2020-12-21 Peer Programming Call

First we discussed deploying the Utility Emissions Channel Project on Kubernetes and Amazon Web Services, based on work that Robin Klemens , Pavel Krovelets, and Michael Bauer have been doing. Please see video for the technical discussion. Kamlesh Nagware also shared this repository from Accenture for fabric deployment on Kubernetes used in production. There's also the Amazon EKS console documentation.

Then we discussed issues for going live with a deployment of the Utility Emissions Channel in the new year.

For emissions factors, we're going to use the European Energy Agency data in addition to the US EPA EGrid database. Although the European database is only at the country-level, because the countries themselves are smaller, it's probably comparable in granularity to the US database, which breaks down to the state- and ISO-level. We had also hoped to get data for India, but it seems that similar data just for utility emissions is not available.

Then we talked about where to store the emissions factors. Although OrbitDB is a promising decentralized database, it seemed it would take quite a bit of work to set up. Given that they're very far from a 1.0 release, we thought for now we would share an Amazon dynamodb instance among the nodes. However, I've left this JIRA task open for anyone who's interested in or familiar with OrbitDB.

Finally we talked about where to store the documents, such as utility bills, proving the utility data on the ledger is correct. We considered these approaches:

1. Each node stores just a hash of the document on the ledger and then the document wherever it chooses – ie local serverless or cloud s3 of its own set up. The node could be requested to present a document, which could be checked against the hash. However, the node could lose the document.
2. All documents are stored on a centralized S3 repository, shared by all the nodes.
3. All documents are encrypted and stored on IPFS and available to all nodes.
4. Using a private data collection in Fabric.

The problem is that GDPR requires that the documents should be deletable, and no identifiable information should be stored. This seems to rule out IPFS as it is a permanent archive. Storing the document in some way on a centralized S3 repository or in some way across the ledger also presents problems with GDPR compliance: Some of the parties may not be subject to GDPR and therefore not necessarily responsive to GDPR requests to delete data. Furthermore, there may be data regulations in other countries, and it would become very complicated if all nodes had to adhere to all the other nodes' regulations. Therefore, we decided the cleanest way for now is to store only the hash on the ledger and let each node store the documents according to its regulations.

Vatsal then shared an interesting project called Blockstack and XODrive, which is a way to store data locally but share them across a network. I've created a JIRA task to research this solution.

Related to this is the question: Are utility account numbers "identifiable information"? It seems the answer is "NO" since you cannot get additional information from someone with just their account number.