Compartir data usando blockchain

Referentes

Lime Scan

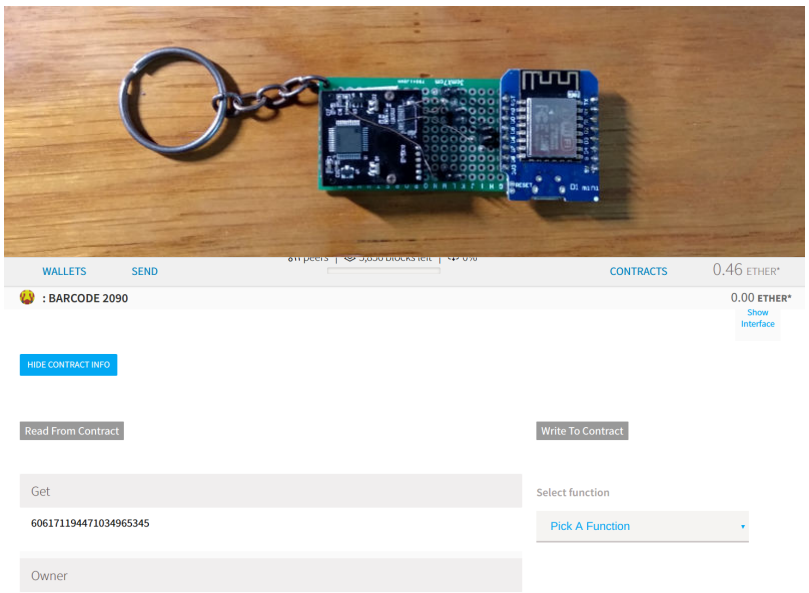
https://www.crowdsupply.com/lime-micro/limenet-micro/updates/limescan-blockchain-demo
https://limescan.net/about

The intention is for LimeSCAN to become a public resource for crowdsourced radio spectrum information, where anyone is free to operate a probe that uploads data, and to make use of the available data — both via the web interface and an open data API. Which raises a number of questions, including, how can we verify data and be certain that it came from a given source?

A popular mechanism for verifying files and data is to create a cryptographic signature or “digest” using a hash function such as SHA256. However, then you need to be sure that the digest you use to validate your data came from a trusted source. One way of doing this in a decentralised manner is to store it on a blockchain, where you can be certain that a given account was responsible for the transaction which recorded that digest, and that it will also be immutable.

In this short video we show an Ethereum node running on a pre-production LimeNET Micro v2 board and participating in a private blockchain network. Spectrum power measurements are made using the SDR hardware, the SHA256 digest is computed, and this is then recorded on the blockchain via a simple smart contract. Thereby demonstrating just one possible way of verifying scan data and its source, and serving as a nice demo of the LimeNET Micro integrated compute.

Yes, You Can Put IoT on the Blockchain using Python and the ESP8266



As a compromise, lets consider an architecture where various IoT devices communicate via MQTT using some level of security with a server that holds the public and private keys for the various devices on the network (probably an on-site server). It would be possible to store the keys on each device, but since the server would need to have them in plaintext at some stage anyway, we may as well avoid transmitting the keys repeatedly. I could also add external key management hardware to an ESP8266 and sign transactions with reasonable security, but that's a fair amount of work and beyond the scope of a quick test.

Anyhow, this architecture lets us handle all the 'blockchain stuff' from a single point per deployment, and leaves the IoT devices more or less as usual. We haven't adhered to complete decentralization here, but I think the tradeoff is worth it. In any case, I previously built an Internet-connected barcode scanner, and given the interest in using blockchain for supply chains, it seems like a fun data source (more importantly, it means my toaster stays intact).

https://hackaday.com/2019/03/01/yes-you-can-put-iot-on-the-blockchain-using-python-and-the-esp8266/