CrossChain: A Visual Monitor for Blockchain on IoT

Ryan Xu, Hakkyung Lee
McKelvey School of Engineering, Washington University in St. Louis
ryanxu@wustl.edu, hakkyung@wustl.edu

I. Introduction

It has been more than a decade since a blockchain was introduced to the world for the first time. With the emergence of Bitcoin, blockchain itself has been developed in various ways, and many trials have been going through to integrate blockchain into various fields. Meanwhile, the Internet of Things (IoT), a concept of connecting physical devices mutually via the Internet, started to appear around a similar period [1]. Since then, tech companies, regardless of their size, have dived into the IoT field and started releasing devices to connect the world like never before. With two new fields explosively arising over the last decade, a question naturally arises of what possibilities are there to combine these two fields? There exist many exciting proof-of-concept applications but rarely could we find actual steps on how to set up blockchain on an IoT device nor a researcher friendly platform for exploration [2, 3, 4, 5].

We present our study and experience of configuring current state-of-the-art blockchain packages on Raspberry Pi 3 Model B+ (RPi) and introduce CrossChain - a platform to help visualize and monitor the system impacts of different consensus mechanisms on RPi. Using CrossChain, a user can easily keep track of some blockchain information such as current block number along with a generator of the block, number of transactions within the generated block, and the size of the block. System information such as CPU usage and network data can be retrieved and visualized as well.

II. Compatibility of Distributed Consensus Algorithms on the IoT device

While trying to host several blockchains with a couple of configurations on RPis, the following consensus algorithms are considered to be successful on RPis: Proof-of-Work (PoW), Proof-of-Stake (PoS), and Proof-of-Authority (PoA). A brief overview is shown in Table I.

Ethereum Geth on Raspbian Stretch 32-bit (PoW) Geth, a Go implementation of Ethereum client, was installed without any error. By connecting each other, a private network was successfully configured and transactions could be sent from one to another. However, mining on the RPi was not possible due to an out-of-memory and memory allocation failure error. We are reasonably certain as of now Geth requires 64-bit for mining [6].

Ethereum Geth on Ubuntu 64-bit (PoW) To test mining, Ubuntu 18.04 for ARM64 was installed. After following the same previous steps, a private network was successfully set up. This time, mining on the RPi was possible but was extremely slow. In a private blockchain of three mining RPis, only 13 blocks were mined in the span of 18 hours, even with the initial difficulty of 0. The issue was that the RPis could not handle the default scaling difficulty so Geth had to be modified to have a hardcoded constant difficulty value. This allows us to adjust the block generation rate at a reasonable rate.

Ethereum Prysm on Ubuntu 64-bit (PoS) Prysm, the only PoS-supported Ethereum client, was also tested. Installing Prysm can be done in two ways: Docker and Bazel. Docker method did not work due to architecture compatibility. while compiling Bazel was successful via bootstrapping, building Prysm with Bazel was impossible due to the out-of-memory issue. Even after setting up a swap memory, RPi crashed in the middle of the building process.

Qtum on Ubuntu 64-bit (PoS) Qtum, a blockchain based on bitcoin, officially supports ARM architecture, so installing was relatively simple. However, to set up a private network, the regtest mode had to be used, which comes with slightly different settings with the mainnet. After the first 501 blocks, which have to be created manually, a block is created every 30 seconds. Also, some unexpected behaviors are noticed when tested with only one staking node in regtest.

Hyperledger Fabric on Raspbian Stretch 32-bit (CFT) Hyperledger Fabric, a popular permissioned blockchain for enterprise, uses Docker images to build the binaries. Unfortunately, up-to-date images built for ARM architecture was not found. Hyperledger Fabric community also stated that as of now Hyperledger Fabric does not officially support 32-bit [7]. On top of that, we recently noticed that the official wiki page recommends at least 4GB of memory for running Hyperledger Fabric, which is another obstacle for RPi. Since RPi does not meet the hardware requirement, running on different OS was not considered.

TABLE I: Current state of support of blockchain on RPi
(O : Supported △ : Partially supported X : Not supported)

	Ethereum (Geth)	Ethereum (Parity)	Ethereum (Prysm)	Qtum	Hyperledger Fabric
Availability	△	O	X	O	X
Private network	O	O	X	△	O
Consensus	PoW	PoA	PoS	PoS	CFT-based
Operating System	Ubuntu ARM64	Ubuntu ARM64	N/A	Ubuntu ARM64	N/A
Limitations	Processing power	Chain specific	Architecture, Memory	Chain specific	Architecture, Memory

Ethereum Parity on Ubuntu 64-bit (PoA) Parity, a Rust-based Ethereum client, supports PoA. Since Parity is officially distributed via Snap, an Ubuntu packaging manager, we moved to Ubuntu. Running a full node with setting up a private network was successful without any constraints. However, a validator must be selected beforehand, which invites trust and security questions in a real-world setting.

III. CrossChain Framework Design

Taking in our findings of the various blockchain implementations, we sought to create an introductory infrastructure to assist researchers in exploring blockchain and IoT. Introducing CrossChain, this platform supports examples of three consensus algorithms present - PoW, PoS, and PoA - on multiple RPis with Ubuntu 18.04 for ARM64. As explained before, Geth, Qtum, and Parity are used as clients, respectively. CrossChain has several aspects that can be useful to researchers.

First, thorough documentation found in the repository [8] helps how to install, set up, and run blockchains on the RPis. With limitations and incompatibilities in linking many conventional blockchains to RPis, this is a reliable and up-to-date tutorial and resource. This hopefully will allow researchers to quickly get a network up and running without going through trial-and-error with numerous chains.

Secondly, a basic Graphical User Interface (GUI) was created to help visualize some attributes that would be helpful to researchers. For each blockchain, a block number along with a miner, difficulty, number of transactions, and the size of each block can be fetched and graphed. Besides, the CPU usage of RPi in terms of percentage is graphed to comprehend the computing load on the device. Network data such as bandwidth, packets, latency, and jitter are also tracked. These attributes can be analyzed to determine how well blockchains operate on the RPis.

A user can add RPis that have been set up with the chains to CrossChain GUI and choose which attribute to be graphed. This helps visualize data in a comprehensible manner. Additionally, there is an option to log all the data collected, as well as to adjust the sample rate of network and system data. This data is packed into a CSV file that is categorized by the node, type of data, and consensus algorithm. Finally, switching between blockchains can be done via GUI. This will aid in making informed comparisons between different blockchains. Although GUI at the current stage is rather rudimentary, it is a viable starting point for exploring. A logic diagram can be found in figure 1 and figure 2 shows the GUI in action.

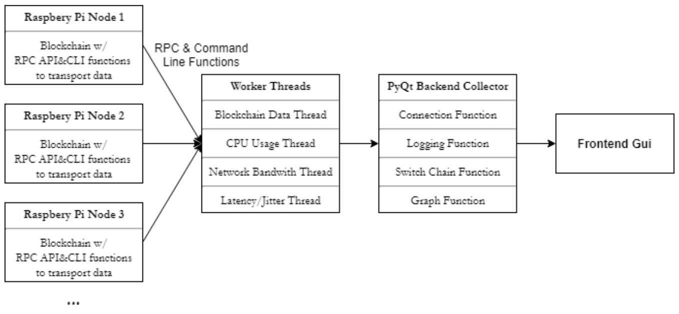

Fig. 1: CrossChain Logic Diagram

It should be noted with the current setup of CrossChain, the difficulty attribute is to be taken with a grain of salt. In PoW, since Geth is modified, the difficulty of the chain is fixed as a constant, thus the difficulty magnitude is the same for all blocks. In PoS, the difficulty does not affect the mining rate due to the feature of regtest mode. In PoA, the protocol does not use a difficulty parameter at all and the value is simply a placeholder. However, it is still possible to monitor when the nodes are connected to the outside network, such as mainnet and testnet.

IV. Summary and Future Work

We presented our experiences trying to configure popular blockchains on RPis. While some trials did not succeed for various reasons, we managed to install a blockchain for each of the three main consensus algorithms. A detailed explanation of how to set the chains up can be found in the project repository. We also provided a visualization tool, CrossChain, to aid researchers in exploring the reactions of the RPis while running blockchains.

There are a few avenues of future work. The first is adding more functionality to the GUI to make it more robust and automated. Secondly, as IoT devices become more powerful, it is intriguing to see how improved hardware can help the devices mine blocks more efficiently. At the time of writing this paper, the Raspberry Pi Foundation has released the Raspberry Pi 4, which comes with 4 GB of RAM. Knowing how much performance increases with hardware improvements may be important in future IoT and blockchain development.

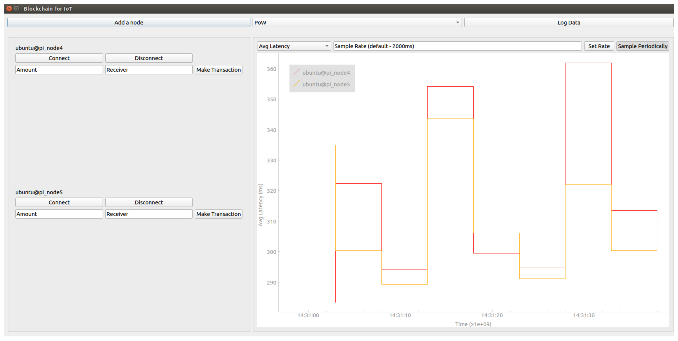

Fig. 2: Screenshot of the GUI graphing

References

[1] Evans, D. (2011). The Internet of Things - How the Next Evolution of the Internet Is Changing Everything. [online] Cisco. Available at: https://www.cisco.com/c/dam/en_us/about/ac79/docs/innov/IoT_IBSG_0411FINAL.pdf

[2] Conoscenti, Marco, Antonio Vetro, and Juan Carlos De Martin. "Blockchain for the Internet of Things: A systematic literature review." 2016 IEEE/ACS 13th International Conference of Computer Systems and Applications (AICCSA). IEEE, 2016.

[3] Malik, Sidra, et al. "TrustChain: Trust Management in Blockchain and IoT supported Supply Chains." arXiv preprint arXiv:1906.01831 (2019).

[4] Tang, Bo, et al. "IoT Passport: A Blockchain-Based Trust Framework for Collaborative Internet-of-Things." Proceedings of the 24th ACM Symposium on Access Control Models and Technologies. ACM, 2019.

[5] Kshetri, Nir. "Can blockchain strengthen the internet of things?." IT professional 19.4 (2017): 68-72.

[6] Szilgyi, Pter "Go-Ethereum Issue 14633" Github Repository. Available at: https://github.com/ethereum/go-ethereum/issues/14633

[7] Wang, Jiahua "Failing to build fabric docker images on 32-bit ARM architecture" Hyperledger. Available at: https://jira.hyperledger.org/browse/FAB-10454

[8] Xu, Lee. "CrossChain" GitLab Repository. Available at: https://gitlab.com/shellb34r/crosschain