

Blockchain & Distributed Ledger Technologies

Main title: Running the essentials for the IOTA Blockchain

**Ivo Aeschlimann** (16-920-803), **Alberto** **Masera** (22-738-116),   
**Ali Ahmed** (19-769-025), **Songyi Han** (18-796-847)

Blockchain and Crypto Economics (HS22)

15.01.2023

**ABSTRACT**

The goal of our blockchain project is to implement the basics of the IOTA network. This includes running the network, fixing the fountain, being able to do transactions and having a working spammer on the network.

In this report we give an overview of the IOTA network, the economics and how we managed to run it locally. We achieve all the requirements, expect that we’re unable to do transactions manually. The faucet and spammer are still working as intended and you could likely do transactions, but we couldn’t get to interact with the API.

The project can be seen here: <https://github.com/Ivoooo/uzh-iota>

However, you may need to contact [ivo.aeschlimann@uzh.ch](mailto:ivo.aeschlimann@uzh.ch) for access.

# Introduction TO IOTA & Economics

The IOTA blockchain is based on a distributed ledger technology (DLT) and is designed to provide secure and efficient transactions between connected devices (machine-to-machine) on the Internet of Things (IoT) ecosystem. It is an open-source, decentralized platform that uses a unique consensus mechanism called the “Tangle”, which allows for transactions at almost no cost due to its unique graph structure.

The most important difference between IOTA and traditional blockchains is its use of a directed acyclic graph (DAG) data structure rather than a traditional blockchain. In a DAG, each transaction can be connected to several other transactions, forming a web of interconnected transactions. This structure allows for parallel processing of transactions, increasing the speed and scalability of the network.

Another key feature of IOTA is its unique upcoming consensus mechanism and currently number one priority called “Coordicide”.[[1]](#footnote-2) “Coordicide” is a mechanism proposed by the IOTA foundation to remove the central coordinator node from the IOTA network. Instead, it will be replaced with a new consensus mechanism called the “Fast Probabilistic Consensus (FPC) within Byzantine Infrastructures” algorithm[[2]](#footnote-3). This should make the network fully decentralized and autonomous and should improve scalability and security. The implementation of the “Coordicide” mechanism is expected to greatly improve the network's functionality and open the door to new use cases and applications.

In the “Tangle”, each transaction must confirm two previous transactions before it can be added to the network. This process is called "tip selection" and is performed by the sender of the transaction. By requiring each transaction to confirm two previous transactions the network is always in a consistent state and there is no need for mining or other energy-intensive processes.

IOTA digital currency is called MIOTA. MIOTA is used to do transactions on the IOTA network and can be used to pay for goods and services, as well as to transfer value between individuals and devices. One of the key benefits of using the IOTA network to do transactions is that there are no transaction fees due to the characteristics of the “Tangle”, making it a great currency for small transactions such as the ones that likely will be happening with IoT devices.

IOTA's architecture also allows for the implementation of smart contracts[[3]](#footnote-4) using a technology named “Qubic”[[4]](#footnote-5). “Qubic” allows for self-executing contracts with the terms of the agreement encoded directly on them. Smart contracts allow for the automation of certain processes and can be used to create decentralized applications (“dApps”).

IOTA has several partnerships and initiatives in place to promote the adoption and development of its technology.[[5]](#footnote-6) They are partnered with several big firms and organizations to further explore the use of IOTA in various applications such as supply chain management, digital identity, and autonomous vehicles. For example, Bosch has partnered with them to explore further possibilities of automation in their products.[[6]](#footnote-7)

# Basics of the iota network

The official documentation[[7]](#footnote-8) is the best and often only resource you can consult to get information about the IOTA network and solve your problems. To run the IOTA network, we used the official guide[[8]](#footnote-9) of the IOTA Foundation. The key part was cloning the GoShimmer git repository[[9]](#footnote-10). The GoShimmer repository is updated daily, and you have no guarantee that there won’t be breaking changes and that every part is working. Because of this we recommend using our version if you don’t have any additional needs.

The two most important files are “docker-compose.yml” and “docker-compose.local.yml”, both in the “./goschimmer/tools/docker-network” folder. The first one provides a faucet, one node and all other requirements the network needs to run. Make sure that if you ever extend this project not to overwrite any ports that are on there as otherwise the network will crash.

In general, you should only need to edit the “docker-compose.local.yml” file, however you could need to edit other files if you want to further expand the functionality aside from what we already have provided. The “docker-compose.local.yml” file is used to provide additional nodes. To do that copy the content of “peer-master\_22” and paste it below. Then edit ALL the ports to some unused ones and you will have another node. To access the new node, go to the corresponding “WEBAPI\_BINDADDRESS”.

For more information about how to run the project please refer to the README.

# The goals

## Running the network

Graphical user interface

Description automatically generatedGraphical user interface

Description automatically generated

The network was already running on it’s own. The only thing that was needed to be changed was the ports on the .yml files. Since it’s in a docker container it should be able to be run on any machine (Tested on Ubuntu and Mac). The challenge was figuring out the ports, which took some time.

## The fountain

A picture containing graphical user interface

Description automatically generated

The fountain does work and is on port 8091. It initially didn’t work but through changing some settings in the faucet folder we were able to get it running again. It should be noted that you may only give out funds ONCE per address in its entire lifespan. The only exception is the evil spammer which can auto-request funds if the setting is enabled.

## The spammer

## 

The spammer was very challenging and we’re still not quite sure how we got it to work in the end. It has a lot of features that can be read here[[10]](#footnote-11). The most important once are the rate and rate limiter. We advice staying below 50 transactions per second as otherwise network congestion can be an issue. The other one is auto-requesting funds from the fountain which can by access by “Settings -> Auto funds requesting -> enable”.

## The transactions

Transactions are working on the network as the faucet and the spammer can do transactions, however we’re having problems doing them manually. We know that they’re working since we can get funds from the faucet and the spammer can do transactions. There are 3 ways of doing transactions:

### Using the API

The API[[11]](#footnote-12) can be accessed using the client written in Go. We’re very certain that the API should be working since the faucet and the spammer are likely also using that connection to interact with the network. However, we were all having problems at different points. It should be noted though that none of us got the client to run. If we did it would likely be able do the transactions.

Installing Go and the Client was easy, however we had problems getting it to run. We suspect that we need to edit the GOPATH or make an executable out of the client (curl -x). However, we couldn’t come to a solution in time.

### Pollen-wallet

Pollenwallet[[12]](#footnote-13), also called DevNet-wallet is a react-based app. The problem with this wallet is that the lockfile is faulty and thus we’re unable to “npm install” the necessary libraries. The last update was over two year ago and we suspect that since then requirements for the libraries changed and thus npm is unable to build without having version conflicts.

### ElectricShimmer

ElectricShimmer[[13]](#footnote-14) got its last update half a year ago and would likely work. The problem is there is no documentation and no information on how to run and interact with it.

# Author contributions

The following is a sample text. All authors conceived and designed the project idea. P.M. and C.J.T. developed and wrote the business model. B.S. worked on the regulatory implications.  Y.Z. and X.Y. developed the technical implementation and wrote the technical section. Y.Z. wrote the critical overview of the plaftorm selected. All authors revised and accepted the final version of this document.

1. https://blog.iota.org/coordicide-the-road-ahead-7d89f41b0ba5/ [↑](#footnote-ref-2)
2. https://www.sciencedirect.com/science/article/abs/pii/S0743731520303634?via%3Dihub [↑](#footnote-ref-3)
3. https://blog.iota.org/an-introduction-to-iota-smart-contracts-16ea6f247936/ [↑](#footnote-ref-4)
4. https://blog.iota.org/the-state-of-qubic-63ffb097da3f/ [↑](#footnote-ref-5)
5. https://www.iota.org/solutions/partnerships [↑](#footnote-ref-6)
6. https://www.ccn.com/bosch-bets-big-on-iota/ [↑](#footnote-ref-7)
7. https://wiki.iota.org/goshimmer/welcome [↑](#footnote-ref-8)
8. https://wiki.iota.org/shimmer/goshimmer/tutorials/setup/ [↑](#footnote-ref-9)
9. https://github.com/iotaledger/goshimmer.git [↑](#footnote-ref-10)
10. https://wiki.iota.org/goshimmer/tooling/evil\_spammer [↑](#footnote-ref-11)
11. https://wiki.iota.org/goshimmer/apis/client\_lib/ [↑](#footnote-ref-12)
12. https://github.com/iotaledger/IOTA-2.0-DevNet-wallet/tree/master [↑](#footnote-ref-13)
13. https://github.com/Dr-Electron/ElectricShimmer [↑](#footnote-ref-14)