BTrust: a new blockchain-based trust management protocol for resource sharing

Badr Bellaj, Aafaf Ouaddah, Emmanuel Bertin, Noel Crespi, Abdellatif Mezrioui, Khalid Bellaj

BTrust: a new Blockchain-based Trust Management Protocol for Resource Sharing

Badr BELLAJ[1 2 3], Aafaf OUADDAH[1*],Emmanuel BERTIN[3],Noel CRESPI[2], Abdellatif MEZRIOUI[1] and KHALID BELLAJ[4]

[1] National Institute of Post and Telecommunication (INPT),Rabat,Morocco
[2] Institut polytechnique de Paris, Paris,France
[3] Orange Lab, Caen, France
[4] Mchain, MA
Corresponding e-mail: bellaj.badr@mchain.uk

Abstract. The emergence of blockchain technology and cryptocurrencies opened the possibility for building novel peer-to-peer(P2P) resource allocation and sharing models. However, the trustless nature of these P2P models creates the need for reliable and effective trust and reputation mechanisms to minimize the risk of accessing or interacting with malicious peers. Blockchain technology, which is renowned for ensuring trust in trustless environments, provides us with new mechanisms to overcome the weaknesses of the existing reputation and trust management protocols. This paper proposes BTrust, an innovative decentralized and modular trust management system based on blockchain technology for evaluating trust in large-scale P2P networks. To quantify and assess the trustworthiness of peers and identify malicious peers, BTrust introduces a multi-dimensional trust and reputation model to represent trust and reputation scores in a single value derived from multiple parameters with appropriate weightings. Other contributions of this paper include the combination of recommendation and evidence-based approaches into a single system to provide a reliable and versatile way to compute trust in the network, an optimized trustless bootstrapping process to select trustworthy peers among neighbour peers and an incentive mechanism to encourage truthful feedback. We implement and evaluate the BTrust protocol using simulations and show that BTrust is highly resilient to failures and robust against malicious nodes.

1 Introduction

Peer-to-peer (P2P) systems have evolved from simple file sharing over the internet to advanced distributed resource sharing such as Crowdcloud [17] and blockchain-based resource sharing. Nowadays, with the surge of cryptocurrencies, there has been growing interest in monetizing computing resources on the blockchain[16] [37] [3][32]. Users are willingly ready to rent their unused storage [34] [13], CPUs [21] or event internet bandwidth [33] to get paid in crypto-tokens. However, sharing resources between users demands a certain level of trust, control of the quality of service (QoS) and digitally enforceable contracts for resource use.

To ensure trust within P2P networks, help choose reliable resources and obviate peer misbehaviour, multiple reputation-based trust management systems(RTMS) [14][39], have been proposed. These RTMS build trust by relying on community-based reputations. They help peers to measure the trustworthiness of others and rate the QoS based on their reputation and mutual past experiences. In these schemes, a peer-to-peer overlay trust network is established. Each device (represented interchangeably as node or peer) is considered simultaneously a client and a service provider whose quality of QoS and trustworthiness need to be evaluated.

1.1 Limitations of traditional RTMS

The early traditional RTMSs presented multiple shortcomings related to the storage, update and dissemination of trust data that needed to be addressed effectively. Many of these systems compute a global trust score based uniquely on direct and indirect interactions between peers, ignoring other important factors such as device security or user behaviour(UB). When it comes to the adopted architecture, these systems adopt either the decentralized or centralized approaches to store and compute peers' feedback and reputations. The disadvantages of centralized models [14][11][4], which use centralized servers and databases for trust management, are their negative effects on the scalability and security of the underlying infrastructure. On the other hand, the adoption of a distributed approach [39][19][2][35] solved most of the centralized-based approach issues but it created new ones. The proposed decentralized models store feedback and compute a global trust value for each peer using a trust manager. This latter notion introduces multiple issues.

- First, a malicious trust manager could intentionally affect the computation of the trust value by ignoring some feedback.
- Second, malicious nodes could attempt to discredit a given peer by attacking the trust manager.
- Third, the number of trust managers scales linearly with the number of peers.
- Fourth, the trust manager for a peer may be unavailable, leading to the failure of the trust computation.

1.2 The Blockchain as a solution

With the emergence of the Blockchain, many Blockchain-based RTMS (BRTMS) have been proposed to cope with the aforementioned shortcomings [12]. For instance, in [31], a BRTMS was proposed to store educational records of achievement and credit, such as degree certificates. Another BRTMS have also been used in IoT/sensors networks to manage trust and authentication, as presented in [7], [26], [9]. Other works, such as [36] and [23], proposed a BRTMS for vehicular ad hoc networks (VANETs) environments. In [5], [30] a BRTMS has been proposed for e-commerce domain. Although these solutions present robust alternatives, most proposed solutions are service-orientated and cannot be applied in a different context, thus the need for a service agnostic blockchain-based RTMS.

1.3 Contributions

In this article, we propose a generic BRTMS solution called BTrust that can be applied to different blockchain-based networks. The proposed solution creates a trust overlay network that addresses the issues of dissemination, incentivisation, and the storage and retrieval of feedback in a P2P network without requiring trust managers or centralised operators. BTrust harnesses the underlying blockchain to manage multiple services (such as identification, access control, micropayments, etc.) to provide a resilient and versatile reputation system capturing new dimensions that can help detect compromised devices, even if they behave correctly.

This work extends the existing efforts on combining Blockchain and trust management systems [12]. The main contributions of BTrust could be listed below:

- The novelty of BTrust relies on the adopted trust formula, which combines recommendation and evidence-based approaches to determine the trustworthiness level of peers in large-scale networks. We make use of smart contracts to both manage important services – serving as a basis for evaluating user's behaviour and the security assessment of a device– as well as to compute the level of trust and disseminate it.
- We harness random walks for selecting trustworthy peers among neighbour peers without causing overloads in the network (The cold start problem).
- BTrust relies on Remote attestation (RA) [20] — a mechanism that allows authorized parties to detect the changes that occurred in remote devices– to protect the integrity of a device (hardware and software).
- We propose an incentive mechanism to encourage truthful feedback.

1.4 Paper organisation

The rest of the paper is organized as follows. In Sections 3 and 2, we present the related works and background of trust and reputation management solutions. Our new reputation management model implemented on top of Blockchain Technology is presented in section 4. We outline the BTrust algorithms in sections 5 and 6 and then present the results of the evaluation of the BTrust protocol using simulations in section 7. Finally, in section 8 we summarize this work by identifying some challenges and promising future avenues, in our conclusion.

2 Background

2.1 Blockchain and Smart Contract

Blockchain: Blockchain technology was initially introduced by Satoshi Nakamoto in 2009 [28] as the underlying mechanism of Bitcoin, as a solution to the double-spending problem [28]. Concretely, Blockchain can be defined as a replicated database (ledger) among the participants of a peer-to-peer network (Figure 1)

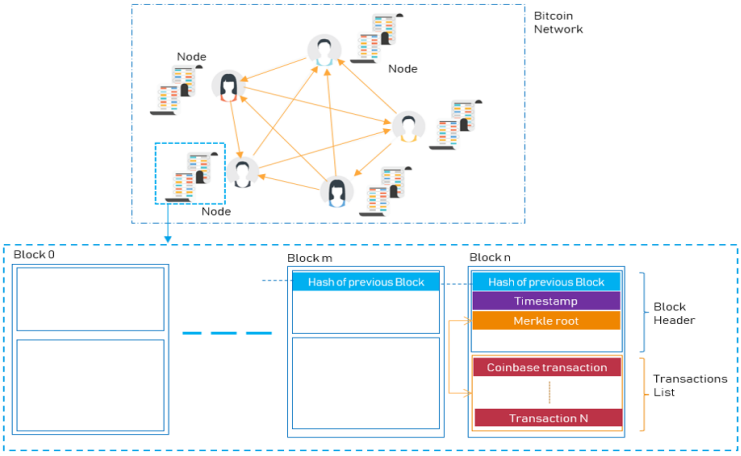

Fig. 1. Basic blockchain network

and managed by a consensus mechanism. Transactions are packed into block units which are chronologically ordered and attached using cryptographic hashes to ensure data integrity. Blockchain technology removes the need for a trusted and centralized entity that would be controlling the network and responsible for establishing trust. The term Blockchain (as a single word) did not appear in Bitcoin's initial paper [28], but the term "chain of blocks" was used instead. This new nomenclature was subsequently coined by projects that came years after [6] to distinguish between Bitcoin and its underpinning mechanisms.

Smart contract: The Smart contract concept is one of the key innovations of Blockchain technology. The concept was proposed even before its emergence by N. Szabo who first coined the term smart contract and defined it as a "computerized transaction protocol that executes the terms of a contract. The general objectives of smart contract design are to satisfy common contractual conditions (such as payment terms, liens, confidentiality, and even enforcement), minimize exceptions both malicious and accidental, and minimize the need for trusted intermediaries" [27]. However, this objective was only truly concretized with the emergence of Blockchain technology. Concretely, a smart contract is a deterministic executable script –written in a high-level programming language– that codifies a given logic (e.g. a business logic) as a set of instructions for manipulating the states recorded in Blockchain. The script's clauses or functions are invoked by external transactions and executed by the validators in the network. The execution result is then recorded in the blocks.

3 Related Works and Comparative analysis

3.1 State of the Art

Thanks to its promising features, Blockchain has been investigated intensively by the research community to build new trust management systems.

Dennis and Owen [8] presented one of the early BRTMS. They proposed a reputation system to store reputation from completed transactions on a new Blockchain network in which transactions are validated by Bitcoin miners using merged mining. They propose a simple schema, where after receiving the correct file, the user sends an encrypted transaction consisting of the reputation score. This score is calculated using a single-dimensional reputation based on the non-satisfactory transactions in which the user received the file they requested. To reduce malicious transactions on the network, they propose a proof-of-stake system, where a user with a low, or no reputation, stakes a small amount of currency (Bitcoins) into a triple signed wallet. However, the use of Bitcoin as a validation network would cause important latency since Bitcoin takes up to 10 minutes to process each block [24] and there is no guarantee that reputation transactions will be mined sequentially in order because miners are free to choose which transaction to validate. Another issue is the ability of a single user to generate multiple identities and promote his reputation since they link the indemnity creation of an identity to the IP address of a user. Moreover, the approach adopted for selecting the peers from which the user will download the file is insecure. The choice of the source depends on the friend's peer reputation, thus malicious nodes could focus their activity on proposing friendship to the newly joining peers and impacting the computation of other peers' reputations. Furthermore, operating this solution on a large scale is unpractical since each peer has to operate a resource-expensive full Bitcoin node. These properties make it unlikely that a network with a high amount of low resourced users, such as IoT devices, would implement this reputation system.

Another BRTMS targeting resource sharing in P2P networks is presented in [15]. The authors proposed a multi-level reputation scoring system for a Cluster Of Non-Dedicated Interoperating Kernels (Clondike). In a Clondike system, each participating user contributes computing performance of their machines and uses the computing performance of the other workstations for his computing. In such systems, there is a need for an RTMS to ensure fair usage of resources among all nodes of an inter-organisation cluster, as well as to identify and eliminate nodes that tend to overuse resources of the whole cluster and do not contribute by their computation resources or contribute by false results. To achieve these goals, they base the trust system on a relation between a node and the system instead of building an interpreted trust based on the feedback of other nodes. Instead of trusting reputation data that single nodes exchange, each node interprets behaviour data, which is stored in a blockchain, with its strategy. This approach presents some drawbacks. The reputation is built only on positive feedback (kudos), therefore peers cannot rate the bad behaviour of their counterparts and this bad behaviour is not logged into the blockchain. Moreover, the authors did

not experiment with their BRTMS on a large-scale network to prove its scalability, since they experimented on a 3 node cluster both 2 fair nodes successfully penalized an abusive one.

In [40], a Proof-of-Trust (PoT) consensus protocol for enhancing data validation and accountability in crowdsourcing services is proposed. The authors introduced a hybrid blockchain architecture, based on a consortium blockchain acting as the underlying deployment network, while a public blockchain is used to ensure validation for the novel consensus protocol. The proposed PoT selects transaction validators to validate collected data based on their trust values while leveraging RAFT leader election and Shamir's secret sharing algorithms. The consortium is connected to the public blockchain through a set of gateways. Each consortium member has a consortium ledger management node and a gateway node. The gateway nodes are situated in a demilitarized zone (DMZ), providing isolation of the private consortium network from the open Internet environment. However, with the limited number of gateways, there is a risk of disconnection of the consortium network from the public blockchain, if the gateways are down or under DoS attack. Consequently, such disconnection will break the consensus within the network.

In [1], the authors proposed a BRTMS for the Autonomous System (ASes). The proposed BRTMS is devoted to evaluating network providers based on their adherence to Service Level Agreements (SLAs) regarding interconnection agreements. The method used to calculate reputation is defined by a pre-agreed and publicly known scoring function and the results are written in a private Blockchain. They propose the use of SLA scores, which are quantified using a smart contract, for helping ASes choose their business partners. They introduce a fair scoring protocol that allows the scores to be deterministically computed from measurements of forwarding performance. The protocol requires each SLA score to be written on the blockchain and achieves privacy preservation by adopting an order-preserving encryption mechanism.

In [29] the authors argue that Distributed Ledger Technology (DLT) can be leveraged to create and manage the trust relationship between peers in a decentralized manner. They propose the LegIoT framework, which utilizes a DLT to store, manage and process trust information, enabling mutually distrusting parties to participate in a network.

In [18], authors leveraged the IoT with Ethereum's Blockchain to provide a reputation-based monetization system for IoT data, whose quality is ensured for consumers through reviews and ratings. They proposed a publish-subscribe model based on smart contracts, whereby a data owner shares information about the topics and subscribers make deposits, consume data and rate the service quality.

In [25], Bitcoin blockchain was proposed to be used as a public platform to manage the trust for decentralized sensor networks, as well as for logging nodes activities. These logs are then used as an indication of a node behaviour and thus a basis from which to compute the node trust score.

In [10], the authors introduced a distributed credit-based Blockchain system with a built-in reputation mechanism. They proposed a distributed ledger –obligation chain– for storing obligations of commitments. The service provider checks the obligation chain and the payment chain (Bitcoin blockchain) to assess the credibility of the obligation issuer by relying on the credit history of consumers and their ability to pay off their obligations.

3.2 Comparative Analysis

In Table 1, we present a comparative study of our work with the state of the art solutions, and show that the proposed solution outperforms other similar approaches.

Table 1. Comparative analysis of BTrust and Related work

Solution	Field of application	Intended Improvement						Performance Measured
		Cold start problem	Incitivation	Assessment of bad behavior	Extensibility of Trust Formula	Metrics of trust	Trust Dissemination	
8	P2P	No	No	Yes	No	Bad or good transactions	Client side	Not Provided
15	Open multi-agent system in P2P clusters	YES	No	Yes	No	Immigration request actively rejected, Immigration request confirmed Task result verified	Consortium Blockchain	Number of kudos delivered
40	Croud-sourcing/ Sensing	YES	yes	Yes	No	Total Transaction Amount validation Times	Blockchain	Accuracy Scalability
1	Autonomous systems	No	No	Yes	No	SLA score of an agreement the SLA score of a network	Oracle and permissioned blockchain	Not provided
18	Monetization of data in IoT	No	No	No	No	Review Rating	Smart Contract on Ethereum Blockchain	Not provided
10 9	IoT /Sensor Network / Edge Computing	No	No	No	No	History of obligation fulfillment	Obligation chain	Reputation bootstrapping delay
25	IoT /Sensor Network / Edge Computing	No	No	Yes	No	Miner approval Blame Payload Renew Payload Ban Payload	blockchain	Trust level
BTrust	Generic	Yes	Yes	Yes	Yes	The feedback a peer obtains from other peers; The total number of transactions that a peer performs; The security assessment of a from less or secure ones peer for discriminating vulnerable devices The user's behavior	Smart contract	Accuracy Effectiveness against dynamic and static Malicious Peers Convergence of BTrust

4 Reputation and trust management in BTrust

Within this section, we present an overview of the proposed BTrust reputation and trust model and we describe the underlying architecture.

4.1 BTrust components

In our model we envision a P2P network, which involves the following 9 entities (Figure 2) playing various roles:

Network operator (NO) A No is a community-based entity operating a service-oriented network. The NO is responsible for the initialization, provisioning and updating of BTrust agents, and for establishing the first nodes in the network. The NO is also responsible for deploying and updating the different BTrust smart contracts. We envision NO as a decentralized entity operating similarly to the maintainers of BC projects such as Ethereum or Bitcoin.

Certificate Authority (CA) The CA is an entity providing valid identities and certificates for the members of the blockchain network. All the actions of the CA (Creation, Validation, Revocation, etc.) are recorded transparently in the blockchain.

Blockchain for reputation (BC) The BC acts as a shared database for storing reputation and computing feedback data. We assume that the BC can store transparent and immutable data as well as execute smart contracts. In BTrust the underlying BC can be either public, private, permissioned or permissionless.

BTrust agent (TA) Every peer-to-peer node hosts an agent that maintains the BTrust protocol rules and evaluates the security of the device through interactions with built-in security tools. This agent ensures the communication between the peers and between a peer and the different BTrust smart contracts (Reputation, Patch, Access control and Identification). The TA is also responsible for managing patches and communicating periodically with the patch distribution server to discover new patch information and rate the peer according to its patching activity.

The device manufacturer (DM) The DM is the entity that creates each device. The DM securely installs in each device the bootstrapping credentials (e.g. the Endorsement Key) needed for the Remote attestation.

Patch distribution server (PDR) A PDR is an entity that informs and communicates to BTrust agents (Patch clients in this case) security patches available for each device using a patch distribution protocol. The PDR manages a patch DB which is updated by the network operator and by security vendors.

Remote attestation server (RAS) The RAS is an entity that manages the entire process of attestation. It contains all the proofs for validating the devices' integrity (such as the BIOS or Boot loader's integrity, and other system

measurements) and checks whether the remote attestation client is trustworthy. Attestation protocols usually assume a single prover, but in our case may involve many provers. In BTrust the remote attestation is not performed on demand but the BTrust agent periodically (and securely) measures and records its own hardware and software state and sends them to the RAS.

Device owner (DO) A DO is the person or entity that physically owns the device and is ultimately responsible for that device and how it is being used. The DO is responsible for onboarding his devices, as well as for transferring ownership of his device to another individual or entity. A device may have only one owner at any given time.

The device user (DU) A DU is the individual or entity that uses a device. In BTrust's network, the DU (or the DO) is represented by his or her cryptographic credentials (an address and public key pairs).

4.2 BTrust smart contracts

To perform its different roles, BTrust relies on four different smarts contracts:

Reputation smart contract (RSC) Builds the trust graph and logs the trust of each peer in the network. Besides, it manages the peers' feedback and the incentivisation process.

Identification smart contract (ISC) Manages the identification and enrolment of the peers, device owners and users as well as the remote attestation. Device identification includes information about a device (the device profile) that helps the patch server send appropriate patches to the BTrust client when new patches are available in a patch DB.

Access control smart contract (ASC) Determines the access rules defined by the DO. Each DO defines an access control policy for his devices, otherwise, a default policy is applied.

Patch and security smart contract (PSSC) Provides the security patches and evaluates if the recommended patches have been applied; stores the evaluation of the security assessment of a device sent by the BTrust agent.

All these smart contracts are developed, deployed and upgraded solely by the NO. In order to utilise BTrust, we assume that the device should have (1) sufficient performance for the required public-key cryptographic operations, (2) a sufficient energy supply to perform the required operations, (3) enough non-volatile storage space to store the blockchain data and cryptographic keys, and (4) hardware features to support remote attestation. If the device lacks the last feature a software-based Attestation mechanism can be used.

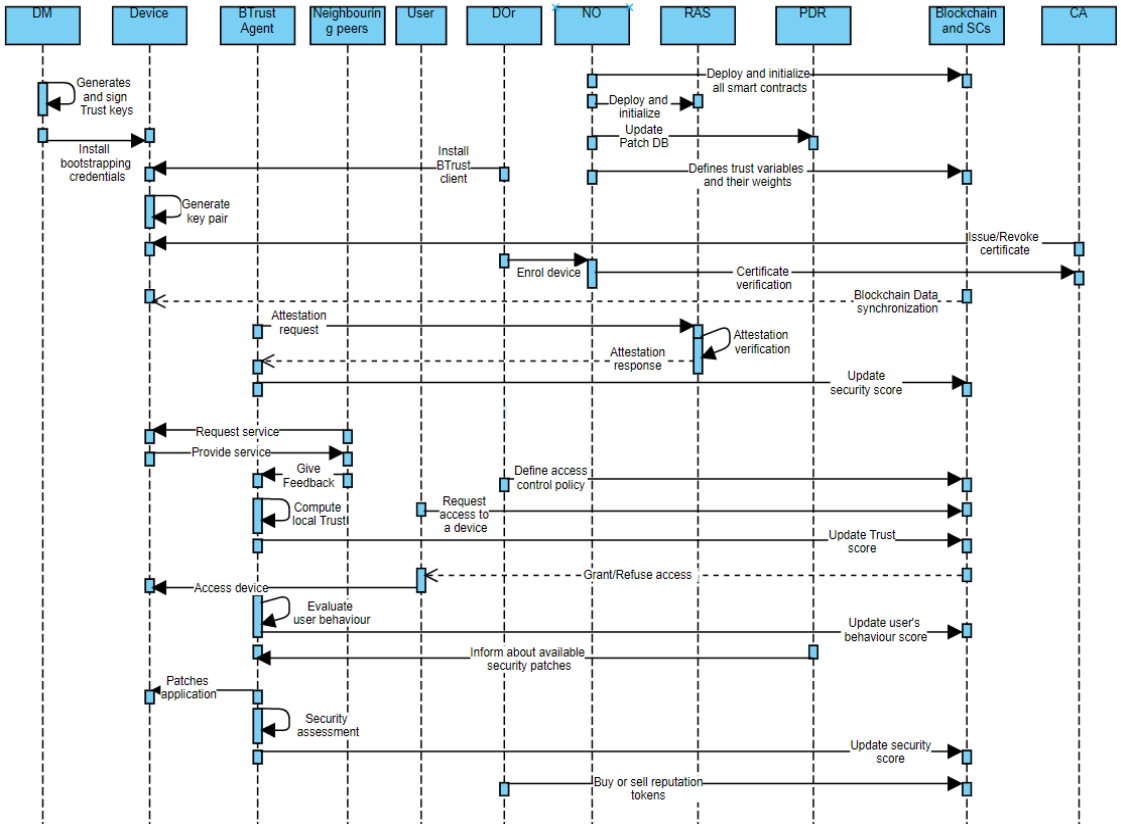

Fig. 2. A simplified overview of interactions between different entities in BTrust architecture

Before joining the network, on-device attestation keys (Endorsement Key) are injected into the device during the manufacturing process and signed by the NO or the CA. The device's Trusted Platform Module (TPM) signs the PCRs (Platform Configuration Registers), and registers for securely maintaining measurements inside the TPM with various attestation Identity keys(AIKs) that it generates. The BTrust agent extends the PCRs at runtime by writing a hash code into them. We consider that at any point in time, the number of active peers may be different, and not known in advance.

4.3 Trust factors

In BTrust, a peer's trustworthiness is defined by a combination of the evaluation of the peer it receives from other peers in the past, alongside behavioural mon-

itoring and detection of abnormal activities. In developing BTrust, we consider four important factors for such evaluation:

- The feedback a peer obtains from other peers;
- The total number of transactions that a peer performs;
- The security assessment of a peer for discriminating vulnerable devices from less or secure ones; and
- The user's behaviour.

Peers feedback In BTrust, trust is based on feedback gained directly and indirectly from other peers, and the total trust score is calculated based in part on the average of all recommendations, weighted by the trust degrees of the other peers.

Number of Transactions The transaction volume is an important factor that reflects the degree of satisfaction among different peers. We consider the ratio of the total amount of satisfactory transactions received over the total number of transactions received by a peer from another peer.

The user's behaviour BTrust aggregates user data from a set of devices to build a reputation score for the users, creating the possibility of tracking a user's behaviour across many devices. At each device, the behaviour is evaluated based on the log of successful and failed events for the services managed by the blockchain using smart contracts. For example, BTrust monitors and evaluates the user behaviour based on three major services:

- Integrity and attestation services: Since devices and their system software can be replaced by malicious users or a device owner, BTrust relies on the Remote attestation (RA) technique, to detect compromised entities. Such that, each device's identity is primarily attested before the device can access a network and afterwards the device is periodically assessed and attested to ensure its integrity.
- Financial transactions: A user can be evaluated regarding their financial transaction records. For instance, a malicious user could try to perform a double-spending attack or launch a DDoS (Distributed Denial of service) attack on the network with malicious transactions.
- Access control service: The user's access activity whether for his own resources or those owned by others is evaluated. If the user access activity deviates from the rules defined by the access control smart contract, he is deemed suspicious and thus the peer reputation.

It is worth noting that other aspects can be considered to evaluate user behaviour such as Spamming activity or behavioural patterns. For simplicity, we choose to adopt only the three aforementioned services.

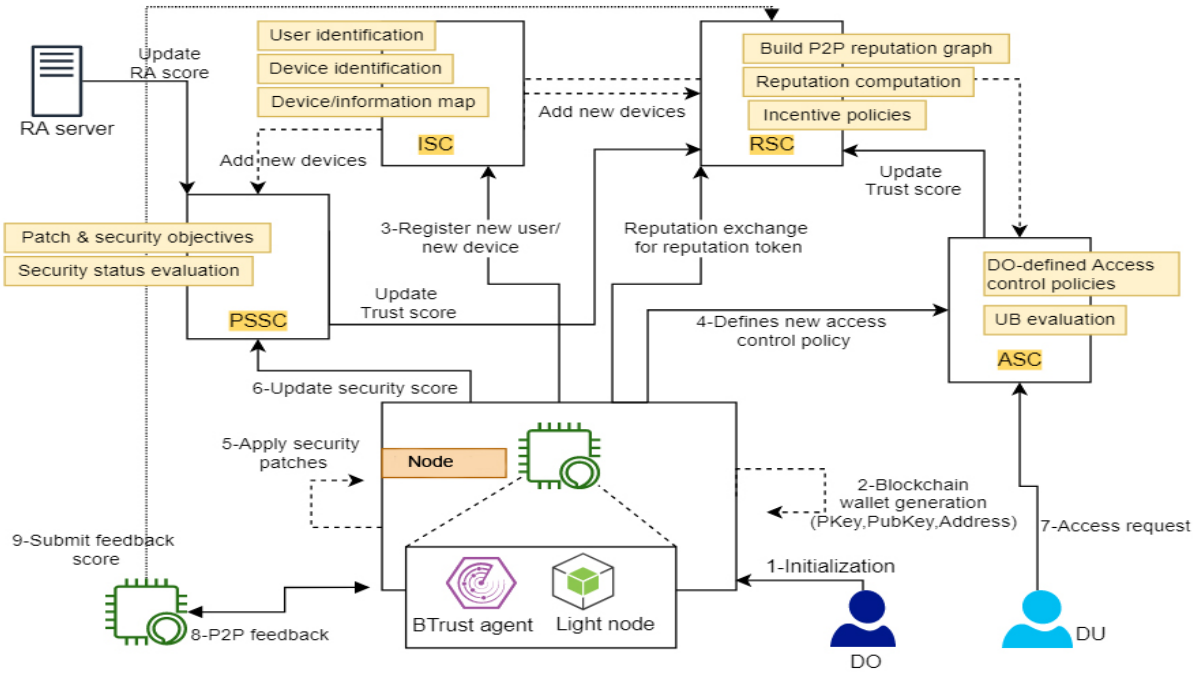

Fig. 3. An overview of the interactions between devices, users and smart contracts in BTrust

The security assessment of the Device We cannot evaluate the trust of a device without considering its security assessment. In this regard, BTrust attempts to determine the security status of a device either through behavioural monitoring, detection of abnormal activities or compliance to required security policies which are updated regularly by the NO (patching security issues, updating vulnerable systems and software, etc.). To conduct the security assessment, each peer is equipped with a BTrust agent, which interacts with a locally hosted anomaly-based IDS (intrusion detection system) to detect anomalies, gather evidence and communicate this information to the RSC to adjust the peer's trust score. The agents also evaluate the compliance of each peer to the security rules and guidelines defined by the NO.

4.4 General Trust Metric

After discussing the importance of the trust parameters involved in BTrust. We formalize the BTrust parameters and present the formula we adopted to compute the trust for each parameter.

P2P networks are decentralized, BTrust builds a virtual trust overlay on top of these networks. Figure 4 shows a trust overlay network. BTrust network

is modelled as a directed edge-weighted graph $G = (V, E)$, where each vertex describes a node in the network, and directed edges are the feedback between peers. We associate a weight to the edges for both directions to express the local trust between the source and the destination peers. To model the peer-to-peer interactions, we assume that peers exchange and rate exchanged transactions. For example, peers can exchange blockchain information (such as block headers) and rate the quality of received data.

In the graph G, the local trust value, a peer accords to others is indicated by the weight of the outgoing edge from the node, whereas the weight of the in-going edges to a peer represents the local trust received from other peers as depicted in Figure 4. Thus, the directed edge P to Q reflects how much P trusts Q. Let $T(P)$ denote the trust score for peer P. The formula is defined in (1) and involves the following parameters:

- $GT(P)$ the global trust value of peer P;
- $T(P, Q)$ the total number of transactions delivered by peer Q to P;
- $UB(P)$ the user's behaviour score; and
- and $ST(P)$ the security assessment score for peer P.

$$T(i) = \alpha * (GT(i) * UB(i)) + \beta * (ST(i)) \tag{1}$$

The coefficients α and β are normalized weight factors that serve to adjust the global trust value.

The adopted trust formula consists of two parts. The first part reflects the degree of trust that other peers have in the subject peer, based on their past experiences, and the second part assesses the security situation of a given peer, reflecting its likelihood of becoming a malicious peer. In the next three subsections, we present how the different parts of the formula are calculated.

P2P Reputation feedback Similarly to most RTMS, we rely on positive and negative feedback to determine a local trust value that helps to compute the global trust of peers. That is, each peer first calculates the local trust values for other peers as described below:

Each time a peer P transacts with another peer Q, it may rate the response received from Q as positive ($tx(P, Q) = +1$) if P is satisfied with the transaction or as negative ($tx(P, Q) = -1$) if the transaction was not satisfactory.

Let $S(P, Q)$ denote the normalized sum of the feedback received by P from Q indicating the amount of satisfaction peer P has with Q. $S(P, Q)$ is a normalized value between 0 and 1. $S(P, Q)$ will be calculated as follow :

First, let $Sum(P, Q)$ denote the sum of the feedback given by Q to P.

$$Sum(P, Q) = \begin{cases} \sum^{T(P,Q)} tx(P, Q) * AR(P, Q), \\ 0, \text{ if } Sum(P, Q) < 0 \end{cases} \tag{2}$$

such that $AR(P, Q)$ denotes the ratio of the satisfactory transactions to unsatisfactory transactions received by P from Q.

$$AR(P,Q) = Ng(P,Q)/T(P,Q) \tag{3}$$

Then, we normalize the result to obtain a score between 0 and 1

$$S(P,Q) = Sum(P,Q)/\sum_{i \in N(P)} Sum(P,i) \tag{4}$$

The local trust value that peer P has about peer Q –denoted as $LT(P,Q)$– can be defined as following :

$$LT(P,Q) = S(P,Q) \tag{5}$$

where $Ng(P,Q)$ denotes the total number of good transactions performed by peer Q with P. Once the local trust with the neighbouring peers is calculated, we can compute the feedback-based global trust value of a peer P as shown in the following formula:

$$GT(P) = \sum_{i \in N(P)} LT(i,P) * (GT(i)/\sum_{j \in N(P)} GT(j)) * (1/1 + e^{-k * Tx_i}) \tag{6}$$

$N(P)$ is the set of neighbouring peers that interacted with P (peers that received transactions from P). GT(P) is not updated unless card(N(P)) is greater than a specific threshold. In the previous formula, Tx_i denotes the total number of transactions provided by peer i to other peers, and k is a parameter that determines how steeply the feedback impact of peers rises with the number of served transactions. The logistic function $(1/1 + e^{-k * Tx_i})$ is leveraged in (6) as a factor to reduce the impact of malicious feedback on honest peers' global trust in the bootstrap phase.

User Behaviour score Various evaluations of the interaction of a user with different services (such as access control or identification), can be incorporated into the metric to help identify malicious users or peers under attack by entities trying to gain illegal access. In other words, an adapted metric that incorporates the users' behaviour, denoted by $UB(P)$ and defined as follows:

$$UB(P) = \sum_{i \in servs(P)} \alpha_i * Us(i) \tag{7}$$

$Us(i)$ represents the score obtained from using the ith service from a set of defined services for the peer P (denoted as $servs(P)$). Different weights α_i can be assigned by the network operator according to the importance of a given network service.

Security assessment score The security assessment score, denoted as $ST(P)$, aims to quantitatively reflect the current security situation of a peer P. We define a peer's security assessment score by aggregating the sum of the required objectives, weighted by the severity of each objective. The $ST(P)$ is defined as follows:

$$ST(P) = \begin{cases} (\sum_{i \in rules(P)} (Fa(i) * Sv(i) * Da(i))) * RA(P) \\ 0, \text{ if } ST(P) < 0 \end{cases} \tag{8}$$

$Sv(i)$ is the severity of an objective i (from 1 to 10), $rules(P)$ denotes the set of security objectives that peer P ought to achieve, $Da(i)$ denotes the patch application delay, and $Fa(i)$ denotes the fulfilment factor, which reflects if a given objective was achieved or not. RA denotes the result of the remote attestation, such that $RA = 0$ if the operation fails, otherwise $RA = 1$.

When a peer first joins the network, it must first fulfil a minimum of security requirements determined by the security policy defined by the NO to acquire an initial trust score (IS). The peer will only be able to interact with and rate other peers if its reputation score is greater than a threshold IS value defined by the NO.

5 BTrust Algorithms

Generally, BTrust proceeds in three phases: an initialization phase, the enrolment phase, and the processing phase. These phases are explained below.

The initialization phase In the initialization phase, the NO builds an information database that includes the security information and available patches for a wide range of devices (models, versions, etc.). It defines the minimal security policy in the security smart contract and the required threshold for a device to start rating other peers. At the same time, the NO defines the initial security actions to be fulfilled by the DO in order to achieve a trusted reputation and indicates their corresponding severities and scores. The NO then defines the exchangeability rules of the BTrust tokens-based incentive mechanism presented in section 6. The manufacturer defines the information about the devices as well as how to evaluate their integrity using remote attestation, whereas the DO installs and sets up a BTrust agent in his devices.

The enrolment phase In the enrolment phase, each user joins the blockchain network by generating a public/private key pair which uniquely identifies each user. Users can then register their devices into the ISC, which maps a device identity (public key or wallet address) to a device ID, an IP address, a blockchain wallet, a DO, and a patch ID. At the end of the initial enrolment phase, the RSC constructs an initial graph of the network composed of the full nodes and the first devices (initially trusted devices).

Algorithm 1: BTrust algorithm for computing peer's trust

Result: $T(P)$
Require α, β, γ,N, $FeedbackThreshold$
initialization;
$ST(P) \leftarrow RetrieveSecurityScore(P)$
$UB(P) \leftarrow RetrieveUserBehaviorScore(P)$
if $N >$ *FeedbackThreshold* **then**
  for $i = 1;\ i < N;\ i = i + 1$ **do**
    $LT(P, i) \leftarrow RetrieveLocalTrust(P, i)$
    $GT(i) \leftarrow RetrieveGlobalTrust(i)$ $T(i) \leftarrow Default$
  end
  while $|GT'(P) - GT(P)| > \gamma$ **do**
    for $i = 1;\ i < N;\ i = i + 1$ **do**
      GT'(P) $\leftarrow \sum_{i \in N(P)} LT(i, P) * (GT(i) / \sum_{j \in N(P)} GT(j))$
    end
  end
end
return $T(P) \leftarrow \alpha * (GT(P) * UB(P)) + \beta * (ST(P))$

The processing phase In the processing phase, the new joining nodes retrieve a list of live peers from a bootstrap DNS server or a cached node list and then select a set of bootstrapping peers using the random walk function defined in algorithm (2). BTrust agents start assessing the device security and the user behaviour, as well as rating other peers using algorithm (1). While computing the trust, the global score is computed iteratively, since the global trust of a peer depends on the global trust of the rating peers, till it converges below a specified threshold(γ). Initially, the algorithm starts with default trust values. As peers obtain feedback from each other, the trust value is updated regularly. Since the trust computation in BTrust is repetitive, we propose that each peer computes its own global reputation, except for the last calculation which should be performed on the smart contract to ensure the veracity of the calculation. In particular, the reputation smart contract calculates and checks the convergence of the global trust –locally computed by the peers. The intermediary values are used to inspect and verify the trust computation performed by the peer.

Figure 3 provides a global overview of different interactions between the components involved in assessing and evaluating a peer's trust in the BTrust network.

5.1 Trust-Based Peer selection using Random walks

In this subsection, we describe how BTrust protocol selects bootstrapping peers or counterparts with whom peers can exchange transactions. Intuitively, an honest peer will tend to interact with the closest peers that have a higher level of reputation and higher trust score. However, this approach will incur a heavy workload for the most reputable and trustworthy nodes in large-scale networks.

To avoid this problem, our selection process considers nodes' reputation and capacity (free inbound connections). Furthermore, we want to assist new joining peers to randomly select their neighbours avoiding peers with a high number of incoming connections (peer's In-degree). The focus of our work is on unstructured P2P systems, where peers select neighbours randomly without any knowledge about the network topology. The selection is performed through random walks over an overlay network based on the reputation graph and peers' degrees.

To evaluate the availability of appropriate peers, we define the pertinence ratio PR as the global trust of a peer 'i' divided by its in-degree (d_i):

$$PR(i) = \begin{cases} T(i)/d_i, & \text{if } d_i > 0 \\ 0, & \text{otherwise} \end{cases} \tag{9}$$

New nodes should preferably connect to peers with similar or higher pertinence ratios. The probability of a new joining node choosing the peer 'i' is defined as the following:

$$P(i) = PR(i) / \sum_{j \in O(t)} PR(j) \tag{10}$$

such that $O(t)$ denotes the set of online nodes at a given time. The probability distribution $P(X = i)$ can be therefore represented as :

$$P(X = i) = P(i) = 1/Z * PR(i) \tag{11}$$

$$Z = \sum_{j \in O(t)} PR(j) \tag{12}$$

As it is impractical to calculate $P(X)$ in wide networks due to the large number of peers, we use a Markov chain Monte Carlo method (MCMC) for sampling from P. More specifically, we use the Metropolis-Hastings (HM) algorithm [38][22] which works by simulating a Markov Chain with a stationary distribution Π. This means that, after enough time, the samples from the Markov chain resemble those of the samples from Π. That is, we model a second overlay P2P network as a connected graph $G' = (V, E)$ with finite node set $V = \{1, 2, .., N\}$ and an edge set E that belongs to V^2. In G', We assign a transition probability and create a self-loop at each node (Fig. 4), such that the total transition probability is 1.

Using HM, we then construct a Markov transition matrix P as :

$$Pij = P(Xn = j / Xn = i) \tag{13}$$

$$Pij = \begin{cases} q(i,j) * \alpha(i,j), & \text{if } j \neq i \\ q(i,j) + \sum_{k \neq i} q(i,k) * (1 - \alpha(i,k)), & \text{otherwise} \end{cases} \tag{14}$$

such that q is the transition kernel, which represents the probability of proposing a move to some state j (peer j) given the current state i(peer i). In our case

$$q(i,j) = 1/(d_i + 1) \tag{15}$$

and

$$\alpha(i,j) = min\{1, p(j) * q(j,i)/p(i) * q(i,j)\} \tag{16}$$

is the acceptance probability for accepting a proposed move from state i (peer i) to state j (peer j). By its definition, the defined MC is reversible, aperiodic and irreducible

By using this algorithm, new joining nodes retrieve a list of neighbouring peers. Each node broadcasts its current connectivity degree to its neighbours to allow other peers to calculate the defined pertinence ratio $PR(i)$ using the global trust retrieved from the reputation smart contract. The pertinence ratio is then assigned as a transition probability to their edges as shown in Figure 4. Next, each node starts multiple walkers, where the number is equivalent to the node's Out-degree. In order to avoid long walks, each walk is limited in time using a TTL value equivalent to the number of iterations in the HM algorithm. The walker moves from one node to another based on the edge probability and the walker's TTL is decremented until it stops ($TTL = 0$). If the node where the walker stops is already connected to the starting node, then the walker moves

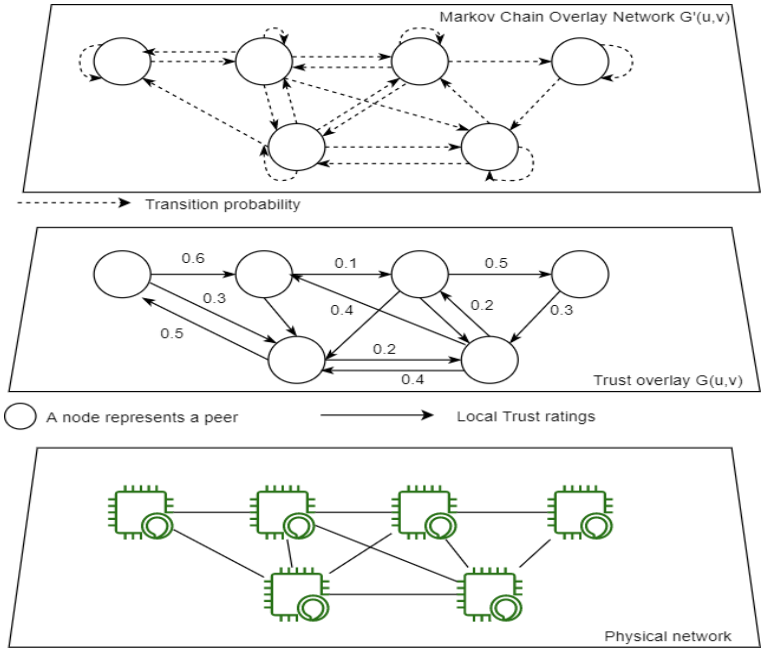

Fig. 4. Trust overlay network in BTrust

Algorithm 2: Random walk algorithm for peer selection

```
Result: PeerList
Require m , P, TTL , Graph G'
Initialization w ← P //The walker starts at P
for i = 1; i < m; i = i + 1 do
    while w is connected to P do
        while TTL > 0 do
            w ← ForwardWalker(TransitionProbability)//Position of the
              walker after moving one step TTL ← TTL − 1
        end
        if w is connected to P and TTL=0 then
            w ← ForwardWalker(TransitionProbability)
        else if w is not connected to P then
            PeerList[i] ← w
    end
end
return PeerList
```

some additional steps. The walk repeats until the walker discovers a suitable node for the new node to join. Each node has a maximum number of allowed inbound and outbound connections.

6 Feedback Quality

Btrust protocol relies partially on p2p feedback rating to compute trust scores. However, two major problems stand in the way of having a reliable feedback exchange; Lack of trustworthy ratings and false feedback. In this section, we propose our solution to address these 2 issues.

6.1 Incentivisation

BTrust proposes a financial incentive mechanism that encourages peers to increase their trust score by behaving honestly and giving correct feedback. The basic idea behind the proposed mechanism is that reputation needs to be monetized by converting it into financial tokens (denoted as reputation tokens). For this specific reason, a peer can choose to convert any value of its trust score into reputation tokens, and once converted, the reputation token can be sold and bought from others by the DO. However, the bought reputation tokens do not promote the reputation of the device owned by the buyer. In this proposed token-based reputation system, losing reputation carries a direct and immediate economic loss. The reputation token and its related operations (conversion, transfer, etc.) are managed by the reputation smart contract.

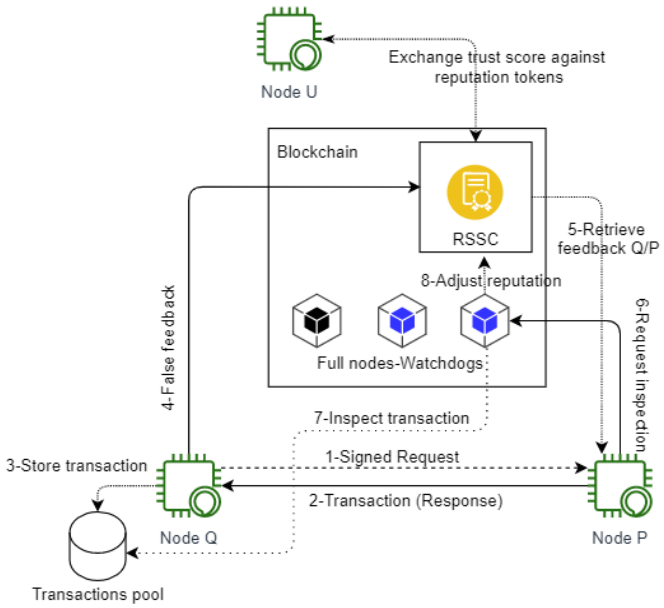

Fig. 5. BTrust watchdog mechanism

6.2 Mitigation of Lack of rating and bad behaviour

In order to limit false feedback and mitigate Bad-mouthing attacks [3], as well as to deter bad behaviour, BTrust relies on special entities we call Watchdogs. The Watchdogs are special BTrust peers –operated by the NO– responsible for inspecting the transactions and evaluating the feedback exchanged between peers. Each WatchDog peer hosts a complete updated copy of the Blockchain. Thus, a sender peer P can request the Watchdog to inspect the delivered transaction and feedback rating. If P does not receive the due rating feedback or it gets false feedback from the requesting peer Q, the watchdog peer can ask peer Q to provide the correct due feedback. This is possible since the reputation feedback provided by the requesting peer Q is stored in the RSC and because service requests and responses are digitally signed by both peers.

In fact, in BTrust when a peer Q requests data from another peer P, the former digitally signs the request. P then responds with a transaction that conveys alongside the requested data, a timestamp, a digital signature (or a message authentication code) of P and the signed request initiated by peer Q. The received transaction is stored in the memory pool of the recipient peer Q, for a limited time –equivalent to the average time needed for a blockchain transaction to be validated–. In the case where a peer is not able to store the received transaction for any reason (e.g. a full pool or unavailable free storage), it can request the watchdog to store them. When peer P doesn't receive the correct feedback it can provide the watchdog with the signed request and response. Afterwards, the

Watchdog can request the peer that missed giving the feedback to provide it to the peer providing the service. If the requesting peer Q refuses the watchdog decreases Q's rating

7 EXPERIMENTAL EVALUATION

In this section, we describe our simulation setup and configurations, and we present the result of the experiments conducted to study the effectiveness of the proposed Protocol. The simulation is based on different experiments that evaluate the proposed trust management algorithm and examine its accuracy in a variety of scenarios.

7.1 Simulation setup

To evaluate the BTrust model, we implemented a simulation model based on the Netlogo 6.1.1 environment [5]. Netlogo is a multi-agent modelling environment for emulating large-scale networks. This environment constructs networks of agents that can be programmed, using a high-level language, to behave and interact with each other according to the defined program. Using Netlogo we designed a cycle-based simulation model for BTrust. For simulation purposes, we assume a P2P file-sharing network as the application scenario of our trust model. At the start of each cycle, we assume that a random number of peers may start a new request for a file, respond to incoming requests, or rate the interactions. For each simulation configuration, we perform five randomized runs for 100 cycles each, on a community of 100 peers. For simulating possible behaviours, we consider three types of peers :

- Honest peer: which always behaves honestly, cooperates with other peers and provides honest feedback afterwards.
- Static malicious peer: which always delivers bad transactions and gives wrong feedback to other peers.
- Dynamic malicious peer: which probabilistically behave maliciously by delivering bad transactions and feedback. We assume that all dynamic malicious peers act honestly in the beginning in order to enhance their trust score.

We use an unstructured P2P architecture with peers responding to incoming requests and providing feedback. We assume the network provides the same service. Table 2 describes the main parameters adopted in our simulations.

In this simulation, nodes are initialized with a global trust value of $1/N$. We assume that dynamic malicious peers act approximately with the same rate denoted as $maliciousness\_rate$. We consider that each node has a maximum of 8 outgoing connections. If one of these outgoing connections is disconnected, the node will try to replace the lost connection by trying to connect to another peer. At the same time, a node may accept up to 50 incoming connections from other peers.

[5] http://ccl.northwestern.edu/netlogo/

Parameters	Default values
α	0.7
β	0.3
$ST(i)$	1
$UB(i)$	1
Maliciousness rate ($maliciousness\_rate$)	100%
Trust score threshold	0.3
Total number of cycles	100
Number of services	1
Peers providing the service	All peers
Convergence threshold (γ)	10^{-4}
Feedback threshold (Witnesses)	5
Percentage of malicious peer	20%
Type of malicious peer	Static

Table 2. Trust simulation default settings

7.2 Experiment 1 : Evaluation of accuracy

In this first experiment, we evaluate the accuracy and effectiveness of BTrust against malicious peers through the calculation of root-mean-square error (RMSE) of aggregated total trust of all peers. RMSE is an estimator of the overall deviations between the predicted and measured values. The RMSE is defined by the following :

$$RMSE = \sqrt{\sum_{i=1}^{N}((T_i - Tc_i)/T_i)^2/N} \tag{17}$$

Where N is the total number of peers in the network, and T_i and Tc_i are the correct and evaluated trust of a peer i, respectively. The RMSE is a good indicator that is inversely proportional to the accuracy of the trust models, such that the lower the RMSE, the more the accuracy of the trust evaluation. The plots in Figure 6 and 7 show respectively the average of RMSE under different rates of malicious static peers and malicious dynamic peers. Within both scenarios, a malicious proportion of 45% produces a low RMSE below a value of 0.2. Thus, the BTrust approach of computing the trust score remains robust when we have a large fraction of dishonest nodes. This result can be explained by the fact that BTrust uses a personalized and adaptive formula to compute trust for each entity. Another reason is the fact that our system, unlike existing decentralized trust systems, replaces score managers with a trustworthy smart-contract based score management system. This obviates the inherent risks of having malicious score managers; i,e., a malicious trust manager could intentionally or under attack affect the computation of the trust value of a given peer. Moreover, In the BTrust model, we leverage watchdogs for feedback verification to filter out malicious feedback.

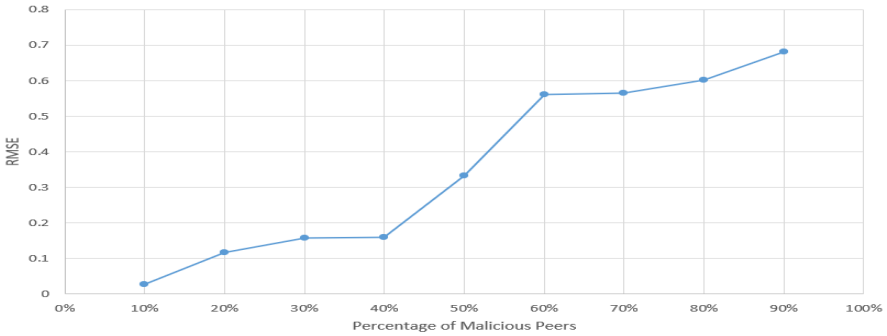

Fig. 6. Trust estimation error (RMSE) in presence of different portions of static malicious peers

7.3 Experiment 2: Effectiveness against dynamic and static Malicious Peers

In this second experiment, we focus on evaluating the effectiveness of the BTrust model in resisting dynamic attacks without leveraging the feedback correction mechanism (Watchdogs). In that regard, we evaluate the successful transaction rate (STR) in the presence of dynamic malicious nodes that oscillate between malicious and honest behaviour at random with a probability of 0.5. The STR is the ratio of the number of successful transactions over the total number of transactions. It is a typical metric used to evaluate the efficiency of trust models. We assume these peers behave honestly up to some time (10th cycle) to accumulate trust before they start behaving maliciously. Within this experiment, we maintain the default percentage of malicious nodes in the network (20%).

Figure 8 illustrates how the level of trust evolves during 100 cycles. We observe that the STR decreases proportionally to the *maliciousness_rate*. In

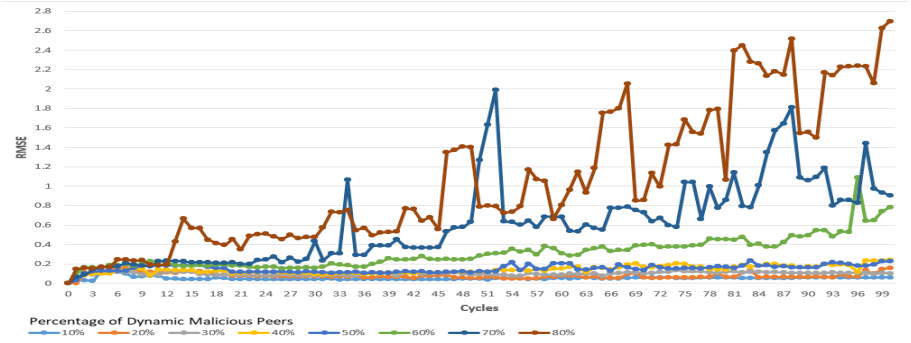

Fig. 7. Trust estimation error(RMSE) in presence of different portions of dynamic malicious peers, during 100 cycles

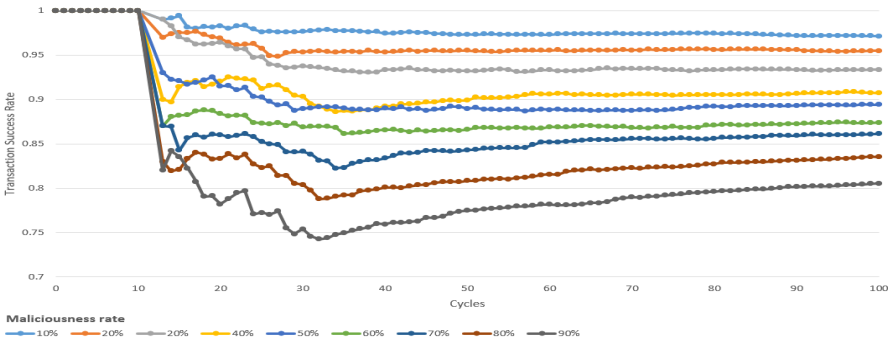

Fig. 8. Successful transaction rates under different scales of dynamic attacks

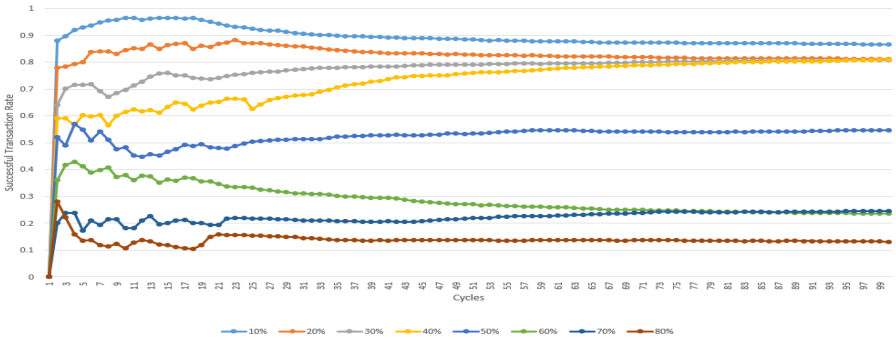

Fig. 9. Successful Transaction Rate(RMSE) in the presence of different proportions of static malicious peers

the context of a low and moderate $maliciousness\_rate < X$ within the interval[0.1,0.2], we observe a weak variation of STR (0.1%). When the malicious peers switch from behaving honestly to behaving maliciously, the STR decreases quickly and, after an average of 40 cycles, the STR stabilizes at a high rate. This cadence can be explained by the fact that the trust score of dynamic malicious peers decreases, even though they behave correctly from time to time in order to regain the trust score. This result reflects that the correctness of the BTrust model is very high even in the presence of dynamic malicious peers. Moreover, the STR remains below 1 as the malicious peers continue to exchange malicious transactions and feedback between them without affecting honest peers.

In a second scenario, we compute the STR in the presence of different proportions of static malicious peers. As shown in Figure 9, a high proportion (over 60%) destabilises the level of trust within the system and thus decreases significantly the STR. Inversely, the STR remains at a high level of 0.8 when we have a malicious proportion of less than 40%. This result reflects the resilience of the BTrust model against a large proportion of static malicious actors.

7.4 Experiment 2 : Convergence of BTrust

In this simulation, we focus on evaluating the convergence speed of BTrust and its scalability with regard to the increasing network. The convergence speed is measured as the number of iterations needed before the trust score converges. Thus, a lower value of the convergence iterations means a higher convergence speed. Interestingly, the results shown in Figure 10, confirm that the convergence speed is very fast since each node needs a maximum of 2 iterations before trust score computation converges. The results prove the scalability of BTrust concerning the number of iterations needed to converge since the latter does not grow substantially with the increase (from 100 to 10000) of the number of peers in the network.

7.5 Load distribution

In this experiment, we evaluate the load of individual peers incurred by BTrust. The metric used for this evaluation is the number of times a particular peer(service) is requested to deliver a transaction. To measure the amount of variation across the network, we compute the standard deviation of the load among all the peers. The simulation is performed after 10000 transactions, the equivalent of an average of 100 transactions per peer, in the presence of a varying-size minority of malicious peers (5-80%). The load computation is performed only on honest peers since they are the preferred targets of other peers and thus the most likely to incur heavy load. Figure 11, shows that the standard deviation of the load distribution is very minimal and does not change significantly as the size of the malicious minority increases. Thus, BTrust does not incur a heavy load on the peers, since the adopted random walk process privileges peers with a high pertinence ratio.

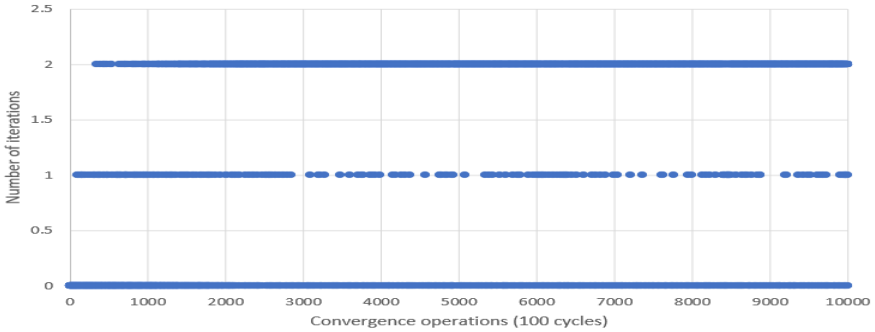

Fig. 10. Number of iterations required for convergence by each node within 100 cycles

8 Conclusion and future work

We have presented a new BRTMS, BTrust that combines multiple determinant factors for quantifying and comparing the trustworthiness of peers based on a P2P feedback system, user behaviour and the security assessment of the devices. We also presented an effective peer selection approach, based on random walks, which does not incur heavy load and computation in large networks. Moreover, BTrust is an extensible and modular protocol, since its underlying concepts, used to compute trust, are generic and thus can be extended. For instance, in order to capture a more accurate evaluation of a user's behaviour, we can incorporate additional services such as Spamming activity or behavioural patterns.

Based on the simulation findings we conclude that our trust model is able to fulfil all design considerations announced in the introduction. Especially the ability to isolate malicious peers in P2P networks without requiring a central authority. Moreover, the results proved the effectiveness and efficiency of the proposed trust model under various attacks.

This paper presents an initial version of the BTrust protocol; our research continues by investigating multiple enhancements. First, to ensure improved privacy, we are investigating the use of Direct Anonymous Attestation instead of sending remote attestation data to a trusted server. Second, we envision a decentralized evaluation of the remote attestation. Third, we are working towards using off-chain models for logging local trust scores, using solutions such as lightning networks, whereby two nodes can mutually rate each other contentiously without overcharging the blockchain with reputation transactions and to reduce validation latency. Fourth, given that secure processing and transmission of trust data were not addressed in this paper, we are working towards implementing such mechanisms.

References

1. Yousef Alowayed, Marco Canini, Pedro Marcos, Marco Chiesa, and Marinho Barcellos. Picking a partner: A fair blockchain based scoring protocol for autonomous systems. *ANRW 2018 - Proceedings of the 2018 Applied Networking Research Workshop*, pages 33–39, 7 2018.
2. HA Kurdi Journal of King Saud University-Computer and and undefined 2015. HonestPeer: An enhanced EigenTrust algorithm for reputation management in P2P systems. *Elsevier*.
3. Z. Banković, J. C. Vallejo, D. Fraga, and J. M. Moya. Detecting bad-mouthing attacks on reputation systems using self-organizing maps. In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, volume 6694 LNCS, pages 9–16, 2011.
4. Johannes Blömer, Jakob Juhnke, and Christina Kolb. Anonymous and publicly linkable reputation systems. In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, volume 8975, pages 478–488. Springer Verlag, 2015.
5. Matthew Buechler, Manosai Eerabathini, Christopher Hockenbrocht, and Defu Wan. Decentralized Reputation System for Transaction Networks.

6. Vitalik Buterin. A next-generation smart contract and decentralized application platform. *Etherum*, (January):1–36, 2014.
7. Marcello Cinque, Christian Esposito, and Stefano Russo. Trust management in fog/edge computing by means of blockchain technologies. In *2018 IEEE International Conference on Internet of Things (iThings) and IEEE Green Computing and Communications (GreenCom) and IEEE Cyber, Physical and Social Computing (CPSCom) and IEEE Smart Data (SmartData)*, pages 1433–1439. IEEE, 2018.
8. Richard Dennis and Gareth Owen. Rep on the block: A next generation reputation system based on the blockchain. *2015 10th International Conference for Internet Technology and Secured Transactions, ICITST 2015*, pages 131–138, 2 2016.
9. Roberto Di Pietro, Xavier Salleras, Matteo Signorini, and Erez Waisbard. A blockchain-based trust system for the internet of things. In *Proceedings of ACM Symposium on Access Control Models and Technologies, SACMAT*, pages 77–83. Association for Computing Machinery, 6 2018.
10. Roberto Di Pietro, Xavier Salleras UPF, Matteo Signorini, and Erez Waisbard. A blockchain-based Trust System for the Internet of Things. *dl.acm.org*, pages 77–83, 6 2018.
11. Ali El Kaafarani, Shuichi Katsumata, and Ravital Solomon. Anonymous Reputation Systems Achieving Full Dynamicity from Lattices. In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, volume 10957 LNCS, pages 388–406. Springer Verlag, 2018.
12. BELLINI EMANUEL, IRAQI youssef, and DAMIANI ERNESTO. Blockchain-Based Distributed Trust and Reputation Management Systems: A Survey. *IEEE ACCESS*, 8, 2020.
13. Ben Fisch, Joseph Bonneau, Nicola Greco, and Juan Benet. Scaling Proof-of-Replication for Filecoin Mining.
14. Lydia Garms and Elizabeth A. Quaglia. A new approach to modelling centralised reputation systems. In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, volume 11627 LNCS, pages 429–447. Springer Verlag, 2019.
15. Josef Gattermayer and Pavel Tvrdik. Blockchain-Based Multi-Level Scoring System for P2P Clusters. *Proceedings of the International Conference on Parallel Processing Workshops*, pages 301–308, 9 2017.
16. B Hamdaoui, M Alkalbani, A Rayes IEEE Internet of Things . . . , and Undefined 2020. IoTShare: A Blockchain-Enabled IoT Resource Sharing On-Demand Protocol for Smart City Situation-Awareness Applications. *ieeexplore.ieee.org*, 2020.
17. Mahmood Hosseini, Constantinos Marios Angelopoulos, Wei Koong Chai, and Stephane Kundig. Crowdcloud: a crowdsourced system for cloud infrastructure. *Cluster Computing 2018 22:2*, 22(2):455–470, 8 2018.
18. Atia Javaid, Maheen Zahid, Ishtiaq Ali, Jalees Ul, Hussen Khan, Zainib Noshad, and Nadeem Javaid. Reputation System for IoT Data Monetization Using Blockchain. *Springer*, 97:173–184, 2020.
19. Sepandar D. Kamvar, Mario T. Schlosser, and Hector Garcia-Molina. The EigenTrust algorithm for reputation management in P2P networks. In *Proceedings of the 12th International Conference on World Wide Web, WWW 2003*, pages 640–651, 2003.
20. Michael Kiperberg, Amit Resh, and Nezer J Zaidenberg. Remote Attestation of Software and Execution-Environment in Modern Machines. Technical report.

21. Kiran Kumar Kondru, R. Saranya, and Annamma Chacko. A Review of Distributed Supercomputing Platforms Using Blockchain. *Lecture Notes in Networks and Systems*, 127:123–133, 2021.
22. KW Kwong, DHK Tsang IEEE/ACM transactions on, and undefined 2008. Building heterogeneous peer-to-peer networks: protocol and analysis. *ieeexplore.ieee.org*.
23. Zhaojun Lu, Qian Wang, Gang Qu, and Zhenglin Liu. BARS: A Blockchain-Based Anonymous Reputation System for Trust Management in VANETs. *Proceedings - 17th IEEE International Conference on Trust, Security and Privacy in Computing and Communications and 12th IEEE International Conference on Big Data Science and Engineering, Trustcom/BigDataSE 2018*, pages 98–103, 9 2018.
24. T McConaghy, R Marques, A Müller ... paper, undefined BigChainDB, and undefined 2016. Bigchaindb: a scalable blockchain database. *git.berlin*, 2016.
25. Axel Moinet, Benoit Darties, and Jean Luc Baril. Blockchain based trust & authentication for decentralized sensor networks, 6 2017.
26. Axel Moinet, Benoît Darties, and Jean-Luc Baril. Blockchain based trust & authentication for decentralized sensor networks. 6 2017.
27. N Szabo First Monday and undefined 1997. Formalizing and securing relationships on public networks. *firstmonday.org*.
28. Satoshi Nakamoto. Bitcoin : A Peer-to-Peer Electronic Cash System. pages 1–9, 2008.
29. Jens Neureither, Alexandra Dmitrienko, David Koisser, Ferdinand Brasser, and Ahmad Reza Sadeghi. Legiot: Ledgered trust management platform for iot. In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, volume 12308 LNCS, pages 377–396. Springer Science and Business Media Deutschland GmbH, 2020.
30. Alexander Schaub, Rémi Bazin, Omar Hasan, and Lionel Brunie. A Trustless Privacy-Preserving Reputation System. *IFIP Advances in Information and Communication Technology*, 471:398–411, 2016.
31. Mike Sharples and John Domingue. The Blockchain and Kudos: A Distributed System for Educational Record, Reputation and Reward. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 9891 LNCS:490–496, 2016.
32. JJ Sikorski, J Haughton, M Kraft Applied Energy, and Undefined 2017. Blockchain technology in the chemical industry: Machine-to-machine electricity market. *Elsevier*.
33. Rajeev Singh, Sudeep Tanwar, and Teek Parval Sharma. Utilization of blockchain for mitigating the distributed denial of service attacks. *Security and Privacy*, 3(3):e96, 5 2020.
34. Shawn Wilkinson, Shawn Wilkinson, Tome Boshevski, Josh Brandoff, and Vitalik Buterin. Storj A Peer-to-Peer Cloud Storage Network. 2014.
35. Li Xiong and Ling Liu. PeerTrust: Supporting Reputation-Based Trust for Peer-to-Peer Electronic Communities NNexus View project Deep Learning on Graph View project. *ieeexplore.ieee.org*.
36. Zhe Yang, Kan Yang, Lei Lei, Kan Zheng, and Victor C.M. Leung. Blockchain-based decentralized trust management in vehicular networks. *IEEE Internet of Things Journal*, 6(2):1495–1505, 4 2019.
37. H Yao, T Mai, J Wang, Z Ji, C Jiang IEEE Transactions on ..., and Undefined 2019. Resource trading in blockchain-based industrial Internet of Things. *ieeexplore.ieee.org*.
38. Ilker Yildirim. Bayesian Inference: Metropolis-Hastings Sampling. Technical report, 2012.

39. Runfang Zhou and Kai Hwang. PowerTrust: A Robust and Scalable Reputation System for Trusted Peer-to-Peer Computing *. Technical report, 2005.
40. Jun Zou, Bin Ye, Lie Qu, Yan Wang, Mehmet A. Orgun, and Lei Li. A Proof-of-Trust Consensus Protocol for Enhancing Accountability in Crowdsourcing Services. *IEEE Transactions on Services Computing*, 12(3):429–445, 5 2019.

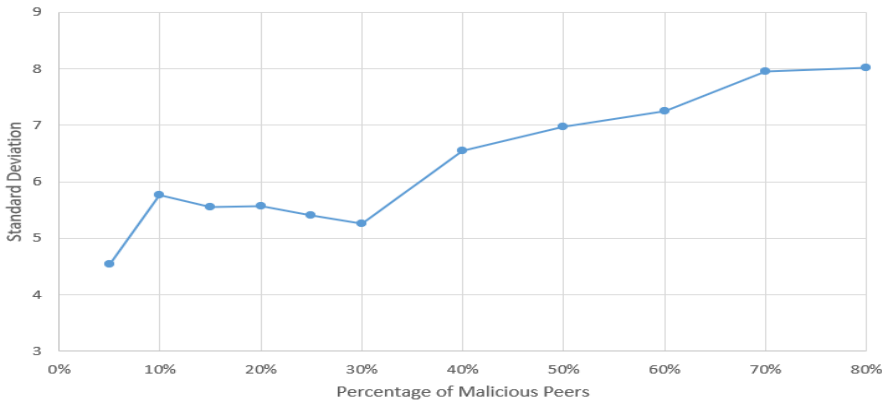

Fig. 11. The standard deviation of the load distribution in the network