Ergo Hack VII Text Presentation

by Analog Ergo | Deathgripson

1. Analog Atomic Swap Market Pre-Alpha Stage

Although not fully complete, Atomic Analogs Swap / Analog Ergo is nearly ready for test-net launch. Throughout Ergo-Hack tremendous progress has been made to push it to a consume-able state. For some perspective, this project was started mainly in the last Ergo-Hack where the accomplishments amounted to conceiving basic Atomic Swap smart contracts for Ergo and Ethereum.

The main goal that time was to put these contracts into completely peer to peer GUIs for people to use, however the complexity of this goal became far too much overhead for a solo developer. I am still going to pursue this as it is a necessary application, however my current application focuses on user experience which I believe will be crucial to getting people to use the application and also to get people interested in improving and contributing to it down the line.

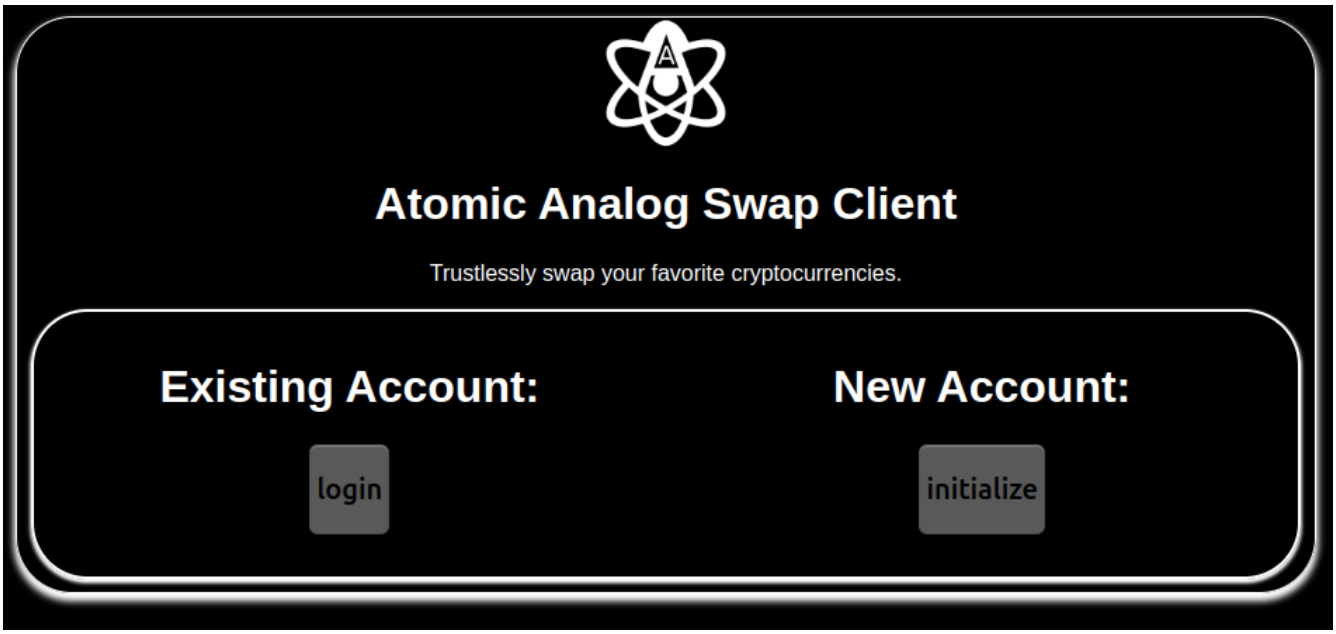
This project focuses on UX by providing an extensive infrastructure for Client (self hosted webUI and REST API) and Server (REST API) interactions, which boils the atomic swap functions down into its basic subroutines and gives each party modularity in how they wish to handle their end of the interactions.

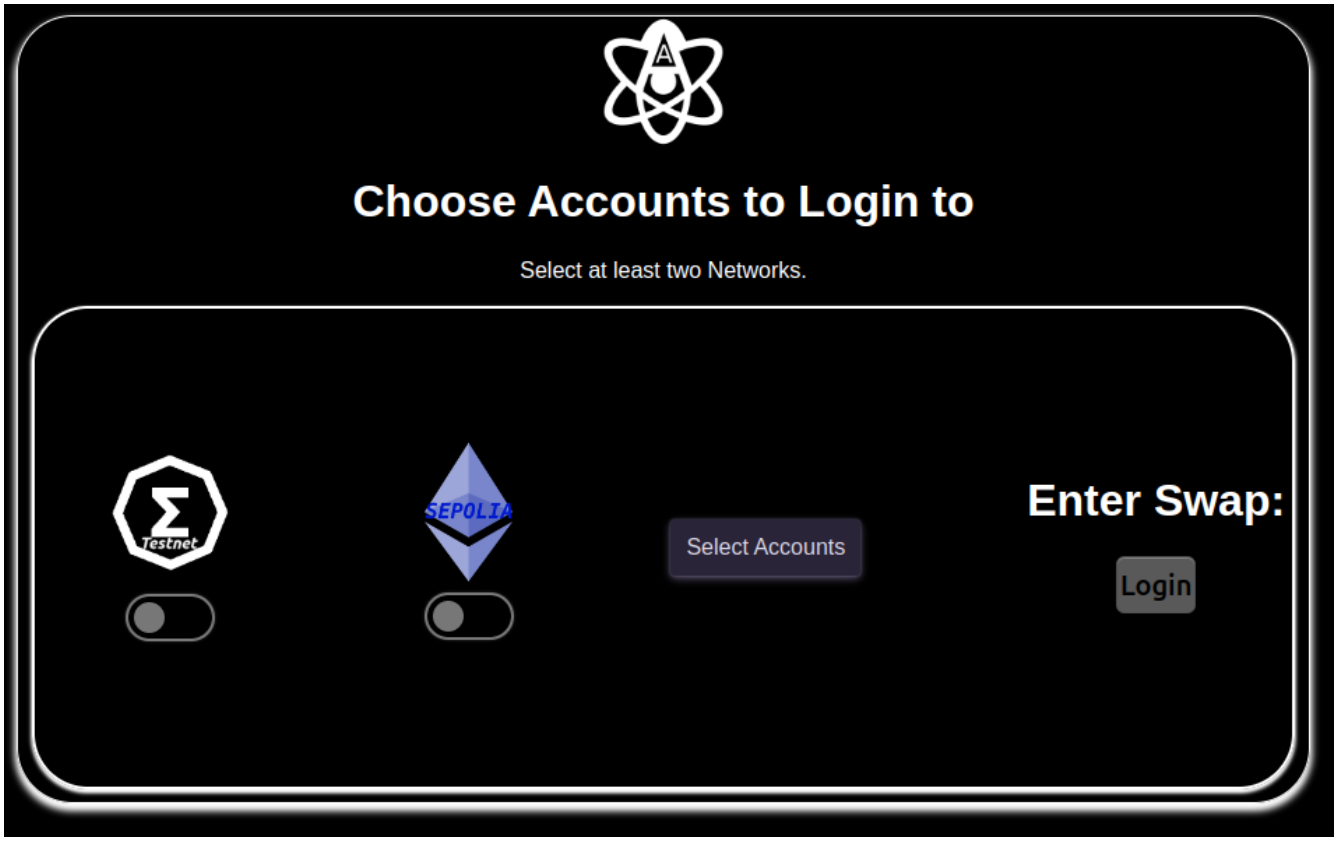
The server also takes on a bulk of the “risk” so to speak by always dedicating its resources to initiating the swap (needs to be online consistently), automating finalization of swaps to limit time wasted waiting for counter-party signatures, and handling the parts of the swap that require the most cross-blockchain interactivity (reduces security assumptions for clients). Effectively our Server handles the bulk of the work involved in this Atomic Swap protocol, which results in the Client having a superior experience by default.

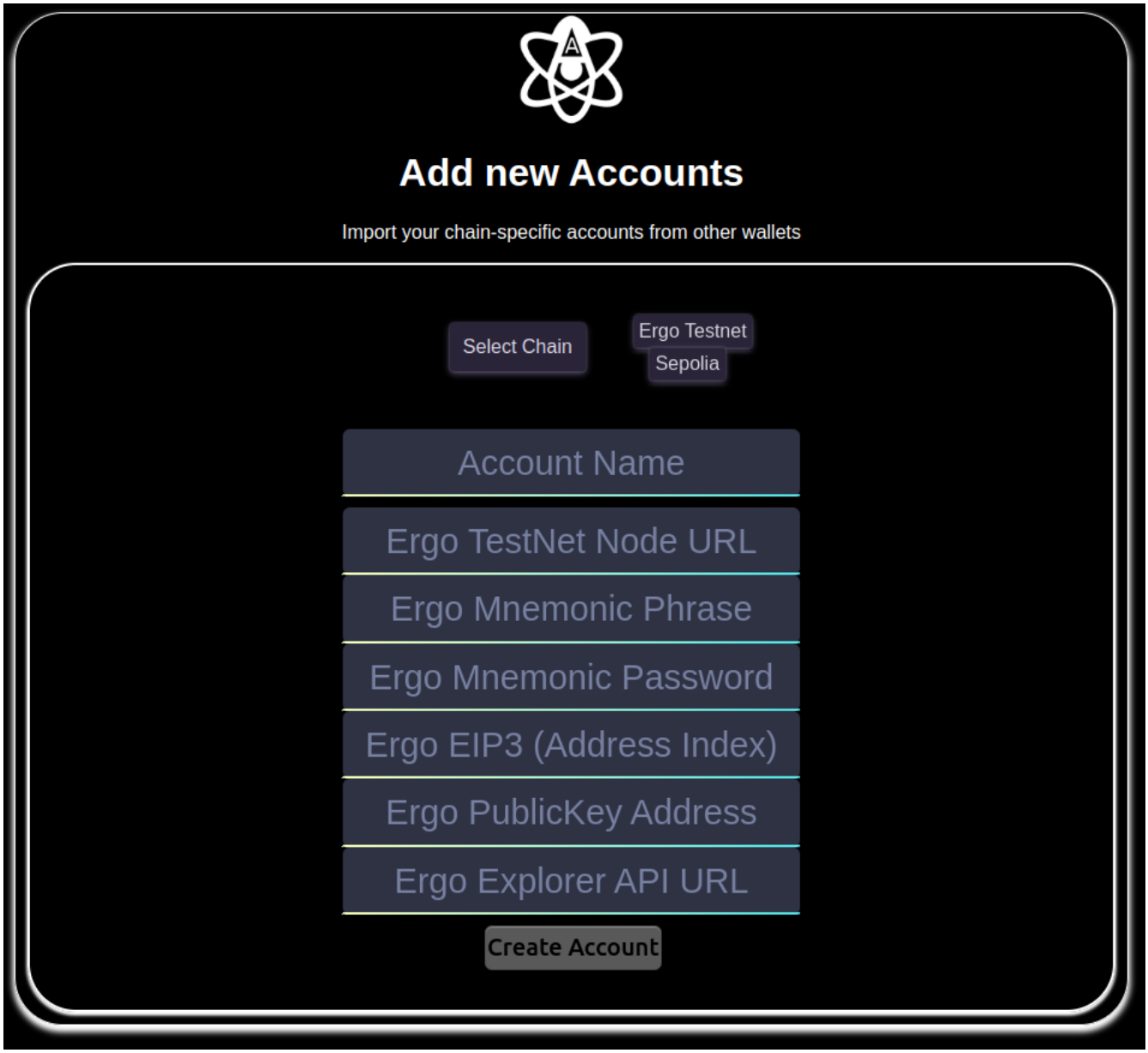
2. Self Hosted Web-UI

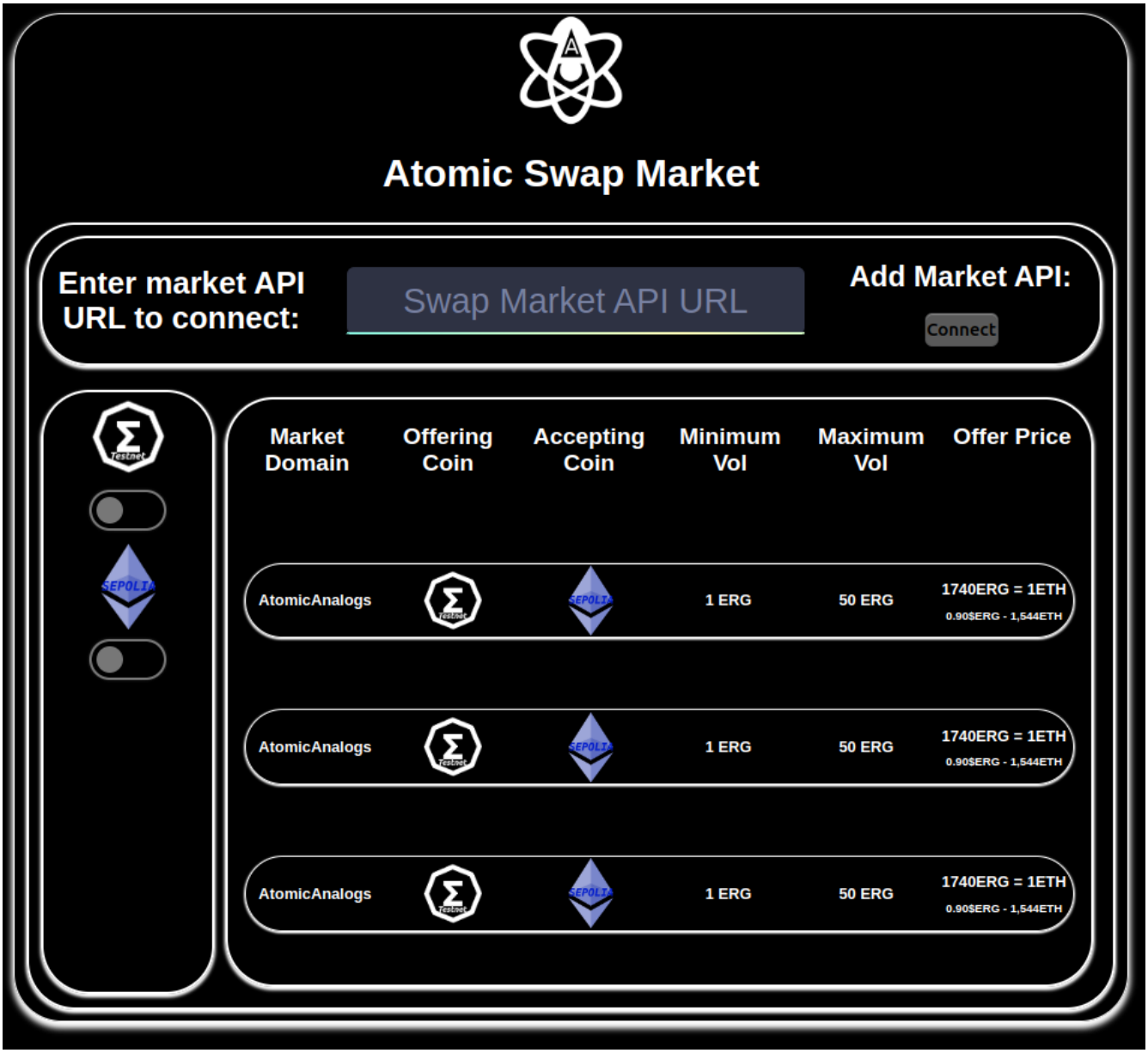
A self-hosted UI is important because atomic swaps require secret data, a web UI is convenient because it is what users are used to in the sense of a browser application (without being integrated into the browser like an add-on) and gives you access to HTML and CSS which are good GUI tools (compared to desktop GUI tooling). The Atomic Analog Swap implementation combines both, currently it is just a Hugo server that runs on the clients private local-host. However it will integrate

some JavaScript to facilitate RESTAPI calls from the webUI to the atomic swap back-end to complete its functionality. The following images are some screenshots of the Web-UI proof of concept along with its GitHub URL






Add new Accounts
Import your chain-specific accounts from other wallets
Select Chain
Ergo Testnet
Sepolia
Account Name
Ergo TestNet Node URL
Ergo Mnemonic Phrase
Ergo Mnemonic Password
Ergo EIP3 (Address Index)
Ergo PublicKey Address
Ergo Explorer API URL
Create Account



GitHub: https://github.com/dzyphr/AtomicAnalogsLocalWebUI

3. Client and Server REST API

Having a REST API gives us many benefits that make the goal of having a nice UI front end and a well automated back end a reality. As stated before, the atomic swap protocol is relatively

interactive, this means there are lots of different kinds of function calls and security checks that a user's application needs to do during the swap. The REST API gives us an interface for automating the interactivity between the client and the server which ends up being the bulk of the bottleneck for this protocol.

Some of the functions included in the Server's REST API include: `publishNewOrderType`, `requestEncryptedInitiation`, and `submitEncryptedResponse`. There are private functions like `publishNewOrderType` which allows the server owner to set order details like price and volume, the client can then retrieve these and represent them on their UI.

There are also public functions like `requestEncryptedInitiation` and `submitEncryptedResponse` which are intended for a Client to use to start and finish their interactions in the atomic swap protocol. These can be called by any client but will require many cryptographic checks in order to actually pass or be processed properly by the server.

An example of a function included in the Client's REST API would be `generateEncryptedResponse` which generates a response to the servers atomic swap initiation. This simply allows the Client to privately generate cryptographic responses through any REST compatible interface (REST makes it easy to integrate into local webUI).

GitHub: https://github.com/dzyphr/Atomic_Analogs_client_RESTAPI

GitHub: https://github.com/dzyphr/Atomic_Analogs_server_RESTAPI

4. Atomic API

Atomic API is the glue that holds together the protocol, its facilitates blockchain operations and cryptographic operations. The choice to write it initially in python was made mainly to get it prototyped as fast as possible, this ended up saving me tons of time in the long run. For future goals, we can re-write it into a faster performing language like Rust or C++ but for now it is functional running through python, and for iteration purposes this is all that matters.

Currently the API is not as fleshed out as it needs to be for main-net usage, however it is only a small number of tweaks away from being ready. At the moment it handles operations optimistically meaning that in the circumstance that the application is being used normally, it is able to perform the test-net atomic swaps to completion. However there are some security features that are crucial to add in that will make the API ready for main-net. The main one is strict time-lock checking and automated refunding, the code for this is already written it is just not integrated into the API yet.

Another security critical feature includes pricing APIs or price Oracles, without these clients and servers would have to trust each-others pricing information which is obviously not secure. While the logic for calculating price is going to be abstracted away from the cryptographic atomic swap layer anyway, it is still useful to be able to easily integrate pricing information into the automated atomic swap protocol so users can verify the value of their trades and ensure a swap wont happen if the price is off.

Atomic API started off as the low level code that went into my previous GUI Atomic Swap project that I put on hold, from there it was turned into functional automated testing scripts, and then finally into generalized functions for high level modularity of the atomic swap protocol. The modularity of the system can and will always be improved, this includes adding complex logic for handling different chains, different kinds of atomic swap protocols, encryption protocol changes, and many more things that would improve user and developer experience.

GitHub: https://github.com/dzyphr/atomicAPI

5. Atomic Analogs Swap – Information Hub / Docs Website

Since the Atomic Swap protocol is fairly complex and will have lots of security assumptions that users will need to know and stay aware of over time, having some website where people can reference official data or documentation about the protocol seems highly necessary. For this we have built a simple Hugo blog style website. Currently it is not being hosted anywhere, however as soon as we officially roll out test-net implementations we will put the site up as well.

On the site there will be instructions on building the software, understanding how to make swaps as a user/client, information about the different kinds of security assumptions and automated precautions implemented, developer information about using core tooling to create other atomic swap based software, hosting their own swap market with existing applications, and specific protocol documentation for properly interacting with the various tools involved in the application.

The following images are some screenshots from the current state of the website, most of it is subject to change.

Cross Chain Atomic Swaps

Welcome to Atomic Analog Swap
Currently Listed: Testnet Ergo, Sepolia Ether

Get Started

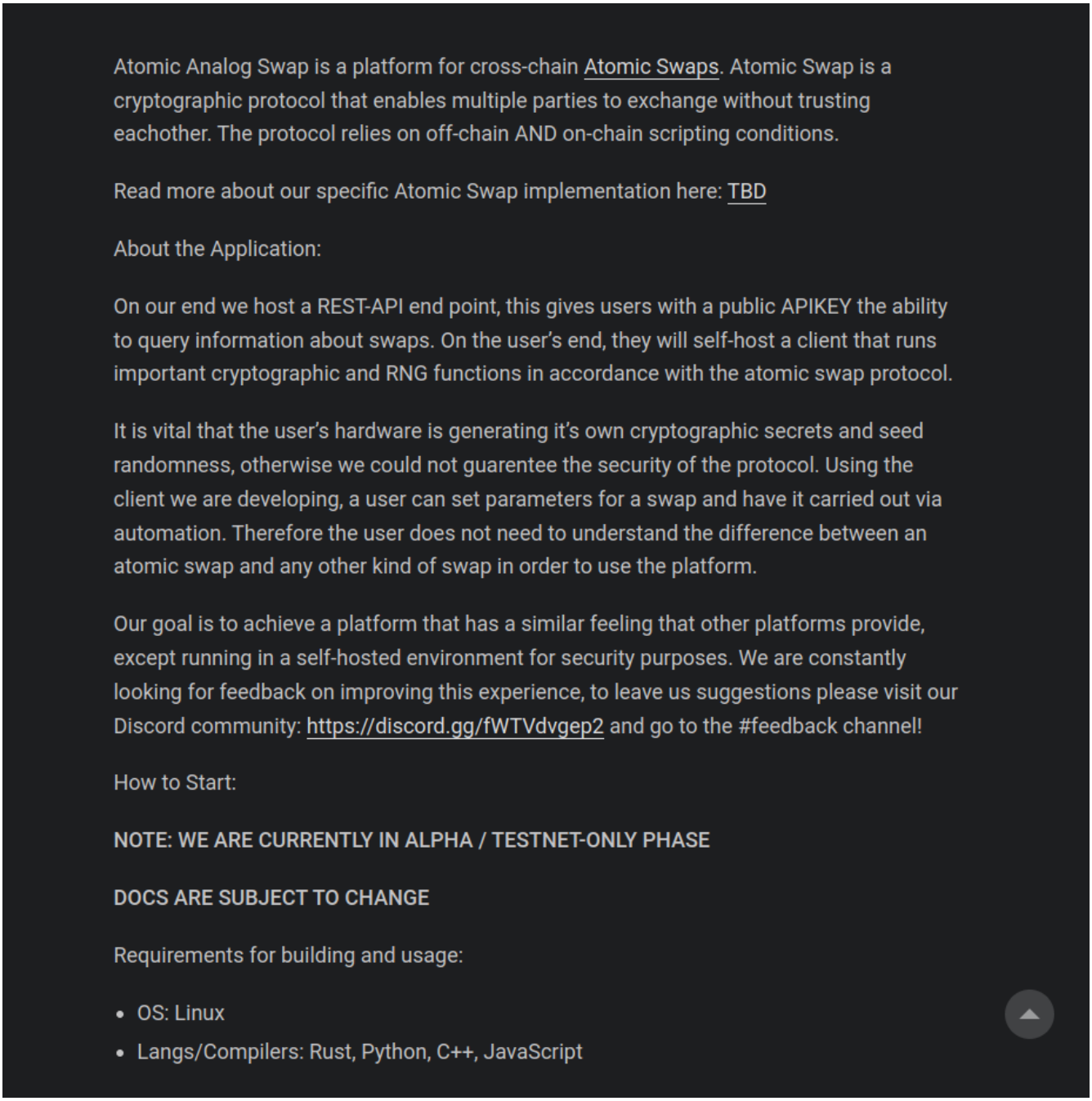
Atomic Analog Swap is a platform for cross-chain Atomic Swaps. Atomic Swap is a cryptographic protocol that enables multiple parties to exchange without trusting eachother. The protocol relies on off-chain AND...

Updated: October 2, 2023 · 2 min · 279 words · Atomic Analog Developers

Get Started

How to use Atomic Analog Swap

Updated: October 2, 2023 · 2 min · 279 words · Atomic Analog Developers | Suggest Changes



https://github.com/dzyphr/AtomicAnalogSwapWebsite

6. Conclusion

Hopefully its pretty clear that a lot of progress has been made towards the goal of making trustless cross chain swaps a reality for Ergo. I have not had a lot of help on this specific project, however I have had some crucial help from ecosystem developers who helped me get started with my first ErgoScript contracts in the last Ergo-Hack and lots of other supplementary assistance like that. To

any of the community members who gave your free time to assist me in learning how to develop on Ergo, thank you so much for your time and patience. Part of my goal for the future is to work on recruiting more interested and engaged developers for the project. This will be a huge step for improving the overall design and usability of the software, nevertheless I am committed to getting it working even if it means developing on my own for the beginning of the projects life cycle.

That being said as soon as Atomic Analog Swap is fully set up, I will be coordinating a test-net roll out for community members who are interested in testing and giving feedback. Those interested can join our Discord [https://discord.gg/r2xkzE3gaW] to stay up to date on the progress of this.

I will also be uploading the Docs / Info website around this time as mentioned earlier, we are resisting uploading this early because lots of the subject matter on there is going to be proof-read and updated for accuracy. Since this is financial tooling, making sure that we are giving sound information to end users is far more crucial than launching an inaccurate website to generate hype or artificial attention.

I have some future goals for the project including supporting more chains like Bitcoin, Monero, Grin, and other mainly proof of work focused Cryptocurrency Networks. I plan to use the majority of any potential prizes earned from this Ergo-Hack to bootstrap liquidity for the Atomic Swap marketplace. Other future projects I am thinking about are layer two applications and bridging applications that leverage the cryptography used in atomic swaps. I also intend to continue investing into Ergo's PoW mining layer, by increasing my personal hash-rate on the network and using those block rewards to fund the marketplace as well. Investing into community centric mining pools or developing them is also something I am interested in down the line. I hope it's clear we are doing this with an Ergo-Positive mindset, and a desire to increase the liquidity, value, and adoption of Ergo with our applications. Anyone interested in innovating in the space of peer to peer trading should absolutely contact me and talk about what is possible with the future of Atomic Swap marketplaces.

Resource Links:

Atomic API GitHub: https://github.com/dzyphr/atomicAPI

Local WebUI GitHub: https://github.com/dzyphr/AtomicAnalogsLocalWebUI

Client REST API GitHub: https://github.com/dzyphr/Atomic_Analogs_client_RESTAPI

Server REST API GitHub: https://github.com/dzyphr/Atomic_Analogs_server_RESTAPI

Docs / Info Site GitHub: https://github.com/dzyphr/AtomicAnalogSwapWebsite

Discord: https://discord.gg/r2xkzE3gaW