Distributed ledger technology for monitoring operations carried out on the embedded generation units

David. VANGULICK
ORES - Belgium
david.vangulick@ores.be

S. J. ESCALONA CORONEL
ULIEGE - Belgium
sjescalona@student.uliege.be

Damien ERNST
ULIEGE - Belgium
dernst@uliege.be

ABSTRACT

Grid monitoring is the process of collecting data from sensors across a distribution grid and sending it to a central system (SCADA) to identify and diagnose problems, improve reliability, and save energy and money. The increasing complexity of power flows and the need to manage them using active network management (ANM) strategies requires accurate data and strong defences against cyberattacks. A proof-of-concept software called "MonitORES" was developed using Hyperledger Fabric to demonstrate how a distributed ledger technology (DLT) such as blockchain can be used to monitor and control generation units within ANM schemes, with improved resilience against cyberattacks.

1 INTRODUCTION

For distribution grid operators, grid monitoring is a crucial task because it enables them to locate and analyze systemic issues. This can lower costs and save energy, while also increasing system dependability.

Grid operators have a variety of tools at their disposal for system monitoring. The general strategy is to employ sensors dispersed throughout the distribution grid, which gather data and transmit it to a central SCADA system., which stands for Supervisory Control And Data Acquisition.

Nowadays, while MV distribution network involve a large portion of energy transition (embedded renewable energy generators, the impact the electrical mobility, an increase in heat-pumps, and other thermal applications...), monitoring gains is of crucial importance as the power-flows in the grid are no longer omnidirectional (from central power plants to end consumer, through transmission and distribution networks). In order to prevent congestion or voltage problems, a distribution system operator (DSO) employs active network management (ANM), a strategy that is based on short-term policies that control the power injected by generators and/or the power removed by loads. This ANM relies firmly on the data provided to and by the generators.

Data collection and monitoring are expensive owing to the required accuracy of the measurements and the needed defence lines against cyber-attacks. Furthermore, the supervision of contractual requirements of generator operators is essential and can barely be realized by the SCADA system. For instance, this includes the obligation to send correct data and to follow up on control commands given by the DSO.

The emergence of distributed ledger technology (DLT), such as blockchain technology, shows that, when it is correctly set up, DLT has an interesting potential regarding data storage and monitoring while offering a high degree of resilience against cyber-attacks.

Can a DLT monitor the contractual obligations of the generation units on behalf of the DSO within the parameters of an ANM scheme and with adequate resilience to cyber-attacks? This is the main question that this paper, which is an update of the master thesis [1], seeks to address.

The considered contractual requirements are the following:

- Send at a determined time step t the status of the generator and its power output (power measures)
- Follow the set point (i.e., output power curtailment) sent by the DSO to an individual generator. This is the normal working practice of the ANM scheme.
- When the DSO enters an “Alert” mode, it decreases the power output to a specified limit. This alert mode is used to ensure that every generator in the network reduced its output
- When DSO enters an “Emergency” mode, all generators have to stop immediately. This mode is used to maintain operational safety when an power surge occurs.

It has to be clear that the ANM scheme is not part of the DLT but operated within the SCADA system. The DLT is part of the communication and supervision layers to ensure that the scheme operates smoothly. All these requirements will be monitored in the DLT by considering them as different types of transaction recorded in the ledger.

We created a proof-of-concept program called "MonitORES" to answer the question raised earlier and demonstrate how this technology could be used to accomplish the desired objectives.

With that aim, this paper is structured as follows: 1) overview of DLT technology and Hyperledger Fabric in particular, 2) set the functional features to be implemented, 3) implementation, 4) cyber security assessment, and 5) conclusions.

2 OVERVIEW OF DLT TECHNOLOGY AND HYPERLEDGER FABRIC

A distributed database on which the state is decided by the nodes of the systems using consensus mechanisms makes up the DLT technology group. Data is stored in an immutable manner once agreement has been reached.

Although blockchains are the most well-known type of DLT, there are other types like acyclic oriented graph DLTs like Tangle [2] or Hedera Hashgraph [3]. Focus will be placed on blockchain for the remainder of the section because it will be the tool used in the prototype developed below.

A blockchain is a type of distributed database that keeps track of blocks, which are a growing collection of records. Each block includes transaction information, a timestamp, and a cryptographic hash of the one before it. This provides a time-stamped, permanent record. Blockchains are made to be resistant to data modification by design. This implies that the data stored in a blockchain can never be altered or deleted by a single party.

Blockchain technology can be broadly categorized into two groups: permissionless and permissioned.
Firstly, anybody is welcome to join a permissionless blockchain, which is an open system that enables participation in peer-to-peer transactions on any given network.
Examples of permissionless blockchains include Bitcoin [4] and Ethereum [5].
Secondly, a closed system that can only be accessed by a select few users is a permissioned blockchain. In healthcare and government applications, permissioned blockchains are typically used to offer secure online access to transaction records between businesses and clients.
One of the most interesting functionalities within DLT technology is the possibility to create a 'smart contract'. With a smart contract, the conditions of the agreement between the buyer and seller are directly encoded into lines of code, making it a self-executing contract. On the blockchain network, the code and the agreements it contains are copied and saved. Smart contracts make it possible to automate the execution of contracts.
Smart contracts were first proposed by computer scientist Nick Szabo in the 1990s as a way to facilitate, verify, and enforce the negotiation or performance of a contract. They are often associated with blockchain technology, as the decentralized and transparent nature of the blockchain makes it an ideal platform for executing smart contracts.

One of the main advantages of smart contracts is that they can automate the execution of contracts, reducing the need for intermediaries and the potential for errors or fraud. This can also make the process of contracting more efficient and cost-effective.

Smart contracts can be used in a variety of industries and applications, such as supply chain management, real estate, and financial services. For example, a smart contract could be used to automatically release payment to a supplier once a shipment of goods has been received and verified, or to automatically transfer ownership of a piece of property once all the terms of a real estate contract have been met.

However, it is important to note that smart contracts are only as reliable as the code they are written in and the systems they are implemented on. It is crucial to thoroughly test and audit smart contracts before deploying them to ensure that they function as intended and do not contain any vulnerabilities.

There are various DLT platforms and technologies such as Cipher, Monax, Wanchain, etc. Our use case has the following characteristics.

- Confidentiality: the transactions/exchange of information between the generator's operator and the DSO need only be shared by these parties as their importance relates solely to the business. A permissionless blockchain is therefore not an appropriate.
- Transaction rate: 100 transaction/s with 95% commit after 20 seconds

These above-mentioned characteristics guide us to a series of three permissioned blockchain: Quorum, Corda and Hyperledger Fabric.
Although the purpose of this paper is not to compare these DLTs, after further assessment (e.g. language and community support), we made the choice to select the Hyperledger Fabric.

An open-source, modular, and extensible platform for creating blockchain applications of the highest caliber is called Hyperledger Fabric. It is a permissioned blockchain, which means that only authorized users are able to access it. It also supports a range of consensus mechanisms, allowing for flexibility in the creation and deployment of applications. Hyperledger Fabric also has a modular architecture, allowing for the plug-and-play of components such as consensus and membership services. It is designed to support the development of a wide range of applications, from supply chain management to digital identity and beyond.

3 FUNCTIONAL FEATURES

To monitor the contractual requirements described in the introduction section, the following functional features have been implemented within the smart contract and the DLT interface.

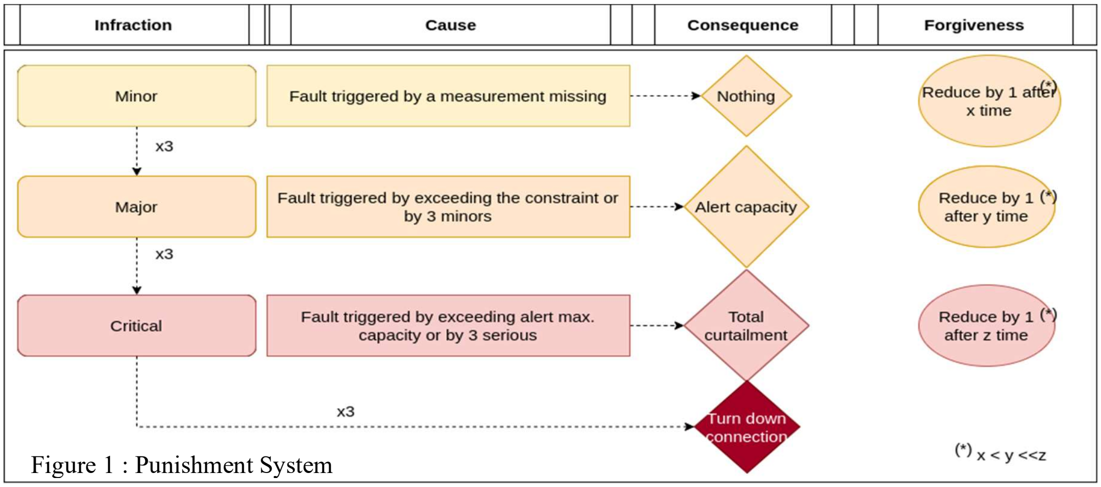
Infraction
Cause
Consequence
Forgiveness
Minor
Fault triggered by a measurement missing
Nothing
Reduce by 1 after (*) x time
x3
Major
Fault triggered by exceeding the constraint or by 3 minors
Alert capacity
Reduce by 1 (*) after y time
x3
Critical
Fault triggered by exceeding alert max. capacity or by 3 serious
Total curtailment
Reduce by 1 (*) after z time
x3
Turn down connection
(*) x < y <<z

Figure 1 : Punishment System

3.1 On-boarding

An automatic service takes care of integrating a generator into the chain and determining if the conditions are right for allowing it to inject generated electricity to the system when it starts up.

It can be broken down into two sections: creating a power generation unit (PGU) and initialization tests. A PGU creation entails creating and assigning a PGU model instance to a fresh customer joining the network. The DSO not only creates and deploys the smart contract that supports the functional features, but also submits all pertinent contractual information to the blockchain.

Once the PGU is recorded in the blockchain, initialization tests can begin. These tests guarantee that the PGU is functioning properly. There are three different tests used:

- Time delay tests: these make sure the PGU can adhere to the frequency of the measures that are imposed,
- Power limit tests: these determine whether the PGU can curtail (reduce) its output of power in response to a constraint.
- Monitoring tests: these ensure that the PGU is communicating the correct data to the SCADA, such as its status and power output.

These tests are run automatically by the smart contract and, once passed, grant to the PGU the access to the network.

3.2 Control/Commands

The DSO can choose to manage the generators by updating their status and submitting a specific constraint to them when the PGU is permitted to inject power into the network. In addition, it can also set a general status for the system. We define three different general statuses: normal state, alert mode and emergency mode. The two last correspond to the contractual requirement described in the introduction section.

The DSO submits an additional constraint to generators running normally when the distribution network is at risk of congestion. Typically, the ANM calculator in the SCADA is sending the restriction (curtailment command). The constraint is correctly sent to the appropriate PGU thanks to the smart contract.

3.3 Monitoring

By continuously sending measurements, the PGU's embedded monitoring system continuously checks that the status and constraints are being respected.

It also enables the PGU operator to indicate the status like ‘normal’, ‘Undergoing maintenance’, or ‘Unplanned disconnection’. The two last statuses are useful for the ANM scheme as it gives information that the unit will not be generating (full) power.

Monitoring and trust are two concepts that are often related, but they can also be distinct from each other.

Monitoring refers to the act of keeping track of something or someone in order to gather information or to ensure compliance with rules or standards. This can be done through various means, such as surveillance, data collection, or regular check-ins.

Trust, on the other hand, refers to the belief in the reliability, truth, ability, or strength of someone or something. Trust is often built over time through positive experiences or through the demonstration of reliability and honesty. Trust is an important component of relationships, as it allows people to feel safe and confident in their interactions with others.

Using monitor features, we have developed a method in the smart contract to ensure confidence (trust) that PGU owners are playing their part fairly. For example, the accuracy of output power measurements is critical, and PGU operators could cheat the system and let other PGU owners manage any reduction in power output in the event of a case of congestion.

Infractions are tracked using a mechanism that is updated in the ledger. The tit-for-tat game theory method [6] that optimizes everyone's rewards is the basis of the system.

This tactic penalizes the opponent right away after a mistake but pardons them after a good deed or after enough time has passed.

We construct three categories of wrongdoing—minor, major, and critical—each with a distinct punishment. Figure 1 depicts the cause, effect, and forgiveness rule for each fault.

Minors errors include a PGU's measurement activity being absent for t minutes. In reality, because the ANM computation needs data that is close to real-time, interrupting the network's observations could result in an inefficient curtailment calculation. After a specific amount of time (x), which is freely chosen by the DSO, these faults are absolved. When experiencing a minor fault, it does not have ant immediate consequences unless more than two minor faults occur in the same forgiveness time window. It acts in a way similar to a yellow card in football.

When the PGU's power output exceeds the ANM computer's limit or when three minor faults occur in a short period of time, major faults are initiated. After y time, which is strictly longer than x, they are pardoned. When a PGU develops multiple major faults, its power output will be severely constrained, typically to 20% of the maximum permissible power.

The last type of fault is a critical fault. This is triggered when the PGU power output exceeds the major fault limitation or by accumulating three major faults within a specific time period. They can be forgiven after z amount of time.

The PGU is completely curtailed when multiple critical faults happen. The PGU is completely cut off from the grid if it develops three critical faults.

To give an illustration about the time period mentioned in this sub-section, t can be equal to 1 minutes, x can be equal to 15 minutes, y can be equal to 4 hours and z equal to 1 month. As an example, if a PGU does not sent measurements three times during a 15 minute period, the major fault counter will be put at 1. If the same problem occurs in the next 4 hours, the major fault counter will be set at 2 and the PGU will have to reduce its power output as penalty.

3.4 Other features

The prototyped system is completed with the following:

- Visualization: The DSO can visualize the condition of the network under investigation using a web interface. The PGU operators can see the state and any penalties that apply to them specifically.
- Interface: It enables one to launch command and control operations from the DSO. It enables the monitoring to be turned on by PGU operators.
- Data mining: The gathered information enables the DSO to assess the network's condition and use it for analysis.

4 IMPLEMENTATION OF « MONITORES »

A high-level view of the architecture is shown in Figure 2. The global prototype architecture is displayed, in which the blockchain is viewed as an integrated, special functional service. In this figure, one can identify four main components:

- Blockchain: This component is made up of the Hyperledger Fabric network itself, a REST API that can be used to help with data queries, and an explorer web-view that shows the blockchain's technical status. The proof-of-concept network was created using the Fablo [7] simulation tool.
- PGU Simulator: This service offers a live PGU simulation. They have a solar or wind source option. The micro-services capabilities of NestJS [8] are used to implement this component. One can choose the source of the simulation data to be an API (OpenWeather API [9]) or historical wind data, such as that from the Belgian TSO (Elia). [10]
- API Gateway: These parts make it easier for PGUs and 'Web-views' to initiate system actions by orchestrating all communications and various protocols that are used.

The code for the smart contract can be found https://github.com/orgs/MonitORES-POC/repositories

We tested all the different features described in the previous section with one PGU mimicking a windfarm and another one to simulate a photovoltaic generator.

5 CYBER SECURITY ASSESSMENT

We consider three types of attacks when evaluating the security of the proposed prototype system: data collection, data transmission, and data storage.

Attacks that target edge devices that are measuring data are referred to as data collection attacks.

Then, a data transmission attack is when the attack occurs in the communication channel between the device and the storage system. Finally, data storage attack is when the attack directly targets the stored data.

As explained in [1], a probabilistic approach is used to tackle this analysis, drawing inspiration from [11] where the authors evaluate their blockchain framework by simulating the likelihood of successful cyberattacks in two scenarios. Both the scenario with a communication system based on blockchain and the current scenario (the normal scenario) have a risk (probability) of the system being compromised. Let's model the likelihood of the system being compromised for each scenario and attack type, where n is the minimum number of nodes required for an attack to be successful and N is the total number of nodes

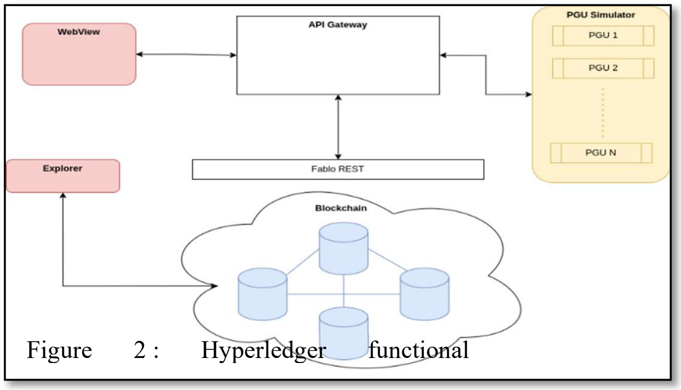

Figure 2 : Hyperledger functional

in the system. Each of the three attack types has its own independent probability. In a typical scenario, we assign each node a probability of being compromised α_i and a probability to each connection with the central system βi where i = 1, 2, ..., N.

A probability η represents the likelihood of the central system being compromised. Finaly, the probability of achieving a successful attack is given by the following equation:

$$P_{normal} = \frac{1}{3}\left(\prod_{i=0}^{n}\alpha_i + \prod_{i=0}^{n}\beta_i + \eta\right) \qquad \text{(EQ 1)}$$

In the blockchain scenario, we assign a probability of hacking αi and a probability of connection βi to each node. the probability ωi for each key used to encrypt a node's communication. In this case, it is necessary to add the voting threshold above which an attacker can seize control of the consensus system. Therefore, in order to calculate the likelihood of communication attacks, we must define K, which stands for the number of channels the attacker must compromise.

$K = ceil(\tau\frac{N(N-1)}{2})$.

Finally, in order to control the consensus and the state of the blockchain, the attacker must hack into M devices, where M = ceil(τ N). Then the probability of achieving a successful attack is given by:

$$\begin{aligned} P_{blockchain} &= \frac{1}{3}\left(\prod_{i=0}^{n}\bar{\alpha_i}\prod_{i=0}^{n}\bar{\omega_i} + \prod_{i=0}^{K}\bar{\beta_i}\prod_{i=0}^{n}\bar{\omega_i} + \prod_{i=0}^{M}\bar{\alpha_i}\prod_{i=0}^{n}\bar{\omega_i}\right) \\ &= \left(\prod_{i=0}^{t}\bar{\alpha_i} + \prod_{i=0}^{K}\bar{\beta_i} + \prod_{i=0}^{M}\bar{\alpha_i}\right)\frac{1}{3}\prod_{i=0}^{n}\bar{\omega_i} \qquad \text{(EQ 2)} \end{aligned}$$

For these attacks, various Monte Carlo simulations are run. The findings demonstrate that, in contrast to the conventional scenario, the likelihood of a successful hack in the blockchain scenario declines more quickly with increased connectivity. This can be explained by the distributed factor, which suggests that in order to change data storage or communication in the system, access to all device keys is necessary. Attacks can be concentrated on fewer communications channels and a central database in the central system model.

6 CONCLUSION

Results show that in terms of security and compliance, a blockchain-based system can reduce the probability of cyber attacks and the disregard for rules dictated by the DSO.

Therefore, a blockchain-based system can be implemented to increase reliability, security and compliance in decentralised production units in the distribution grid.

While our prototype has clearly illustrated the feasibility of a blockchain monitoring system in DSO grids, it raises the question of how well it can be physically implemented and scaled to real-world scenarios.

REFERENCES

[1]Escalona Coronel, Saul Jose, Master thesis : Decentralization of control operations carried out on the decentralized production units connected to the ORES network, 2021

[2] Serguei Yu. Popov. “The Tangle”. In: 2015.

[3] Mance Harmon Dr. Leemon Baird and Paul Madsen. Hedera: A Public Hashgraph Network Governing Council. 2020. url: https://hedera.com/hh_whitepaper_v2.1-20200815.pdf

[4] Satoshi Nakamoto. Bitcoin: A Peer-to-Peer Electronic Cash System. Bitcoin, 2008

[5] Vitalik Buterin et. Al. Ethereum Whitepaper 2020. Feb. 2020. url: https : / /ethereum.org/en/whitepaper/.

[6] Martin Nowak and Karl Sigmund. “A strategy of win-stay, lose-shift that outperforms tit-for-tat in the Prisoner’s Dilemma game”. In: Nature 364.6432 (1993), pp. 56–58.

[7] Hyperledger Labs. Fablo. url:https://github.com/hyperledger-labs/fablo.

[8] NestJS. NestJS. url: https://docs.nestjs.com/.

[9] OpenWeather. Weather API. url: https://openweathermap.org/api.

[10] Elia. Wind power generation. data retrieved from Elia grid data, https://www. elia.be/en/grid-data/power-generation/wind-power-generation. 2022.

[11] Gaoqi Liang et al. “Distributed blockchain-based Data Protection Framework for modern power systems against Cyber attacks”. In: IEEE Transactions on Smart Grid 10.3 (2019), pp. 3162–3173. doi: 10.1109/tsg.2018.2819663.