Exploring blockchain-based management for LoRa IoT nodes

Eloi Cruz Harillo and Felix Freitag[0000−0001−5438−479X]

Universitat Politècnica de Catalunya, BarcelonaTech, Spain
{eloi.cruz}@estudiantat.upc.edu
{felix.freitag}@upc.edu

Abstract. We consider open IoT applications in which the IoT devices, i.e. sensor nodes and gateways, can be contributed by volunteer participants. Since the contribution of sensor deployments and their data is critical for the value of such IoT applications, we propose to account for each participant her donation of both the measured data and the hardware, and translate this information into rewards to create incentives. The system architecture we propose uses a blockchain-based backend and aims to extend properties of the blockchain to the IoT sensor layer in order to achieve the trusted accounting of each contribution. A potential use case that can benefit from this concept are environmental monitoring applications that need to integrate IoT sensor nodes from third parties for increasing their geographic reach.

Keywords: LoRa, Blockchain, IoT

1 Introduction

Today's IoT applications often consist of full-stack applications that integrate sensor nodes which deliver the measured data. While most of the software is hosted at cloud-based services, the sensors are at remote locations, often without Internet access. Specific communication technologies of the IoT are used in order to connect these sensors with gateways. A popular solution for this communication is LoRa, a low power technology which can establish links of several kilometers between nodes and transmits small amounts of measurement data in LoRa packets [1].

Environmental monitoring applications are of a growing need for our society in order to better understand and control the effects of our actions on the surroundings. However, covering all monitoring needs by commercial IoT applications is costly and unfeasible for many stakeholders, such as communities of citizens, regional governments and municipalities. An alternative to a commercial solution is to build systems in which citizens can be involved, such as successfully shown in volunteer computing, where citizens contribute to scientific tasks [2]. For environmental monitoring applications this idea translates into open applications in which anybody can contribute with data and devices. Indeed, many valuable geographic locations of interest for data collection might

be privately owned and are not publicly accessible, and the application would benefit if citizens are enabled to collaborate in the monitoring of the data. This later aspect relates to data collection from volunteer monitoring within citizen science [3].

The trustworthiness of the measured data in such open monitoring applications is critical for the data to be of value. Imagine a person that has the suspicion that a certain area is exposed to an elevated level of a pollutant. An open IoT application might be created driven by a citizen initiative in order to monitor this area. However, in order for the measured data to be of value, the data generated by the sensors needs to be trusted. While sensors can be calibrated to obtain correct measurements such as shown in [4], guarantees for the origin of the data and protection against manipulation, i.e. the immutability of the data, must be given by the design of the components of the system.

The challenge we address in this paper is the design of components for an open IoT application to achieve a trusted accounting of the contributions of sensors and gateways by participants to an IoT network, in order to use this information for being able to incentivize such contributions with rewards. We present the prototype of a system based on the vision of LoRaCoin [5], which consists of a blockchain-based application to manage IoT sensors of a low-power wide-area network. The system aims to account both for IoT sensor nodes that generate the data and the gateways that provide the Internet connectivity to these sensor nodes.

2 Background and related work

One of the successful applications of blockchain is its use for the decentralized accounting of contributions. Several practical applications such as GridCoin[1], to reward volunteer computing contributions, and FileCoin[2], to reward storage contributions, are used in production with real communities of users. The concept is that third parties can contribute their computing resources to a system, the system measures this contribution, and the amount of the contribution is rewarded with coins. Blockchain technology is used to transparently account the contribution and to create specific coins that are rewarded.

Blockchain-based smart contracts can also be used to perform tasks of coordination of resources. In [6] blockchain is proposed to support the management of a pool of edge computing resources in an IoT application. In that work the KubeEdge container orchestration engine integrates an interface with smart contracts in order to provision edge resources for running software components on demand. The benefit are IoT applications which can more flexibly adapt to execution time and power consumption requirements. An operational application example in which the coordination capacity derived from the blockchain is lever-

[1] `https://gridcoin.us/`
[2] `https://filecoin.io/`

aged is the provision of computing resources offered through a marketplace in iExec[3].

Incentivizing the provision of IoT hardware, specificially LoRa gateways, is the objective of the Helium network[4]. Different from the above mentioned examples of GridCoin, Filecoin and iExec, in which computing resources connected to the Internet are contributed at the exchange of coins, in Helium the target is that third parties contribute LoRa gateways, which enable that data from sensor nodes can reach the Internet. It is interesting to mention in this context that LoRa is a technology that does not need any licence or network operator, different for instance to NB-IoT or Sigfox. Due to this fact IoT applications that integrate LoRa networks have the potential to grow organically in the IoT layer with the contributions from everybody.

Most IoT applications which integrate a LoRa-based network apply the LoRaWAN standard [7]. In LoRaWAN, the LoRa network consists of LoRa gateways and LoRa end nodes. LoRa gateways have two network interfaces, one northbound connected to the Internet and one southbound to receive LoRa packets. The end nodes (e.g. sensor nodes) do not have Internet connectivity and use LoRa communication to reach the gateway. The gateway transmits the received data messages to higher levels of the IoT application.

Possibly the largest worldwide deployment of LoRaWAN-based IoT applications has been done through The Things Network (TTN)[5], which reports to have nowadays more than 170k members and more than 20k gateways. While the company offers an Enterprise Edition, the Things Stack Community Edition is popular among makers for registering LoRaWAN nodes and gateways of personal IoT applications which then leverage the centralized backend services of The Things Industries.

Several works have combined blockchain with LoRa-based IoT applications. In [8] a decentralized pollution monitoring system based on blockchain is proposed. The authors detail a list of non-functional requirements that need to be fulfilled in order to achieve trusted data from the IoT nodes. Another IoT pollution monitoring system using LoRaWAN and blockchain is presented in [9]. The focus is on applying smart contracts of the Ethereum blockchain to provide data integrity without the need for a Trusted Third Party. In the work of Ribero et al. [10] a permissioned blockchain is used to replace a component of the LoRaWAN stack, specifically the Join Server, responsible for the management of the keys of the LoRa nodes. The benefit of the solution is avoiding a single point of failure while similar execution and latencies are obtained. In our preliminary work described in [5] we introduced the idea of LoRaCoin. Our current work builds upon that early vision and describes the developed prototype.

[3] `https://iex.ec/`
[4] `https://www.helium.com/`
[5] https://www.thethingsnetwork.org/

3 Application design

3.1 Architectural overview

A LoRa-based IoT application typically consists of the IoT device layer where data is produced, a gateway layer which interconnects the data from the IoT devices with the Internet, and a cloud-based backend where the data is processed, analyzed and stored.

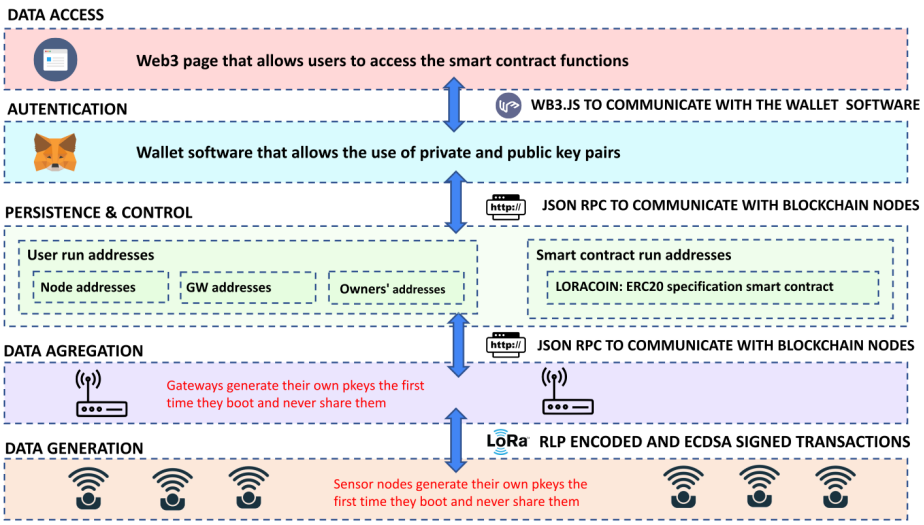

Fig. 1. Overview layered architecture.

Figure 1 gives an overview of the architectural layers on which our application design is based. The two layers at the bottom, i.e. the data generation and data aggregation layer, correspond to the functions performed by the IoT devices of the application. The nodes in the data generation layer obtain sensor data that is transmitted by LoRa packets to the gateway. The gateways, belonging to the data aggregation layer, have Internet connectivity and are able to communicate these data to higher layers of the application. The persistence and control layer is formed by the smart contracts hosted on the blockchain, providing both persistence of data and control of the operations. The authentication layer is required to validate certain write operations performed by the the smart contracts. For the authentication the Elliptic Curve Digital Signature Algorithm (ECDSA) is used. This has an interesting effect as it removes the need for the users to be registered to the platform and thus disclosing personal information. The data access refers to the graphical frontend of the application to perform the user-oriented interactions.

3.2 Functional analysis

We analyze several functions of the application with regards to the interaction of the components in the prototype.

Creating an IoT device: All IoT devices that connect to the smart contract of the application have to be created before. Creating a device is doneby adding the device address to a list of valid devices kept by the smart contract. The function *createDevice* of the communication interface of the smart contract is invoked for this purpose. This call can only be made by the provider of the device. The steps are as follows: The device runs a routine which creates its address that then appears at the OLED screen. Through the application's graphical user interface, the hardware provider registers the address of the device which is stored in a variable of the smart contract. Initially, the authorized hardware providers are registered at the smart contact.

Rewarding a sensor node owner for data contributions: The transactions containing sensor data are produced when LoRa nodes capture data that is send to the gateway and from there to the smart contract. As step 1 it is necessary that a gateway is made available that provides network coverage to the LoRa sensor node. This gateway then is able to receive LoRa packets from its LoRa interface. In case of being it a first interaction between sensor node and gateway, the next two steps are performed: When the node initializes for the first time (step 2), it generates a private key that is saved in the flash memory. The key is used to sign transactions. In step 3 the sensor node sends a signed message to the gateway and from there this message is sent to the smart contract for the gateway to obtain confirmation that the sensor node is a registered device. In step 4 the sensor node sends a signed message to the gateway to validate that the gateway is a registered device. The gateway sends a signed acknowledgement back to the node which in addition contains the signature of the hardware provider, allowing the LoRa node to validate the gateway. After having done this mutual validation, the LoRa node sends a LoRa data packet to the gateway in step 5, which is encapsulated in the http requests that the gateway sends to the smart contract. After registering the data, in step 6 the node owner is rewarded.

Rewarding a gateway owner for the relaying of a data message: When the gateway sends a message to the smart contract, it is validated by the signatures contained in the message and a transaction is written to the blockchain by *sendTransaction* whose hash is returned to the gateway. In order for the gateway to claim the reward, the gateway invokes the method *registerTransaction* with the parameter that contains the previously obtained hash. That allows to validate the contribution of the gateway and then to release the reward.

4 Prototype

We implement a prototype and deploy the frontend and backend in a local environment. As described in section 3.1, the backend of the application is formed by the layer of control and persistence. This layer is implemented by a decentralised

service which hosts the smart contracts. Typically, testnets are used for testing application that use the Ethereum blockchain. In our case, we use the Ganache blockchain environment, which provides to the application the same interfaces as the production Ethereum blockchain.

We use real IoT devices for prototyping the IoT layer, specifically the TTGO T-Beam embedded system board which feature an ESP32 microcontroller and a SX1276 LoRa transceiver. The device is flashed with code to be either a gateway or a sensor node. The sensor nodes are deployed in a star topology around a gateway as shown in Figure 2. The LoRa technology is used for the communication between the sensor nodes and the gateway.

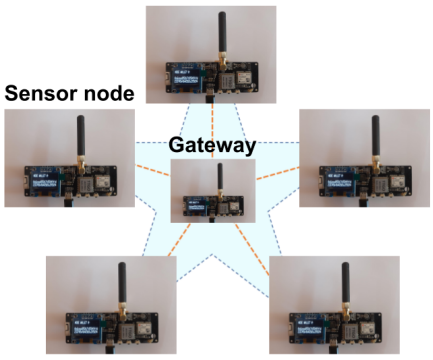

Fig. 2. Deployment of IoT nodes in a star topology.

The smart contracts deployed on the blockchain provide the control and data persistence for the application. Figure 3 contains a code fragment of the smart contract that is implemented in our prototype. First a set of variables can be seen, which contain the state of the application which evolves by the invocation of the methods of the smart contract. Then a set of methods are declared for the management of the devices, corresponding to the different use cases of the application, some of which have been described in section 3.2. Finally, the smart contract contains a set of methods for managing data messages. These methods allow to relate the received data messages to the contributions of the sensor nodes and enable the accounting.

The frontend of the application is given by Web pages and corresponds to the data access layer of the architecture described in section 3.1. Node.js is used to provide the Web page which is shown in Figure 4 (left). It can be seen in this example that three devices have been registered. The devices have both their proper Ethereum address, which is registered by the hardware provider after the device is created (as explained in section 3.2), as well as the owner address in case the device has been purchased, as shown for the first device.

The operations of the smart contract which produce a transaction, e.g. by writing to a variable and changing its state, result in a cost. This cost is paid by

```
contract LoraCoin is ERC20 {

    address public root;

    struct NetworkDevice {
        address owner;
        bytes signature;
        bytes32[] transactions;
    }
    mapping(address=>NetworkDevice) public devices;

    constructor(string, string) ERC20(name, symbol);

    //Device management
    function createDevice(address _address, bytes memory _signature) public returns (bool);
    function registerDevice(address _address, uint8 v, bytes32 r, bytes32 s) public returns (bool);
    function getDevices() public view returns (address [ ] memory)
    function getOwnerDevices() public view returns (address [ ] memory);

    //Data management
    function sendTransaction(bytes memory data) public returns(bool);
    function registerTransaction(address _sensorAddress, bytes32 transactionHash) public returns(bool);
}
```

Fig. 3. Code fragment of the implemented smart contract.

the account corresponding to the Ethereum address which desires to generate the transaction. To perform the transaction it is needed that the request is signed. We use the Metamask wallet, which manages the private keys belonging to the users' Ethereum accounts. Figure 4 (right) shows the Metamask interface for such an Ethereum account. It can be seen that this account has already made operations in the prototype application, since 10 LRC (LoRaCoins) have been gained as a reward.

The result of the prototype experimentation is the demonstration of a proof-of-concept implementation of the application. The design, which assigns Ethereum addresses to the involved actors, i.e. nodes, gateways and owners, along with private and public key pairs associated to each of these accounts for the signing and authentication of messages, allows to attribute the operations performed on the smart contracts with their respective accounts, and thus the accounting of each actor's contribution.

The design decisions taken, however, have also implied a trade-off on how open the application actually is to the different types of participation. Principally, the application is open for participants which desire to contribute as data

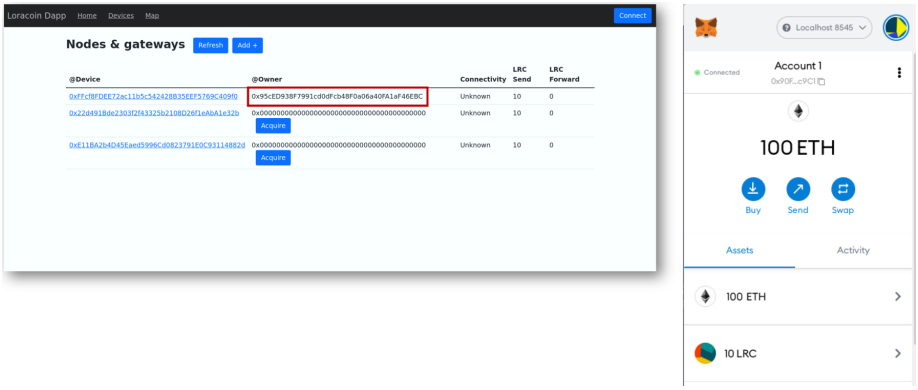

Fig. 4. Left: Web application frontend. Right: Metamask wallet of a rewarded device owner.

and gateway providers. They can participate if they have an Ethereum account, and through the application they can become the owner of a device (Figure 4 left) and contribute data. Our design, however, has achieved less flexibility with regards to the registration of any IoT hardware. Each IoT node that operates in the application has to be created beforehand by a registered hardware provider. On one hand, this registration enables that the node obtains an Etherum address and a public and private key pair, allowing the application to validate for any operation that it is performed by a valid node. On the other hand, this solution does not allow that a user can register any new hardware, but only those IoT devices that are known to the application can be used.

5 Conclusions

This paper addressed the concept of creating a blockchain-based accounting system for incentivizing third party contributions to open IoT applications. The scenario considered are IoT applications open to participation, to which anybody can contribute by providing IoT hardware or IoT data. Environmental monitoring applications were given as a case for which the participatory contribution of a large number of IoT nodes for delivering measurements can be specifically useful.

A prototype of the system was implemented and demonstrated by the deployment of a proof-of-concept. A blockchain-based backend was used for the control and data persistence layer of the application. A key design decision was to create Ethereum addresses together with their associated public and private key pairs for all the involved actors. This allowed that the operations done with the application through the smart contracts could be correctly attributed to the specific actor involved, and as a consequence, the contribution could be accounted for being rewarded.

While the developed system focused on being able to account contribution in terms of hardware and measured data for incentivizing participation, it could be interesting to further research how to give value to these measured data and how valuable each sensor node is. The current system design assures that a received message with measured data is indeed sent from a registered sensor node. If in addition the value of these data was known, then the reward given could be tailored to the data sent, encouraging thus not only data contributions, but also fostering the contributions of those nodes which are more useful.

Acknowledgements This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 957228 — TruBlo and was partially supported by the Spanish Government under contracts PID2019-106774RB-C21, PCI2019-111851-2 (LeadingEdge CHIST-ERA), PCI2019-111850-2 (DiPET CHIST-ERA).

References

1. Haxhibeqiri, J., De Poorter, E., Moerman, I., Hoebeke, J.: A survey of lorawan for iot: From technology to application. Sensors **18**(11) (2018)
2. Anderson, D.: Boinc: A platform for volunteer computing. J Grid Computing **18** (2020) 99–122
3. Wiggins, A., Wilbanks, J.: The rise of citizen science in health and biomedical research. The American Journal of Bioethics **19**(8) (2019) 3–14 PMID: 31339831.
4. Ferrer-Cid, P., Barcelo-Ordinas, J.M., Garcia-Vidal, J., Ripoll, A., Viana, M.: Multisensor data fusion calibration in iot air pollution platforms. IEEE Internet of Things Journal **7**(4) (2020) 3124–3132
5. Cruz Harillo, E., Freitag, F.: Poster abstract: Loracoin: Towards a blockchain-based platform for managing lora devices. In: IEEE INFOCOM 2022 - IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS). (2022) 1–2
6. Zhou, M.T., Shen, F.G., Ren, T.F., Feng, X.Y.: Blockchain-based volunteer edge cloud for iot applications. In: 2021 IEEE 93rd Vehicular Technology Conference (VTC2021-Spring). (2021) 1–6
7. Almuhaya, M.A.M., Jabbar, W.A., Sulaiman, N., Abdulmalek, S.: A survey on lorawan technology: Recent trends, opportunities, simulation tools and future directions. Electronics **11**(1) (2022)
8. Lücking, M., Kannengießer, N., Kilgus, M., Riedel, T., Beigl, M., Sunyaev, A., Stork, W.: The merits of a decentralized pollution-monitoring system based on distributed ledger technology. IEEE Access **8** (2020) 189365–189381
9. Niya, S.R., Jha, S.S., Bocek, T., Stiller, B.: Design and implementation of an automated and decentralized pollution monitoring system with blockchains, smart contracts, and lorawan. In: NOMS 2018 - 2018 IEEE/IFIP Network Operations and Management Symposium. (2018) 1–4
10. Ribeiro, V., Holanda, R., Ramos, A., Rodrigues, J.J.P.C.: Enhancing key management in lorawan with permissioned blockchain. Sensors **20**(11) (2020)