Distributed Internet of Things Applications Experimentation Tool

Jonas L. de Vilas Boas, Antônio M. Alberti, Joel J. P. C. Rodrigues

Abstract— **The complementarity of distributed ledger technologies such as Blockchain and the Internet of Things has gained increasing attention in the literature. In order to facilitate the immutable validation and registration of the data collected by the devices, the so-called smart contracts have been used and distributed applications have been developed to increase the adhesion of the actors. Thus, this work presents a tool for experimentation to evaluate the performance and scalability of distributed applications in the context of the Internet of Things. The proposed tool is flexible and allows experimentation with different technologies and configurations. This work contributes to the convergence among immutable information recording, deterministic computation, and sensing of physical world quantities.**

Keywords— **Blockchain, Distributed Ledger Technology, Internet of Things, Smart Contracts, Distributed Applications.**

I. Introduction

Distributed Ledger Technologies (DLTs) have revolutionized the storage and distribution of digital information. In the financial sector, this technology has been used to create a network of trust among participants, removing the need for a central regulatory authority. Since its first implementation, proposed in [1], the characteristics of data immutability and transparency, as well as self-regulation and consensus among the participants were already present and are evolving with each new solution.

In the context of the Internet of Things (IoT), the use of DLT can be very advantageous. IoT applications bring weaknesses and vulnerabilities that can be addressed by using DLTs. For example, Peer to Peer (P2P) networks maintained by DLTs solve the single point of failure problem common in Cloud-IoT architecture and the problem of attacks for interception and data tampering can be solved by encryption mechanisms and consensus promoted by DLT nodes [2]. Despite the integration problems of these two technologies, such as the hardware limitation of the devices and the computational requirement of the mechanisms of the DLTs [3], standards have been proposed to bring good practices in the use of DLTs in applications of IoT [4].

On the other hand, the second generation of DLTs [5] brought the Smart Contracts (SCs), allowing the programming of deterministic routines (which cannot be changed after being published on the network) to facilitate the registration and consultation of data. Among the most diverse applications, SCs have been used to mediate communication between IoT solutions and DLTs [6]. In addition, Distributed Applications (DApps) are based on SCs to deliver a user-friendly solution for monitoring chain actors to interact with data in the DLT [7].

In research on 6G networks, which considers scenarios with a large number of active devices, the complications of IoT applications are exponentially increased [8]. DLTs nodes can be deployed at the edge and fog of 6G networks [9] and have the potential to meet the security, decentralization, and immutability requirements needed for these networks [10]. In Brasil 6G project [1], DLT nodes will be deployed at the edge of the network to provide different services, especially for IoT data management.

Despite the number of proposals involving DApps and SCs in the IoT scenario, there is still a need for simulation, experimentation, and evaluation tools for these solutions. These tools evaluate the performance and scalability of solutions in a controlled environment, which is essential for testing the operation of applications in general, but also for surveying infrastructure requirements, identifying critical situations, and developing contingency plans. Different solutions have already been studied and analyzed in the literature [11], but they have limitations, such as results that are far from reality and low flexibility, for example. In this context, this work presents a simple experimentation tool to evaluate the performance and scalability of DApps in the context of IoT. The goal is to bring the benefits of DLTs to the IoT world flexibly and practically. Thus, the proposed tool meets the following requirements:

- Flexibility: It allows for different device configurations and network topologies, as well as different DLTs. Setting up different experiments should be simple, just configuring the startup parameters.
- Easy instantiation: It runs the necessary procedures and set up the experiment environment automatically, given the input parameters, without the need for manual configurations.
- Similarity with a real scenario: It closes to the maximum of the operation in a real environment, so that the collected results can more accurately reflect the performance of the applications.

To meet these requirements, the proposal combines different technologies and tools, such as Docker, the Cooja simulator,

J. L. Vilas Boas, Information and Communications Technology Laboratory, Instituto Nacional de Telecomunicações, Santa Rita do Sapucaí - MG, e-mail: jonas.lopes@dtel.inatel.br; A. M. Alberti, Information and Communications Technology Laboratory, Instituto Nacional de Telecomunicações, Santa Rita do Sapucaí - MG, e-mail: alberti@inatel.br; J. J. P. C. Rodrigues, Senac Faculty of Ceará, Fortaleza-CE, Brazil; Instituto de Telecomunicações, Portugal, e-amil: joeljr@ieee.org. Este trabalho foi parcialmente financiado pela RNP, com recursos do MCTIC (01245.010604/2020-14).

[1]https://inatel.br/brasil6g/

Shell Script, Python commands, and the Web3 library. With this, it is possible to assemble a complete scenario of a DLT-based IoT application in a flexible and easy-to-configure way.

The organization of the rest of this work is described below. Section II presents some important concepts for understanding the proposal. In Section III some related works are discussed. The IV section presents the proposed tool and the V section presents a usage scenario, validating the tool and illustrating the convergence between IoT, DLTs, and SCs. Finally, Section VI presents the final considerations of this work.

II. Background

The concepts of IoT and DLT are introduced in this section, as well as aspects of SCs and DApps and how each of these concepts connects. In addition, some tools and implementations used in the scope of this work are presented.

The term IoT [12] is used to describe a series of information and communications technologies organized in an architecture that allows everyday objects to collect data related to the physical properties of the environment in which they are inserted and transmit this data to human users or other objects. Furthermore, this data can be applied to control these objects, changing their state and behavior. The components that support this architecture are generally simple low-cost microcontrollers with embedded software, equipped with batteries, sensors, actuators, and communication technologies, usually wireless.

In the proposed tool, the Contiki Cooja simulator is applied, an open-source operating system for IoT devices with limited resources, low cost, and low power consumption. Several different Contiki libraries can be compiled and loaded in the same Cooja simulation, representing different types of sensor nodes, forming heterogeneous networks. Node firmware can be implemented to send data externally using TCP/UDP ports.

DLT [13] was proposed to serve as a decentralized digital ledger, that is, a kind of database maintained by multiple participants. This makes data breaches more difficult as each participant has a copy of all records. This technology is distributed across multiple users using P2P networks (overlay networks where computers communicate and exchange data with each other directly) so that all changes in the ledger are reflected in all copies on the network. What makes the technology interesting is the possibility of establishing a consensus protocol, applying transaction rules to avoid conflicting information, and allowing the nodes of its distributed computer network to self-inspect the entire operation, without the presence of a central server or certification authority, quickly and globally. The Blockchain technology [1] is considered a DLT, which is distinguished by using a specific data structure, which consists of a chain of blocks, where each block is linked to the hash code generated from the content of a previous block. Currently, there are also proposals for DLT based on Directed Acyclic Graph (DAG), where transactions are linked directly, without the need to form blocks. The objective is to make the rate of transactions accepted per second higher, in addition to making the consensus mechanisms lighter.

SCs [14] are tools for structuring rules in programming code, registered in a DLT, such as Blockchain. The main purpose of SCs is to automate the execution of rules and clauses, in addition to allowing those involved to follow the process of execution of each clause. SCs work in such a way that when a predetermined event occurs, another predetermined action will occur automatically, safely, and immutably. In practical terms, the SC "code" is compiled, generating an Application Binary Interface (ABI), and is recorded in a DLT as a transaction. Thus, each node that has access to the records can interact with this interface, executing SC methods. Because they run from DLT, SCs offer a higher level of security, as they cannot be changed at runtime (this concept is also known as deterministic execution, since in current operating systems there is no guarantee that the running program has not changed).

In this scenario, without centralized servers, data can be accessed and manipulated by a DApp. These applications are usually installed on the device of the interested users, avoiding the need for a hosting server, and can implement communication mechanisms with any node on the network directly, accessing the information registered in the wallet (ledger) or entering new transactions, using the credentials of users on the network in question, usually mediated by SCs.

When building the tool, the DLT nodes and applications, such as the Cooja simulator, are instantiated using Docker containers. Docker is a set of platform-as-a-service products that use operating-system-level virtualization to deliver software in packages called containers. Containers are isolated from each other and bundle their programs, libraries, and configuration files. This technology allows for automation of the configuration and initialization of the networks and components used in the experimentation processes.

III. Related Works

Other works in the literature have already proposed to implement test networks to analyze the performance or scalability of solutions in this scenario. The most famous solution is BlockSim [15]. Although the solution has the advantage of being much lighter than full scenarios, such as testbeds, and being suitable for performance analysis of consensus mechanisms, it has some disadvantages such as difficulty in extension, failure to monitor all relevant aspects, and important metrics in a Blockchain system. In [16], an extension to BlockSim was proposed, called BlockPerf. The extension is based on six layers (Application, Contract, Incentive, Consensus, Node/Data, and Network) and, despite achieving more realistic results compared to BlockSim, the solution still has some limitations. It does not implement support for other non-Blockchain-based data structures such as DAG. In addition, the solution also does not support testing with Smart Contracts, which are very important in current applications.

In [17] a tool for experimenting with proposals for future Internet architectures in the IoT context is presented, where the eXpressive Internet Architecture (XIA) was evaluated. In the proposal, Docker images were used to implement a network and monitor data traffic between client nodes and IoT gateways, emulated through the Cooja tool. In [18] an improvement of the tool was proposed to make the architecture used in the experimentation more flexible and a significant increase in the scale of the experiments. The proposed solution presents a flexible and extensible structure, being able to be adapted for tests with different architectures and network topologies.

IV. Proposed Tool

The tool proposed in this work assists in the configuration, execution, and monitoring of an experimentation environment for DApps in the IoT context. The goal is to evaluate the scalability of a solution that is based on SCs deployed in a DLT to intermediate the registration and query of data provided by IoT devices. The tool is based on open-source programs and Linux shell script commands. In addition, the tool is an adaptation of the solution presented in [18], taking advantage of the network structure and making the necessary modifications to meet the DLT scenario. The source code of the tool are available on github [2].

Before running the tool, it is necessary to include the SC ABI and bytecode and to create Docker images of the components. After that, the configuration parameters are given, and the tool performs the activities necessary to create the experimentation environment, starts the experiment, and collects the results, as Figure 1 shows. Tool activities are coded in Shell Script and parameters are provided at the command prompt during code execution, followed by indication flags, as shown in Table I. Given the example values shown for the parameters, Figure 2 illustrates the architecture of the environment generated by the tool. The design decisions are presented below, as well as the details of each of the tool components.

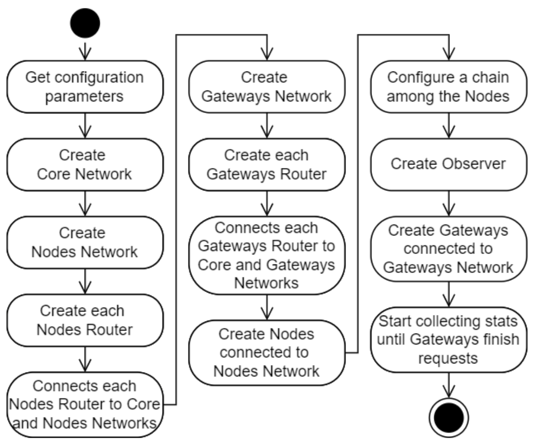
Get configuration parameters
Create Core Network
Create Nodes Network
Create each Nodes Router
Connects each Nodes Router to Core and Nodes Networks
Create Gateways Network
Create each Gateways Router
Connects each Gateways Router to Core and Gateways Networks
Create Nodes connected to Nodes Network
Configure a chain among the Nodes
Create Observer
Create Gateways connected to Gateways Network
Start collecting stats until Gateways finish requests

Fig. 1. Proposed tool activities.

A. Design decisions

The proposal provides scalability and flexibility of experimentation scenarios, efficiently using hardware resources to validate the performance of DApps in scenarios with a large number of participating nodes. For this, the Docker platform was adopted to perform the virtualization of each component of the architecture. Containers can share resources among themselves and have performance and resource utilization optimization advantages compared to virtual machines. Furthermore, they can be easily instantiated and replicated.

[2]https://github.com/jonaslopess/diotappet

TABLE I

Tool configuration parameters.

Flag	Parameter	Value
-N	Number of routers in the DLT network	1
-n	Number of DLT nodes per router	1
-G	Number of routers in the gateways network	1
-g	Number of gateways per router	1
-m	Number of motes per gateway	9
-r	Number of requests	100

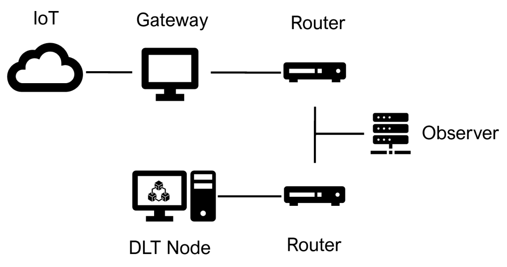

Fig. 2. Architecture of an example environment generated by the tool.

To emulate the IoT devices, the Cooja tool was used, which implements the Contiki operating system in the IoT nodes. With this tool it is possible to run different programs on the sensor nodes, allowing flexibility both in the topology of the simulated device network and in the behavior of each of the emulated nodes. The selected link-layer technology was 6LoWPAN, to facilitate the communication with the motes. The scripts to enable communication between devices and DLT nodes and to automate the deployment of SCs were implemented in Python, using the web3.py library.

Finally, to implement each of the nodes of the DLT network, IOTA's open source software was chosen. The nodes will form the network needed to deploy the DApp support SCs and perform the consensus mechanisms for recording transactions. This DLT was chosen because of its great synergy with applications in the context of IoT, allowing for lightweight consensus mechanisms, high transaction rates per second, and fee-less. Currently, at version 2.0, the IOTA network is evolving into a truly decentralized version (early versions use nodes maintained by the IOTA foundation that serve as coordinators). In version 2.0, network nodes, called validators, will form committees to carry out the consensus mechanisms. Wasp is the implementation of IOTA 2.0 validator nodes, created to enable the deployment of SCs. It is possible to form a network of Wasp nodes, with committees for the validation of transactions and to connect this network with the public network of IOTA through Goshimmer nodes. The Goshimmer nodes form the main network that keeps the network state up to date, enabling communication between different Wasp networks. In this tool, a dummy Goshimmer node was used to represent the main net and a private Wasp network was created to deploy the SCs.

B. Components

The tool is composed of four components built as Docker images. Starting from these images, the components can be instantiated as containers and the topology can scale with the replication of these containers. All images are based on Ubuntu operating system. Docker networks are created to allow connection between them. The components are described following.

- DLT Node: Runs the DLT software to communicate with other network nodes to perform the validation and consensus mechanisms. Provides communication interfaces for interacting with SCs.
- Router: Implements network routing functions to forward packets from IoT devices to DLT nodes. The two types of routers needed in the experimentation environment are instantiated from the same image based on different input parameters for configuration. Images are configured to send data to routers, ignoring default configuration.
- Observer: It is configured as a Docker volume, used as a file repository shared between containers. Test results are stored in this repository, as are configuration files. For architectural simplification purposes, this component is also used as a Remote Procedure Call (RPC) provider, necessary to provide interaction routines with the DLT nodes.
- Gateway: Runs the IoT network simulation tool, composed of IoT Motes that emulate Contiki nodes. Motes are configured as echo servers, that is, they receive and return the same data, just to check network latency. One of the IoT Motes is configured as an edge router and communicates with the Gateway through the tunslip6 application. Also, when creating the Gateway, scripts are executed to deploy an SC in the DLT. Each Gateway interacts with a specific SC. Finally, an application in the Gateway serves as an echo client, sending requests to random Motes and, upon receiving the Mote's response, interacts with the SC, through its address, to register the data in the ledger maintained by DLT nodes. All interaction with the DLT happens through the RPC provider present in the Observer.

V. Use Case

This section illustrates the application of the tool in an experimentation environment. The DApp to be analyzed in this experiment simply records dummy data collected by the devices' sensors to keep a track of some environmental conditions. This DApp can be used in the logistics of products with restrict transport and storage conditions. For this, a SC must be developed with methods that help in these operations. Two SCs were developed to have their performance compared. The diagram in Figure 3 shows SC 1, only with methods to record and return the temperature value. The diagram in Figure 4 shows the more flexible SC 2, in which each device can have a list of properties to be monitored (sensors such as temperature, brightness, and humidity, for example). The properties are set by the *addMonitoredProperty* function. SC can update the value of a given property (*setMonitoredProperty*) or return the value of that property (*getMonitoredProperty*). With the proposed tool, it is possible to analyze the impact on the scalability of this DApp with the use of each SC.

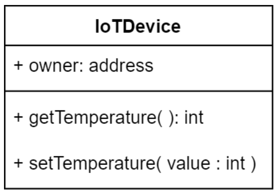

Fig. 3. Smart Contract 1 applied to the proposed scenario.

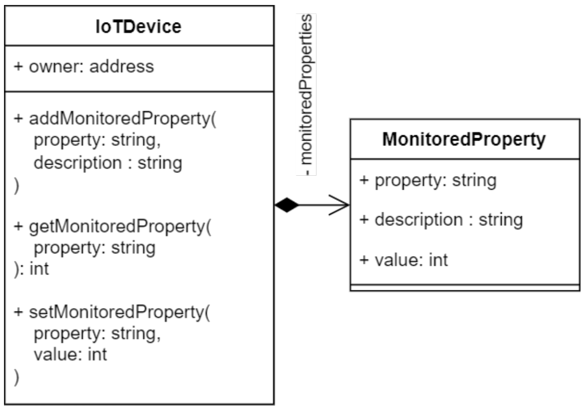
IoTDevice
+ owner: address
+ addMonitoredProperty(property: string, description : string)
+ getMonitoredProperty(property: string): int
+ setMonitoredProperty(property: string, value: int)
- monitoredProperties
MonitoredProperty
+ property: string
+ description : string
+ value: int

Fig. 4. Smart Contract 2 applied to the proposed scenario.

The first step to run the experiment is to insert the SC to be evaluated in the system. The SC must be compiled, generating the files with the ABI and the bytecode, necessary in the process of deploying SC replicas and interacting with their functions. Both files are placed in the gateway files repository to be copied during image creation and later replicated in each container. The Python script of the gateway must also be modified to define which SC method(s) should be called. In this case, for the first SC to be analyzed, it was defined that the *setTemperature* method must be called on each request, passing a random integer value as a parameter. For the second SC, it was defined that the *addMonitoredProperty* method must be called initially, passing the "Temperature" parameter and the *setMonitoredProperty* method must be called on each request, passing the "Temperature" property and a random integer value as parameters.

Subsequently, the Docker images of each component must be generated. Images can be assembled only once and used for different experimentation scenarios, except for the gateway image, which must be generated again whenever a new SC is to be evaluated. With the images mounted, the configuration parameters are adjusted for the desired experimentation environment. Table I shows the parameters used in the proposed scenario. In the example, a simple scenario was defined with only one DLT node, connected to a router. One gateway will be instantiated, will be connected to another router, and nine motes will be initialized in the Cooja application. The Python script will perform 100 requests to random motes and send data to the SC.

Once the tool is started, the containers are instantiated and the network between them is established. Then the settings to establish a chain between the nodes of the DLT are made. When starting each of the gateways, the SCs are deployed and a new monitored property is added. Then, the gateway application starts to interact with the motes and with the SC, to record the fictitious temperature in the DLT. As soon as the last request is made, the tool ends, recording the data about the system, collected during the experiment, and removing all created containers.

Table II shows the results collected during the execution of the experiment for each of the analyzed SCs. As shown, the second SC takes an average of 5.65 seconds to register temperature data on the DLT, taking 799 seconds to register all 100 requests, while the first only takes an average of 0.7 seconds, taking 293 seconds in total. In addition, if the DLT adopted has transaction costs, a difference of 4331.48 GAS [3] on average between the two SCs. The volume of data transmitted is also bigger in the second scenario. Although the second SC is more flexible and can be applied in scenarios with different sensors, the first SC is more efficient.

TABLE II

COLLECTED RESULTS.

Parameters	**SC1**	**SC2**
Average delay to register data to DLT	0.7 s	5.65 s
Average cost to register data to DLT	28811.32 GAS	33142.8 GAS
Experiment total time	293 s	799 s
Total data flow on Rpc	1620 kB	1786 kB

VI. CONCLUSION

A new experimentation tool for DApps in the context of IoT was proposed in this work. The tool is flexible and allows the use of different device configurations and network topologies. The use case shows that the tool allows for quick setup and startup. Just by entering the ABI and bytecode of the SC to be analyzed, informing the methods that must be invoked, and informing input parameters, the experiments can be easily modified and scaled. The collected results can be analyzed easily.

Although the tool allows the use of different DLTs, this work only considers the software of the IOTA network. It is necessary to evaluate the use of the tool for other DLTs in future works. By fixing the topology, the number of components, and the SC structure, it will be possible to compare the performance of different DLTs for a given application.

ACKNOWLEDGEMENTS

Acknowledgment to Ramon P. Chaib, Tiago B. da Silva and Antonio M. Alberti for providing the code that served as the basis for this work. This work was partially supported by RNP, with resources from MCTIC, Grant No. No 01245.010604/2020-14, under the 6G Mobile Communications Systems project of the Radiocommunication Reference Center (Centro de Referência em Radiocomunicações - CRR) of the National Institute of Telecommunications (Instituto Nacional de Telecomunicações - Inatel), Brazil; by FCT/MCTES through national funds and when applicable co-funded EU funds under the Project UIDB/50008/2020; and by Brazilian National Council for Scientific and Technological Development - CNPq, via Grant No. 313036/2020-9.

[3]GAS is the unit used in Ethereum to represent the amount of computational power used to register a transaction on the ledger.

REFERENCES

[1] S. Nakamoto *et al.*, “Bitcoin: a peer-to-peer electronic cash system (2008),” 2008.

[2] B. Farahani, F. Firouzi, and M. Luecking, “The convergence of IoT and distributed ledger technologies (DLT): Opportunities, challenges, and solutions,” *Journal of Network and Computer Applications*, vol. 177, p. 102936, Mar. 2021.

[3] A. Reyna, C. Martín, J. Chen, E. Soler, and M. Díaz, “On blockchain and its integration with IoT. challenges and opportunities,” *Future Generation Computer Systems*, vol. 88, pp. 173–190, Nov. 2018.

[4] A. Panarello, N. Tapas, G. Merlino, F. Longo, and A. Puliafito, “Blockchain and IoT integration: A systematic survey,” *Sensors*, vol. 18, p. 2575, Aug. 2018.

[5] S. Aggarwal and N. Kumar, “Blockchain 2.0: Smart contracts,” in *Advances in Computers*, pp. 301–322, Elsevier, 2021.

[6] Z. Gao, Z. Zhuang, Y. Lin, L. Rui, Y. Yang, C. Zhao, and Z. Mo, “Selectstorage: A new oracle design pattern on blockchain,” in *2021 IEEE 20th International Conference on Trust, Security and Privacy in Computing and Communications (TrustCom)*, pp. 1177–1184, IEEE, 2021.

[7] W. Cai, Z. Wang, J. B. Ernst, Z. Hong, C. Feng, and V. C. M. Leung, “Decentralized applications: The blockchain-empowered software system,” *IEEE Access*, vol. 6, pp. 53019–53033, 2018.

[8] R. Sekaran, R. Patan, A. Raveendran, F. Al-Turjman, M. Ramachandran, and L. Mostarda, “Survival study on blockchain based 6g-enabled mobile edge computation for IoT automation,” *IEEE Access*, vol. 8, pp. 143453–143463, 2020.

[9] T. Nguyen, N. Tran, L. Loven, J. Partala, M.-T. Kechadi, and S. Pirttikangas, “Privacy-aware blockchain innovation for 6g: Challenges and opportunities,” in *2020 2nd 6G Wireless Summit (6G SUMMIT)*, IEEE, Mar. 2020.

[10] T. Hewa, G. Gur, A. Kalla, M. Ylianttila, A. Bracken, and M. Liyanage, “The role of blockchain in 6g: Challenges, opportunities and research directions,” in *2020 2nd 6G Wireless Summit (6G SUMMIT)*, IEEE, Mar. 2020.

[11] R. Paulavicius, S. Grigaitis, and E. Filatovas, “A systematic review and empirical analysis of blockchain simulators,” *IEEE Access*, vol. 9, pp. 38010–38028, 2021.

[12] J. Gubbi, R. Buyya, S. Marusic, and M. Palaniswami, “Internet of things (IoT): A vision, architectural elements, and future directions,” *Future Generation Computer Systems*, vol. 29, pp. 1645–1660, Sept. 2013.

[13] D. Burkhardt, M. Werling, and H. Lasi, “Distributed ledger,” in *2018 IEEE International Conference on Engineering, Technology and Innovation (ICE/ITMC)*, pp. 1–9, IEEE, June 2018.

[14] S. Rouhani and R. Deters, “Security, performance, and applications of smart contracts: A systematic survey,” *IEEE Access*, vol. 7, pp. 50759–50779, 2019.

[15] C. Faria and M. Correia, “BlockSim: Blockchain simulator,” in *2019 IEEE International Conference on Blockchain (Blockchain)*, IEEE, July 2019.

[16] J. Polge, S. Ghatpande, S. Kubler, J. Robert, and Y. L. Traon, “BlockPerf: A hybrid blockchain emulator/simulator framework,” *IEEE Access*, vol. 9, pp. 107858–107872, 2021.

[17] R. P. Chaib and A. M. Alberti, “A simple solution for iot experimentation in the context of future internet architectures,” in *Anais do VIII Workshop de Pesquisa Experimental da Internet do Futuro*, SBC, 2017.

[18] T. B. da Silva, R. P. dos Santos Chaib, C. S. Arismar, R. da Rosa Righi, and A. M. Alberti, “Toward future internet of things experimentation and evaluation,” *IEEE Internet of Things Journal*, vol. 9, pp. 8469–8484, June 2022.