Link Blockchain Documentation

Release 1.0

Jonathan Brown <jbrown@bluedroplet.com>

Jun 28, 2017

Contents

Link is an uncontrolled linked data ecosystem living on the blockchain. It is fully public. It cannot be censored and no one can be prevented from participating. It is fully programmable at every level. It is a development platform for interconnected distributed content apps that empower the individual.

The blockchain was activated on Thursday 27th April 2017 09:15:23 AM (UTC).

CHAPTER 1

Useful links

- Blog
- Twitter feed
- Subreddit
- Gitter channel
- Block explorer
- Network stats
- Solidity APIs
- Issue tracker

CHAPTER 2

Table of contents

How does it work?

Link is an Ethereum blockchain. A blockchain is a shared database that has a built-in cryptocurrency that is used to financially incentivize the neutrality of the database. No-one can be in charge of a blockchain.

The first blockchain was called Bitcoin. Pretty much the only thing in the Bitcoin database is how much Bitcoin each account has.

Ethereum takes this concept much further. Computer programs called smart contracts can be uploaded into the blockchain and these programs can then store data in the database that everyone has a copy of.

Many big projects are deployed on the Ethereum blockchain. But this presents a problem. One must assume that every software project has fatal bugs that will need to be fixed once they are discovered. Bitcoin had fatal problems early on that needed to be fixed and so did Ethereum.

Unfortunately, when the projects that are deployed on Ethereum need to be fixed they will have a very hard time because they will need to convince the entire blockchain to deploy their fix. This was the problem with The DAO crowdfunding project. An attacker stole $150m because of a bug in the smart contract. In attempting to fix this problem the Ethereum was split into two blockchains: Ethereum and Ethereum Classic.

The only purpose of the Link blockchain is to run Link. This means that whenever there is a problem that can only be fixed with a hard fork, it will not be difficult to convince the Link community to adopt it.

The cryptocurrency of the Link blockchain is also called Link.

BlobStore

The principle smart contract on the Link blockchain is *BlobStore*. It uses IPFS as the underlying storage layer. Anyone can store a blob of data and get back a blobId. This data is publically readable to everyone.

BlobStore has a rudimentary revisioning system so blobs can be updated while retaining their blobId and revision history.

Google Protocol Buffers

In order for the data stored in BlobStore to have meaning, there must be some sort of schema. Google Protocol Buffers (Protobuf) is ideal for this. It has an upgradable schema system. This means that Link has a system of content type inheritance. A standardized story type could be extended with new fields and still be readable by software that only knows how to read standard stories.

Anyone can make a new type. Example types are user profiles, media metadata, tweets, reddit posts, blog posts, business information, reviews, etc.

Protobuf encodes the data very tightly. This data is then compressed with Google Brotli. This minimizes the cost of storing information on Link.

Additional smart contracts

Anyone can deploy additional smart contracts onto the Link blockchain. An important example would be a feed contract that would provide functionality akin to RSS / Twitter / YouTube.

Other examples would be contracts to vote for content or tag content.

Cryptofuel

Name: Link

Code: LINK

Symbol:

BIP44 coin type: 76 / 0x8000004c

All transactions on the Link Blockchain must be paid for with Link. Transaction size is measured in "gas". Transactions can be prioritized by paying a higher gas price.

Issuance

Project revenue of 55 million LINK (equivalent to 5 years of mining) is pre-allocated into a smart contract that will release it over 2000 days at a rate that decreases every 200 days.

The purpose of the revenue is to provide funds for re-investment into the Link ecosystem and to provide profit for the developer.

The block reward is the same as Ethereum. 5 LINK per block / 30k LINK per day.

Issuance rate

Cumulative issuance

Post-revenue inflation rate

Revenue live status

Configuration

Network ID: 76

Chain ID: 76

Port: 30313

RPC Port: 8645

WS Port: 8646

```
git clone https://github.com/link-blockchain/link-blockchain.git
```

Link Blockchain can be synchronized with either Geth or Parity:

Geth

Install Geth as described here: https://github.com/ethereum/go-ethereum/wiki/Building-Ethereum

```
geth --config link.toml --datadir ~/.link-geth init genesis.json
geth --config link.toml --datadir ~/.link-geth --rpc
# In a separate terminal launch the console.
geth attach ~/.link-geth/geth.ipc
```

Parity

Install Parity as described here: https://parity.io/parity.html

```
parity --chain link.json --port 30313 --jsonrpc-port 8645 --geth
```

Netstats

To have your node listed on http://stats.link-blockchain.org/ do the following (requires Node.js):

```
git clone https://github.com/cubedro/eth-net-intelligence-api
cd eth-net-intelligence-api
npm install
sudo npm install pm2 -g
```

Edit app.json:

```
     {
       "NODE_ENV"        : "production",
       "RPC_HOST"        : "localhost",
-      "RPC_PORT"        : "8545",
-      "LISTENING_PORT"  : "30303",
-      "INSTANCE_NAME"   : "",
+      "RPC_PORT"        : "8645",
+      "LISTENING_PORT"  : "30313",
+      "INSTANCE_NAME"   : "MY_NODE_NAME",      // Customize this.
       "CONTACT_DETAILS" : "",
-      "WS_SERVER"       : "wss://rpc.ethstats.net",
-      "WS_SECRET"       : "see http://forum.ethereum.org/discussion/2112/how-to-add-
↪yourself-to-the-stats-dashboard-its-not-automatic",
+      "WS_SERVER"       : "ws://stats.link-blockchain.org:3000",
+      "WS_SECRET"       : "welcometothelinkedworld",
       "VERBOSITY"       : 2
     }
```

Then:

```
pm2 start app.json
```

Filter bubbles

500 million tweets are published every day, but if you go to Twitter you will only see a small subset of these. Every platform has its own filter system to decide which content you see.

On Twitter you see the latest tweets from people you follow (minus those who have been banned), some sponsored tweets, with a "While you were away..." section at the top generated by unknown means. People are sometimes banned entirely from the platform.

On Reddit, you see content from subreddits you are subscribed to. Typically ordered by an opaque algorithm called "hot". Users can vote, but there is no real way to see how this affects what content is displayed. Large sections of the site will sometimes disappear.

On Medium content creators are sometimes ordered to change their content under threat of it being removed altogether.

Each of these communities is siloed from each other. They control the filter bubble within their platform.

Link is fully programmable at every level. Anyone can take existing content on the system and develop new ways to select which content each user should see, or even utilize data from another filter bubble, for example analysing existing votes in a new way.

Light client

Currently to interact with Link in a fully decentralized way you need to have a fully syncronized node running locally. It can take a long time to sync and will take up a lot of storage space.

This will all change with Ethereum's light client functionaity that is rapidly nearing maturity. Both Geth and Parity will be able to serve light clients and act as light clients themselves.

It will also be possible for Android and iOS devices to have light client functionality.

BlobStore

BlobStore is the principle smart contract system for Link. All base content is registered with BlobStore. IPFS is used as the storage layer.

Useful links

- Source code
- Issue tracker
- Solidity API

Properties

BlobStore has the following properties:

- **Timestamped** The approximate time that every blob is published is recorded on the blockchain.
- **World-readable** Every blob published can be read by anyone. The only way to avoid this would be to encrypt the blob before publishing it.
- **Immutable** While a blob can be "retracted", it can never really be deleted because the transaction that created it will be archived for eternity on full nodes.
- **Revisioned** BlobStore has a rudimentary revisioning system built-in where a blob can have multiple revisions, e.g. for editing posts. More sophisticated revisioning systems can be built on top of BlobStore where each blob is a revision.
- **Ownership** Each blob can have an owner. Only the owner can modify a blob, change blob settings, or transfer ownership to another address.
- **Configurable** Each blob has the following flags that can be set:
  - **Updatable** The contents of the blob can be changed. Once disabled this flag cannot be re-enabled.
  - **Enfore Revisions** When updating the blob a new revision must be created. It is not possible to retract revisions. Once enabled, this flag cannot be disabled.
  - **Retractable** The blob in its entirety can be retracted. This is unaffected by Enforce Revisions. The blobId of a retracted blob can never be used again. Once disabled this flag cannot be re-enabled.
  - **Transferable** The blob can be transfered to another user (if they accept it), or disowned completely. Once disabled this flag cannot be re-enabled. At creation time blobs can also be flagged as anonymous to not have an owner associated. An alternative to transferable blobs is to use a proxy account with transferable ownership as the blob owner.
- **Scalable** Only a bare-minimum of information is stored in contract state. IPFS is used for actual content storage.
- **Upgradability** BlobStore is an upgradable system. Due to a security vulnerability, new storage system, or gas cost improvements a new BlobStore contract may be deployed.
- **Unit tests** BlobStore has tests written in Solidity using the Dapp framework.

blobId

Each blob has a 20 byte blobId that is unique to the contract that generated it. The blobId can be further classified to indicate which contract and blockchain the blob is on.

- **Off-chain** There is considered to be an ordered list of BlobStore contracts. For example, the first contract would be #0. The next one would be #1. 20 byte blobIds can be prefixed to indicate which contract the blob is in, or this can be assumed from context.

  Client software should have a hard coded whitelist of BlobStore contracts that its knows how to read from.

- **On-chain** New BlobStore contracts register with the BlobStore registration contract. When a blobId is stored in a smart contract, it must be stored with the 12 byte BlobStore contractId. This way the smart contract can look up the address of any BlobStore contract in the registration contract. This is essential to future-proof smart contracts that need to communicate with BlobStore contracts.

Deployments

All current deployments of BlobStore contracts on Link were compiled with Solidity 0.4.11 with optimizations enabled.

BlobStoreRegistry contract address: `0x767f8a42f169038d18416b66f7241bf70fffc329`

BlobStore #0

IPFS with SHA256 hash.

contract address: `0x73d1bdf1550cb6506b5728c9f11cd1acacfb2095`

BlobStore contractId: `0x28ccad938027378a568fb84f`

Obtain LINK

In order to publish content on the Link network, or execute any kind of transaction, you must spend some LINK. LINK can be obtained either by mining it or by purchasing it.

You need to have a Link account. If you do not have one already, go to the Link console and run `personal.newAccount();`.

Alternatively an Ethereum account can be created with MyEtherWallet and that can be used for Link.

Mine LINK

CPU Mining

CPU mining with Geth is very easy. Run the regular Geth command, but add the `--mine` option. Use `--etherbase` to specify which account will receive the mining rewards. It defaults to using all processor cores. This can be changed using the `--minerthreads` option.

GPU Mining

Use ethminer-genoil.

Ubuntu 15.10 or Newer. OpenCL only (for AMD cards)

```
sudo apt-get update
sudo apt-get -y install software-properties-common
sudo add-apt-repository -y ppa:ethereum/ethereum
sudo apt-get update
sudo apt-get install git cmake libcryptopp-dev libleveldb-dev libjsoncpp-dev␣
↪libjsonrpccpp-dev libboost-all-dev libgmp-dev libreadline-dev libcurl4-gnutls-dev␣
↪ocl-icd-libopencl1 opencl-headers mesa-common-dev libmicrohttpd-dev build-essential␣
↪-y
git clone https://github.com/Genoil/cpp-ethereum/
cd cpp-ethereum/
mkdir build
cd build
cmake -DBUNDLE=miner ..
make -j8
```

Ubuntu 15.10 or Newer. OpenCL + CUDA (for NVIDIA cards)

```
wget http://developer.download.nvidia.com/compute/cuda/repos/ubuntu1404/x86_64/cuda-
↪repo-ubuntu1404_7.5-18_amd64.deb
sudo dpkg -i cuda-repo-ubuntu1404_7.5-18_amd64.deb
sudo apt-get -y install software-properties-common
sudo add-apt-repository -y ppa:ethereum/ethereum
sudo apt-get update
sudo apt-get install git cmake libcryptopp-dev libleveldb-dev libjsoncpp-dev␣
↪libjsonrpccpp-dev libboost-all-dev libgmp-dev libreadline-dev libcurl4-gnutls-dev␣
↪ocl-icd-libopencl1 opencl-headers mesa-common-dev libmicrohttpd-dev build-essential␣
↪cuda -y
git clone https://github.com/Genoil/cpp-ethereum/
cd cpp-ethereum/
mkdir build
cd build
cmake -DBUNDLE=cudaminer ..
make -j8
```

CD to the `ethminer` subfolder and run the following command

```
./ethminer -G -F http://localhost:8645
```

Purchase LINK

LINK from the revenue smart contract will be sold on exchanges once they support it. Until then, LINK can be purchased directly.

If you wish to purchase LINK please email purchase@link-blockchain.org. It is currently available at a price of 0.01 USD per LINK. This price is subject to change at any time. Payment can be made via any cryptocurrency.

Team

Jonathan Brown - Founder

Jonathan was a participant in the Drupal web development community for ten years where he created the blockchain Drupal modules Coin Tools and Ethereum. As he fosters the growth of the Link development community, he endevours to replicate the vibrancy of the Drupal development community.

James Drummond - Dapp developer

James is a senior developer from London with over ten years experience. He has worked in the finance sector for clients such as HSBC and as senior front end developer for ALLSAINTS - an e-commerce site with one million + unique visits / month and frequently £1,000,000+ per day in sales.

More recently he has worked as a contract developer in London, including as senior developer for RangeRoom - a fashion sector marketplace that he developed from the ground up.

He is founder and developer of Smoothbook - an online appointments system for yoga and fitness studios which currently manages 45,000+ appointments / month.

Events

Upcoming

Kuala Lumpur Blockchain Evening

Friday, July 14, 19:00 - 22:00
Einstein Boardroom
WORQ
Level 3A, Glo Damansara

No. 699, Jalan Damansara
Kuala Lumpur

Previous

Bangkok Satoshi Square

Monday, June 26, 18:30 - 21:30
American Bar & Grill
4/26 Sukhumvit Soi 8
Bangkok

CHAPTER 3

Indices and tables

- genindex
- modindex
- search