MultiDasher
A Drupal-based MultiChain blockchain manager

22 October 2018

At its simplest, MultiDasher provides a graphical interface to MultiChain blockchains, allowing users to quickly and simply create or join blockchains, issue and send digital assets (cryptocoins or tokens instantiated by the blockchain) and manage sharing of content through the blockchain. But because it is build using Drupal it has the potential for much more.

MultiDasher uses MultiChain, a well-established open source permissioned blockchain package based on the Bitcoin source code, and continually undergoing development. MultiChain supports both digital assets and the transfer of data through streams.

Although MultiDasher allows you to get started with blockchains quickly and simply, as the back-end is built on a decoupled Drupal installation, it provides the potential for linking human-readable content to the blockchain. For example, an address book allows you to send coins to others by specifying their name, rather than their blockchain address. Furthermore, we see the possibility of linking Drupal sites together, even across low-trust boundaries.

With *MultiDasher*, Drupal sites can begin to share data in a consensus building manner across low-trust boundaries, and true interaction between different Drupal sites becomes a reality.

MultiDasher brings the untapped value of blockchain to Drupal.

The front-end is implemented in Angular, which allows customization of the system to support additional user interaction with further functionality.

MultiDasher is open source software licensed under the GPLv3 software license. Its development is sponsored by Chainfrog Oy.

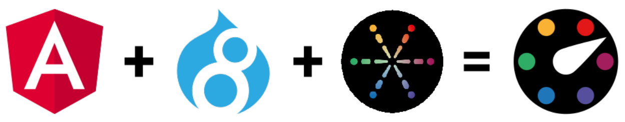
Introduction

Blockchains are commonly viewed as one of the most promising new innovations to emerge onto the tech scene. Unfortunately they are also esoteric and complicated, relying on the mathematics of cryptography, peer-to-peer communication, and fault tolerance schemes. Furthermore, the deployment of a successful blockchain requires not only the development of software code, but also an understanding of business cases, economics and social/psychological issues.

A search of the Drupal website and the internet in general reveals that very little activity has taken place in the Drupal world with regards to blockchain. The few projects there are focus predominantly on handling bitcoin payments, and general blockchain projects appear to be stalled or no longer maintained.

With MultiDasher, we present a fully functional Drupal 8 installation that interfaces with MultiChain to allow Drupal content to interact with blockchain assets, and for Drupal to benefit from tamper-proof data storage and retrieval, automatic peer-to-peer connectivity between different Drupal sites, data transfer possibilities, as well as standard digital asset creation and transfer.

Read on to understand the MultiDasher architecture better, find out how it connects Drupal to MultiChain, and what the future road-map for the project is.



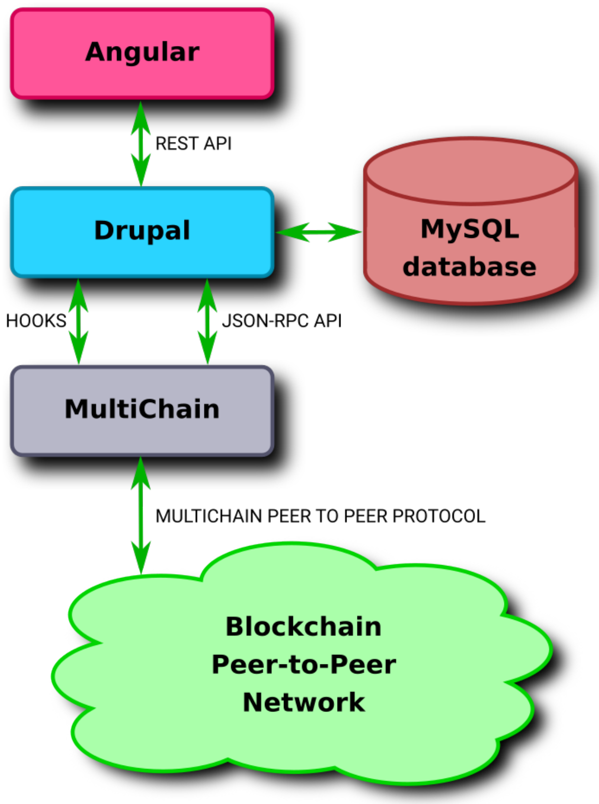
Summary

Angular Front-end: Angular, licensed under the MIT license, provides the structural framework for a dynamic web front-end.

Decoupled Drupal Back-end: Drupal connects the Angular front-end to the MultiChain blockchain transport and asset layer, and through its content management capacities, allows meta-data and associated data to be stored and linked to or updated with data retrieved from the blockchain. Because of the number of modules that Drupal supports, almost any kind of conceivable functionality can be added to the system. For example, Drupal with OAuth provides user authentication and login, allowing usable access to the blockchain back-end. Similarly, the Drupal user space means that abstract blockchain addresses can be associated with those users.

MultiChain Engine: MultiChain blockchains provide both the connectivity for the system through a peer-to-peer network, IAM (identity and access management) capabilities, the possibility of issuing and transferring digital assets, as well as data integrity. MultiChain has an extensive API (application program interface), and comes with a CLI (command line interface) to interact with chains. However, the sheer number of commands, parameters and concepts that it embodies can be somewhat overwhelming for casual users.
MultiChain allows multiple blockchains to be run on the same machine for different purposes. There is, however, no current means to transfer assets from one to the other. Data can be transferred between different chains by a trusted party using, for example, Drupal.

The figure below illustrates how the various components of MultiDasher communicate.


Angular
REST API
Drupal
MySQL database
HOOKS
JSON-RPC API
MultiChain
MULTICHAIN PEER TO PEER PROTOCOL
Blockchain Peer-to-Peer Network

Before we dive into a deeper description of the system structure, let's look at what MultiDasher could be used for.

Use Cases

Simple Tokens

MultiChain allows the creation of bitcoin-like tokens, known as digital assets. If you use MultiDasher to spin up your own blockchain, you can issue as many different tokens as you like, for whatever purpose. Each token has all the transaction properties of bitcoin - you can send them to other users, receive them, check balances for them, knowing all the time that due to the properties of blockchain they are unique, unforgeable digital entities with clear indisputable ownership. With MultiDasher this capability is enable through a graphical interface rather than long command line interface strings.

Use cases for digital tokens include loyalty points, local currencies, accounting units, transferable session keys, tamper-proof records of ownership, license keys, university course achievement credits, in-game money ... the list goes on and on. You may well be thinking of numerous different applications in your own area of expertise or industry.

The main point is that the tokens issued are digitally secured through the blockchain, so no one (other than the issuer, if they choose to set them up in a manner that allows more to be issued) can create more, and in any case if someone owns a token, no one can take it off them - they are in full control of what they own.

Communication Efficiencies

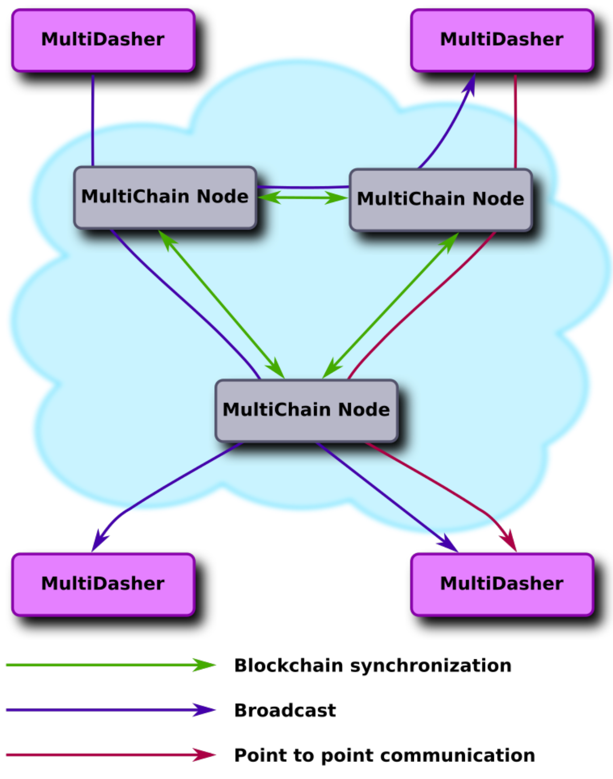
A significant change in the usual communication model of Drupal installations with the application of a blockchain is the shift from Pull or Push data transfer to a Push-then-Pull model.

Take a standard online store, for example. The site needs to know how much stock, if any, of a particular item is available to sell. The store owner does not want customers to select items, only to find out at checkout (or even worse, after checkout) that the goods are not available. In the traditional model, the site will poll suppliers (a Pull model) through an API at regular intervals to see if stock is available. This model is terribly inefficient - for busy commerce systems the site is basically asking every five minutes if a particular pair of shoes are still available for sale. Multiply this polling activity by ten, a hundred or even more, and you'll understand how much pointless polling activity is taking place.

The Push model is equally fraught with problems - the site owner has to open the site up to potential security vulnerabilities in order to allow the supplier to push changes in stock level over to the shop. Furthermore, the supplier now has to connect to and inform a large number of sites each time a sale is made one one particular site. In practice accurate levels of stock are not communicated, which also means that "this item is selling quickly - get your order in now" promotions are not possible.

With a Drupal installation blockchain-enabled through MultiDasher it becomes possible to adopt a Push-then-Pull model. A shop pushes a record of a sale onto the blockchain, the supplier pulls this off the blockchain (noting that the transaction has occurred because they are subscribed to a stock data stream). The supplier confirms the reservation of the item, and pushes a new stock level onto the blockchain, which all other store pull off the blockchain. All interested parties are informed within seconds or at most minutes, and no pointless API calls are ever made.

In summary, using MultiDasher, communication only occurs when a relevant change has happened. The figure below illustrates how the various components of MultiDasher communicate.



Consensus

If you have ever had to deal with multiple Drupal installation, where users have to sign up for each site, and their content is scattered across sites, you are dealing with a problem that MultiDasher may be able to solve.

In blockchain reference is sometimes made to "low-trust boundaries"; the situation where two different entities or sites, for whatever reason, can't open up all their data to each other. It can be down to political, social, or simply paranoid reasons, but it results in a headache for system administrators.

Introducing Frogchain

To allow you to try out the system fully, we have spun up a blockchain called *Frogchain*, the connection details of which are present in each MultiDasher installation. You can choose to connect or not. Connecting automatically connects you to all other Frogchain users. Initially you will be awarded two sets of tokens - "votes", which allow you to vote on the priority list for future features, and "reputation", which at some point in the future should be hooked up to a StackOverflow-style reputation system (but unlike that system, you can actually transfer or "spend" your reputation points, independently of the blockchain administrators - that's us).

Build up enough reputation and you'll receive permission to issue your own tokens on Frogchain, mine blocks, create data streams, and who knows, even receive administrator rights.

Frogchain also has a native bitcoin-style currency, which we've called *tadpoles*. We haven't worked out what to use them for yet, but we'll be handing them out to project contributors as we see fit. At the moment they are there more for educational purposes than anything else.

Architecture and Code

We will now move on to a code-level description of the system. If you visit the MultiDasher Github repository you will see that the code has been divided into three folders:

```
angular
drupal
example-database
nginx
```

There is also a bash installer script, namely `multidasher_server.sh` which automates the process of installing MultiDasher.

The `angular` directory contains all Angular code for the front-end interface, including routes and functions connecting the front-end to Drupal. Look in `src/app/` for the relevant files.

The `drupal` directory contains Drupal files required by the installer script to install MultiDasher. For developers, the files of particular interest can be found under `web/modules/custom/multidasher/src/` in the `Controller/` directory.

The `example-database` directory contains a copy of a MySQL database containing data structures and tables for the sample Drupal installation to work, including *Frogchain* connection details, allowing you to get started quicker.

The `nginx` directory contains required settings files for the NGINX web server. If you know what you are doing is is perfectly possible to swap this out for an Apache web server or other web server.

No MultiChain binaries or code are present in the repository, as MultiChain is downloaded during the installation, and runs as a separate standalone binary. In a sense, it forms a transport layer such as TCP/IP or HTTP (which also do not need to be downloaded or edited for application layer software to work).

Drupal

Drupal is the "glue" that binds the system together.

Future Road-map

First, a confession: although we've been thinking about the issues and features described above at Chainfrog for over a year now, we've only been working on MultiDasher since 4 October 2018. Hopefully we've outlined all the potential of a fully functional system, but as of today only the following blockchain functionality is available in the Github codebase:

- View existing chains
- Create your own chain
- Handle requests to join your chain
- Request to join a chain
- Start/stop node connection to a chain
- Browse available assets
- View asset balance
- Issue assets
- Send/Receive assets
- Associate Drupal user data with chain addresses

Obviously there is still plenty more to do, but the above already allows you to play with blockchains and get a feeling for how they work and what they can do. We have a lot of ideas about what to do next, but would like to get input from the community and some support too.

That's why we've set up the votes system, and listed our hoped-for future functionality. You have the opportunity to priorities the list, or even add further elements to the list. Visit the MultiDasher Github repository at and go to the Wiki for more details on the following:

- Chain visualizer
- Creating and managing data streams
- Publish Drupal data through streams
- GDPR issues
- Security issues
- Multiple users for same node/Drupal instance
- Separate Drupal modules for MultiDasher functionality
- Documentation
- Automating the voting system
- Thinking about the reputation system
- Test automation

What Next?

We believe that MultiDasher has significant potential for improving the Drupal experience. If you're managed to read this far we hope you agree with us and are perhaps thinking about joining in. Why not start out by visiting the MultiDasher website and the Github repository, and install the software to try it out. The MultiDasher Wiki in particular has a lot of resources to get you started.

Once you've done that, please feel free to reach out to us, raise issues, or start improving the code with us.

About Chainfrog Oy

Chainfrog Oy was founded in 1996, and invents and researches blockchain technology. Visit our website or contact us at info@chainfrog.com for more information.