The convergence of Blockchain and Machine Learning for Decentralized Trust Management in IoT Ecosystems

Tharindu Ranathunga
Munster Technological University
Cork, Ireland
tharindu.ranathunga@mycit.com

Alan McGibney
Munster Technological University
Cork, Ireland
alan.mcgibney@mtu.ie

Susan Rea
Munster Technological University
Cork, Ireland
susan.rea@mtu.ie

ABSTRACT

The EU data strategy postulates that by 2025 there will be a paradigm shift towards more decentralized intelligence and data processing at the edge. The convergence of a large number of nodes at the IoT edge along with multiple service providers and network operators exposes data owners and resource providers to potential threats. To address cloud-edge risks, trust-based decentralized management is needed. Blockchain technology has created an opportunity to decentralize IoT ecosystems, through its intrinsic properties and together with machine learning (ML) it can be used to provide a trusted backbone for managing IoT ecosystems to support automated and adaptive trust management. This paper presents a novel approach for cross-layer intelligent trust computation modelling leveraging ML and Blockchain for decentralized trust management in IoT ecosystems. The effectiveness of the proposed approach for flow-based trust assessment is demonstrated using the Hyperledger Framework and the Cooja-based simulation environment. Finally, an initial evaluation is presented to understand the performance in terms of scalability and trust convergence of the proposed model.

CCS CONCEPTS

• **Networks** → **Network management**; *Network monitoring*; **Cross-layer protocols**; • **Security and privacy** → **Malware and its mitigation**.

KEYWORDS

Blockchain, Machine Learning, Internet of things, IoT Ecosystems, Trust, Hyperledger

ACM Reference Format:
Tharindu Ranathunga, Alan McGibney, and Susan Rea. 2021. The convergence of Blockchain and Machine Learning for Decentralized Trust Management in IoT Ecosystems. In *The 19th ACM Conference on Embedded Networked Sensor Systems (SenSys'21), November 15-17, 2021, Coimbra, Portugal.* ACM, New York, NY, USA, 6 pages. https://doi.org/10.1145/3485730.3493375

1 INTRODUCTION

IoT ecosystems consist of IoT nodes, network services, and network participants such as organizations, consumers, governments, and businesses. The Next-Generation IoT initiative supports the implementation of the European Strategy [9]for data management across the cloud-edge-IoT continuum and identifies the need to manage security, verify trust and ensure the privacy and confidentiality of data while also managing the heterogeneity of nodes, networks, and data. Due to the growing number of connected heterogeneous nodes associated with multiple parties such as data owners, service providers and network operators and inherent risks associated with them, trust management in these ecosystems is challenging.

An IoT trust management mechanism should first prevent untrusted nodes from joining the network and otherwise identify any untrusted nodes as early as possible during network operation. An untrusted node is a node that performs a malicious activity on the network to gain economic and strategic advantage or can be a malfunctioning node that causes problems in the network operation. Untrusted nodes can cause security threats and attacks that can compromise data, nodes and entire networks such as distributed denial of service (DDOS), Spoofing, Routing attacks, among others. Ensuring trust is key to mitigating these threats in IoT ecosystems, which in turn contributes to supporting data veracity for IoT nodes. The National Institute of Standards and Technology (NIST) has identified five requirements that define trustworthiness in IoT ecosystems as part of their Smart Grid and Cyber-Physical Systems Program's CPS Framework 2016 [21]: *a)* security (confidentiality, integrity, and availability), *b)* privacy, *c)* resilience, *d)* reliability, and *e)* safety. The key objective of any IoT trust management mechanism should be to address these requirements during the design and operational phase.

According to recent studies, Blockchain and other Distributed Ledger Technologies (DLT) have the ability to address a number of unsolved issues in IoT trust management [8]. Blockchain offers decentralized operation and is already being leveraged to manage multi-stakeholder networks in IoT ecosystems. Furthermore, its inherent properties, such as immutability, transparency, and the underlying secure-by-design architecture, can be leveraged as a trusted backbone to enable decentralized trust management across the IoT edge-cloud continuum. DLT can record ownership and identities of nodes along with their interactions, thereby creating traceable footprints of their actions across layers, which can be used in addressing the security and resilience elements of trust. Moreover, Smart Contracts (SC) can be used to specify and enforce reliable behaviour in IoT systems (e.g., access control and reputation calculation) and automate compliance.

Previous studies have been conducted to leverage these concepts in IoT trust management, particularly in SC-based IoT trust reputation [8, 16]. These works rate the IoT service-providing nodes based on their interactions using predefined criteria. The existing models

SenSys'21, November 15–17, 2021, Coimbra, Portugal
© 2021 Association for Computing Machinery.
ACM ISBN 978-1-4503-9097-2/21/11…$15.00
https://doi.org/10.1145/3485730.3493375

do not consider the complexities and dynamics of next-generation IoT ecosystems associated with machine-driven, hyper-connected societies where demand for data continues to grow. Trust formation features in these ecosystems can change rapidly due to network and application requirements, and a trust model should be able to capture these changes seamlessly. Therefore, cognitive trust modelling and automated workflows that intelligently adapt to dynamic application, resource, and network requirements and states are required. Machine Learning (ML) is already being used to bring intelligence to these networks in terms of managing security and optimizing performance [2**?**]. This paper provides a novel solution for decentralized trust management in IoT ecosystems through the convergence of ML and Blockchain, enabling cross-layer intelligent trust computation modelling. The contributions of this work are threefold. First, we introduce a decentralized, trusted-off-chain workflow to manage ML models deployed in different layers in the IoT network. This allows us to swap in and out different ML models across the layers in the network in a trusted manner. Second, we propose an SC-based reputation mechanism to compute IoT node trust levels based on ML-based evidence. We also implement the proposed trust model and present an initial evaluation to understand the challenges in transaction scalability and convergence of the trust model.

The remainder of this paper is structured as follows: a brief review of related work is presented in Section 2, while Section 3 introduces the proposed trust model and the related workflow. In Section 4 we present a use case to demonstrate the implementation of the trust model. The performance of the proposed model is discussed in Section 5, followed by concluding remarks in Section 6.

2 RELATED WORKS

Trust management in IoT has been be an active area of research prior to the application of Blockchain, where trust was addressed from the perspective of peer to peer reputation systems for IoT services. A survey by Guo et al. [12] provides a comprehensive overview of these IoT trust reputation mechanisms. Two types of trust compositions (factors considered in quantifying trust) are identified: *a)* QoS: trust computation based on network quality of service [3], and *b)* Social: trust derived from the social relationship between users and IoT nodes [4]. One of the major concerns for these models is the impact of trust computations on the performance of resource constrained nodes.

With the emergence of edge-cloud architecture in IoT systems, node-level computations are offloaded to either the edge or cloud layers, including the associated trust computations [20]. The arrival of Blockchain initiated a paradigm shift towards more decentralized management of IoT systems. Novel Blockchain-based trust management models started to emerge, addressing issues related to centralized trust computation. Gunter et al. [8] proposed a trust-based authorization model for IoT leveraging Blockchain using smart contract-based trust reputation for access control evaluation. Trust levels are measured by evaluating positive and negative interactions between IoT nodes. This method relies solely on access policy violations to establish trust. Djamel et al. [16] presents a novel consensus mechanism at the FoG layer to agree upon the trust reputations of IoT nodes. Reputations are numerically calculated from user satisfaction ratings based on predefined criteria. In this solution trust computation mechanism along with trust model parameters are pre-loaded into the edge layer and therefore the flexibility in adapting to dynamic network conditions is reduced. Dedeoglu et al. [7] propose a Blockchain-based trust model that evaluates trust reputation based on observations at the data layer. This mechanism aims to address data veracity however this requires redundancy in the form of multiple sensors doing the same observations. Jayasinghe et al. [15] use ML to classify the trusted and malicious nodes based on the interactions of nodes and users. However, they do not discuss how the trained models can be secured and how network dynamics can affect the model accuracy.

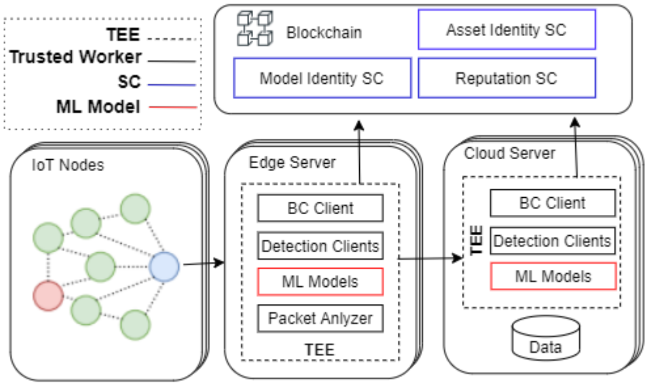

Figure 1: On- and off-chain components of the trust model.

The work presented in this paper addresses multiple concerns that the previous works have not considered. We present a comprehensive trust model that is capable of considering trust formations across different layers in the network. This is enabled by the proposed off-chain trusted computation workflow, which allows us to perform complex trust computations on multiple layers securely. In addition it is proposed to utilise ML to create a cognitive trust model that can adapt to network dynamics. This is achieved via a decentralized workflow for ML model management, enabling the required configurability in model training, deployment, and reputation calculation to facilitate adaptations to network dynamics.

3 TRUST MODEL METHODOLOGY

The proposed trust model follows the IIC TIoTA [14] reference architecture, which provides a specification for Blockchain and IoT integration. TIoTA identifies 3 layers for modern Blockchain-based IoT applications: *a)* Asset Layer (Nodes and Edge servers); *b)* Cloud Layer; *c)* Blockchain Layer; and the integration patterns between these layers. The proposed trust model is distributed across these three layers, where continuous trust assessments are done based on each layer's observations (evidence). In the asset layer, we detect malicious nodes based on the network QoS characteristics. The cloud layer is responsible for the detection of anomalies in data generated from IoT nodes. The Blockchain layer records the interactions and the evidence during the assessments in the other two layers and computes trust levels for the IoT nodes.

3.1 Main Components

The trust model approach consists of two type of components: *a)* off-chain Trusted Workers; *b)* on-chain Smart Contracts (SC); as shown in Figure 1. Off-chain trusted workers are deployed in edge and cloud layers inside a Trusted Executions Environment (TEE). These TEEs, along with a Blockchain-based workflow discussed in the Section 3.2 make sure that the executions of these workers are secured and traceable, and therefore they can be considered as trusted. These trusted workers are used to assess untrusted behaviours in the network continuously. This is done through ML clients and trained ML models deployed as trusted workers. The models in the edge layer are trained such that they can detect malicious nodes trying to perform network levels attacks. This is achieved through QoS observations on the edger layer. The trained models in the cloud layer are responsible for detecting and reporting anomalies on the data generated by IoT nodes. In both layers, Detection Clients shown in the Figure 1 are accountable for model execution and pre-processing, which involves feature extraction. The models are trained prior to the deployment through simulation, and can be updated when necessary. The model training process is discussed in Section 4.2. The BC clients in both layers push the outputs of the Detection Clients along with the observations used to the Blockchain through the use of on-chain SCs.

On-chain SCs are used to manage the identities of IoT assets and trusted workers, record the outputs of the trusted workers and compute IoT node trust reputations based on those outputs. This will create an immutable and traceable footprint of these transactions, ensuring resilience and integrity. The following SCs are defined to enable this workflow.

Asset Identity SC. Encapsulates all the attributes and functions of **asset** ID management. This includes the identity of the IoT end nodes, edge servers, and gateways. The Blockchain ledger act as a registry that keeps track of the changes occurring to asset attributes and status in their life cycle, which is bound to an immutable ID.

Model Identity SC. Exposes functions to register trained ML models that are deployed off-chain. This records model performance metrics (Precision and Recall), model hash value, deployment details and off-chain URIs to test data. This information is used in validating ML model integrity, where integrity in this context refers to ensuring the ML model has the stated precision and recall and ensuring that the model was not modified post deployment.

Reputation SC. Numerically calculate trust reputations based on the outputs of the registered model on chain. This is triggered by off-chain Blockchain clients periodically. This records node reputation on the ledger along with the ML-based evidence (input to the model and the detection result) that the reputation calculation was based on. The optimal frequency of these trust update transactions needs to be experimentally determined considering the performance of the Blockchain system, hosted infrastructure.

3.2 Trusted Work Flow Management

Next, we discuss how these components interact to ensure a trusted lifecycle for IoT nodes and how secure trust computation is done. Trusted execution of the SCs on-chain is guaranteed by its design through public-key cryptography, distributed execution, and consensus mechanisms. However, this needs to be explicitly maintained when off-chain executions are done. On the other hand, adding trusted off-chain execution to a blockchain network improves performance in terms of throughput and latency because computationally intensive executions can be off-loaded. Off-chain trusted workers are executed in TEEs which guarantees that code and data used within the TEE are protected.

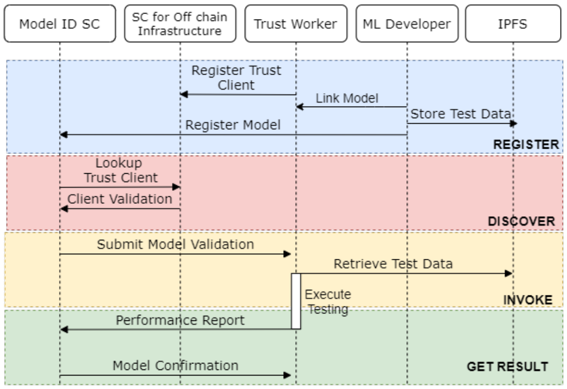

Figure 2: ML models and trust workers provisioning.

We propose a trusted execution workflow for our trusted workers leveraging the Enterprise Ethereum Alliance (EEA) Off-Chain Trusted Compute Specification [10]. The specification propose four steps for trusted off-chain execution: *a)* Register; *b)* Discover; *c)* Invoke; and *d)* Get Result.The sequence diagram in Figure 2 presents the proposed workflow for trusted provisioning, i.e. the secure deployment of the trusted workers on one of the layers in the network.

Registration of the trust workers is done through the asset and model identity SCs. This makes sure that workers deployed across the network layers are uniquely identified and traceable. Importantly, every ML model that will be deployed across the layers needs to be registered through the model identity SC as discussed in the 3.1. Each ML model is associated with a test data set that can verify the model performance metrics. These test data sets can be stored either on-chain or off-chain. A distributed interplanetary file system (IPFS) is recommended for off-chain storage to minimise risk of a single point of failure. When a SC needs to validate the performance metrics of a particular model which is triggered periodically, it can query in the Blockchain for the trusted worker responsible for executing the ML model and submit a validation request. The trusted worker will then retrieve the corresponding test data set, do the performance metrics validation, and send the

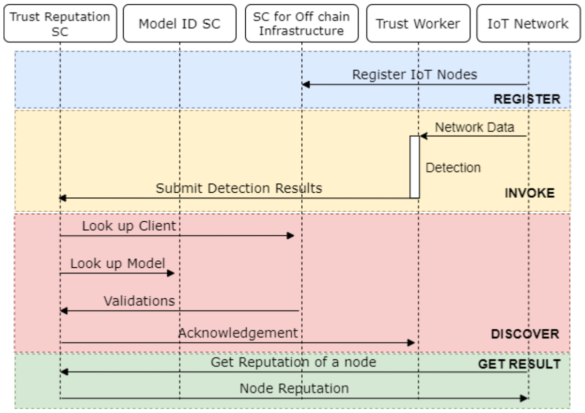

Figure 3: Trust updating work flow.

result to the SC that invoked the validation. This process ensures the integrity of the ML models.

The trust updating process takes place according to a similar execution work flow and is shown in Figure 3. Initially, IoT nodes need to register on the Blockchain through the Asset Identity SC and this is similar to the trusted worker registration process. Upon registration nodes are monitored by trust workers across IoT network layers (e.g., monitor network traffic in the edge layer). Trust workers push the output of ML models based on the network observations, to trust reputation SC to invoke reputation calculation functions. This SC does all the validations through identity SC and updates the trust levels of the node according the algorithms discussed in Section 3.3.

3.3 Trust Computation

The trust model has been designed to support the detection of multiple attacks and anomalies based on specific ML models, *M = {M₁, M₂, M₃,..Mₖ..,Mₘ}*, that can be dynamically plugged into different layers in the IoT network. The possible outcome of these models are binary, i.e., either positive or negative for a certain observation window. Two trust scores have been defined, i.e., $\mathrm{M}^{\mathrm{pos}}$ and $\mathrm{M}^{\mathrm{neg}}$ for the two outcomes as follows :

$$M^{\mathrm{pos}} = W^{\mathrm{pos}}\alpha^{\mathrm{pos}}, M^{\mathrm{neg}} = W^{\mathrm{neg}}\alpha^{\mathrm{neg}} \tag{1}$$

Where α is a predefined score where α^{pos} is the maximum score given to a node after a positive interaction and α^{neg} is the minimum score given to a node after a negative (malicious or anomaly) interaction. α^{pos}=1 and α^{neg}=-1 so that a trust score lies between -1 and 1. $W = \{W_1, W_2, W_3,..W_k..,W_m\}$ is used to weight α based on the accuracy of the ML model (M) which the interaction was evaluated.

One of the best measures that reflect model accuracy is *F-Score* [11]. It is calculated using the model's precision (*P*) and recall (*R*), where the precision is the number of true positives divided by the number of all positive classifications including those not classified correctly, and the recall is the number of true positive results divided by the number of all samples that should have been identified as positive. We use the *F-Score* measure to weigh the trust score of a given ML model as follow:

$$W = (1+\beta^2)\frac{PR}{\beta^2 P + R} \tag{2}$$

β is chosen such that recall is considered β times as important as the precision. W^{pos} and W^{neg} is calculated using positive and negative *F-Scores* respectively. β is set such that $\beta^{\mathrm{pos}} > \beta^{\mathrm{neg}}$ in order to invoke a higher importance to recall when a positive interaction happens. Less recall on positive detection means chance of that detection to be negative is higher, so we make sure trust score is increased weighted by a lesser F-score. So growing trust will more depend on the model recall.

The Trust Score of an IoT node *i* according to a given model M_j during the n^{th} observation window is given by the Model Score, *MS* function:

$$MS(n, \mathrm{M_j}, i) = \mathrm{M_j}^{\mathrm{x}} \tag{3}$$

where,

$$\mathrm{M_j}^{\mathrm{x}} = \begin{cases} M_j^{pos}, & \text{if } \mathrm{M_j(n, i)} \text{ is positive.} \\ M_j^{neg}, & \text{if } \mathrm{M_j(n, i)} \text{ is negative.} \end{cases} \tag{4}$$

The trust reputation (R) of a node *i* according to a given model M_j on the n^{th} observation window is calculated using the following function :

$$R(n, M_j, i) = (1-\gamma)R(n-1, M_j, i) + \gamma MS(n, M_j, i) \tag{5}$$

Where γ (0< γ <1)is an aging parameter used to consider trust decay over time. i.e., the decay of the trust reputation on previous observation window and the contribution to the latest observation.

4 IMPLEMENTATION

This section discusses the implementation aspects of the proposed trust model methodology including the architecture and model training procedure.

4.1 Architecture

The Blockchain layer is developed using the Hyperledger Fabric(HLF) framework [1]. HLF is preferred over other platforms due to its permissioned nature, scalability, pluggable consensus mechanisms and active developer community. The ordering service in HLF plays an important role in the ledger consensus and HLF supports RAFT [22] and Kafka [23] ordering services. RAFT is the preferred mechanism because:*a)* it is easy to setup, *b)* has native support, *c)* has the ability to decentralize governance, *d)* there is a possibility that RAFT will implement Byzantine Fault Tolerance (BFT) in the future [6]..

We use Hyperledger Avalon along with the Secure Container Environment (SCONE) [13] to orchestrate the off-chain components of our trust model. Avalon provides a framework to manage trusted workers, adhering to the EEA Trusted Computation Specification. It helps to maintain a registry of trusted workers that operate across layers, manage work-order between on-chain and off-chain components and preserves a log of work orders and acknowledgments. Currently, Avalon supports Intel SGX [19] TEEs, and SCONE allows us to run our trust workers as a docker container inside the SGX TEEs.

4.2 Simulation and Model Training

The Contiki-Cooja [5] simulator was used to simulate the IoT networks to test and validate the trust model. In this paper, only models operating at the edge server are considered. Initially, Cooja is utilised to produce a network traffic data set that is leveraged to train the ML models. Then these models are deployed on the edge tier as discussed in the Section 3.1. Finally, the overall trust model is validated through additional simulations.

Simulation Scenario Attack Model. The simulation scenario used to demonstrate the proposed model is an RPL-based node cluster where a set of publisher nodes publish data to a sink node periodically using the Routing Protocol for Low-Power and Lossy Networks (RPL). RPL is vulnerable to multiple types of attacks targeting resource consumption, topology, and traffic [18]. We have considered *a)* Flooding attack (resource consumption attack); *b)* Version number attack (topology based attack); *c)* Decreased rank

Attack	Model	Malicious		Benign	
		P	R	P	R
Flooding	AE	0.99	0.98	0.96	0.96
Version Number	SVM	0.66	0.94	0.89	0.51
Decreased Rank	LightGBM	0.65	0.88	0.95	0.82

Table 1: Performance metrics of trained models.

attack (traffic based attack); which covers all three types of RPL related attacks in validating the trust model.

Simulation Network Configuration. 3 simulation environments consisting of 10, 20 and 30 nodes are used. The type of nodes that were used in these simulations were Sky Motes, programmed to publish data to a sink node using the MQQT-SN protocol every 10 seconds. The networks' radio medium was based on the unit disk graph model (UDGM) with a predefined radio range of R, and the topology positioning was random. Each attack was simulated for each of these three network configurations for a duration of 30 minutes in order to create the training data sets. In each simulation, 10% of the nodes were changed to be malicious by changing the firmware of the motes to perform the attacks described above. In the end, the data from the three networks were fused to create one large data set for each type of attack with a unique identifier for each node. So 3 packet capture (PCAP) files were generated consisting of the network-level interactions during the attacks for the training process.

The model training workflow proposed in [2] was followed. An RPL specific PCAP processor was implemented based on [17] and a general feature extraction module. The feature extraction module can be extended to extract attack specific features. The general set of features extracted here are: *a)* highest layer protocol packet count; *b)* transport layer protocol packet counts; *c)* packet length; *d)* RPL control messages count (DIS, DIO and DAO [18]); *e)* received packet count; *f)* transmitted packet count; for each transmission window (i.e. any continuous period of time after the network started). The training transmission window was set 10 seconds following the approach in [2] that demonstrated model accuracy. Both the packet analyzer and the feature extraction modules are used in the model training process, and it is deployed in the gateway layer as a trusted worker during the network operation.

Different models were evaluated as part of the training process from simple classification models such as support vector machine (SVM) to deep learning based models such as Autoencoders (AE). The model performance metrics for the best performing models corresponding to the three type of attack scenario are shown in Table 1. These metrics are registered on the Blockchain during the model registration process as described previously. The model trust scores are autonomously calculated on chain (e.g., for the trained AE M^{pos} = 0.96 and M^{neg} = -0.98 based on the R and P values).

5 EVALUATION

Next, we discuss the initial evaluation of the proposed approach based on understanding the constraints/challenges of using a Blockchain based solution as well as an initial evaluation of the ML based trust model.

To evaluate the performance of the Blockchain-based approach when executing our trust model, a benchmark performance assessment was conducted using the Hyperledger Caliper benchmark tool [24] on the HLF network setup depicted in Figure 4. Each instance of the network was a n1-highcpu-16 (16 vCPUs, 14.4 GB memory) Google Compute instance. HLF 2.1.0 was used with its default configurations. The metrics considered in the model performance evaluation are: *a)* throughput: the number of transactions that the Blockchain can execute and add the outcome of the execution to the ledger per second (*TPS*), *b)* latency: time taken from transaction invocation until the transaction is added to the ledger..Based on our test setup a maximum throughput of 190 *TPS* was achieved in relation to trust update transactions as shown in the left of Figure 5 with the average latency being less than 5 seconds (while maintaining 100% throughput). Throughput increases linearly when the transaction send rate increases until the maximum possible throughput is reached. This provides an upper limit in terms of the model's scalability, which defines the limit for trust update requests, in this case it is a maximum 190 per second from off-chain DLT clients for the specific HLF network configuration.

The convergence of the trust model based on the trained AE (which detects flooding nodes) is studied by varying the ageing parameter(γ) through the Cooja simulation environment as discussed in Section 4.2. β^{pos} was set to 2 to make recall twice as significant as precision. The trust reputation of a benign and a malicious node is calculated for the trained AE as shown in the centre and right of Figure 5 respectively. The ground truth in both scenarios is known as we know which ones are trusted and which ones are malicious (i.e., nodes that flood the network). As defined in 3.3 reputation of a fully trusted node is 1, and fully un-trusted node is -1. The reputation of a node joining the network is set to 0.5 through the asset identity SC. This assigns an immutable ID to the node through a verification process that is out of this paper's scope. The trust convergence is compared for 3 different aging parameters: γ=0.3, 0.5, 0.8. In both scenarios the benign and malicious nodes trust convergence is faster when γ is closer to 1. As seen in Figure 5 (center) the trust reputation based on the trained AE converges towards the ground truth in the benign scenario. However, in the malicious scenario, it takes approximately 75 seconds to show an impact on the trust reputation shows some converging behaviour as it shifts to an un-trusted state. This is because the malicious node only becomes active and advertises its position and joins the network after this period. The trust reputation oscillations in 5 (right) between 150-200 seconds is due to false-positive detection of the ML model. A mechanism to slow the trust growth and decay (i.e. by using a growth function like Gompertz function) during false negatives and positives needs to be further investigated.

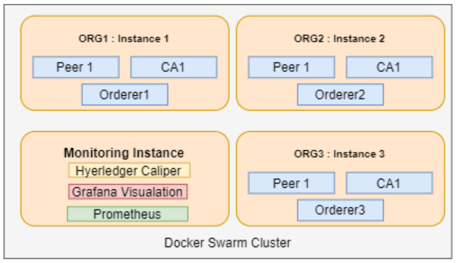

Figure 4: Blockchain network.

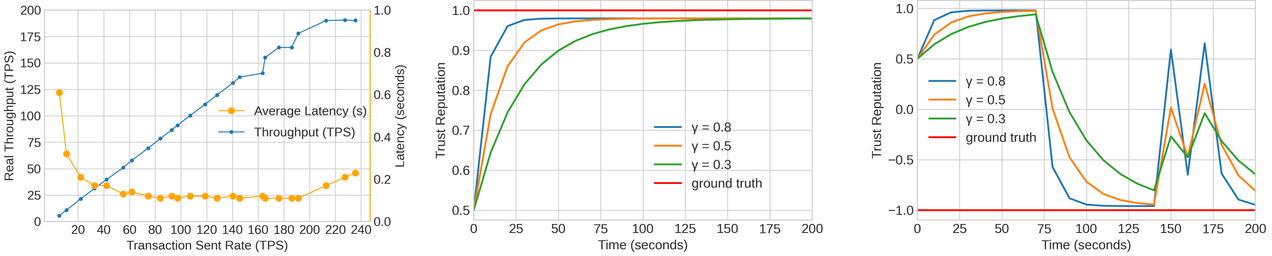

Figure 5: Evaluation results: throughput (left), benign TR (center), and malicious TR (right).

6 CONCLUSION AND FUTURE WORK

In this paper, a Blockchain-based approach for cross-layer intelligent trust computation in IoT ecosystems was presented. We propose a trusted ML-based reputation calculation mechanism that enables us to consider cross-layer observations for trust computation. We discuss the implementation aspects of on-chain and off-chain workflows, which demonstrate the qualitative feasibility of our solution. In addition, we presented our initial evaluation to understand the BC integration challenges and trust convergence of the model. As future work, we will study the behaviour of our model under more network dynamics, more attacks on other layers of the network and further investigate how recall and precision affect the trust reputation. In addition, we will investigate how trust aggregation can be done across the edge-cloud continuum as well as exploring the trigger conditions for ML model retraining.

Acknowledgements. This work has been part funded by the Sustainable Energy Authority of Ireland under the SEAI Research, Development and Demonstration Funding Programme 2018, Grant number 18/RDD/262, and by the SFI Research Centres 2017 PhD Awards Program under SFI CONNECT 13/RC/2077.

REFERENCES

[1] Elli Androulaki et al. 2018. Hyperledger Fabric: A Distributed Operating System for Permissioned Blockchains. In *Proc. of EuroSys*.

[2] Semih Cakir, Sinan Toklu, and Nesibe Yalcin. 2020. RPL attack detection and prevention in the Internet of Things networks using a GRU based deep learning. *IEEE Access* 8 (2020), 183678–183689.

[3] Dong Chen et al. 2011. TRM-IoT: A Trust Management Model Based on Fuzzy Reputation for Internet of Things. *Comput. Sci. Inf. Syst.* 8, 4 (2011).

[4] Ray Chen, Fenye Bao, and Jia Guo. 2015. Trust-based service management for social internet of things systems. *IEEE T DEPEND SECURE* 13, 6 (2015), 684–696.

[5] Contiki OS Community. [n.d.]. Contiki-OS. contiki-os.org. Version 3.1, Online, 2021-08-21.

[6] Christopher Copeland and Hongxia Zhong. 2016. Tangaroa: a Byzantine Fault Tolerant Raft.

[7] Volkan Dedeoglu et al. 2019. A Trust Architecture for Blockchain in IoT. In *Proc. of MobiQuitous*.

[8] Guntur Dharma Putra et al. 2021. Trust-based Blockchain Authorization for IoT. *arXiv e-prints* (2021), arXiv–2104. https://arxiv.org/pdf/2104.00832.pdf

[9] Horizon Europe. 2021. *Digital, Industry and Space.* Technical Report. Horizon Europe.

[10] Bill Gleim et al. 2018. Enterprise Ethereum Alliance Off-Chain Trusted Compute Specification V1. 1. https://entethalliance.org/wp-content/uploads/2019/05/EEA_Off_Chain_Trusted_Compute_Specification_V0_5.pdf. Enterp. Ethereum Alliance 2019.

[11] Cyril Goutte and Eric Gaussier. 2005. A probabilistic interpretation of precision, recall and F-score, with implication for evaluation. In *Proc. of ECIR*. Springer.

[12] Jia Guo et al. 2017. A Survey of Trust Computation Models for Service Management in Internet of Things Systems. *Comput. Commun.* 97 (2017).

[13] Mujtaba Idrees. 2021. *Hyperledger Avalon with SCONE.* Technical Report. T-Systems Multimedia Solutions.

[14] IIC. 2019. IIC LIAISON: TRUSTED IOT ALLIANCE (TIOTA) REFERENCE ARCHITECTURE. https://hub.iiconsortium.org/portal/IndividualContribution/5db03a83f7679b000f0e762f. 2020-02-17.

[15] Upul Jayasinghe, Gyu Myoung Lee, Tai-Won Um, and Qi Shi. 2018. Machine learning based trust computational model for IoT services. *IEEE Transactions on Sustainable Computing* 4, 1 (2018), 39–52.

[16] Djamel Eddine Kouicem et al. 2020. A decentralized blockchain-based trust management protocol for the internet of things. *IEEE Trans Dependable Secure Comput* (2020).

[17] Gobinath Loganathan, Jagath Samarabandu, and Xianbin Wang. 2018. Sequence to Sequence Pattern Learning Algorithm for Real-time Anomaly Detection in Network Traffic. In *Proc. of CCECE*.

[18] Anthea Mayzaud, Remi Badonnel, and Isabelle Chrisment. 2016. A Taxonomy of Attacks in RPL-based Internet of Things. *Int. J. Netw. Secur.* 18, 3 (2016), 459–473.

[19] Frank McKeen et al. 2016. Intel® software guard extensions (intel® sgx) support for dynamic memory management inside an enclave. In *Proceedings of HASP*.

[20] Suneth Namal et al. 2015. Autonomic trust management in cloud-based and highly dynamic IoT applications. In *2015 ITU Kaleidoscope: Trust in the Information Society (K-2015)*. IEEE, 1–8.

[21] NIST. 2016. *CPS PWG Cyber-Physical Systems (CPS) Framework Release 1.0.* https://pages.nist.gov/cpspwg/

[22] Diego Ongaro and John Ousterhout. 2014. In search of an understandable consensus algorithm. In *Proc. of USENIX Annual Technical Conference.*

[23] Parth Thakkar et al. 2018. Performance benchmarking and optimizing hyperledger fabric blockchain platform. In *Proc. of MASCOTS*.

[24] The Linux Foundation. [n.d.]. Caliper. https://www.hyperledger.org/projects/caliper. Version 0.4.2, Online, 2021-08-21.