GruPur Platform

Release 0.0.2a1

Sep 02, 2020

Table of Contents

CHAPTER 1

Description

GruPur is a free open-source decentralized social media platform focused on public group interaction, democratic moderation, issue polling, and media discussion.

The entire GruPur platform can be downloaded and modified by anyone, there is no *central hosting place* for the GruPur network so it does not rely on any one company or individual to host the data. This keeps any entity from attempting to sabatoge our network with DDoS attacks, database hacks or code exploits. Everyone is entitled to everything that GruPur is and everyone is responsible for holding themselves and others accountable for what they do with it.

We believe in the right to free speech as well as the individual responsibility to maintain and moderate that right.

CHAPTER 2

Technicial

The GruPur web client production environment is hosted on the IPFS network. Data can be pinned - based on a tokenomic services model - to be permanently stored on the cluster. IPFS enables decentralized web hosting and supports the management of real top level domains.

As for our database, we store all data using BigChainDB. BigchainDB allows developers and enterprise to deploy blockchain proof-of-concepts, platforms and applications with a blockchain database, supporting a wide range of industries and use cases.

CHAPTER 3

Data

GruPur utilizes a decentralized blockchain database. Any data created or stored using GruPur is immutable, meaning it can never be modified and it can never be deleted. *Every individual is responsible for their own data on GruPur*. If you lose your passphrase, you lose your ability to verify ownership of the data.

GruPur is designed to be a permenant public record. All data is accessible by anyone and can never be deleted. Data owners can append extra metadata to the assets updating values, creating the transaction chain seen in Blockchain applications.

CHAPTER 4

Disclaimer

ALL DATA ON GRUPUR IS PUBLIC AND ACCESSIBLE AS PLAIN TEXT. DO NOT SUBMIT ANY PRIVATE DATA. EVERY INDIVIDUAL IS THEMSELVES RESPONSIBLE FOR THE CONTENT THEY SUBMIT.

CHAPTER 5

Web Client

5.1 Recent Changes

- Now hosting all files on ipfs directly

To host web client on local node with ipfs do ipfs get /ipns/www.grupur.com ipfs add -r ./www.grupur.com

5.2 Hosting

Official HTTPS The official build is hosted on the IPFS Phantom dApp which is a decentralized website hosting solution. Our files are uploaded to the IPFS filemanager. The official link and currently the only way to reach the official client via HTTPS is at https://ipfs.io/ipns/www.grupur.com/

Official Domain Our official domain GruPur.com was purchased from GoDaddy and the DNS A record @ points to 184.168.131.241 **which is a centralized DNS host at GoDaddy** the root domain is forwarded to *http://www.grupur.com* which is a CNAME record pointing to gateway.ipfs.io our official domain should be considered centralized and for the most secure connection you should use the Official HTTPS domain.

Unofficial Domains To point your own domain at our web client, create a CNAME record of any name and point it to gateway.ipfs.io then create a TXT record named _dnslink.**<CNAME name>** and point it to dnslink=/ipns/www.grupur.com

At this point there are no Unofficial Domains that have been registered.

Unofficial Clients Users may download the source and modify it as they like while still choosing to use GruPur.com's official database. Applications range from secure private hosting of the client, to heavily modifiying the client to suit your needs, to experimenting with programming. If you find bugs or make improvements you would like to see implemented into the official client, you can make a pull request on the dev branch. Users can also choose a different app id from the official client's to create their own database for other communities that don't use any of the official GruPur data.

5.3 Interface

The web client interface consist of a few main areas: home, login, register, account and feed pages.

Home The home page is only seen by non-logged in users. It's purpose is to give new users a summary of what the website is about and incentive to login.

Login The login page is where users can authenticate themselves verifying their idenity using their 12 word mnemonic passphrase. This will then send them to their account page.

Register The register page is where users can obtain a randomly generated 12 word mnemonic passphrase which can be downloaded to your computer.

Account The account page when visted by the owner of the account will show your recent broadcasts and mentions as well as controls for you to modify your account. By adding the name= parameter along with a user handle you will pull up their account page showing their broadcasts, mentions and other data.

Feed The feed page by default will show all global broadcasts from the current calendar week. By adding the tag= parameter you will pull up the associated group from the tag. By adding the week= parameter you can pull up the global broadcasts from that calendar week. By adding both tags you can view just the broadcasts using a tag from a certian week. By adding the sort= parameter you can sort the post by new, top or relevant. You can pull up a single post and all replies with the post= parameter.

5.4 Code

The code is written in Javascript and HTML and hosted at BitBucket

5.5 Dependencies

- AllOrigins - We use AllOrigins to get around the Cross-Origin policy of modern browsers.
- Nude.JS - Internal nudity filter
- FontAwesome - Icons
- BigChainDB - Database storage
- IPFS - Interplanetary File System
- textance.herokuapp.com - Generate titles from websites.
- api.letsvalidate.com - Generate website thumbnails.
- nsfw.haschek.at - External nudity filter
- GoDaddy - DNS for grupur.com

CHAPTER 6

Account

6.1 Description

Accounts are public-private key pairs generated using a 12 word Mnemonic. The public key is used as the account identifier with the user name handle being the first 10 characters of the public key. The private key is used to sign transactions to the BigChainDB Blockchain assets.

An account asset is created on the register page when a user presses the "Register" button. The *doRegister()* method is called which generates a random 12 word mnemonic which is displayed to the user along with a link for the user to download the file with the passphrase.

An account asset is created with the app id, year, week, timestamp, asset type, and full public key. The metadata of the asset creation transaction consists of a description, timestamp, week and year.

Everytime an account is logged in, a self transfer transaction is created updating the metadata of the login timestamp. The public key is stored as a cookie in the browser and if the user toggled to stay logged in the private key is also stored as a cookie in the browser.

6.2 Asset Structure

Asset

```
// This is an account asset
var accountAsset = {
    appId: APP_ID, // APP_ID is a global in grupur.js
    year: "year-" + new Date().getWeekYear(),
    week: "week-" + new Date().getWeek().doubleDigitString(),
    type: ASSET_ACCOUNT, // Asset types are globals in grupur.js
    account: user.publicKey, // You will need the user public key generated from the␣
↪mnemonic
    timestamp: new Date().toString()
};
```

Creation Metadata

```
// This is the account creation metadata
var creationMetadata = {
    appId: APP_ID,
    year: "year-" + new Date().getWeekYear(),
    week: "week-" + new Date().getWeek().doubleDigitString(),
    type: DATA_LOGIN,
    description: "Account creation",
    timestamp: new Date().toString()
};
```

Login Metadata

```
// This is the account login metadata
var loginMetadata = {
    appId: APP_ID,
    year: "year-" + new Date().getWeekYear(),
    week: "week-" + new Date().getWeek().doubleDigitString(),
    type: DATA_LOGIN,
    description: "Login timestamp",
    timestamp: new Date().toString()
};
```

Account Asset

```
// This is an account asset
const accountAsset = {
    appId: APP_ID,
    year: "year-" + new Date().getWeekYear(),
    week: "week-" + new Date().getWeek().doubleDigitString(),
    type: ASSET_ACCOUNT,
    account: user.publicKey,
    timestamp: new Date().toString()
};
```

Account Data

```
// This is a login data type
const loginTimestamp = {
    appId: APP_ID,
    year: "year-" + new Date().getWeekYear(),
    week: "week-" + new Date().getWeek().doubleDigitString(),
    type: DATA_LOGIN,
    data: {
        description: "Login timestamp",
        timestamp: new Date().toString()
    }
}

// This is a user info data type
const userInfo = {
    appId: APP_ID,
    year: "year-" + new Date().getWeekYear(),
    week: "week-" + new Date().getWeek().doubleDigitString(),
    type: DATA_USER,
    data: {
        description: "User data",
```

(continues on next page)

(continued from previous page)

```
        timestamp: new Date().toString(),
        displayName: "",
        avatar: "",
        bitcoin: ""
    }
}
```

CHAPTER 7

Asset

7.1 Assets are created by the web client and are transfered to the BigchainDB network

asset An asset is a javascript object that holds static information about the asset including individual identifiers and maximum tokens available. Once created, an asset can never be deleted nor can any more tokens be created. However, ownership of the indivdual tokens can be transferred from one peer to another.

metadata Metadata is the extra data attached to the transactions of on asset. Meaning this data is new for each transfer. This is how you would update dynamic variables in an asset. To create an updated metadata, create a self-transfer transaction.

broadcast A broadcast is an asset that is created by an account. It is essentially a signed message with an array of hashtags. Any tag used in a broadcast will place that post in the group feed. Owners of group relevance can transfer their relevance assets to increase the position of a post in the global and group feeds. The author of the post will now be able to transfer their earned relevance points to other users. Each broadcast made generates one status asset which can also bre transferred to other users to increase their broadcasts positions in the top sorting.

group A group is created when a user claims a group which has not already been claimed. Searching for a hashtag or going directly to the group page will show you a feed of all broadcasts using the hashtag sorted by relevance.

relevance The creator of a group will recieve 240 relevance tokens for the group. Users with Relevance tokens can transfer those tokens to users who have posted relevant information to improve the positions of their post. Once a user has spent all of their relevance, they can only gain more by them being transferred back from the new owners.

status Status is generate once per broadcast. Users can transfer these points to other users to increase their broadcasts positions in the Top sorting pages. They can then retransfer those status points to whomever they desire.

CHAPTER 8

Broadcasts

8.1 Description

A *broadcast* is an immutable public signed message created by an account asset. Broadcasts can use group tags to be displayed in a group feed and be awarded relevance tokens from group relevance owners that find the content relevant to the group.

A *broadcast* is the core database asset that is used as consumer content for the Global, Group, and Account feeds. They are messages created through the *broadcast* area and signed by accounts. These messages can be 240 characters long, any handle characters are excluded from the character count as well as any portion of a url other than the root domain.

If a handle (**#**, @ or **>**) is used in a *broadcast* it will automatically link to the coorosponding asset. A **#** is used to tag a group or week, an @ is used to tag an account and a **>** is used to reply to another *broadcast*.

If a url is used in a *broadcast* it will be parsed, linked and stripped to the root domain. The first url in any *broadcast* will be parsed and embeded. If no content is found at the link, a screenshot of the webpage will be shown along with the title of the website.

Broadcasts will be filed under the week it was created. All broadcasts can be found in the Global Feed of their week, in all groups they tag and on the author's account page. A *broadcast* with a mention will also show up in the mentions section of their account page. Any reply to a broadcast will be shown on the individual post page when selected.

Any broadcast posted generates one status point for the author and can be used to transfer to other authors for their content.

To toggle the *broadcast* area on a compatible page, tap the bullhorn icon. Your message preview should update as you type.

8.2 Broadcast Asset

```
// This is a broadcast asset
const broadcast = {
    appId: APP_ID,
    year: "year-" + new Date().getWeekYear(),
    week: "week-" + new Date().getWeek().doubleDigitString(),
    type: ASSET_BROADCAST,
    account: user.publicKey,
    timestamp: new Date().toString()
}
```

8.3 Broadcast Data

```
// This is a message data type
const message = {
    appId: APP_ID,
    year: "year-" + new Date().getWeekYear(),
    week: "week-" + new Date().getWeek().doubleDigitString(),
    type: DATA_MESSAGE,
    data: {
        description: "Broadcast message",
        timestamp: new Date().toString(),
        message: "This is a #test #message."
    }
}
```

CHAPTER 9

Database

9.1 Data

GruPur utilizes a decentralized blockchain *database*. Any data created or stored using GruPur is immutable, meaning it can never be modified and it can never be deleted. Every individual is responsible for their own data on GruPur. If you lose your passphrase, you lose your ability to verify ownership of the data.

GruPur is designed to be a permenant public record. All data is accessible by anyone and can never be deleted. Data owners can append extra metadata to the assets updating values, creating the transaction chain seen in Blockchain applications.

9.2 Ed25519Keypair

BigchainDB JavaScript driver allows you to create a keypair based on a seed. The constructor accepts a 32 byte seed. One of the ways to create a seed from a string (e.g. a passphrase) is the one used by bip39, specifically the function mnemonicToSeed.

9.3 Mnemonic

A mnemonic is a combination of words from a word bank to be used as a seed for generation of your Private and Public Ed25519Keypair to sign transactions to the database. The word bank is comprised of 2048 words and when an account is registered for GruPur you are given 12 random words from the bank as your passkey to generate your unique KeyPair.

CHAPTER 10

Groups

Groups are created when a user visits a tag page, which has not already been claimed, and claims it for themselves. Tag pages are linked from broadcasts when a # handle is used in the message of a broadcast. On the tag page, every broadcast with the same tag in its message will be displayed in the feed. When a user claims ownership of a group, they are awarded relevance points they can transfer to other users for their quality broadcasts and increase their positions in relevance-sorted feeds.

CHAPTER 11

Handles

11.1 Description

Handles are special strings that will be parsed into links when rendering pages and posts. They can be included in broadcast messages to hot link around the site and include broadcast in other feeds like Mentions or Group feeds.

11.2 Types

There are a few different types of handles:

- # is for group or week tags. Using this in a post will make the post visible in that group but will not place the post in a tagged week
- @ is for mentioning an account this will hyperlink to their account page.
- > is for replying to a specific broadcast this will hyperlink the to the broadcast you are referencing.

11.3 Examples

When an external message is to be displayed it may look like the following.

```
var rawMessage = "Check out this #gif @aB73jMwl18 https://www.example.com/media/
↪archive/2017/10/1/this.gif";
```

All external string are sent through parseRawMessage() for handles and urls to append html code or strip it.

```
var parsedMessage = parseRawMessage(rawMessage);
body.innerHTML = parsedMessage;
```

And so the body of the page would have

```
Check out this <a href='.\feed.html?tag=gif'>#gif</a> <a href='.\account.html?
↪name=aB73jMwl18>@aB73jMwl18</a> <a href='https://www.example.com/media/archive/2017/
↪10/1/this.gif'>example.com</a>
```

11.4 Notes

For character counting purposes the html stripped parsed message is counted.

So for the above example when counting characters the algorithm will only count we get 41 since we include whitespace.

```
Check out this gif aB73jMwl18 example.com
```

CHAPTER 12

Tools

ReadTheDocs - reStructured Text.

BigChainDB - https://docs.bigchaindb.com/projects/server/en/latest/http-client-server-api.html - https://github.com/bigchaindb/js-bigchaindb-driver - https://docs.bigchaindb.com/en/latest/ - https://www.bigchaindb.com/developers/guide/tutorial-car-telemetry-app/ - https://www.bigchaindb.com/developers/guide/tutorial-rbac/

Hosting - https://ipfs.io

CHAPTER 13

Transactions

There are two different possible transactions. Asset Creation and Asset Transfer. Using these two types of transactions you can create assets you own and transfer their values to other accounts for them to transfer themselves to others. This is core to the immutability of the block-chain database that BigChainDB is.

13.1 Metadata

Metadata is data that is included in the creation and transfer transactions that allows a developer to use these transactions to store and manipulate data.

CHAPTER 14

Transfers

A transfer is an asset that has tokens that can be divided and distributed to other accounts. The only two current transferable assets are called *relevance* and *status*. Transfers help increase broadcasts positions in the Relevant and Top sorted feeds. Any transfer made to a broadcast not made in the current week will not change the position but will still transfer to the account for their usage.

14.1 Relevance

When an account goes to a **#** handle tag page, that has not been claimed, they can type a permanant description for the tag and claim the ownership for themselves. When the tag is claimed, 240 relevance points are awarded to the owner for the tag. Any broadcast that has the tag in the message will be displayed on the page in cronological order. The owner of relevance points for the group can award quality broadcasts with points to increase it's position on the Relevance sorted feed. The users who have been rewarded relevance points can then award other users posts to increase their positions.

Relevance Asset

```
//This is a group asset
const group = {
    appId: APP_ID,
    year: "year-" + new Date().getWeekYear(),
    week: "week-" + new Date().getWeek().doubleDigitString(),
    type: ASSET_RELEVANCE,
    tag: "",
    account: user.publicKey,
    timestamp: new Date().toString()
}
```

Relevance Data

```
//This is a vote data type
const vote = {
    appId: APP_ID,
```

(continues on next page)

(continued from previous page)

```
    year: "year-" + new Date().getWeekYear(),
    week: "week-" + new Date().getWeek().doubleDigitString(),
    type: DATA_TRANSFER,
    data: {
        description: "Group vote share transfer",
        timestamp: new Date().toString()
    }
}
```

14.2 Status

When an account sends a broadcast of any kind they will generate a single status point for their account. This can be used to transfer to other accounts for their quality posts which will increase the broadcast position in top-sorted feeds. The broadcast assets themselves are used as the tokens for status as opposed to creating a completly seperate asset each time a broadcast is made.

CHAPTER 15

Week

GruPur weeks span between Monday and Sunday every week. Activity in these weeks will be recorded. Posts made in a week will be displayed on that weeks global feed. Only Relevance and Status gained in a week will increase it's position in the feed, but all Relevance and Status will still be counted in totals no matter the time frame of the transaction. GruPur new year will always start on the first Monday of every January. Any week can be refrenced by it's handle for example #2019-12 would span the week of Monday March 18, 2019 - Sunday March 24, 2019.