating into GitHub using their Open Authorization (OAuth) token credentials. Contributors authenticate themselves with the GitToken token contract using the GitToken server authentication URL associated with an organization. If a contributor has not yet verified their identity with the contract, their contribution rewards will be held by the contract until their identity has been verified.

The Ethereum ecosystem has adopted a de facto contract interface for transacting value on top of the Ethereum network, the ERC20 protocol. The ERC20 protocol allows tokens to be exchanged over-the-counter (OTC) with private parties using Ethereum contracts. While the standard is still evolving, many developers have used the ERC20 token to represent utility or rights in their projects and have offered tokens to the public to raise funding for open source development. (Aitken 2017)

GitToken will offer GTK tokens to represent contributions made to the organization's GitHub repositories. A portion of tokens issued are automatically auctioned to bidders by the GitToken contract upon each event.

GitToken Services

Web API Server

GitToken provides an Express Node.JS application for handling HTTP post requests from GitHub web hook services.

The GitToken server consists of composable GitToken Node libraries and packages; including an Express middleware library that can be used in existing Express applications.

Additionally, GitToken provides a Dockerfile and Docker image to quickly deploy a GitToken server instance, using an environmental variables file and Docker Compose (`docker-compose up --build -d`) to deploy the server.

Deploying with Docker

> Docker provides a way to run applications securely isolated in a container, packaged with all its dependencies and libraries.

Docker uses Linux containers to define a standard environment for an application to run within.

GitToken provides a Docker image and Dockerfile for deploying a GitToken server instance. The following sections provide an overview of using `docker-machine` and `docker-compose` command line tools with the GitToken Docker server image.

Machine

`docker-machine` is a command line interface (CLI) tool provided by Docker to deploy containers and images on remote servers and virtual machines.

Using `docker-machine` with `docker-compose` is the preferred method for setting up a GitToken server instance.

Configuring a Docker Machine

Configuring a Docker machine requires a remote server to deploy the GitToken server to. The following example demonstrates configuring a basic generic (host agnostic) server.

```
# Create a new docker machine
docker-machine create \
  # Specify the `generic` driver
  --driver generic \
  # Supply the public network IP address of the machine
  --generic-ip-address=___.___.___.___ \
  # Provide the path to the private SSH key of the
  # current user for authenticating into the machine
  --generic-ssh-key ~/.ssh/id_rsa \
  # Name the machine
  machine_name
```

Dockerfile

```
# Example Dockerfile

# Use NodeJS Docker image
FROM node:6.11.0

# Install yarn to quickly install cached dependency files
RUN npm i -g yarn

# Run the following commands inside the gittoken-server directory
WORKDIR /gittoken-server

# Clone the git-token/express-server repository
RUN git clone https://github.com/git-token/express-server.git .

# Install dependencies
RUN yarn install

# Build source files
RUN yarn run build-src

# Start the GiToken Server
ENTRYPOINT yarn run start

# Expose port 1324 for the express application;
# Expose port 1325 for the web socket server.
EXPOSE 1324 1325
```

Docker Image

Docker images are cached snapshots of built Docker containers. They provide a minimal amount of data, which may be composed of immutable layers of cached snapshot data from other images, to form a container environment runnable by docker machines.

GitToken provides public Docker images for services on Docker Hub.

Using an Image

GitToken provides a NodeJS Express application for handling GitHub web hook events and distributing tokens for git contributions.

From within a Docker machine, running the CLI `docker pull gittoken/express-server` will grab the GitToken `express-server` image for running a GitToken server instance.

The image is intended to be run with `docker-compose` where an environment variables (`.env`) file is provided to the `docker-compose.yml` `env_file` field.

Environment Variables File

The `.env` file customizes the values for the GitToken server instance.

```
# Example environment variables (.env) file

# NOTE Web3 Provider must be a public IP address,
# and not associated with localhost.
#
# i.e. The IP address must not be 127.0.0.1 or 0.0.0.0
#
# If a Docker container is running an Ethereum client on the machine,
# use the IP Address associated with that container's public network IP add
ress
# or the bound port of the service on the machine.

WEB3_PROVIDER                  = "http://___.___.___.___:8545"

IS_GITHUB_WEBHOOK              = true

GITTOKEN_DIRECTORY_PATH        = "/gittoken-server"

GITTOKEN_KEYSTORE_FILENAME     = ".keystore"

GITTOKEN_CONTRACT_FILE         = "contract.json"

GITTOKEN_FAUCET_ACTIVE         = false

GITTOKEN_CONTRACT_OWNER        = "0x00000000000000000000000000000000000000000
0"

GITTOKEN_CONTRACT_OWNER_EMAIL  = "your_email@your_organization.website"

GITTOKEN_CONTRACT_ORGANIZATION = "Your Organization"

GITTOKEN_CONTRACT_SYMBOL       = "SYMBOL"

GITTOKEN_CONTRACT_DECIMALS     = 8

GITTOKEN_API_SESSION_SECRET    = 'SOMETHINGFANTASTIC'

GITHUB_API_ID                  = "YOUR_GITHUB_APPLICATION_CLIENT_ID"

GITHUB_API_SECRET              = "YOUR_GITHUB_APPLICATION_CLIENT_SECRET"

GITHUB_CALLBACK_URL            = "https://your_organization.website/auth/gi
thub/callback"
```

Environment Variables File

The `.env` file customizes the values for the GitToken server instance.

```
# Example environment variables (.env) file

# NOTE Web3 Provider must be a public IP address,
# and not associated with localhost.
#
# i.e. The IP address must not be 127.0.0.1 or 0.0.0.0
#
# If a Docker container is running an Ethereum client on the machine,
# use the IP Address associated with that container's public network IP add
ress
# or the bound port of the service on the machine.

WEB3_PROVIDER                   = "http://___.___.___.___:8545"

IS_GITHUB_WEBHOOK               = true

GITTOKEN_DIRECTORY_PATH         = "/gittoken-server"

GITTOKEN_KEYSTORE_FILENAME      = ".keystore"

GITTOKEN_CONTRACT_FILE          = "contract.json"

GITTOKEN_FAUCET_ACTIVE          = false

GITTOKEN_CONTRACT_OWNER         = "0x00000000000000000000000000000000000000000
0"

GITTOKEN_CONTRACT_OWNER_EMAIL   = "your_email@your_organization.website"

GITTOKEN_CONTRACT_ORGANIZATION = "Your Organization"

GITTOKEN_CONTRACT_SYMBOL        = "SYMBOL"

GITTOKEN_CONTRACT_DECIMALS      = 8

GITTOKEN_API_SESSION_SECRET     = 'SOMETHINGFANTASTIC'

GITHUB_API_ID                   = "YOUR_GITHUB_APPLICATION_CLIENT_ID"

GITHUB_API_SECRET               = "YOUR_GITHUB_APPLICATION_CLIENT_SECRET"

GITHUB_CALLBACK_URL             = "https://your_organization.website/auth/gi
thub/callback"
```

Compose

`docker-compose` is a command line interface (CLI) tool for composing docker service deployments.

Deploying a GitToken server instance is intended to be done using a `docker-compose.yml` file to define the configuration of the service.

Use the `gittoken/express-server:v1` Docker image to build the GitToken service instance from.

Configuration parameters of a GitToken server instance are passed to the container using an environment variables file.

GitToken server instance requires both ports 1324 and 1325 to be publicly exposed and available to the host machine. The host machine may map the default ports to different ports, according to the needs of the developer. For example, the developer may instead map ports `3000:1324` to bind the service to port 3000 on the host machine.

```yaml
# Example docker-compose.yml configuration using the
# GitToken express-server Docker image

# Use docker-compose v3
version: '3.0'
# Define services
services:
  # Define gittoken service
  gittoken:
    # Use the GitToken Docker image to build from
    image: "gittoken/express-server:v1"
    # Define an environment variables file path
    # relative to the Docker machine file system
    env_file:
      - /gittoken-server/gittoken.env
    # Expose port 1324 for the Express Server
    # Expose port 1325 for the WebSocket Server
    ports:
      - 1324:1324
      - 1325:1325
    # Mount the local volume of the Docker machine
    # to the container volume
    volumes:
      - /gittoken-server/:/gittoken-server/
```

Ethereum Smart Contracts (`Solidity`)

GitToken.sol

GitTokenLib.sol

Token Auctioneering

Ethereum ERC20 tokens have recently provided a new mechanism for funding projects. While tokens have brought liquidity to start-ups, it has also brought mis-pricing and speculation.

Many of the projects that have offered tokens to build their projects have used Git and GitHub to manage and develop software collaboratively, yet independently.

There is a clear relationship between the number of git contributions a project has and the market capitalization of the project.

GitToken Exchange

test

Evaluation of Open Source Projects

```
TESTER = document.getElementById('test')
Plotly.plot(TESTER, [{
  x: [38438497236, 18401040701, 1376206596, 235578951, 228864420, 87398581, 56
23927],
  y: [14409, 8680, 8299, 3958, 258, 1492, 706],
  name: 'Contributions',
  text: ['Bitcoin', 'Ethereum', 'Ethereum Classic', 'Golem', 'Gnosis', 'Sta
tus', 'Aragon'],
  textposition: 'top center',
  marker: {
    size: [38.43, 18.40, 1.37, .23, .22, .08, .05]
  },
  mode: 'markers+text'
}],{
  height: 600,
  margin: { t: 100, l: 50 },
  title: 'Git Commits Vs. Market Capitalization'
});
```

GitToken maps the total supply of the token to contributions made to the project.

GitHub Integrations

GitToken provides a Docker image and Dockerfile for configuring a NodeJS Express application for listening to incoming GitHub contribution events via HTTP POST requests made by an organization's GitHub webhook service.

Additionally, the GitToken Express application establishes Open Authorization (OAuth) endpoints for authenticating contributors with their GitHub login credentials and verifying the contributor with the GitToken contract instance.

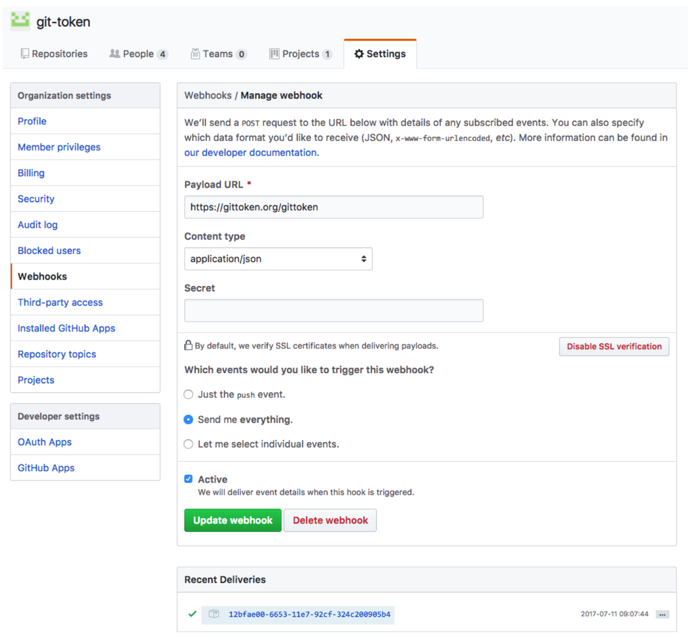
Request data is parsed and signed by the GitToken middleware handler, and sent to the GitToken contract to create and distribute tokens to contributors.

GitHub Webhook Events

Configuring a Webhook

> We'll send a POST request to the URL below with details of any subscribed events. You can also specify which data format you'd like to receive. More information can be found in our developer documentation.

Under the **settings** tab in an organization's GitHub dashboard, click **Webhook** on the left navigation section and add a new webhook.



Webhooks / **Manage webhook**

We'll send a `POST` request to the URL below with details of any subscribed events. You can also specify which data format you'd like to receive (JSON, `x-www-form-urlencoded`, *etc*). More information can be found in our developer documentation.

Payload URL *

https://gittoken.org/gittoken

Content type

application/json

Secret

By default, we verify SSL certificates when delivering payloads.

Which events would you like to trigger this webhook?

- Just the `push` event.
- [✓ Send me **everything**.]
- Let me select individual events.

[✓] **Active**
We will deliver event details when this hook is triggered.

Recent Deliveries

✓ `12bfae00-6653-11e7-92cf-324c200905b4` 2017-07-11 09:07:44

GitToken, by default, sets the webhook endpoint to be `/gittoken`. This endpoint is customizable in the configuration file of the GitToken Docker service.[1]

Enter the url of the organization's GitHub webhook endpoint in the `payload URL` field of the webhook settings page. This is the endpoint that will receive POST requests when a contribution is made to any of an organizations' repositories.

Additionally, the organization can filter for which events they would like to receive from the web hook service. The organization may wish to listen only for `push` events, or they may wish to receive individual events, or all events. GitToken provides event handling for all event types and recommends listening for all events.

Ping Event

The `ping` event is the first event sent by the GitHub webhook service. Its purpose is to test the endpoint configuration and, in the context of GitToken, establish the keystore and contract for the GitToken server.

The webhook service will display a checkmark if the endpoint responds with a `200` success status. A successful response will include JSON data about the keystore account, contract creation transaction receipt, and current details about the Ethereum blockchain the GitToken server is connected to.

```
{
  "accounts": { ... },
  "contract": { ... },
  "blockchain": { ... }
}
```

Otherwise, the webhook service will display an error message with either a `400` or `500` error status.

Upon receiving a ping event, the GitToken server checks if a keystore and GitToken contract already exist. If either does not exist, the GitToken server attempts to create the keystore and deploy a contract with the configured parameters provided to the GitToken server instance.

One common reason a contract may not deploy may be due to inadequate funds in the Ethereum account tied to the keystore. For the purpose of the GitToken alpha on the Ropsten testnet a faucet can be used to provide the minimum amount necessary to create a contract.

The server will respond with an error message if the contract could not be created.

Event Types & Reward Values

The following events are monitored by the GitHub webhook. These events can be filtered on the webhook setting page for submitting POST requests to the endpoint GitToken service.

Additionally, the following reward values can be set by the contract owner by submitting a signed transaction to the `setRewardValue` method of the contract.

```
/**
 * @dev GitToken contract owner(s) may set individual reward values for
 * contribution events
 * @param _rewardValue uint256 Value of reward type
 * @param _rewardType string Name of reward type
 * @returns success bool Returns true after successfully updating the
 * contract
 */
function setRewardValue(
  uint256 _rewardValue,
  string _rewardType
) onlyOwner public returns (bool success) {
  gittoken.rewardValues[_rewardType] = _rewardValue;
  return true;
}
```

Event	**Reward Value**	**Description**
*	0	Any time any event is triggered (Wildcard Event).
commit_comment	250	Any time a Commit is commented on.
create	2500	Any time a Branch or Tag is created.
delete	0	Any time a Branch or Tag is deleted.
deployment	5000	Any time a Repository has a new deployment created from the API.
deployment_status	100	Any time a deployment for a Repository has a status update from the API.

fork	5000	Any time a Repository is forked.
gollum	250	Any time a Wiki page is updated.
installation	250	Any time a GitHub App is installed or uninstalled.
installation_repositories	1000	Any time a repository is added or removed from an installation.
issue_comment	250	Any time a comment on an issue is created, edited, or deleted.
issues	500	Any time an Issue is assigned, unassigned, labeled, unlabeled, opened, edited, milestoned, demilestoned, closed, or reopened.
label	100	Any time a Label is created, edited, or deleted.
marketplace_purchase	0	Any time a user purchases, cancels, or changes their GitHub Marketplace plan.
member	1000	Any time a User is added or removed as a collaborator to a Repository, or has their permissions modified.
membership	1000	Any time a User is added or removed from a team. Organization hooks only.
milestone	15000	Any time a Milestone is created, closed, opened, edited, or deleted.
organization	1000	Any time a user is added, removed, or invited to an Organization. Organization hooks only.
org_block	0	Any time an organization blocks or unblocks a user. Organization hooks only.
page_build	500	Any time a Pages site is built or results in a failed build.

project_card	250	Any time a Project Card is created, edited, moved, converted to an issue, or deleted.
project_column	50	Any time a Project Column is created, edited, moved, or deleted.
project	1000	Any time a Project is created, edited, closed, reopened, or deleted.
public	10000	Any time a Repository changes from private to public.
pull_request_review_comment	250	Any time a comment on a pull request's unified diff is created, edited, or deleted (in the Files Changed tab).
pull_request_review	100	Any time a pull request review is submitted, edited, or dismissed.
pull_request	1000	Any time a pull request is assigned, unassigned, labeled, unlabeled, opened, edited, closed, reopened, or synchronized (updated due to a new push in the branch that the pull request is tracking). Also any time a pull request review is requested, or a review request is removed.
push	1000	Any Git push to a Repository, including editing tags or branches. Commits via API actions that update references are also counted. This is the default event.
repository	2500	Any time a Repository is created, deleted (organization hooks only), made public, or made private.
release	5000	Any time a Release is published in a Repository.
status	200	Any time a Repository has a status update from the API

team	2000	Any time a team is created, deleted, modified, or added to or removed from a repository. Organization hooks only
team_add	2000	Any time a team is added or modified on a Repository.
watch	500	Any time a User stars a Repository.

GitHub Open Authorization (OAuth)

To receive tokens, contributors must verify her/his identity with the organization's GitToken contract.

The GitToken NodeJS Express application provides a GitHub OAuth endpoint for contributors to authenticate themselves with the organization's GitToken contract instance.

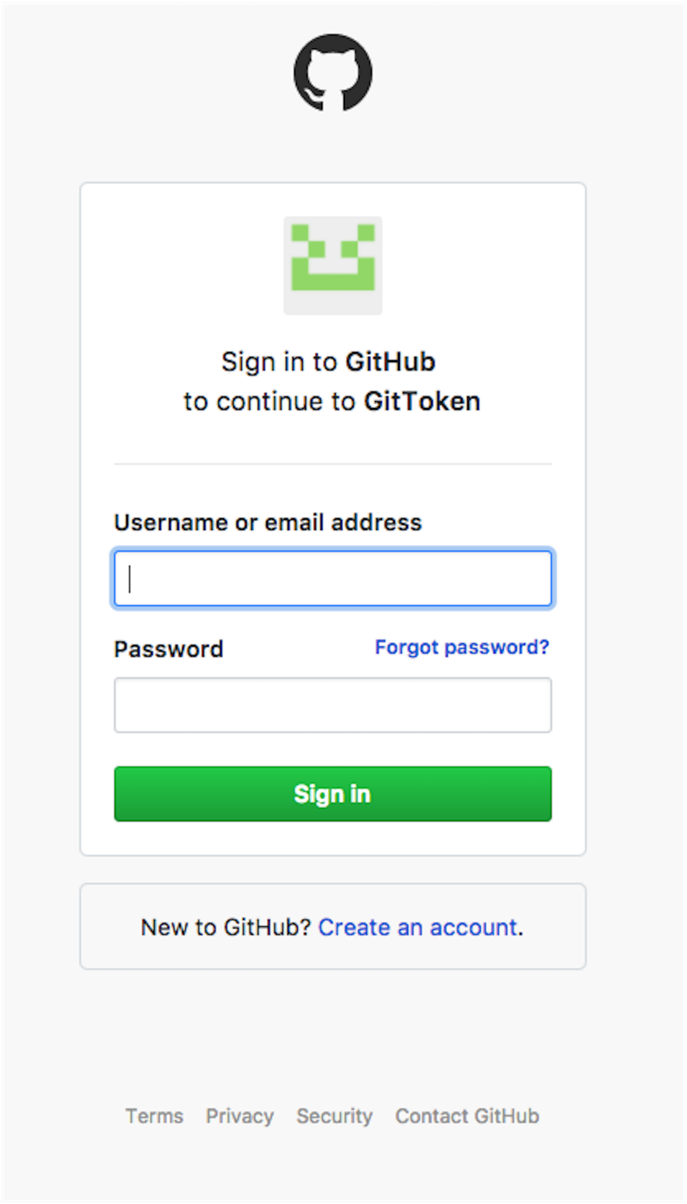
Additionally, GitToken provides a web application user interface to enable contributors to easily verify their identity.

By default authentication requests are handled by the GitToken server service at the `/auth/github` endpoint.[2]

A contributor verifies her/his identity by associating an Ethereum address to her/his OAuth GitHub credentials using the provided GitToken web application or desktop user interface(s).

The contributor's GitHub login session is stored by the GitToken Express application.

Using GitHub's OAuth strategy to authenticate contributors enables the


Sign in to GitHub
to continue to GitToken
Username or email address
Password
Forgot password?
Sign in
New to GitHub? Create an account.
Terms Privacy Security Contact GitHub

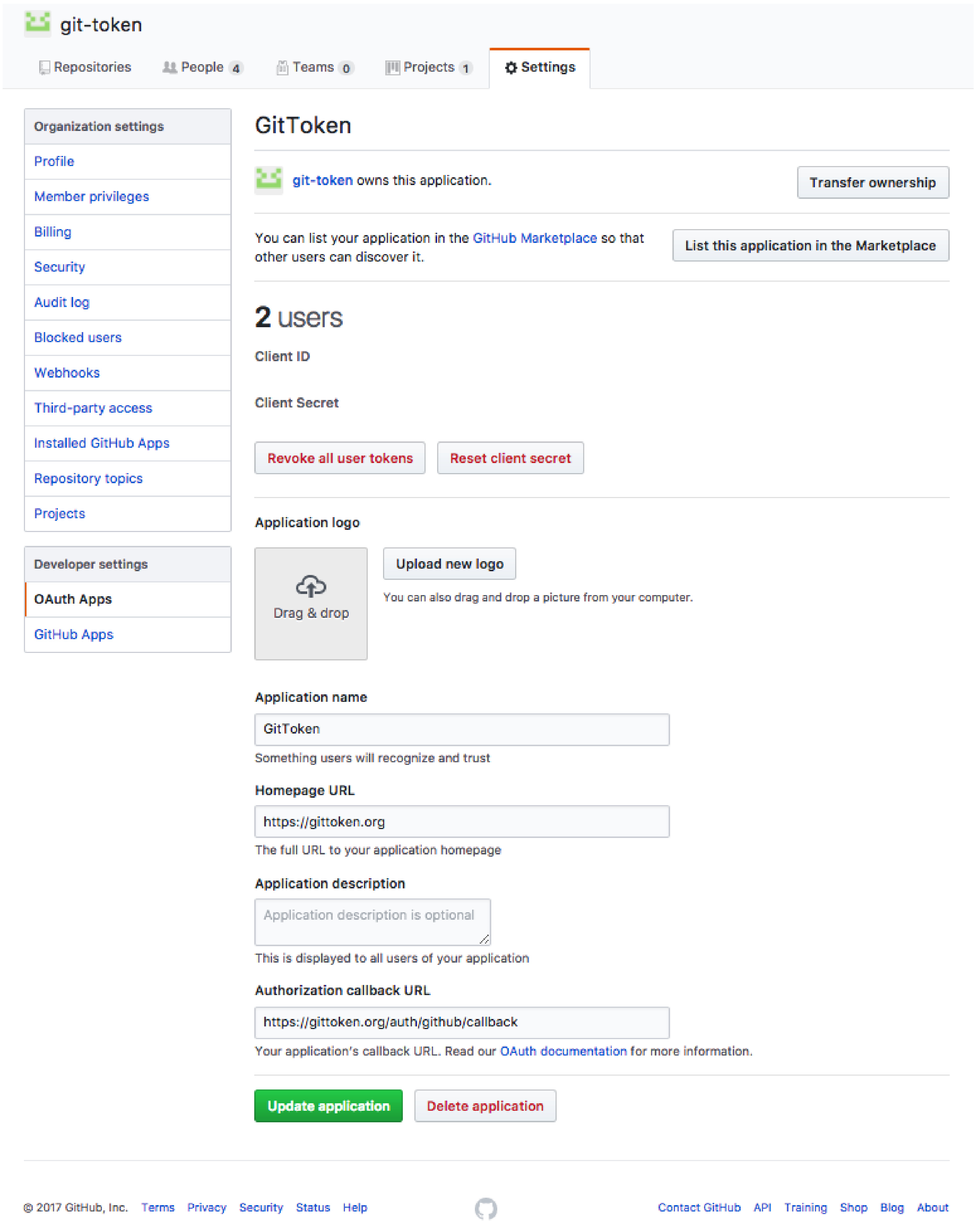
GitToken services to map the Ethereum address of the contributor to the contributor's GitHub username, thereby successfully configuring the service to distribute tokens to the contributor when GitHub web hook events are triggered.

Configure a GitHub OAuth Application

To configure GitHub Open Authorization for a GitToken server instance, click **settings** under the organization's main GitHub landing page. Select **OAuth Apps** on the left side navigation section.

Create a new OAuth application for the GitToken application. The configuration requires an **Authorization Callback URL**. This URL is provided by the GitToken server instance at the endpoint `/auth/github/callback`.[3]

Enter the full callback URL for the organization in the provided field. For example, https://your_organization.website/auth/github/callback


git-token

Repositories | People 4 | Teams 0 | Projects 1 | Settings

Organization settings
- Profile
- Member privileges
- Billing
- Security
- Audit log
- Blocked users
- Webhooks
- Third-party access
- Installed GitHub Apps
- Repository topics
- Projects

Developer settings
- OAuth Apps
- GitHub Apps

GitToken

git-token owns this application. [Transfer ownership]

You can list your application in the GitHub Marketplace so that other users can discover it. [List this application in the Marketplace]

2 users

Client ID

Client Secret

[Revoke all user tokens] [Reset client secret]

Application logo

Drag & drop [Upload new logo]

You can also drag and drop a picture from your computer.

Application name

GitToken

Something users will recognize and trust

Homepage URL

https://gittoken.org

The full URL to your application homepage

Application description

Application description is optional

This is displayed to all users of your application

Authorization callback URL

https://gittoken.org/auth/github/callback

Your application's callback URL. Read our OAuth documentation for more information.

[Update application] [Delete application]

© 2017 GitHub, Inc. Terms Privacy Security Status Help Contact GitHub API Training Shop Blog About

References

Aitken. 2017. “Gnosis' Prediction Market Scores $12.5M in 'Record-Breaking' Crypto Auction.” *Https://Www.forbes.com/Sites/Rogeraitken/2017/04/24/Gnosis-Prediction-Market-Scores-12-5m-in-Record-Breaking-Crypto-Auction/#3afec93ce87d*. Accessed July 11.

Buterin et. al. 2017. “A Next-Generation Smart Contract and Decentralized Application Platform.” *Https://Github.com/Ethereum/Wiki/Wiki/White-Paper*. Accessed July 12.

GitHub et. al. 2017. “Celebrating Nine Years of GitHub with an Anniversary Sale.” *Https://Github.com/Blog/2345-Celebrating-Nine-Years-of-Github-with-an-Anniversary-Sale*. Accessed July 12.

———. 2016. “The State of the Octoverse 2016.” *Https://Octoverse.github.com/*. Accessed December 31.

---

1. e.g., the webhook endpoint for GitToken's repositories is `https://GitToken.org/gittoken` ↩
2. e.g., The authentication endpoint for GitToken's GTK contract is `https://GitToken.org/auth/github`.↩
3. e.g., The authorization callback url endpoint for GitToken's GTK contract is `https://GitToken.org/auth/github/callback`.↩