SymbIoT: Towards An Extensible Blockchain Testbed for IIoT

John Hayes
john.hayes@bcu.ac.uk
Birmingham City University
Birmingham, UK

Adel Aneiba
adel.aneiba@bcu.ac.uk
Birmingham City University
Birmingham, UK

Mohamed Gaber
mohamed.gaber@bcu.ac.uk
Birmingham City University
Birmingham, UK

ABSTRACT

This paper presents SymbIoT, an extensible hybrid simulation-emulation testbed to investigate the integration of blockchain and distributed ledger technology (DLT) within the Industrial Internet of Things (IIoT) continuum. By adopting a joint software and hardware-based approach, we amalgamate the flexibility of software solutions and the real-world applicability offered by integrating comparable IoT hardware. The versatility of SymbIoT lies in its extensibility, offering flexibility in parameters including consensus algorithms, block size, node count and topology, throughput limitation, and use-case application deployment. SymbIoT facilitates comprehensive empirical studies of blockchain implementations within IIoT, focusing on performance, scalability, and security considerations. The testbed provides a platform for innovative and pragmatic experimentation in blockchain and IIoT integration, holding promise for shaping future applications and solutions in this cross-disciplinary field. We also present results from preliminary experimentation, indicating the applicability of the testbed for IIoT and broader IoT-to-cloud scenarios.

CCS CONCEPTS

• **Computer systems organization** → **Peer-to-peer architectures**; • **Information systems** → **Data structures**; • **Software and its engineering** → **Development frameworks and environments**.

KEYWORDS

Blockchain, IoT, Testbed, IIoT

ACM Reference Format:
John Hayes, Adel Aneiba, and Mohamed Gaber. 2023. SymbIoT: Towards An Extensible Blockchain Testbed for IIoT. In *Enhanced network techniques and technologies for the Industrial IoT to Cloud continuum (IIoT-NETs '23), September 10, 2023, New York, NY, USA.* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3609389.3610565

1 INTRODUCTION

The Industrial Internet of Things (IIoT) has brought about automation and efficiency by connecting machines, devices, sensors, and people [12]. However, the associated increase in interactions across these networks introduces significant security vulnerabilities and performance concerns [17]. Each interaction becomes a potential avenue for malicious attacks and resource exhaustion, emphasising the requirement for robust and well-considered integration frameworks.

Furthermore, the ongoing paradigm shift towards Industrial 5.0 promises further advances in connectivity and integration scale [2]. The convergence of IIoT with blockchain technology, renowned for its characteristic decentralisation, transparency, and immutability, presents diverse opportunities to revolutionise these modern industrial systems [21]. This amalgamation promises to bring about enhanced security, traceability, and decentralisation to IIoT, addressing some of its most significant challenges.

However, the seamless integration of blockchain with IIoT is not straightforward; the inherent complexities of blockchain, such as consensus algorithms, transaction speeds, and energy consumption, are significant obstacles in realising its full potential within the IIoT domain [16]. Therefore, in this evolving landscape, understanding the practical implications of combining these two powerful technologies is critical for developing optimised and well considered solutions [6]. In particular, the utilisation of Blockchain technology for enabling trustworthy interactions and information exchange between users and systems is becoming a commonplace occurrence, with various security-centric applications across varied deployment domains adopting this approach [3]. Within the IIoT domain, the frequent and operation-critical inter/intra-factory communication provides an integration environment paragon for ensuring the secure and trustworthy operation of industrial systems while meeting strict reliability constraints.

Despite the theoretical benefits, the practical application of integrating blockchain with IIoT infrastructure remains relatively under-explored [1]. This limitation is due in part to the absence of adaptable, scalable, and customisable environments to conduct in-depth empirical analyses and experimentation [6]. Beyond public Blockchain environments, such as Ethereum [22] and Bitcoin [14], which are unsuitable for lightweight scenarios, per-device performance metrics and comparisons are not readily available nor are operational parameters fully customisable, limiting scope for prototype optimisation.

Addressing this gap, we introduce SymbIoT, an extensible blockchain integration testbed purpose-designed for the IIoT continuum, architecturally situated within the network edge. SymbIoT bridges the gap between theory and practice, providing a platform where characteristics of the blockchain-IIoT integration can be thoroughly explored. It stands out due to its extensive customisability in consensus algorithms, block size, node count, and use-case specifics. This flexibility facilitates the deployment of testing environments tailored to specific use-case requirements.

This paper outlines the design and functionality of SymbIoT, emphasising its potential to catalyse advanced research and development in the blockchain-IIoT domain. Through preliminary

IIoT-NETs '23, September 10, 2023, New York, NY, USA

ACM ISBN 979-8-4007-0302-7/23/09
https://doi.org/10.1145/3609389.3610565

empirical study conducted, we demonstrate its capability to shed light on the practical implications, including performance, scalability, and security considerations of integrating blockchain within IIoT. This work does not raise any ethical issues.

Contributions: This paper makes several key contributions to the study of blockchain integration in the Industrial Internet of Things (IIoT) landscape.

- Firstly, we introduce SymbIoT, a highly extensible testbed designed to facilitate in-depth empirical investigation into the use of blockchain and Distributed Ledger Technology (DLT) within the IIoT.
- Secondly, we provide a comprehensive exploration of SymbIoT's theoretical underpinning, architecture and software functionalities, delving into how its design allows for robust testing and analysis of various aspects of the blockchain-IIoT integration.
- Thirdly, we present a preliminary empirical study carried out using SymbIoT to highlight its potential in shedding light on the practical considerations and challenges of integrating blockchain within the IIoT.

2 RELATED WORK

The continued adoption of IoT within smart factories and the broader IIoT paradigm has prompted the development of substantial simulation and, to a lesser extent, emulation solutions. The principal ambition of these being to prototype and evaluate the viability of diverse use-cases and optimise deployment parameters. While not necessarily positioned directly within the industrial context, the adoption of standardised edge and device-to-cloud architectures within this domain has signalled their suitability.

Fogify [20] is an emulation framework for IoT service modelling and deployment within fog computing architectures. The framework facilitates the modelling, experimentation, and evaluation of IoT services and accounts for use-case parameters and fault simulation. Similarly, MockFog [10] provides an edge-to-cloud emulation framework, enabling the deployment of both edge and cloud resources.

Kaala [8] is a scalable end-to-end IoT simulator, offering integration with real-world cloud providers. The simulator is capable of simulating large numbers of devices and corresponding events to evaluate IoT-to-cloud systems. However, they do not consider advanced parameters in their evaluation, only CPU and latency, nor do they implement any blockchain evaluation methods.

Similarly, the IoTNetSim [18] architecture presents a simulation-based IoT framework. Unlike Kaala, each device requires a new virtual machine, leading to potentially significant resource exhaustion issues, particularly within lightweight deployment scenarios.

From the existing literature, only one simulation framework, FobSim [4], actively supports blockchain technology. However, their blockchain implementation is extremely barebones and does not reflect the technologies embedded within cutting-edge modern blockchains, such as digital signature schemes or block merkle trees. This limits the applicability of the solution for evaluating such integrations in a meaningful way. Furthermore, the solution is an entirely simulation-based platform, with emulation element considered.

In summary, the current literature provides invaluable insight into the theoretical benefits and potential applications of blockchain integration within the IIoT ecosystem. However, a notable gap persists in the availability of a comprehensive solution to fully evaluate this integration in practical terms. The majority of existing research predominantly concentrates on simulation-based approaches, which, while offering certain advantages, fail to capture the full complexity and idiosyncrasies of real-world system interactions. In particular, the lack of an extensible, scalable and customisable platform that can support both emulation and simulation hinders comprehensive and practical research in this crucial field.

3 SYSTEM DESIGN

Based on the shortcomings identified above and the notable lack of a suitable hybrid emulation platform for DLT-IIoT investigation, we propose a novel approach termed SymbIoT. This approach is guided by the need to bridge the gap between theoretical potential and practical realities in DLT-IIoT integration. SymbIoT is strategically positioned within the edge/fog layers in an IoT-to-cloud environment, a placement synonymous with enhanced intermediary data analysis and computation. In the following sections, we present a detailed formal overview of the primary components that constitute SymbIoT.

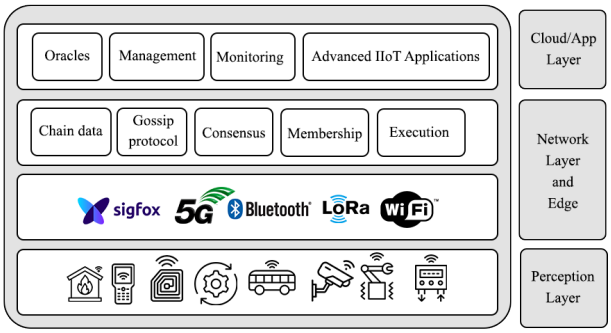

Figure 1: SymbIoT high-level deployment architecture

3.1 System Architecture

We define a high-level architectural overview of the system design to provide operational context. Fig.1 demonstrates this architecture, with distinct layers for end-devices, network, edge, and cloud resources. We consider a variety of device and communication protocol types to reflect the varied nature of IIoT deployments.

3.2 Configuration

To ensure the extensible nature of the solution, we facilitate the configuration of critical blockchain and network parameters.

3.2.1 Topology. Dynamic topology design is critical for representing the diverse scenarios associated with IIoT deployments. Nodes are provided with an initial peer-list of 2 or 3 neighbours, corresponding to the local network address of the collaborating peers.

The topology itself is highly dynamic but can be modelled as a connected (but not complete) graph of nodes N, with a corresponding adjacency matrix A. In this context, we assume that each edge (i, j) in the graph corresponds to a different TCP connection with its own throughput value NT_{ij}.

3.2.2 Throughput. To simulate the data rates associated with varied communication protocols, we enable arbitrary enforcement of throughput caps on inter-node communications, as outlined in Table 1. However, hardware limitations dictate a 1Gbps maximum throughput.

Protocol	Enforced Data Rate
Wi-Fi (802.11ac)	850 Mbps
5G	1 Gbps[1]
4G LTE	15 Mbps
LoRa	25 Kbps
Bluetooth 5.0	1 Mbps
Zigbee	100 Kbps

Table 1: Implemented data rates based on suitable values for popular communication protocols.

The adjacency matrix A can be transformed into a capacity matrix C where each entry C_{ij} is the throughput NT_{ij} (for i ≠ j). If no direct connection exists between i and j, C_{ij} can be set to zero. Assuming a multicast gossip protocol, we can define the maximum flow F between a source node s as follows:

$$\begin{aligned} \underset{F}{\text{maximise}} \quad & \sum_{i \in N} F_{si} \\ \text{subject to} \quad & F_{ij} \leq C_{ij} \quad \forall (i,j) \in N \times N, \\ & \sum_{i \in N} F_{ij} = \sum_{i \in N} F_{ji} \quad \forall j \in N, \\ & F_{ij} \geq 0 \quad \forall (i,j) \in N \times N, \end{aligned} \tag{1}$$

Due to the probabilistic nature of gossip-based protocols, the maximum flow as formulated here represents an upper bound. However, it provides a useful benchmark for comparing different network configurations or communication protocols.

3.2.3 Blockchain Parameters. A blockchain $\mathcal{B}$ can be denoted as a sequence of blocks $b_0, b_1, b_2, ..., b_n$, where n is the blockchain length. Let $h : 0,1^* \rightarrow 0,1^d$ be a cryptographic hash function that maps arbitrary length binary strings to fixed d, and let Tx be a set of transactions included in a block. Then each block b_i can be represented as:

$$b_i = \{h(b_{i-1}), Tx_i, t_i, nonce_i\} \tag{2}$$

where $h(b_{i-1})$ is the hash of the previous block, Tx_i is the set of transactions, t_i is the block creation timestamp, and nonce_i is a nonce. Denoting block size as $S(b_i)$, the size of a single transaction as $S(tx)$ (assuming an average size for simplicity), and the maximum block size as $S_{\max}$. The number of transactions $N_{Tx}(b_i)$ that can be included in a block b_i is then limited by the size of the block:

$$N_{Tx}(b_i) = \left\lfloor \frac{S_{max} - S_{overhead}}{S_{(tx)}} \right\rfloor \tag{3}$$

Here, S_{overhead} represents the size of the block overhead, and $\lfloor x \rfloor$ denotes the floor function, which rounds x down to the nearest integer. This equation indicates that the number of transactions that can be included in a block is determined by the maximum block size and the size of each transaction, both of which are variable.

The block generation time can be denoted as T_{block} and the number of blocks in a given time period T as $N_{\text{blocks}}(T)$. Assuming a constant block time, the relationship between these variables can be expressed as:

$$N_{blocks}(T) = \frac{T}{T_{block}} \tag{4}$$

where T represents the total time period, T_{block} is the block time, and $N_{\text{blocks}}(T)$ is the number of blocks created. The total number of transactions $N_{\text{total}}(T)$ that can be finalised in the time period T can then be defined as:

$$N_{total}(T) = \frac{T}{T_{block}} \times \left\lfloor \frac{S_{max} - S_{overhead}}{S(tx)} \right\rfloor \tag{5}$$

3.3 Accounts

Accounts are a critical aspect of both public and private blockchain networks, enabling transaction exchange between network participants. These accounts are typically associated with a keypair (PK, SK) which is used for transaction signing and verification. Consider a set of accounts $\mathcal{A}$ and a mapping function $\mathcal{K}$ that maps an account $a \in \mathcal{A}$ to its public and private keys, this relationship can be be represented as follows:

Let $PK : \mathcal{A} \rightarrow \mathcal{K}$ and $SK : \mathcal{A} \rightarrow \mathcal{K}$ be two functions that map an account a to its public key $PK(a)$ and private key $SK(a)$, respectively.

$$\forall_a \in \mathcal{A}, PK(a), SK(a) = \mathcal{K}(a) \tag{6}$$

Here, $\mathcal{K}(a)$ generates the pair of keys for account a. The address $addr(a)$ of an account a, which is used when a secondary address wants to transact, is usually derived from the public key $PK(a)$ through a hashing function $\mathcal{H}$ and a checksum function C for error detection.

$$addr(a) = C(\mathcal{H}(PK(a))) \tag{7}$$

3.4 Transactions

Transactions are a fundamental aspect of blockchain systems, both for the exchange of tokens and data between the accounts detailed above. In the case of private IIoT networks, transactions are the medium of data and information exchange between nodes. The owner of an account a can sign a transaction tx using their private key $SK(a)$ with a digital signature algorithm $\mathcal{S}$ (typically ECDSA [11]):

$$sig = \mathcal{S}_{SK(a)}(tx) \tag{8}$$

Any other node in the network can verify the signature using the public key $PK(a)$ of the account. Here, $\mathcal{V}$ represents the verification function, which checks whether the signature sig on the transaction tx is valid. If $verify = true$, the signature is valid; otherwise, the signature is invalid.

$$verify = \mathcal{V}_{PK(a)}(tx, sig) \tag{9}$$

Two principal approaches exist for managing the store and transfer of token balance, each of which is outlined briefly below.

3.4.1 Account-Based. In an account-based model, each account has a balance, and transactions directly transfer value between these balances. The balance of an account a at the time t can be denoted as $B(a, t)$. A transaction T from account a to account b with value v at time t can be expressed as:

$$B(a, t+1) = B(a, t) - v,$$
$$B(b, t+1) = B(b, t) + v$$

3.4.2 UTXO. In a UTXO-based model, transactions consume unspent outputs from previous transactions as their inputs, and produce new unspent outputs. We denote a set of UTXOs as U, and a transaction T that consumes a subset of UTXOs $u \subset U$ as inputs and creates a set of new UTXOs v. This can be expressed as:

$$U_{\text{after}} = (U_{\text{before}} - u) \cup v$$

3.5 Consensus

Consensus mechanisms form the foundational backbone of both public and private blockchain networks. They serve as the underlying algorithm that enforces security, finality, and guarantees system integrity. This is accomplished through an intricate interplay of cryptographically secured, algorithmic protocols that coordinate agreement among participating nodes, ensuring robustness against threats and manipulation.

Therefore, we implement the majority of popular and robust consensus protocols within the system. As consensus protocols underpin the security, scalability, and sustainability of blockchain networks, We briefly formalise the differing functionality of each protocol below, providing a base theoretical understanding.

3.5.1 PoW. Proof of Work is primarily focused on computational effort. In the simplest sense, it can be described in terms of finding a nonce that solves a specific mathematical problem. Let H be a hash function, B a block, n a nonce, and d a difficulty target. PoW seeks to find n such that:

$$H(B \parallel n) < d \tag{10}$$

3.5.2 PoS. Proof of Stake (PoS) [19] consensus is related to the amount of the native token an account holds. Where S_u denotes the stake of user u, and S_{total} as the total stake in the network. The chance P_u that user u will mine the next block can be represented as:

$$P_u = \frac{S_u}{S_{\text{total}}} \tag{11}$$

3.5.3 DPoS. Delegated Proof of Stake (DPoS) is an adapted PoS algorithm whereby stakeholders elect a certain number of delegates that create blocks and confirm transactions, rather than involving every network node. Let V_u represent the votes for user u, and D be the set of all delegates. The user u becomes a delegate if:

$$u \in D \iff V_u > \min V_i | i \in D \tag{12}$$

Essentially, user u becomes a delegate if they are in the top n users by number of votes, where n is the number of delegates.

3.5.4 PBFT. Practical Byzantine Fault Tolerance [7] (PBFT) is a consensus algorithm that tolerates up to $\frac{n-1}{3}$ Byzantine nodes out of n total nodes. PBFT involves multiple rounds of voting and the consensus is reached when more than $\frac{2n}{3}$ nodes agree on a value for a given round.

3.5.5 PoA. In a Proof of Authority (PoA) network, the validators are pre-selected, and they take turns to create new blocks. If we denote V the set of all validators and t the current time, the validator v creates the block at time t if:

$$v = t \mod |V| \tag{13}$$

3.5.6 PoET. Proof of Elapsed Time (PoET) [5] seeks to randomly select a leader to create the next block, with each participant waiting a random amount of time. The participant with the shortest wait time gets to create the next block. If we consider T_u as the wait time for user u, and $T_{\min}$ as the minimum wait time among all users, then user u is selected if:

$$T_u = T_{\min} \tag{14}$$

3.5.7 SCP (FBA). Stellar Consensus Protocol (SCP) [13] makes use of a Federated Byzantine Agreement system (FBAS). Nodes in SCP decide to commit or abort a value based on their quorum slices. The protocol reaches agreement when all honest nodes accept the same value. Let V be the set of all nodes, $Q(v)$ the quorum slice for a node v, and q a quorum. The network agrees on a value if for all honest nodes $v \in V$:

$$Q(v) \subseteq q \tag{15}$$

3.6 Smart Contracts

Smart Contracts facilitate the network-wide execution and evaluation of logical applications when called by users [9]. Contract execution can be viewed as a deterministic function that takes a contract's current state and a set of inputs, and produces a new state and a set of outputs dependent on purpose. Let C denote a smart contract, S its current state, and I the inputs to the contract. The execution of the smart contract can be represented as a function f that takes C, S, and I, and produces a new state S' and a set of outputs O:

$$(S', O) = f(C, S, I) \tag{16}$$

3.6.1 Oracles. Oracle services provide smart contracts with off-chain data. Simply, they can be viewed as functions that take an off-chain data request as input and provide the requested data as output. Let's denote an oracle O, a request R, and the corresponding off-chain data D. The oracle function could be represented as:

$$D = O(R) \tag{17}$$

3.7 Use-case

In the context of IIoT-DLT, a use-case can be considered system that takes some inputs (sensor data, user commands, etc.), performs some function(s) on these inputs (data processing, decision making,

etc.), and then generates some outputs (actuator commands, data for storage or transmission, etc.).

Let $\mathcal{I} = i_1, i_2, ..., i_n$ denote the set of all possible inputs, where each i_j could be a set of sensor data or other inputs

$$i_j = \{s_1, s_2, \cdots, s_k\} \tag{18}$$

where s_l represents a sensor reading or other data point. Let $f : \mathcal{I} \rightarrow O$ denote the function performed by the IIoT use-case. This could involve multiple sub-functions:

$$f(i) = f_m(f_d(i)), tx_{out1}(f_d(i)), tx_{out2}(f_d(i)), \cdots, tx_{outp}(f_d(i)) \tag{19}$$

Here, tx_{out} denotes a function that creates an outgoing transaction based on the processed input data. The outputs include both the results of the device's internal functions and the transactions that it creates:

$$o = \{f(i), tx_{out1}, tx_{out2}, \cdots, tx_{outp}\} \tag{20}$$

4 IMPLEMENTATION

Based on the design and blockchain components described above, we implement a highly modular and extensible hybrid platform. This implementation can be categorised into three distinct development areas: software, hardware, and orchestration.

4.1 Software

Our blockchain framework, implemented in Python 3.8, utilises JSON [15] and HTTP for efficient inter-node communication. To ensure robust monitoring, configuration, and operation of the chain state, the framework exposes an assortment of API endpoints, divided into two broad categories: Blockchain Network Metrics Node-specific Performance Metrics. Given the criticality of performance evaluation in our testbed, we have incorporated a comprehensive monitoring suite that leverages Grafana, in combination with telegraf. This suite allows for real-time tracking of key node performance metrics, facilitating dynamic and detailed analyses of the system behaviour under different operational conditions. ECDSA has been utilised as the signature scheme, and a rudimentary Python eval execution environment used to simulate contract capability.

This software architecture, demonstrated in Figure. 2, is designed to enable extensive, flexible, and meaningful interactions with the blockchain system, providing a dynamic platform for comprehensive performance monitoring and system management.

4.2 Hardware

To accurately emulate real-world edge devices, we have deployed our software on a 31-node cluster, each node equipped with a Raspberry Pi 4B. The choice of Raspberry Pi 4B is motivated by its high performance-to-cost ratio and its ubiquity in IoT applications. Fig.3 depicts the layout of this cluster, which is designed to mirror the device capabilities typically seen in the intermediary edge/fog layers of IoT-to-cloud architectures.

Specific roles are assigned within the cluster: one node functions as the cluster head, coordinating the activities of the other nodes; one node is dedicated as the monitoring server, collecting and analysing performance metrics; node operates as the oracle service, interfacing with simulated and real external data sources and services; and one node handles cloud interactions.

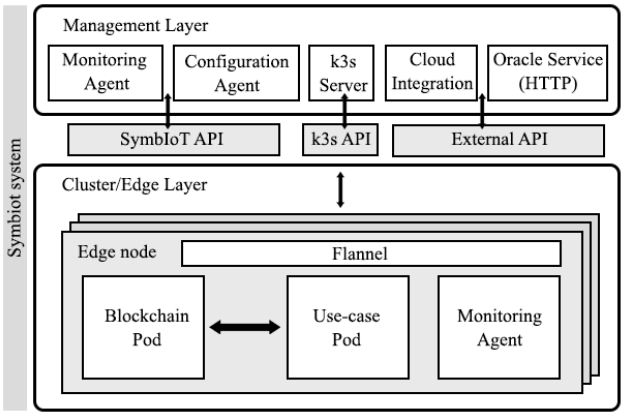

Figure 2: Software interaction overview

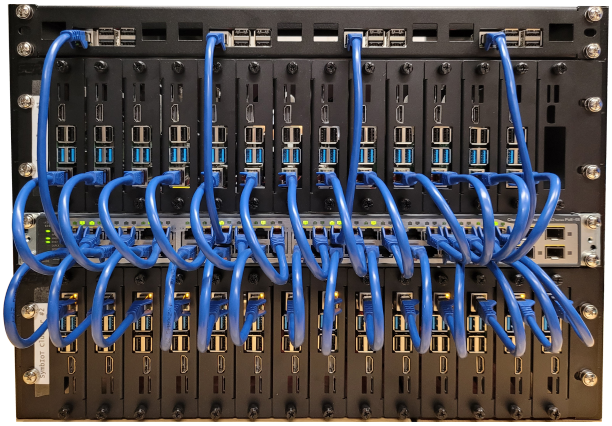

Figure 3: SymbIoT Cluster with 31 Raspberry Pi 4B devices connected via a Cisco Switch.

This physical setup, in conjunction with our software architecture, provides a comprehensive and realistic platform for evaluating blockchain and IIoT integration. The emulation environment simulates real-world conditions and constraints, offering invaluable insights that extend beyond theoretical analysis.

4.3 Orchestration

Given the necessity for efficient software deployment in our system, we leverage an orchestration-based approach. However, considering the resource constraints of our emulation environment, we cannot employ resource-intensive orchestration solutions such as standard Docker or Kubernetes distributions. Instead, we opt for k3s, a lightweight Kubernetes distribution, purposefully designed to operate optimally within limited-resource environments. This approach ensures the efficient deployment of software and use-case containers across the hardware cluster without overburdening system resources, thereby enabling comprehensive and efficient experimentation.

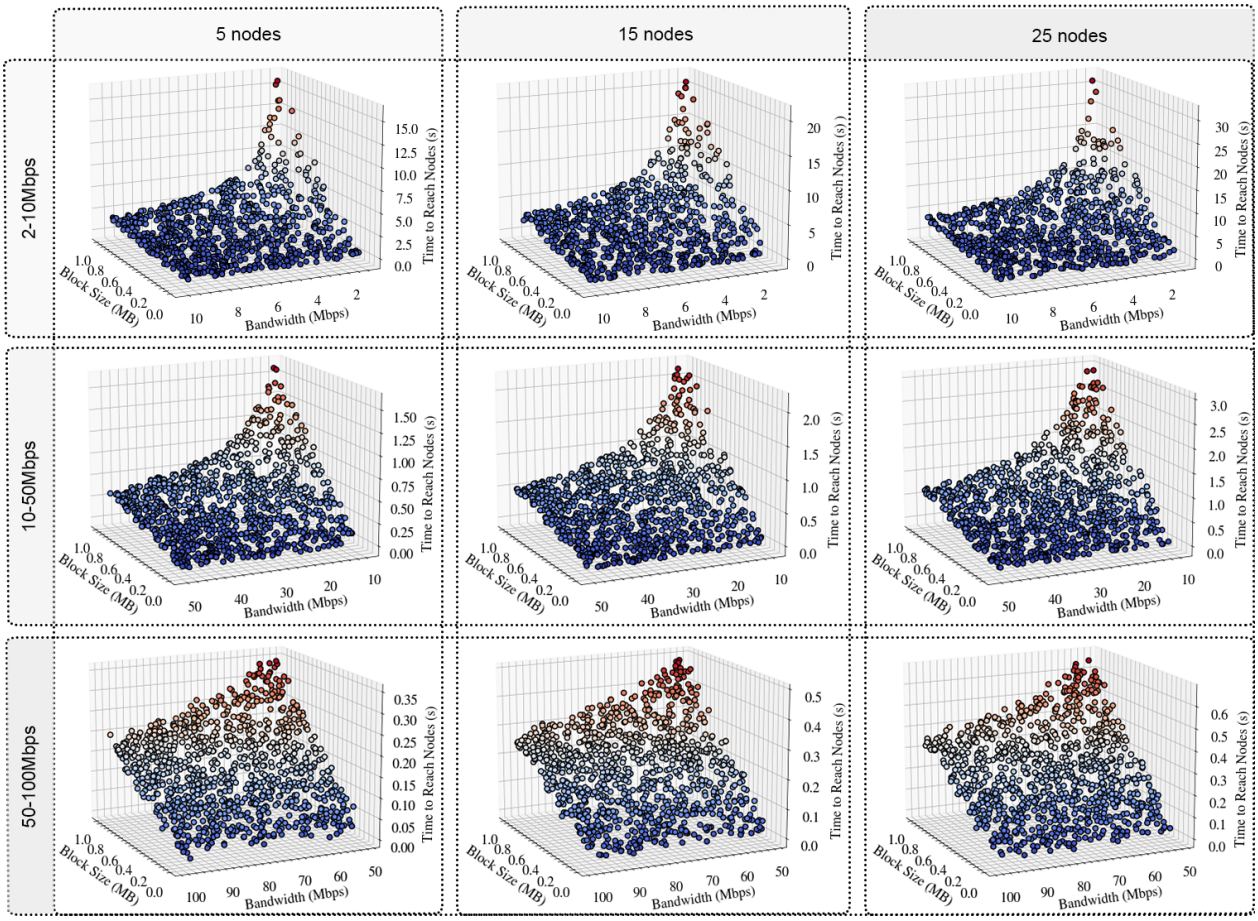

Figure 4: Scatter plots visualising network convergence rate in seconds for a block, given block sizes (0.1-1MB), available bandwidth (1-100Mbps) and node count (5/15/25), with each node assigned 2-3 neighbours.

5 DISCUSSION AND EVALUATION

5.1 IIoT Case Study

To highlight the applicability of this solution to the IIoT domain, we briefly evaluated a real-time asset tracking scenario in a manufacturing facility where hundreds of components are regularly transported for various production processes. Secure, real-time tracking of these assets can be challenging but asset loss can lead to significant delays and financial loss.

In this scenario, a simple use-case pod was deployed to submit regular transactions to the blockchain at varying rates of 1 to 10-per-second. The blockchain framework was configured with various different block sizes and node data rates, indicating the consensus convergence time at several thresholds. This study directly addresses the theme of using decentralisation (DLTs) to improve trust, within the IIoT context. It provides a practical demonstration of how varied chain parameters can impact secure consensus and network convergence time in a real-world implementation.

The results in Fig. 4 demonstrate the increasing convergence time associated with lower data rates and greater block sizes. It is therefore important to consider the optimal parameters for a given deployment scenario to ensure security and finality for transactions, as selecting large block sizes with lower throughput protocols can significantly impact convergence performance and, by extension, chain security. At higher data rates, in particular those above 50Mbps, the convergence time is of little concern regardless of block size.

6 CONCLUSION

In this paper we presented SymbIoT, a hybrid simulation/emulation testbed for deploying and evaluating blockchain and DLT technologies within IIoT environments. By implementing and deploying the solution onto a bespoke IoT tesbed, we were able to demonstrate the efficacy with some preliminary experimentation in the IIoT domain. The results indicate the viability of the solution and highlight the potential value such a solution offers for prototyping. Additionally, the inclusion of in-depth real-time evaluation metrics is invaluable for informed decision making.

In future work, we intend to further extend capability by introducing additional consensus approaches, zero-knowledge proofs, and power usage analysis. In addition, we intend to release an open-source iteration of this solution and full build guide.

REFERENCES

[1] Mwrwan Abubakar, Zakwan Jarocheh, Ahmed Al-Dubai, and Xiaodong Liu. 2022. A Survey on the Integration of Blockchain and IoT: Challenges and Opportunities. In *Advanced Sciences and Technologies for Security Applications*. Springer International Publishing, 197–221. https://doi.org/10.1007/978-3-031-04424-3_11

[2] Amr Adel. 2022. Future of industry 5.0 in society: human-centric solutions, challenges and prospective research areas. *Journal of Cloud Computing* 11, 1 (Sept. 2022). https://doi.org/10.1186/s13677-022-00314-5

[3] Nikolaos Alexopoulos, Sheikh Mahbub Habib, and Max Mühlhäuser. 2018. Towards Secure Distributed Trust Management on a Global Scale: An Analytical Approach for Applying Distributed Ledgers for Authorization in the IoT. In *Proceedings of the 2018 Workshop on IoT Security and Privacy* (Budapest, Hungary) *(IoT SP '18)*. Association for Computing Machinery, New York, NY, USA, 49–54. https://doi.org/10.1145/3229565.3229569

[4] Hamza Baniata and Attila Kertesz. 2020. FoBSim: An extensible open-source simulation tool for integrated Fog-Blockchain systems. (11 2020). https://doi.org/10.36227/techrxiv.13148390.v1

[5] Mic Bowman, Debajyoti Das, Avradip Mandal, and Hart Montgomery. 2021. On Elapsed Time Consensus Protocols. Cryptology ePrint Archive, Paper 2021/086. https://eprint.iacr.org/2021/086 https://eprint.iacr.org/2021/086.

[6] Francesco Buccafurri, Gianluca Lax, Serena Nicolazzo, and Antonino Nocera. 2017. Overcoming Limits of Blockchain for IoT Applications. In *Proceedings of the 12th International Conference on Availability, Reliability and Security* (Reggio Calabria, Italy) *(ARES '17)*. Association for Computing Machinery, New York, NY, USA, Article 26, 6 pages. https://doi.org/10.1145/3098954.3098983

[7] Miguel Castro. 2001. Practical Byzantine Fault Tolerance. (04 2001).

[8] Udhaya Kumar Dayalan, Rostand A. K. Fezeu, Timothy J. Salo, and Zhi-Li Zhang. 2022. Kaala: Scalable, End-to-End, IoT System Simulator. In *Proceedings of the ACM SIGCOMM Workshop on Networked Sensing Systems for a Sustainable Society* (Amsterdam, Netherlands) *(NET4us '22)*. Association for Computing Machinery, New York, NY, USA, 33–38. https://doi.org/10.1145/3538393.3544937

[9] Primavera De Filippi, Chris Wray, and Giovanni Sileno. 2021. Smart contracts. *Internet Policy Review* 10, 2 (April 2021). https://doi.org/10.14763/2021.2.1549

[10] Jonathan Hasenburg, Martin Grambow, and David Bermbach. 2023. MockFog 2.0: Automated Execution of Fog Application Experiments in the Cloud. *IEEE Transactions on Cloud Computing* 11, 1 (jan 2023), 58–70. https://doi.org/10.1109/tcc.2021.3074988

[11] Don Johnson, Alfred Menezes, and Scott Vanstone. 2001. The Elliptic Curve Digital Signature Algorithm (ECDSA). *Int. J. Inf. Secur.* 1, 1 (aug 2001), 36–63. https://doi.org/10.1007/s102070100002

[12] Avish Karmakar, Naiwrita Dey, Tapadyuti Baral, Manojeet Chowdhury, and Md. Rehan. 2019. Industrial Internet of Things: A Review. In *2019 International Conference on Opto-Electronics and Applied Optics (Optronix)*. 1–6. https://doi.org/10.1109/OPTRONIX.2019.8862436

[13] David Mazières. 2015. The Stellar Consensus Protocol: A Federated Model for Internet-level Consensus.

[14] Satoshi Nakamoto. [n. d.]. Bitcoin: A peer-to-peer electronic cash system. ([n. d.]).

[15] Felipe Pezoa, Juan L Reutter, Fernando Suarez, Martín Ugarte, and Domagoj Vrgoč. 2016. Foundations of JSON schema. In *Proceedings of the 25th International Conference on World Wide Web*. International World Wide Web Conferences Steering Committee, 263–273.

[16] Ana Reyna, Cristian Martín, Jaime Chen, Enrique Soler, and Manuel Díaz. 2018. On blockchain and its integration with IoT. Challenges and opportunities. *Future Generation Computer Systems* 88 (Nov. 2018), 173–190. https://doi.org/10.1016/j.future.2018.05.046

[17] Ahmad-Reza Sadeghi, Christian Wachsmann, and Michael Waidner. 2015. Security and privacy challenges in industrial Internet of Things. In *2015 52nd ACM/EDAC/IEEE Design Automation Conference (DAC)*. 1–6. https://doi.org/10.1145/2744769.2747942.

[18] Maria Salama, Yehia Elkhatib, and Gordon Blair. 2019. IoTNetSim. In *Proceedings of the 12th IEEE/ACM International Conference on Utility and Cloud Computing*. ACM. https://doi.org/10.1145/3344341.3368820

[19] Fahad Saleh. 2018. Blockchain Without Waste: Proof-of-Stake. *SSRN Electronic Journal* (2018). https://doi.org/10.2139/ssrn.3183935

[20] Moysis Symeonides, Zacharias Georgiou, Demetris Trihinas, George Pallis, and Marios D. Dikaiakos. 2020. Fogify: A Fog Computing Emulation Framework. In *2020 IEEE/ACM Symposium on Edge Computing (SEC)*. 42–54. https://doi.org/10.1109/SEC50012.2020.00011

[21] Md Ashraf Uddin, Andrew Stranieri, Iqbal Gondal, and Venki Balasubramanian. 2021. A survey on the adoption of blockchain in IoT: challenges and solutions. *Blockchain: Research and Applications* 2, 2 (June 2021), 100006. https://doi.org/10.1016/j.bcra.2021.100006

[22] Gavin Wood. [n. d.]. Ethereum: A secure decentralised generalised transaction ledger. ([n. d.]).